Journal of applied logics (Print) ISSN 2631-9810
Journal of applied logics (Online) ISSN 2631-9829

Journal of Applied Logics

The IfCoLog Journal of Logics and their Applications

Volume 12 • Issue 6 • October 2025

Special Issue To Celebrate Dov Gabbay's 80th Birthday

Guest Editor
Jane Spurr

Available online at
www.collegepublications.co.uk/journals/ifcolog/
Free open access

Journal of Applied Logics - IfCoLog Journal of Logics and their Applications

Volume 12, Number 6

October 2025

Disclaimer

Statements of fact and opinion in the articles in Journal of Applied Logics - IfCoLog Journal of Logics and their Applications (JALs-FLAP) are those of the respective authors and contributors and not of the JALs-FLAP. Neither College Publications nor the JALs-FLAP make any representation, express or implied, in respect of the accuracy of the material in this journal and cannot accept any legal responsibility or liability for any errors or omissions that may be made. The reader should make his/her own evaluation as to the appropriateness or otherwise of any experimental technique described.

ISBN 978-1-84890-492-7
ISSN (E) 2631-9829
ISSN (P) 2631-9810

College Publications
Scientific Director: Dov Gabbay
Managing Director: Jane Spurr

http://www.collegepublications.co.uk

Editorial Board

Scope and Submissions

This journal considers submission in all areas of pure and applied logic, including:

pure logical systems
proof theory
constructive logic
categorical logic
modal and temporal logic
model theory
recursion theory
type theory
nominal theory
nonclassical logics
nonmonotonic logic
numerical and uncertainty reasoning
logic and AI
foundations of logic programming
belief change/revision
systems of knowledge and belief
logics and semantics of programming
specification and verification
agent theory
databases

dynamic logic
quantum logic
algebraic logic
logic and cognition
probabilistic logic
logic and networks
neuro-logical systems
complexity
argumentation theory
logic and computation
logic and language
logic engineering
knowledge-based systems
automated reasoning
knowledge representation
logic in hardware and VLSI
natural language
concurrent computation
planning

This journal will also consider papers on the application of logic in other subject areas: philosophy, cognitive science, physics etc. provided they have some formal content.

Submissions should be sent to Jane Spurr (jane@janespurr.net) as a pdf file, preferably compiled in LaTeX using the IFCoLog class file.

Contents

REMINISCENCES

ARTICLES

Editorial

Jane Spurr

I have been working with Dov now for almost half his lifetime (which in itself is worth a mention). What he has achieved in these 38 years is quite staggering.

Over that time I've had the honour of being a sounding board for his ideas, as well as converting reams[1] of handwritten pages into a reader-friendly typewritten manuscript. Of course, there was the little issue of having to learn to use LaTeX first!!

I have had the privilege of launching a myriad of new ventures with him.

First and foremost have been the Handbooks. One contributor to this issue has suggested that there have been over 50 of them. Then there are the Journals. The *Journal of Logic and Computation* was started from scratch, published by Oxford University Press and is now in its 35th year of publication. The *Logic Journal of the IGPL* was born from the Interest Group in Propositional Logic – another brainchild of Dov's – and has also been published by Oxford University Press since 1999.

Then, there has been the charity. IfCoLog, which sponsors the publication of the *Journal of Applied Logics. The IfCoLog Journal of Logics and their Applications.* The *IfCoLog Journal* was launched in 2014 and, following permission from Elsevier to take over the *Journal of Applied Logic*, morphed into its current format in 2018.

Along the way, we managed to develop College Publications into the publishing house that it is today, with some 600 titles. This was a combination of timing and opportunity. Print-on-demand publishing was just beginning, and we were approached by a colleague asking for help, when they found that their "traditional" academic publishing house no longer wanted to publish multi-authored volumes of post-conference proceedings. Hence, the birth of College Publications!

And, while these enterprises were bubbling along, Dov found the time and energy to lecture, nurture and travel. Quite remarkable achievements.

This volume is a special issue to mark Dov's 80th birthday. The response to the invitation that I put out (at quite short notice) was wonderful, and I sincerely thank

[1] Literally! Dov's writing isn't small, he doesn't write on lines, and sometimes he manages to get 40 or 50 words on a page.

all those who have been able to make a contribution. My special thanks to Dov's wife, Lydia, for drawing the image of him for the front cover.

It's very sad to have to acknowledge an ever-growing list of Dov's colleagues and collaborators whose absence here will be noted. Most specifically, John Woods, who died last year, and Jörg Siekmann, whose health prevents him from making a contribution. We are also sad to announce that Dimitrij Skvortsov passed away shortly before the publication of this issue.

I have enjoyed reading the contributions, especially in the Reminiscences section, and none of you will be surprised to read about Dov's academic achievements, but I do wonder how many of you knew about his skills with a football? Another hidden talent is his ability to paint woodwork! After having a very large bookcase/desk arrangement built at home (probably 30 years ago), Dov came over for a day to help me paint it. Many hands made light work!

Finally, I would like to thank Dov for his endless encouragement and belief in me over the years, sentiments that he has extended to my entire family.

So, on this personal note...

Happy birthday, Dov.

Received September 2025

Memories of our Father

Jamie (Murdoch) Gabbay, Mike (Michael) Gabbay and Medan Gabbay

Jamie Gabbay

When I was about ten years old I was playing lego on the floor of my bedroom when my father came in and tried to explain Modus Ponens to me.

If you know “A” and you know that “$A \to B$” then you can deduce “B”, he said, and he wrote it down on a sheet of paper:

1. A (assumption)
2. $A \to B$ (assumption)
3. B (from 1 & 2 by MP)

I remember thinking “For this they pay him money?”. I showed no interest and got back to playing with my lego. Deflated, my father gave up.

(It only occurred to me many years later that lego is quite similar to what he was trying to teach me.)

Yet, it bothered me what he had said. If we know A and we know that A implies B then of course we know B. I wondered: why is this non-obvious enough to be worth spelling out? Why write down the assumptions? Why number them? When we apply MP to deduce B, why list the numbered assumptions? If A and $A \to B$ are true then they are true, and if they are not true then why would we assume them?

Every so often, for years, I would remember this and think about it.

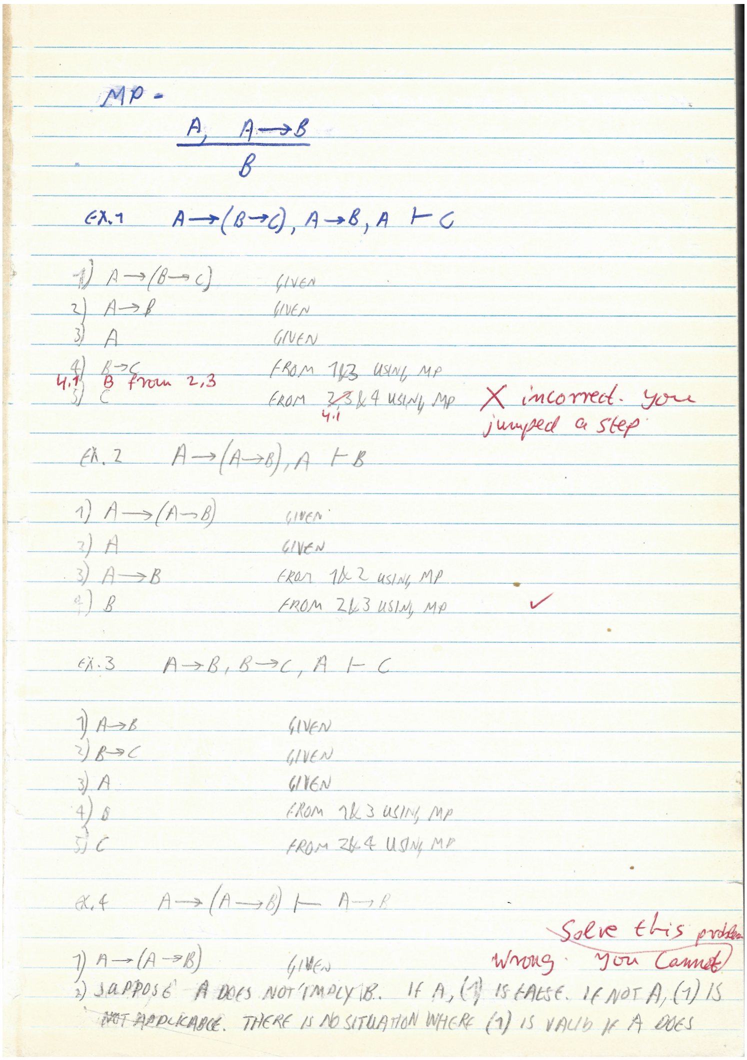
In due course I went to university to study maths. There was only one logic course and it was quite brief, but it contained A and $A \to B$ and Modus Ponens, and I recognised them as old friends. By then I could appreciate them better. I took to the course and went into an academic career in logic and foundations of mathematics, and published many papers on A and $A \to B$.

I wonder how much of what I do today was seeded by that day in London in the early 1980s, when I, surrounded by my lego, was introduced to logic. I didn’t appreciate it but it must have planted a concept in my head, because I still remember it.

By some miracle, I even have one of the jotter notebooks that my father and I used. I found it during a house clearout. I attach a photo.

I took another lesson from this: when I tell my son something and he appears not to understand, I am patient. Some ideas take time to sink in. Some ideas take a lifetime.

Thank you Dad.



MP -

$$\frac{A, \quad A \to B}{B}$$

EX.1 $A \to (B \to C), A \to B, A \vdash C$

1) $A \to (B \to C)$ GIVEN
2) $A \to B$ GIVEN
3) A GIVEN
4) $B \to C$ FROM 1&3 USING MP
4.1 B from 2,3
5) C FROM ~~2~~ 4.1,3 & 4 USING MP

X incorrect. you jumped a step

EX. 2 $A \to (A \to B), A \vdash B$

1) $A \to (A \to B)$ GIVEN
2) A GIVEN
3) $A \to B$ FROM 1&2 USING MP
4) B FROM 2&3 USING MP ✓

EX.3 $A \to B, B \to C, A \vdash C$

1) $A \to B$ GIVEN
2) $B \to C$ GIVEN
3) A GIVEN
4) B FROM 1&3 USING MP
5) C FROM 2&4 USING MP

EX.4 $A \to (A \to B) \vdash A \to B$

Solve this problem

1) $A \to (A \to B)$ GIVEN

Wrong. You Cannot

2) SUPPOSE A DOES NOT IMPLY B. IF A, (1) IS FALSE. IF NOT A, (1) IS ~~NOT~~ APPLICABLE. THERE IS NO SITUATION WHERE (1) IS VALID IF A DOES

Mike Gabbay

My mother would occasionally comment, on a footballing afternoon, that my father used to be an excellent footballer. A claim that was backed up on one or two occasions when my father would display some ball skills: nothing fancy or inspiring, just some dexterous ground control on the spot before passing the ball back to me.

I also recall my mother commenting on body shape. She'd observe that he had sportsman's legs, and would note with irony that hers, despite having been put to extremely good sporting use (field hockey, karate), were not as suited to such activities as his would have been.

Despite these hints here and there regarding my father's physical potential, I never thought of him as being, or even capable of being, a sportsman. He took up tennis in his 50s I recall, and as a teenager I would approach this and other sporting excursions very competitively. It was, after all, my only opportunity to express some dominance.

In my early 20s having spent my whole life dismissing my father's athletic prowess. I was at a conference with him, perhaps an ESSLLI, at which there was organised a lecturers vs students football match. That particular year the turnout from the lecturers was rather meagre, so a team of mostly lecturers was formed to play a team of students. Even more unusual was that my father allowed himself to be coerced into playing, making up the numbers on the lecturer's team. I hardly ever played football since the early years of secondary school, but was nevertheless firmly part of the student team.

I regarded myself as a decent footballer, able to contribute as a defender, not in the modern style, but in the vintage style common till the late 80s, where the main skills required of the back four were persistent obstruction and reckless slide-tackling. I was then most amused to see my father, then well into his 50s, not only take to the field but take position in goal. The amusement turned to surprise and then awe as I saw him, from the other side of the pitch, make save after save. Not just deflections, but full bodied dives and drops to the ground, completely smothering and keeping the ball, after which he would deftly rise and pass it out to one of his teammates. The fearlessness with which he took to the ground was humbling, he was a natural. Even as a child I, and many of my friends, would have been hesitant to make dives like that. And for sure, as adults, the possibility of a painful landing would have deterred most of those present from playing in goal at all, let alone from playing with such wholehearted physicality. It is an enduring vision for which I will be forever grateful.

Perhaps a year later when my department decided to form a football team and were desperately short of players, I agreed to join them. Unsurprisingly, nobody wanted to play in goal, so I, with my father's example in mind, took up that position and proceeded to make, a number of daring saves just like he did (although I had a slightly better haircut). I continued to play in that position in the occasional inter-departmental fixture for a couple of years, and each time drew inspiration from that one game at ESSLLI.

This inspiration has filtered down to my own child, to whom I try to provide a strong and fit example. She has inherited the physique and athleticism of her mother and also my father, and the option of a sporting career remains as realistic to her as does an academic one. The motivation for me to improve myself to motivate her derives substantially from that day, seeing my father amaze everyone with his goalkeeping skills.

But on the other hand, I can't help but look back at that moment and think of it in the context of the odd hint my mother would drop about my father's sporting potential. The man could have been a sportsman, a serious contender in his chosen sport, a star player. And with it, very very wealthy. I see the riches bestowed on top athletes now and think that I've missed out on the luxurious life of a member of a top sportsperson's family. I imagine their sons and daughters with their mansions and sports cars, and just like my older brother, but with more irony, I think to myself:

Thank you, Dad!

Medan Gabbay

I never understood my father. I remember the feeling of awe looking at a man I knew touched a greatness I would perhaps never understand. He wasn't fixing bikes or telling jokes (at least not funny ones); his mind lost somewhere between theorem and eternity. I do recall him saying as a young man he faced a choice, growing up in Israel he almost became a Rabbi. . . but instead he chose to honour his God through the purity of mathematics and over 80 years his unbroken prayer of logic has been written across hundreds of books and papers. Watching him I imagined Moses staring at the burning bush — with no understanding of what he was seeing but knowing it was magnificent and somehow changing the very meaning of our universe. Although I could never quite follow the formulas, his relentless dedication and life's work has illuminated paths for so many great minds. I have witnessed the journey with my ever-childlike understanding of that path and with the unwavering love of a child for his father.

Received September 2025

The Influence of Dov on My Work

Arnon Avron
School of Computer Science, Tel Aviv University, Tel Aviv, Israel
aa@tauex.tau.ac.il

I met Dov for the first time about 45 years ago, when I was still working on my Ph.D in mathematics under the supervision of Prof. Haim Gaifman. It is remarkable that although Dov is only seven years older than me, he was then already a full professor of logic at Bar-Ilan university and known worldwide as a great logician.

My Ph.D was devoted to relevant and paraconsistent logics. Since Dov was practically the only specialist in Israel on (among other subjects...) non-classical logics at the time I was a Ph.D student, it was very natural that I would try at a certain point to meet him and talk with him about what I am doing and about problems that I have. Our discussions since that time have always been very fruitful for me. Thus, in one of our first meetings I described to him a certain system I have reached, and he suggested to me to look at its possible connections with Dummett's system **LC** (now usually known as Gödel's fuzzy logic **G**). Following this suggestion, I first looked up at the literature in order to learn what **LC** is, since I had never heard about it before. After doing that I discovered that the system I told Dov about is actually equivalent to **LC**! From that point **LC** has become very important for my research. In particular: I have developed the proof-theoretical framework of hypersequents in order to find cut-free calculi for it and for the strongly related semi-relevant system **RM**. (Interestingly, my proof system for **LC** had been the starting point of the area of proof theory for fuzzy logics, to which Dov himself contributed later. See [4].)

Another, even more important, influence of Dov on my research at that time and ever since was made through his book [2]. From that book I learned for the first time the notion of consequence relation (CR), as well as the difference between Tarski's CRs and Scott's CRs. Dov's excellent presentation of the subject at that book completely changed my whole view on logics, and since then the notion of CR became central to my work.

The pattern of my cooperation with Dov that started when I was a Ph.D student remains the same after that. I have always benefited from our discussions and the suggestions or hints that Dov has given me in them. Therefore I have never missed an opportunity to meet him, whether it was in the many times that I happened to be in London, or when Dov was visiting Israel. What is more, although we have never managed to write a joint paper (despite the several times we were planning to do so), many of my scientific publications were born as a result of Dov pushing me to contribute to the many handbooks, or collections of papers, or series of books that he was editing. It started with the contribution he asked me to give to [3], and culminated with the book [1], that Dov was his publisher.

Finally, I should mention how grateful I am to the ways Dov has affected my academic career: he served as one of the referees of my Ph.D Thesis; he wrote letters on my behalf in some of the stages of the long process of getting the position of full professor at the department of computer science at Tel Aviv university (including the most crucial one: getting *some* position there after my postdoc period at Edinburgh); and making me one of the editors of some of the several journals that he is editing.

Thank you, Dov, for everything. I wish you many more years of being active and productive as you have always been and still are. AD 120![1]

References

[1] A. Avron, O. Arieli, and A. Zamansky, **Theory of Effective Propositional Paraconsistent Logics**, Studies in Logic (Mathematical Logic and Foundations) vol. 75, College Publications, 2018.

[2] Gabbay D. M., **Semantical Investigations in Heyting's Intuitionistic Logic**, Synthese Library vol. 148, Springer, 1981.

[3] Gabbay D. M. (editor), **What is a Logical System**, Studies in Logic and Computation vol. 4, Oxford University Press, 1994.

[4] Metcalfe G., Olivetti N. and Gabbay D. M., **Proof Theory for Fuzzy Logics**, Applied Logic Series vol. 36, Springer, 2009.

[1] This is the usual blessing in Hebrew for birthdays. It means: may you live like Moses till the age of 120 years.

Received October 2025

Homage to Dov Gabbay on his 80th Birthday

Rolf Nossum

Dov Gabbay's career seems, to me at least, to have been an unbroken series of conquests, ever accelerating along the path to deeper knowledge of symbolic logic. Mine has been nothing of the sort. More than once I have abandoned efforts that ceased to yield results and taken up new research interests. It was at one of those impasses that I met Dov.

We were in a project funded by the European Commission, headed up by Imperial College in London (where Dov was then), and encompassing research groups from over a dozen different countries. To my admission that I felt like a student of the subject matter at hand, and needed some guidance to get started, Dov simply replied «Well, we are all students of this, that or the other», and suggested a couple of things for me to look into. Our working relations were pleasant and fruitful, and papers started to come out. When the Scandinavian AI Conference in Stockholm gave us their «Best Paper» award, we knew we were on the right track. I spent two sabbaticals in London, where Imperial College was kind enough to bend a rule or two in my favour. Dov reciprocated by occasionally visiting my regional college in Agder, Norway, and Kristiansand Municipality made available some funds to support his visits, as part of their programme for Agder College to rise to university status.

I was visiting one of the other research groups from the European project I mentioned above, in Trento, Italy, when I got a phone call to inform me that I had been promoted to full Professor. The citation had mentioned my joint papers with Dov Gabbay as the main basis for promoting me. That evening, the excellent cuisine of the Alto Adige, and its magnificent wines, contributed to a memorable celebration with my Italian colleagues. I was almost two decades older than Dov was when he became a Professor in Tel Aviv.

Agder College finally became the University of Agder and decided that they would award a small number of doctorates honoris causa once every five years. It was a great pleasure to be Dov's designated host during the festivities where he was among the first to receive his honorary doctorate from the University of Agder, in well-deserved recognition of his contributions to transforming a Norwegian regional college into a research university. It is with gratitude and admiration that I salute Dov Gabbay on his 80th birthday.

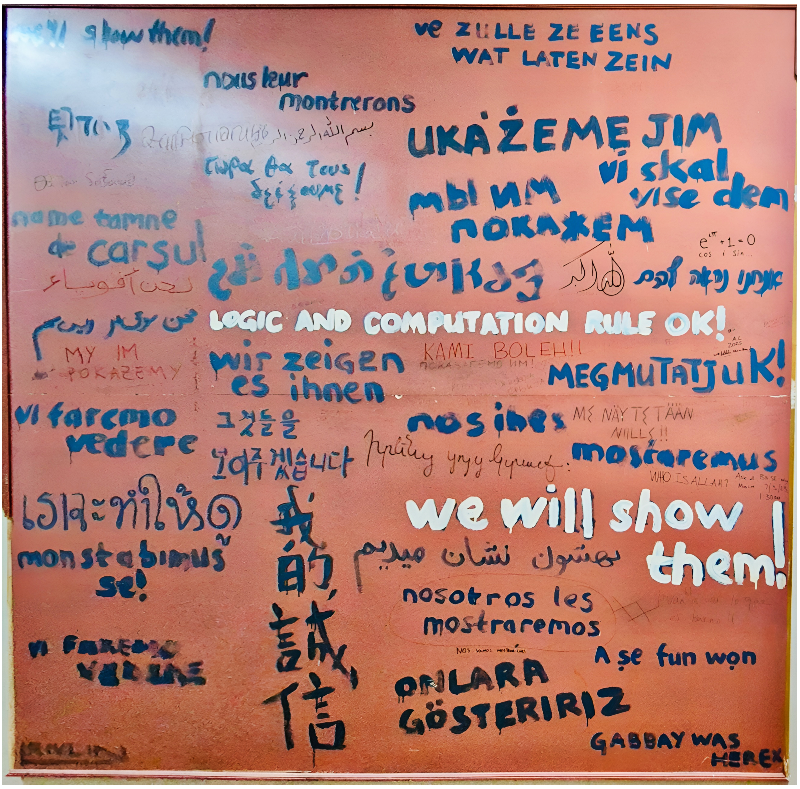
Received August 2025

Dear Dov – A note of appreciation on your 80th birthday

Odinaldo Rodrigues
Kings College London
odinaldo.rodrigues@kcl.ac.uk

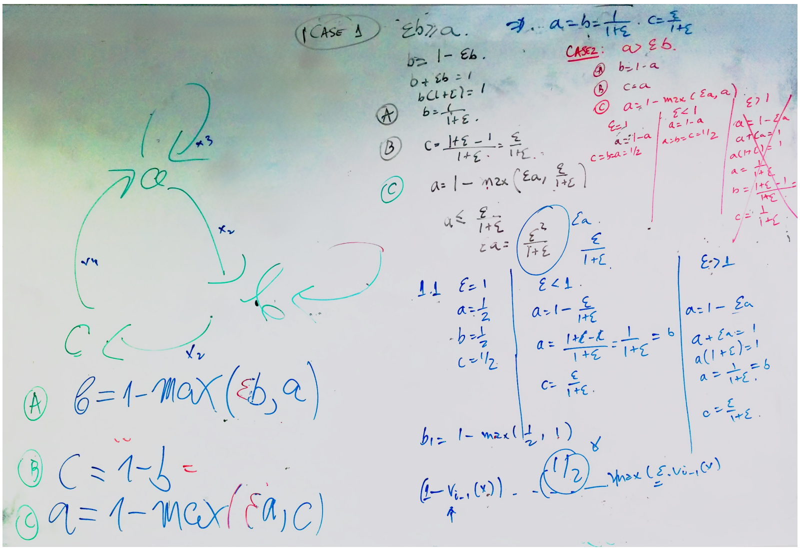
I arrived in London 31 years ago and we started working soon afterwards (I had originally been assigned to another supervisor). Those were very early days with my broken English and a lot to catch up with, but you were always very patient and supportive. Despite your already tremendous status in the academic community at that time, you never made any of your students feel inadequate or failed to lend an ear to hear about a technical problem or a personal issue.

Your early days at King's were pioneering. It is worth remembering that you were the first to transfer from the west of London to build and attract, and indeed you marked the beginning of a new phase for the then Department of Computer Science, paving the way for what is now the well-established Department of Informatics. You did "show us" and although the cardboard cutout of Seven of Nine is long gone, history is still on the walls of the Main Building.



Your interactions with your classes in the logic module were truly remarkable. I can only hope to approximate your rapport with my own students one day. You have always been very inspiring for your truly visionary approach, focus on the research that really matters in the long run, and for the love of the craft. I often think of your quiet but steady support during my PhD studies and those long hours in the years since, proving theorems together and I endeavour to carry the same passion with me.

Below is a little reminder of our proving one of those theorems (this one took us a few months, but we got there in the end).



Most people who have met you will have noticed how intelligent you are and how easily you can navigate through complex theories in mathematics, physics and computer science, allowing you to see so many things through the eyes of a true logician. A few will have known about your unique sense of humour too! I have been fortunate enough to have worked with you in some projects, many articles, and a couple of books, but above all to have always been treated as a friend and a member of your own family. I would like to think you know that I think of you in the exact same light, as one of my own family.

May you continue to collect“the little speckles of logic sprinkled by God everywhere” so you can explain what they mean to us all and may your legacy be honoured as they so rightly deserve. Happy 80th!

Odinaldo Rodrigues

Received October 2025

A Letter to Dov

Gadi Rosenberg

I would like to write about my collaboration with Professor Dov Gabbay, an extraordinary individual and a brilliant mind. Over the years, we have worked together on several groundbreaking articles that seamlessly integrated fields such as psychology, criminology, sexual offending, cognitive distortions, and therapy with logic, mathematics, and innovative model-building. Our work has yielded fascinating and novel frameworks that continue to inspire and challenge conventional thinking in these disciplines.

Collaborating with Dov has been nothing short of thrilling and deeply enriching. His intellectual curiosity and unparalleled creativity make every interaction stimulating and thought-provoking. Beyond the academic and professional realm, our conversations often extended to discussions about family, politics, and both rational and less rational solutions to life's challenges. These exchanges added a unique personal dimension to our collaboration, making it even more meaningful.

I am honoured and humbled to have had the opportunity to work with such a talented individual. Dov's generosity in sharing his knowledge and insights has profoundly influenced my work and thinking. His tireless dedication to problem-solving, often even during his sleep, is a testament to his passion and commitment. It is not uncommon for Dov to wake up with innovative ideas and groundbreaking solutions that leave me in awe.

If there were more people in the world like Dov, I firmly believe it would be a much better place. His creative energy, coupled with his remarkable intellect, has made a lasting impact on me, both professionally and personally. I remain grateful for the privilege of knowing and working alongside him.

Sincerely,
Dr. Gadi Rosenberg
Head of the Clinical Criminology Track

Received May 2025

An Appreciation to Dov Gabbay

Matthias Thimm
Artificial Intelligence Group, University of Hagen, Germany

It is difficult to overstate Dov Gabbay's influence on the field of logic in AI—not only through his many scientific contributions but also through the ways he has shaped and inspired the research community. Like many others, I encountered Dov not only in print, but in person: at conferences, in discussions, and, most memorably, in moments of sharp critique and generous encouragement. What follows is a brief personal recollection of three encounters with Dov that left a lasting impression on me.

The first time I met Dov was at the *Eleventh European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU)* in Belfast in 2011.[1] I obtained my PhD just shortly before and I was still quite new to the whole academic business. I also did not know Dov at all (in particular not personally and I had no idea how he looked like). At ECSQARU, I presented a paper about some work related to my thesis, an approach for probabilistic reasoning in fragments of first-order logic (co-authored with Gabriele Kern-Isberner and Jens Fisseler). In the first row at my presentation, there was a guy apparently taking a nap. At least, he did not seem to be much interested in the work I presented and had his eyes closed. I did not feel quite good about that, but I think I did a good presentation nonetheless. Anyway, when I was finished with my presentation and people started clapping, the guy (Dov) was startled and seemed to wake up. He looked, for like 2 seconds, at my final conclusions slide and raised his arm for a question. The chair (Guillermo Simari) called him up and the guy stood up, faced the audience, and made 2-minute monologue on the relationship between my work and some work he did in the 80s. I cannot recall much of what he talked about, but I can safely say that there were some good arguments in it (but also some not-so-good ones). In any case, due to the quite humorous way he did it, I was not really angry with him. Later I learned that was in fact Dov Gabbay. This was the first time I have met him.

The second time I met Dov was at the *20th European Conference on Artificial Intelligence (ECAI)* in Montpellier in 2012. Again, I presented some work and he

[1] I am 95% sure it was ECSQARU'11, but I may be wrong.

was in the audience. My talk was again about probabilistic reasoning, but this time for abstract argumentation frameworks. The paper I presented is one of my better works and was generally also well-received. I cannot recall the discussion section of my presentation and whether Dov asked a question (I think he did). But after the session, Dov directly approached me and gave me his appreciation. “This is very good work, continue with it!” is what he said. By then I knew a bit better who he was and this compliment felt very good. It inspired me to indeed continue the work on probabilistic reasoning with formal argumentation approaches. This was the second time I have met Dov.

The third time I met Dov was at the *Fifth International Conference on Computational Models of Argumentation (COMMA)* in the Highlands of Scotland in 2014. I was presenting some work on the relationship of reasoning with abstract argumentation on the one hand and reasoning with conditional logics on the other (joint work with Gabriele Kern-Isberner). The approach was more a technical exercise and highly preliminary. We developed a very simple approach for abstract argumentation to relate the two formalisms and it had not a very good intuitive interpretation on the argumentation side. Dov absolutely destroyed the work in the discussion. I was taken aback in light of the quite unfiltered criticism (which was mostly correct, but also quite blunt) and it took some time afterwards to regain my confidence. However, I also remember quite some nice and long discussions with Dov at the very same conference (mostly non-scientific though), which in general was one of the nicest conferences I have been to. This was the third time I met Dov.

I am not sure whether Dov recalls all three occasions (and, in particular, whether he recognized that it was me in all three instances). These encounters occurred during the early stages of my academic career, when every interaction felt magnified in significance and consequence. Of course, I met him quite more often afterwards, but these three encounters remain remarkably vivid in my memory. Despite some of these interactions being rather challenging—even bruising my academic ego at times—I can still say that I remember them with genuine fondness. There is something profoundly valuable about those moments when a senior scholar such as Dov takes the time to engage seriously with your work, even when that engagement comes in the form of rigorous critique.

Happy Birthday, Dov!

Matthias

Received May 2025

The Dov Gabbay Prize for Logic and Foundations
On a Birthday Gift

Emil Weydert
University of Luxembourg

1 Prelude

According to the available evidence, Oct.12, 2020 may be considered the birth date of the "*Dov Gabbay Prize for Logic and Foundations*". The team in Luxembourg was planning a Logic event in collaboration with our Chinese partner Beishui Liao from Zhejiang University, and my colleague Leon, while enjoying a sabbatical in Vienna, was wondering how to celebrate Dov's 75th birthday on Oct.23, considering the restrictions linked to the pandemy. For more than a decade Dov had been a frequent visitor and close collaborator of our lively Logic group in Luxembourg. He had escaped just in time to the Isle of Man when things started to deteriorate. I made several suggestions, among them:

> *We could launch the "Dov-Gabbay-Prize for the Foundations of Logic."*

In fact, I had already played with the idea of launching a scientific prize 30 years ago (then not about Logic), but without enough people sharing my enthusiasm this project was dropped, a wise decision. Nevertheless, a flower was waiting to bloom. This time the reactions were more positive.

Beishui: *Launching the"Dov-Gabbay-Prize for the Foundations of Logic" is a great idea.*
Leon: *If it's good for Emil it is good for me (:*

So the decision was made to proceed, and I took over the job to implement it. I choose as my co-organizer Alex Steen, a dynamic representative of the young generation of computational logicians, formerly a postdoc in Luxembourg, then assistant professor in Greifswald. Although the details obviously could not be settled until Oct.23, we wanted to present our gift-idea to Dov on his 75th birthday. For this occasion we planned a video session and made some slides to prepare the revelation (emphasizing that the responsibility for the content is entirely mine).

Dov 75
And Now for Something Completely Different
A Birthday Gift Experiment

ICR and friends
Luxembourg/Vienna/Zhejiang, Oct 23

Dov's 75th birthday

Goal: Celebration
Action sequence: What to do/offer?
???: Ok, let's discuss some ideas ...

Traveling?

Maybe: ... a trip for 2 persons to Vienna, the eternal City of Logic?
Mmh: ... better to wait for Dov 80, if we are lucky,
the Abstract Dialectical Framework fad will then be over!

Books?

Maybe: ... a hot logic book Dov doesn't yet possess?
Mmh: ... yes, but egoistic Emil wants to keep it for himself!

Girls?

Maybe: ... drinking tea and chatting about logic with Gina L.?[1]
Mmh: ... interesting idea, but it comes 55 years too late!

The real thing

Maybe: ... let's listen to Dov himself:

- *I am a logic, each one of us is a logic* (Dov 1994)

[1] Very, very famous Italian actress from the 60/70s.

Mmh: ... so the best way to celebrate Dov may be just

- *to celebrate and promote Logic,*

which as long as AIs are still more like Artificial Animals
means feeding natural logicians hammering in the basement ...

Our Idea

Let's launch the ...

Dov Gabbay Prize
for the
Foundations of Logic

The prize

The Dov Gabbay Prize for the Foundations of Logic (DGPFL) is an annual research prize which will mark, reward, and encourage exceptional contributions to the Foundations of Logic. The Prize seeks outstanding work ideally combining deep foundational insight and conceptual innovation with sophisticated formal analysis and an interdisciplinary perspective. Its importance should be communicable to a well-educated public and promise to inspire future generations of scientists.

Dov

Following Dov ("*I am a logic*"), Logic is here understood in its broadest sense, encompassing Mathematical, Applied, as well as Philosophical Logic in their orthodox and less orthodox incarnations, as for instance described and discussed in the more than 50 Logic handbooks edited by Dov. Dov's 75th birthday is a great occasion to honour his huge, multifaceted, and original work as well as his lifelong and unique dedication to Logic and its foundations by a prize reflecting his ideals.

Here we have been inspired by Dov's own words: *Any comprehensive theory modelling the processing in our heads is a logic.*

After addressing some procedural details, we also introduced a special symbol trying to encode the essence of Dov's work.

Design

> *Characteristic icon*: $\dashv\!|\!|\!\!\sim$
>
> This expression states monotonic logical entailment on the left, fixing the past, and defeasible plausible inference on the right, guessing the future. The icon thus combines two important research threads in Dov's life: temporal reasoning and nonmonotonic consequence.

Conclusion

HAPPY BIRTHDAY DOV

Fortunately, Dov liked the idea, and even more importantly, his wife Lydia also found very kind words. And in Dov's universe this may be considered the ultimate earthly blessing. So the main preconditions were met to launch the prize!

One may acknowledge here the role of the pandemy. Without the forced isolation calling for creativity we might well have settled for a more conventional present, and not have broken the – sad – tradition that people whose names decorate prizes are usually no longer around to enjoy them.

2 The First Steps

The initial plan was to start in 2021. But the challenges of the pandemy sucked away quite some energy and created a suboptimal environment. So we decided to postpone until 2022, when Dov would conveniently turn 77. A 1-pager from 2021 summarized the essentials of the prize. It was intended to provide basic guidance for the organizers, as well as to inform and convince potential jury members, future nominators, and also generous minds of any kind.

In addition to the motivation, the scope, and the goals, sketched above, and bread and butter details, like the timing and the submission rules, there were four major questions to address:

1. *Which organization?*
2. *Which target group?*
3. *Which jury?*
4. *Which reward?*

2.1 The Organization

The Dov Gabbay prize is special in several respects. Scientific prizes are typically created by well-established scientific societies, academic institutions, philanthropic organizations, or wealthy individuals wishing to give something back to society, maybe pushed by an ounce of self-promotion. None of this did apply here.

Because the prize was initially conceived as a birthday gift of our group, teaming up with a financially and organizationally potent partner was not an option for the first rounds, where we preferred to stay independent. It thus became a kind of experiment, trying to find out how to create a research prize from scratch, and an opportunity to gain first-hand experience in the academic philanthropy business.

The downside of such a grassroot approach is the lack of resources. Fundraising is an option, but helping academics to reward scholars from an obscure formal discipline does not look particularly attractive for most potential sponsors. Furthermore, to generate sufficient interest there must be evidence for relevant past achievements. In other words, at the begin you are on your own.

The most important task is to organize the selection process. Initially, to be prepared for high numbers of nominations from a multitude of logical subdiciplines, we foresaw a more complex structure. The actual decision was meant to be in the hands of an independent core jury of (at least) six renowned logicians expected to represent the three major realms of Logic: Mathematical, Philosophical, and Computational Logic. They would be supported by a management team and an active scientific advisory forum. To ensure transparency and trust, the main jury would have access to all the submissions and comments, however without being restricted by them.

With a manageable number of nominations, the multi-board structure was eventually dropped, leaving us with a single, independent jury, a lean management committee, and a small informal advisory group not involved in the selection procedure. In practice this was fully sufficient.

One of the trickier tasks is the management of the financial dimension, especially the acquisition of funds and the handling of cash rewards. As soon as money is flowing one is confronted to anxious banks and state agencies looking out for dubious business, fees and tax income, which creates a number of hurdles. It is recommendable to operate within the legal framework of a non-profit organization, necessary for a committed bank account, and radiating seriousness. This requires quite some efforts, and may not even be sufficient. Learned societies have been debanked and struggled to find a new financial host. This concerns especially organizations with a international/global perspective, or smaller entities which may not have the resources to handle these issues effectively.

The Dov Gabbay prize is currently hosted by ILOAF asbl, a non-profit organization registered under Luxembourgish law. ILOAF is a young, independent academic association which aims at promoting research, education, and outreach in domains related to Logic and Foundations, with an interdisciplinary perspective. It sees itself mainly as an incubator - inspiring, launching, developing, and accompanying corresponding activities together with suitable partners and experts. The Dov Gabbay Prize has been its first major project.

2.2 The Target Group

As stated in the most recent call for nominations from 2025, the Dov Gabbay Prize is an international research award *aimed at outstanding and inspirational contributions in Logic and Foundations. It targets active researchers combining foundational insight and conceptual innovation with sophisticated theoretical analysis.*

In the first year the call was – quite ambitiously – directed at the whole community of logicians. Confronted to a broad range of excellent candidates from very different areas, the Jury and the organizers then however decided, for reasons of fairness, comparability, and communicability, to focus each year on a specific realm, starting with Computational Logic, to be followed by Mathematical and Philosophical Logic. The Jury does the classification, sometimes complicated by the interdisciplinary character of the field.

In practice the chosen formulation, and notoriety, privilege nominations of more senior researchers. There are also different dimensions, life-time achievement, or extraordinary recent contributions. We had awardees from both shores. Selecting in this context clearly presupposes a strong and fearless jury.

When targeting the top, there is of course always a certain risk of redundancy, the usual suspects being honoured again and again. Because of its independent multi- and interdisciplinary perspective, the Dov Gabbay Prize enjoys some additional freedom. An award can also be an opportunity to pull into the limelight excellent researchers/research who are thought to deserve more or special attention.

2.3 The Jury

The heart of an academic prize is its jury. Assembling an excellent team is especially important when starting a new prize. The jury has to guarantee credibility and independence, and it offers experience and visibility. We had three main criteria for members: First, we were seeking well-recognized and well-connected leading researchers. Secondly, we wanted representatives from different sub-communities of Philosophical, Computational, and Mathematical Logic, ideally with interdisciplinary interests. Thirdly, we also aimed at some geographical and sociological diversity in line with the seniority requirement.

For the founding jury, to smoothen the process, I gave priority to people I have been in contact with in the past. If a stranger asks you to join the jury of a newly created prize, the reaction may be a bit reserved. On the other hand, relying on academic buddies sitting next door definitely conveys the wrong message. But these problems tend to evaporate when the prize is established.

The actual jury acquisition process started on April 27 in 2022 with the mail "A question", and ended 2 months later on July 1 by informing Dov with the mail "The glass is full". We were unexpectedly lucky to be able to win overall 7 extraordinary scholars for our project (6-7 in each round), who accepted despite countless other obligations. In alphabetical order:

1. John T. Baldwin (math.logic)
   University of Illinois, Chicago, United States

2. Christoph Benzmueller (comp.logic)
   University of Bamberg, Germany

3. Johan van Benthem (phil.logic)
   Univ. of Amsterdam/Stanford/Tsinghua, NL/US/China

4. Agata Ciabattoni (comp.logic)
   Vienna University of Technology, Austria

5. Laura Giordano (comp.logic)
   University of Eastern Piedmont, Italy

6. Hannes Leitgeb (phil.logic)
   Ludwig-Maximilians-University, Munich, Germany

7. Philip Welch (math.logic)
   University of Bristol, United Kingdom

In the first three years, the jury chose Philip Welch as its chairman. The organizer also used an informal advisory group – the scientific forum – of seven interdisciplinary experts to provide support and advice, but without being involved in the selection process. Especially in the early phase, this has been very helpful.

2.4 The Reward

A reward has several purposes. First of all, it shows and materializes recognition for awardees. Secondly, it documents the commitment of the organizers to the prize. Thirdly, a cash prize, or other substantial offers, help to gain attention in a competitive environment. On the other hand one may keep in mind that a juicy financial perspective is usually not what makes logicians tick. It can be useful if one wants to promote research on a particular question, a cherry on top of the search for insight and fame. The best illustration may be the Millenium Problems of the Clay Mathematics Institute (1 Mio.$). But what may count more in ordinary life is the standing of the prize in the respective research community, all the rest being just a pleasant side-effect. But this arguably depends on the career stage.

We went for the middle ground, offering 2001 EUR for each of the first three editions. Besides the reference to a memorable SF movie, this guarantees a prize sum above 1000 EUR even when the prize is split, as it was in the first two years: two individual candidates in 2023, and a team of two in 2024. This was an order of magnitude manageable by the sponsors of the birthday gift.

Non-cash rewards can make sense in the academic context, but they may still produce costs. A common idea is to invite the winner to give a talk at some relevant scientific event, to offer a special slot in an academic publication, or more generally, to make a special promotional effort for the rewarded scientific work. An original idea has been to create a travelling trophy, inspired by the soccer champion business. Here one may mention the Ernst-Zermelo-Ring, which is temporarily assigned every four years by the DVMLG, the German Association for Mathematical Logic and the Foundations of the Exact Sciences. Maybe a venue to explore.

3 The First Three Years

The first three years, aimed at implementing the birthday gift, can be seen as a test phase. The actual nominations, their evaluation by the Jury, and the communication with the community, they all helped to shape the prize and to suggest ideas for its future.

Because the original target theme "*... Foundations of Logic*" carried the risk to be interpreted too narrowly, it was replaced later on by "*... Logic and Foundations*", more in line with the actual intentions.

Year 1: The first edition was an occasion for the organizers and the Jury to clarify the goals and to adapt the procedure. As mentioned before we decided to focus in the first year on Computational Logic, Dov's main area of interest in the past decades, and in the following two years on Mathematical, resp. Philosophical Logic.

The first call for nominations, still with a broad scope, was made in October 2022, after Dov's 77th birthday. As existence means existence on the web, we also launched a modest website: https://dgp.iloaf.org.

We got a two-digit number of submissions, from which the Jury selected those fitting Computational Logic. Begin of July 2023 the Jury revealed that the prize had been awarded jointly to

Dale Miller and **Mirek Truszczynski**,

for their long-standing important contributions to practice and theory of logic programming, theorem proving and related important strands in Computational Logic.

Dale Miller (Inria-Saclay and LIX/Ecole Polytechnique, France) has done pioneering and agenda-setting research bringing together and advancing logical proof theory and computational logic in the areas of higher-order logic programming and higher-order theorem proving. His research spans the full range from innovative foundational theory to the design and implementation of state-of-the-art working systems.

Mirek Truszczynski (University of Kentucky, USA) has done pioneering and agenda-setting research in logic programming, non-monotonic reasoning, answer-set programming, and preference in computational choice, where his seminal contributions are widely recognized. His research has succesfully run the gamut from mathematical foundations to industrial applications.

The online ceremony took place on October 25. Christoph Benzmueller and Laura Giordano presented the respective laudatios, followed by talks from the awardees and Dov himself.
- Dale Miller: *A system of inference based on proof search*
- Mirek Truszczynski: *The road to answer set programming*
- Dov Gabbay: *Negation as failure and failure in general*

Year 2: The second call for nominations, aimed at Mathematical Logic, was launched begin of March 2024. Dov being less visible in that community, there was a lower number of nominations, but with high-profile candidates. In August 2024 the Jury could again announce two winners,

David Asperó and **Ralf Schindler**,

for their work in the foundations of set theory, and in particular for their work connecting determinacy principles and so-called strong forcing axioms, both impinging on the nature of the continuum hypothesis (supporting assumptions suggesting that the continuum has size $\aleph_2$). Principally the award is given for their solution to a decades old problem in the area by showing that there is a concrete bridge between these two rather different approaches to the foundations of set theory. The paper appeared in the Annals of Mathematics: *Martin's Maximum implies Woodin's Axiom.* Ann. Math. 193(3), 793-835 (2021).

David Asperó (University of East Anglia, UK) is well known for his work in set theory, in particular for his contributions to forcing and forcing axioms.

Ralf Schindler (University of Műnster, Germany) has made significant contributions to the theory of inner models of set theory under strong theoretical axioms of infinity and hypotheses of the determinacy of infinite games.

At the online ceremony on November 28, Joan Bagaria's laudatio was followed by a talk from the awardees, and the event concluded with interesting historical reflections by Dov. In the audience Adrian Mathias enjoyed hearing that Dov was once so fascinated by one of his papers that he continued reading on the floor of the library although his chair had broken down.

Year 3: The third call for nominations was directed at Philosophical Logic. In May 2025 the Jury awarded the prize for the first time to a single researcher,

Alexandru Baltag.

Alexandru Baltag (ILLC, University of Amsterdam, NL) is honoured for his multiple and seminal contributions to Philosophical Logic. He has been one of the fathers of the paradigm of Dynamic-Epistemic Logic, a powerful extension of existing epistemic logics that can describe the precise information flow in a wide range of scenarios, something that did not exist before.

Baltag has become a leading architect of this research program, whose applications continue to increase in number, starting out from philosophical topics in epistemology concerning knowledge change (e.g., those relating to the famous Fitch Paradox), belief revision, and subjective probability, but also reaching out to practical areas such as epistemic planning in AI, or social epistemology. Noteworthy also is his innovative work with Sonja Smets in the philosophy of science offering a fresh perspective on the interface of quantum logic and quantum mechanics in terms of a dynamic logic of quantum measurement in quantum information systems.

In the last decade, Baltag has furthermore become one of the main movers in Epistemic Topology, the branch of formal epistemology which uses logical-topological models for analyzing age-old philosophical questions such as the Problem of Induction, and the issues posed by observation in the empirical sciences where measurements are approximate.

His ability to combine deep and broad philosophical thought with a rigorous mathematical analysis offers a role model for the field, and – last but not least – it also reflects well Dov Gabbay's own style of work.

4 The Future

After three years of honouring exceptional logicians from a broad range of domains, the Dov Gabbay Prize has now a solid standing, and one may claim that the birthday gift has been successfully delivered! This initial phase, which will be concluded with a ceremony for Alexandru Baltag, has taught us a number of lessons, which will guide the future development. However, the overall goal to celebrate and promote excellent logic and logicians in different areas, with an affinity for interdisiplinarity and foundations in line with Dov's philosophy, will stay the same.

After the incubator phase, there are a number of important questions and interesting venues to explore, so as to make sure that the Dov Gabbay Prize will continue to promote Logic, the Science of Reasoning, and thereby honour Logic's most forceful voice: Dov.

Received September 2025

Logic, Probability and Causality

Jon Williamson
Department of Philosophy, University of Manchester

Dov Gabbay is well known for the breadth of his contributions to logic—it is therefore apt that he was appointed Augustus de Morgan Professor of Logic at King's College London, as de Morgan himself viewed logic as a broad enterprise, encompassing both deductive logic and probability theory [8]. I found Dov's broad view of logic inspiring during my time as a postdoc with Donald Gillies in the philosophy department at King's College London and I have fond memories of our collaborations. Two of these collaborations particularly stand out for me.

In 2002, Dov and I organised the fourth Augustus de Morgan workshop on the theme of combining probability and logic, at King's College London. This workshop became the first in a fertile series of 'Progic' workshops on combining probability and logic—a series that recently held its 12th instalment at Carnegie Mellon University. Speakers at the 2002 workshop represented philosophy (Colin Howson, Henry E. Kyburg Jr), computer science (James Cussens, Peter Flach, Joseph Y. Halpern, Jürg Kohlas, Stephen Muggleton), engineering (Rachel Bourne), medical informatics (John Fox) and mathematics (Jeff Paris), and this breadth of disciplines has been characteristic of subsequent progic workshops. Some of the papers presented at the 2002 workshop can be found in a special issue of *Journal of Applied Logic* 1(3-4) [9].

A second collaboration concerns connections between causality, Bayesian networks and self-fibring networks [10]. In the philosophical literature on causation, it has not been widely recognised that causal relationships can themselves act as causes or effects. For example, *smoking causing cancer* causes governments to restrict tobacco advertising, which prevents smoking and thereby prevents cancer. Once we recognise this phenomenon, the question arises as to how to model these nested causal relationships in order to reason about them more effectively. Bayesian networks are widely used to model networks of causal relationships, and in our paper we generalised Bayesian networks to 'recursive Bayesian networks', which can take Bayesian networks as nodes. We showed how to use a recursive Bayesian net to define a joint probability distribution over the domain, and explored connections with self-fibring networks, which relate to Dov's work on fibring logics and fibring neural networks [4]. This research on recursive Bayesian nets was subsequently developed and discussed by [2]; [3]; [5]; [7]; [1] and [6]. This line of work relates to the important problem of how to model mechanisms with causal cycles, a problem to which standard Bayesian networks, which require directed acyclic graphs, seem poorly suited.

This provides a good opportunity, then, to celebrate not only Dov's 80th birthday but also the breadth of his vision of logic.

References

[1] Casini, L. (2016). How to model mechanistic hierarchies. *Philosophy of Science*, 83(5):946–958.

[2] Casini, L., Illari, P. M., Russo, F., and Williamson, J. (2011). Models for prediction, explanation and control: recursive Bayesian networks. *Theoria*, 26(1):5–33.

[3] Clarke, B., Leuridan, B., and Williamson, J. (2014). Modelling mechanisms with causal cycles. *Synthese*, 191(8):1651–1681.

[4] Garcez, A. S. d. and Gabbay, D. M. (2004). Fibring neural networks. In *Proceedings of the 19th National Conference on Artifical Intelligence*, AAAI'04, page 342–347. AAAI Press.

[5] Gebharter, A. (2016). Another problem with RBN models of mechanisms. *Theoria*, 31(2):177–188.

[6] Gebharter, A. and Leuridan, B. (2024). Modelling cyclic causal structures. In Illari, P. and Russo, F., editors, *The Routledge Handbook of Causality and Causal Methods*, pages 269–280. Routledge, New York and Abingdon.

[7] Gebharter, A. and Schurz, G. (2016). A modeling approach for mechanisms featuring causal cycles. *Philosophy of Science*, 83(5):934–945.

[8] Rice, A. (2003). 'Everybody makes errors': The intersection of de Morgan's logic and probability, 1837–1847. *History and Philosophy of Logic*, 24(4):289–305.

[9] Williamson, J. and Gabbay, D. (2003). Combining probability and logic: editorial. *Journal of Applied Logic*, 1(3–4):135–138.

[10] Williamson, J. and Gabbay, D. (2005). Recursive causality in Bayesian networks and self-fibring networks. In Gillies, D., editor, *Laws and models in the sciences*, pages 173–221. King's College Publications, London. With comments pp. 223–245.

Received September 2025

Extending Depth-Bounded Reasoning to First-Order Logic

Marcello D'Agostino[*]
University of Milan

Costanza Larese[†]
University of Milan

Sanjay Modgil
King's College, London

Abstract

This paper discusses how the framework of Depth-Bounded Boolean Logics and their underlying informational approach can be extended to first-order logic, outlining an intuitive "informational" semantics for quantifiers, a corresponding hierarchy of approximations, and a proof-theoretical characterization via suitable natural deduction rules.

1 Introduction

Classical logic models *logically omniscient agents*, that is, agents—whether human, artificial, or hybrid — who can recognize all valid consequences of their assumptions. This normative ideal, however, clashes with the reality of reasoning under limited cognitive and computational resources. As Gabbay and Woods observe:

Our warm thanks go to Jane Spurr for her unfailing patience and generous support in accompanying this paper from its first draft to its final form.

[*] This research was funded by the Department of Philosophy "Piero Martinetti" of the University of Milan under the Project "Departments of Excellence 2023– 2027" awarded by the Ministry of University and Research (MUR), and also by MUR under the PRIN 2022 Project "CORTEX" (cod. 2022ZLLR3T).

[†] This research was funded by the Department of Philosophy "Piero Martinetti" of the University of Milan under the Project "Departments of Excellence 2023– 2027" awarded by the Ministry of University and Research (MUR). I also thankfully acknowledge the generous support of MUR under the FIS1 Advanced Grant scheme G53C23000510001 (Reasoning with Data).

> A logic is an idealization of certain sorts of real-life phenomena. By their very nature, idealizations misdescribe the behaviour of actual agents. This is to be tolerated when two conditions are met. One is that the actual behaviour of actual agents can defensibly be made out to approximate to the behaviour of the ideal agents of the logician's idealization. The other is the idealization's facilitation of the logician's discovery and demonstration of deep laws [8, p. 158].

The *approximation problem* is therefore to define, for a given logic L, a hierarchy of systems that (i) converge to L, (ii) preserve rational standards at each stage, and (iii) provide realistic models of the deductive capacities of resource-bounded agents.

The *depth-bounded approach* offers a systematic solution. Its central insight is the distinction between two kinds of information:

- **Actual information:** information explicitly possessed by an agent or feasibly accessible;
- **Virtual information:** information not actually possessed but temporarily assumed *as if* it were possessed, in order to advance inference[1].

The aim is to minimize recourse to virtual information, thereby controlling combinatorial explosion[2].

This approach, which stems from [6], has been thoroughly investigated for classical propositional logic in a series of papers (e.g. [2, 5, 4]) and in a recent book by the present authors with Dov Gabbay [11]. At depth 0, reasoning is restricted to propositional inferences that can be justified in terms of actual information alone. This yields a tractable subsystem whose semantics can be captured in several equivalent ways, including a non-deterministic three-valued matrix first noted (but never properly investigated) by Quine [14] to represent what he called "the primitive meaning

[1] An intuition pump: suppose you've been informed that either your login is incorrect or your password is incorrect. You do not possess the information as to which is the case. But you can, as it were, branch into two virtual information states, in one of which you assume that your login is incorrect, another in which you assume that it is correct and therefore your password is incorrect. Continuing to "hold in mind" each of these two virtual information states so as to draw further inferences in each, exacts greater cognitive effort that may reveal further implicit information. For example, suppose that given other information to hand, you infer from the assumption that your login is incorrect that you've been hacked (someone changed it), and you infer from the assumption that it is correct and so your password is incorrect that you've been hacked (someone changed it).

[2] In proof-theory the use of virtual information is apparent in the "discharge rules" of Gentzen-style natural deduction. As argued in [11], these rules are not suited to provide an adequate measure of depth for deductive reasoning

of the logical operators" (see [3]) and later rediscovered by Crawford and Etherington [1] in connection with unit resolution. The 0-depth logic validates exactly those inferences that can be justified without recourse to virtual assumptions.

Higher layers of the hierarchy are generated by controlling the nested uses of virtual information. Increasing the permitted depth of nested applications of this rule yields a sequence of sound and paracomplete systems of growing inferential power and computational complexity, which in the limit converge to full classical logic [4]. The depth parameter thus measures the cognitive and computational effort required to *dig out* implicit consequences from the premises.

From the semantic distinction between actual and virtual information, one obtains a parallel proof-theoretical distinction, captured by *intelim rules* (introduction and elimination rules for Boolean operators) together with a single branching rule (RB). This is a structural rule—essentially a restricted cut, related to the principle of bivalence of classical logic—that governs the introduction of virtual information. Depth-0 reasoning is captured by intelim sequences, while deeper approximations are represented by intelim trees. Each system satisfies a subformula property and supports analytic proof search, so that tractability arises as a consequence of the informational perspective rather than from an *ad hoc* syntactic restriction.

Conceptually, the depth-bounded framework provides a fresh account of the informational content of deduction. An inference is of depth 0 when possessing actual information about its premises suffices, without supplementation, to possess actual information about its conclusion. At higher depths, inference becomes conditional on simulating virtual extensions of the actual information state. This perspective dissolves the traditional "scandal of deduction" (see [6]), since it recognizes that logical inference is rarely trivial even at the propositional level: it typically involves a non-trivial deployment of virtual information, which incurs real cognitive and computational costs.

The aim of the present paper is to extend the depth-bounded approach, originally developed for Boolean logic, to the domain of *first-order reasoning*, pursuing a line of research started in [7]. We outline an intuitive informal semantics for the quantifiers that mirrors the distinction between actual and virtual information, and combine it with a depth-bounded proof theory to obtain a hierarchy of approximations to classical first-order logic. This is an exploratory Festschrift piece, emphasizing ideas over formal details or technical results, that continues Gabbay's long-standing interest in resource-sensitive and controlled derivation. Formal aspects and results will be investigated in a subsequent paper.

Section 2 reviews the propositional framework of depth-bounded Boolean logics (DBBL). Section 3 discusses the informational meaning of the quantifiers focusing on the information we actually possess about *objects*, rather than propositions. The exposition is informal and technical details are omitted. Section 4 presents the intelim rules and the RB rule for first-order inference, elaborating on some ideas first put forward in [12]; again the exposition is informal and concept-oriented. Section 5 integrates the Boolean and quantificational components into a bivariate measure of depth, proposing a unified view of bounded reasoning. Section 6 concludes with remarks on conceptual implications and directions for further research.

2 An overview of Depth-bounded Boolean Logics

In DBBL the underlying semantic notions are not classical truth and falsity, but *informational* truth and falsity. A formula A is *informationally true* if we actually possess the information that A is true, *informationally false* if we actually possess the information that A is false, and *informationally indeterminate* otherwise. By "actually possessing" a piece of information we mean that either we possess it explicitly as part of our data or it is feasibly accessible to us by a chain of simple inferences each of which descends immediately from the *informational meaning* of the Boolean operators—that is, that part of their meaning that can be specified solely in terms of informational truth and informational falsity (see [11, Chapters 1-2]). For conceptual clarity we use *signed formulas*, i.e., expressions of the form $T\,A$ and $F\,A$ where the former means that A is informationally true and the latter that A is informationally false. Information states are typically partial and therefore it cannot be assumed that for every information state and for every formula A, either $T\,A$ or $F\,A$.

The *intelim rules* for DBBL express the most basic inferences that can be immediately justified solely in terms of informational truth and falsity. While all the rules that are sound for informational truth and falsity are also sound for their classical "metaphysical" counterparts that obey the principle of bivalence, some of the rules that are sound for classical truth and falsity are obviously unsound for their informational counterparts. For example, under the informational reading of the signs T and F, from $T\,A \vee B$ it does not follow that either $T\,A$ or $T\,B$. Indeed, very often we are informed that a certain disjunction is true without holding any specific information about either disjunct ("either the password or the username is incorrect"). Similarly, we may well be informed that a conjunction is false without holding any specific information about either conjunct; hence $F\,A \wedge B$ does not imply that either $F\,A$ or $F\,B$. On the other hand, the intelim rules can be easily seen as sound under the informational interpretation of the signs.

The intelim rules for the four standard Boolean operators are illustrated in Table 1. Their unsigned version in Table 2 is sufficient for all practical purposes, albeit

$$\frac{F\,A}{T\,\neg A}\,F\neg I \qquad \frac{T\,A}{F\,\neg A}\,T\neg I \qquad \frac{T\,\neg A}{F\,A}\,T\neg E \qquad \frac{F\,\neg A}{T\,A}\,F\neg E$$

$$\frac{T\,A}{T\,A\vee B}\,T\vee I_1 \qquad \frac{T\,B}{T\,A\vee B}\,T\vee I_2 \qquad \frac{\begin{array}{c}F\,A\\ F\,B\end{array}}{F\,A\vee B}\,F\vee I$$

$$\frac{\begin{array}{c}T\,A\vee B\\ F\,A\end{array}}{T\,B}\,T\vee E_1 \qquad \frac{\begin{array}{c}T\,A\vee B\\ F\,B\end{array}}{T\,A}\,T\vee E_2 \qquad \frac{F\,A\vee B}{F\,A}\,F\vee E_1 \qquad \frac{F\,A\vee B}{F\,B}\,F\vee E_2$$

$$\frac{F\,A}{F\,A\wedge B}\,F\wedge I_1 \qquad \frac{F\,B}{F\,A\wedge B}\,F\wedge I_2 \qquad \frac{\begin{array}{c}T\,A\\ T\,B\end{array}}{T\,A\wedge B}\,T\wedge I$$

$$\frac{\begin{array}{c}F\,A\wedge B\\ T\,A\end{array}}{F\,B}\,F\wedge E_1 \qquad \frac{\begin{array}{c}F\,A\wedge B\\ T\,B\end{array}}{F\,A}\,F\wedge E_2 \qquad \frac{T\,A\wedge B}{T\,A}\,T\wedge E_1 \qquad \frac{T\,A\wedge B}{T\,B}\,T\wedge E_2$$

$$\frac{F\,A}{T\,A\to B}\,T\!\to I_1 \qquad \frac{T\,B}{T\,A\to B}\,T\!\to I_2 \qquad \frac{\begin{array}{c}T\,A\\ F\,B\end{array}}{F\,A\to B}\,F\!\to I$$

$$\frac{\begin{array}{c}T\,A\to B\\ T\,A\end{array}}{T\,B}\,T\!\to E_1 \qquad \frac{\begin{array}{c}T\,A\to B\\ F\,B\end{array}}{F\,A}\,T\!\to E_2 \qquad \frac{F\,A\to B}{T\,A}\,F\!\to E_1 \qquad \frac{F\,A\to B}{F\,B}\,F\!\to E_2$$

Table 1: Intelim rules for the four standard Boolean operators.

semantically less transparent. (See [5] for a generalization to arbitrary Boolean operators.)

The intelim rules—or equivalently, the informational semantics mentioned in the introduction and discussed in [11]—characterize the basic layer of 0-depth inference. Inferences based only on the intelim rules can be arranged in sequences such that each signed formula in the sequent is either a premise or follows from previous formulas by applying one of the rules.

To characterize classes of increasingly deeper inferences and, asymptotically, full classical propositional logic we only need to add a single branching rule:

$$\begin{array}{cc}\diagup & \diagdown\\ T\,A & F\,A\end{array} \quad \text{or} \quad \begin{array}{cc}\diagup & \diagdown\\ A & \neg A\end{array} \quad \text{for unsigned formulas.}$$

RB governs the use of virtual information, i.e., information that we do not actually possess, but must be simulated in order to reach deeper conclusions: in each branch

$$\dfrac{A}{\neg\neg A}\,\neg I \qquad \dfrac{\neg\neg A}{A}\,\neg\neg E$$

$$\dfrac{A}{A \vee B}\,\vee I_1 \qquad \dfrac{B}{A \vee B}\,\vee I_2 \qquad \dfrac{\begin{array}{c}\neg A\\ \neg B\end{array}}{\neg(A \vee B)}\,\neg\vee I$$

$$\dfrac{\begin{array}{c}A \vee B\\ \neg A\end{array}}{B}\,\vee E_1 \qquad \dfrac{\begin{array}{c}A \vee B\\ \neg B\end{array}}{A}\,\vee E_2 \qquad \dfrac{\neg(A \vee B)}{\neg A}\,\neg\vee E_1 \qquad \dfrac{\neg A \vee B}{\neg B}\,\neg\vee E_2$$

$$\dfrac{\neg A}{\neg(A \wedge B)}\,\neg\wedge I_1 \qquad \dfrac{\neg B}{\neg(A \wedge B)}\,\neg\wedge I_2 \qquad \dfrac{\begin{array}{c}A\\ B\end{array}}{A \wedge B}\,\wedge I$$

$$\dfrac{\begin{array}{c}\neg(A \wedge B)\\ A\end{array}}{\neg B}\,\neg\wedge E_1 \qquad \dfrac{\begin{array}{c}\neg(A \wedge B)\\ B\end{array}}{\neg A}\,\neg\wedge E_2 \qquad \dfrac{A \wedge B}{A}\,\wedge E_1 \qquad \dfrac{A \wedge B}{B}\,\wedge E_2$$

$$\dfrac{\neg A}{A \to B}\to I_1 \qquad \dfrac{B}{A \to B}\to I_2 \qquad \dfrac{\begin{array}{c}A\\ \neg B\end{array}}{\neg(A \to B)}\,\neg\!\to I$$

$$\dfrac{\begin{array}{c}A \to B\\ A\end{array}}{B}\to E_1 \qquad \dfrac{\begin{array}{c}A \to B\\ \neg B\end{array}}{\neg A}\to E_2 \qquad \dfrac{\neg(A \to B)}{A}\,\neg\!\to E_1 \qquad \dfrac{\neg(A \to B)}{\neg B}\,\neg\!\to E_2$$

Table 2: Intelim rules for the four standard Boolean operators (unsigned version).

we suppose a jump to a hypothetical refinement of the current information state, in which we actually possess either the information that A is true or the information that A is false.

Intelim rules can be combined with RB to yield *intelim trees*. These are downward-growing trees in which each branch corresponds to a possible information state. Branching occurs only through RB, which introduces the virtual assumptions $T\,A$ and $F\,A$ on two sibling branches.

Intelim trees An *intelim tree* for a set X of signed formulae is a finite tree such that each node is either: (i) an assumption from X, (ii) the conclusion of an application of an intelim rule to earlier nodes on the same branch, or (iii) a virtual assumption introduced by an application of RB. The *depth* of an intelim tree is the maximum number of nested applications of RB.

A branch is *closed* if it contains both $T\,A$ and $F\,A$ for some formula A; otherwise it is *open*. A signed formula sA (where s is either T or F) is *deducible* at depth k

from a set X of signed formulas if there is an intelim tree of depth k whose open branches all end with sA.

Note that according to the above definition, if an intelim tree of depth k is closed (i.e., every branch is closed), then any sA is deducible from the premises. Hence, k-depth consequence is *explosive*. However, for each k there are classically inconsistent sets of formulae that are not inconsistent at depth k.

Example: The classically inconsistent set

$$\{T\,A \vee B, T\,A \vee \neg B, T\,\neg A \vee B, T\,\neg A \vee \neg B\}$$

is consistent at depth 0.

Hence, depth-bounded consequence remains monotonic and reflexive but not complete with respect to classical entailment.

Figure 1 shows three examples of intelim trees of depth 0, 1 and 2 respectively. Formulas are labelled with the name of the rule from which they are obtained; unlabelled formulas are premises or virtual information introduced via RB. Note that in the third tree the conclusion u follows from the premises because it occurs in all open branches. Note also that intelim trees can also be used to prove tautologies directly from no premise. An example of a derivation of the excluded middle law is shown in Figure 2.

3 The informational meaning of quantifiers

When the informational perspective is extended from propositional to quantified reasoning, the focus shifts from information about *propositions* to information about *objects*. From this standpoint, the objects that are introduced by quantifier elimination are not metaphysical individuals but *informational objects*: bundles of properties providing a partial description of unknown objects in the domain. For ease of exposition, we shall call them *informons*.

An informon carries information about how such unknown objects are partially characterized within an information state, and thus represents the objectual dimension of information. Each informon represents an agent's current information about a certain object: the properties that are known to hold of it and those that remain undetermined. A property expressed by a signed formula in one free variable $T\,A(x)$ or $F\,A(x)$ holds of an informon ι if $T\,A(x)$, respectively $F\,A(x)$, belongs to the bundle of properties associated to ι. Informons then provide *approximations* to unknown objects that are typically refined as new information is acquired either

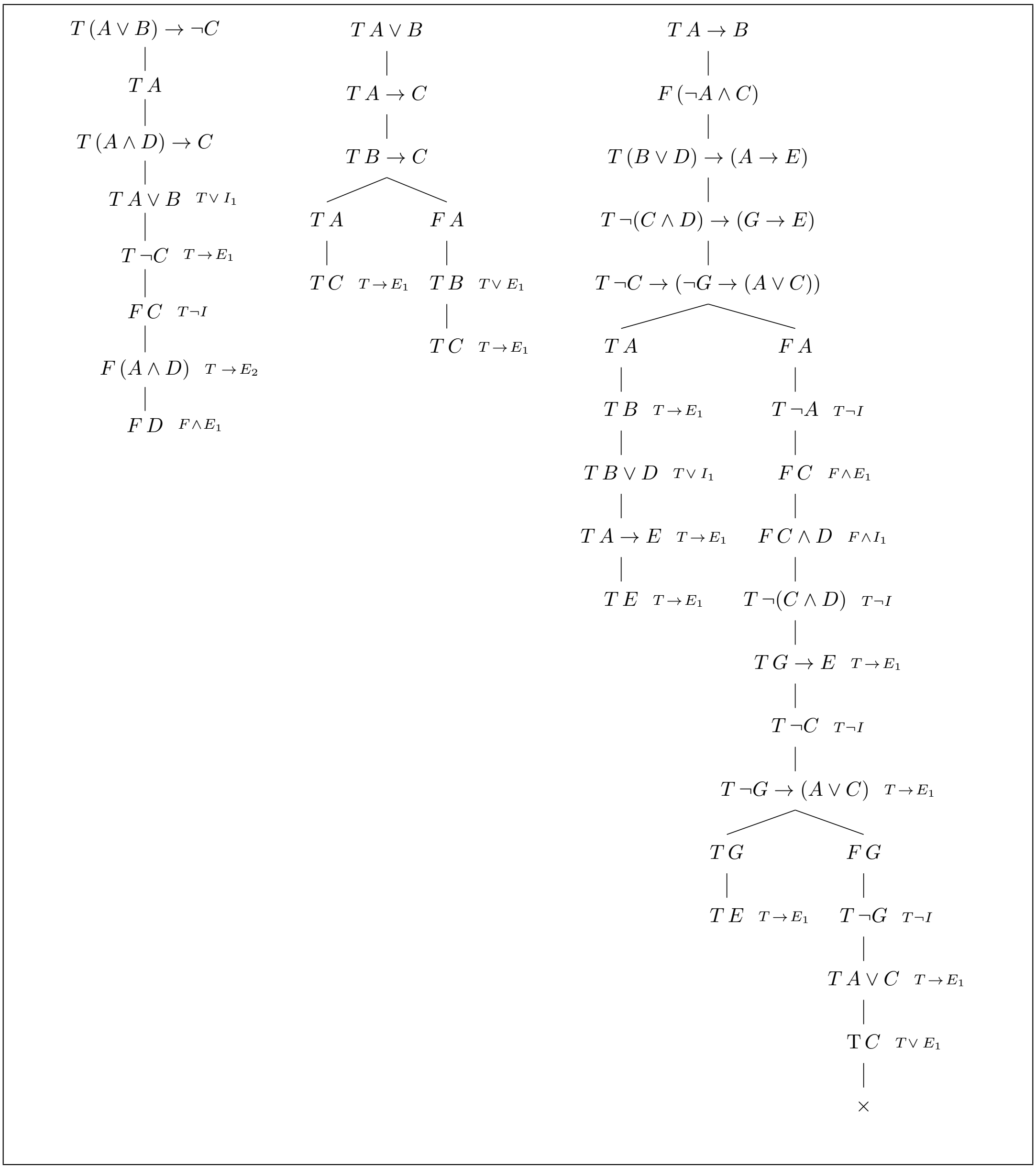


Figure 1: intelim trees of depth 0, 1 and 2 respectively.

from an external source or through (depth-bounded) inference. In a purely informational perspective, the unknown objects themselves could be identified with *ideal*

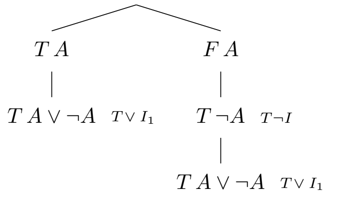


Figure 2: A direct proof of the excluded middle law.

informons: complete (possibly infinite) descriptions in which all properties that can be expressed in the given language are either informationally true or informationally false. This is the limit to which the approximations converge. A “real” informon, on the other hand, is always an approximation of some ideal informon, consisting of all its properties that can be determined in a given information state.

To emphasize the approximation process, we use the same fictitious name (a *parameter*) for the successive approximations of the *same* ideal informon. A parameter thus persists as the identifier of a single ideal informon, while its associated “real” informon refines over time as new information is acquired. For example, if we are informed that some creature has a heart, we may choose a parameter c to denote an unknown ideal informon that provides a complete description of one of the individual creatures which have a heart. At this stage we have only a minimal approximation for c, namely the informon ι_1 whose bundle contains $T\,\mathsf{HasHeart}(x)$. Learning that all creatures with a heart have kidneys puts us in a disposition to assent to $\mathsf{HasKidneys}(c)$; the bundle for c therefore expands to the informon ι_2 containing both $T\,\mathsf{HasHeart}(x)$ and $T\,\mathsf{HasKidneys}(x)$, which represent the best approximation to c available in the current information state. Hence we are in a disposition to assent to $\mathsf{HasKidneys}(c)$.

To summarize:

- parameters statically denote largely unknown objects (ideal informons) that live in *complete* information states, purely *ideal* entities that are highly inaccessible;
- the information content actually carried by a parameter c is dynamically determined by the current information state and consists of an informon ι associated to c that refines as our information grows[3].

[3] This picture fits the classical *monotonic* view of object-level information. In a non-monotonic view, informons can also *shrink*.

We say that $T\,Ax$ $(F\,A(x))$ *holds* of an informon ι to mean that the property expressed by the signed formula in one free variable $T\,Ax$ $(F\,A(x))$ belongs to the bundle of properties that characterize ι. We also write $\iota_1 \sqsubseteq \iota_2$ to mean that ι_1 is an *approximation* of ι_2, that is, $\iota_1 = \iota_2$ or is less defined than it [4].

In general, the following holds:

Informon inheritance property Given any pair of informons ι_1 and ι_2, and any property expressed by the signed formula $T\,Ax$ $(F\,A(x))$,

$$\iota_1 \sqsubseteq \iota_2 \quad \Longrightarrow \quad \text{if } T\,Ax\ (F\,Ax) \text{ holds of } \iota_1, \text{ then } T\,Ax\ (F\,A(x)) \text{ holds of } \iota_2.$$

A remarkable feature of depth-bounded reasoning is that an informon which is consistent at a given depth, may turn out to be inconsistent at a greater depth, although we shall not elaborate on this aspect in the context of this preliminary informal exposition. An example will suffice: if our information state is such that the following set of signed formulas

$$\{T\,A(c) \vee B(c), T\,A(c) \vee \neg B(c), T\,\neg A \vee B(c), T\,\neg A \vee \neg B(c)\}$$

hold in it, the informon currently associated with c is always consistent if the allowed propositional depth is 0, but becomes inconsistent at depth 1.

At the propositional level, on the other hand, information is carried by discrete *information tokens* (or, more briefly, *veritons*), which record the status of propositions (informationally true, informationally false, or indeterminate). Informons and veritons therefore play similar roles in the informational hierarchy: the former track how properties attach to (otherwise undetermined) partial descriptions of ideal informons, while the latter track how informational truth and falsity attach to (otherwise undetermined) propositions. Their interaction will underpin the extension of depth-bounded reasoning to first order logic.

To summarize, informons are partial specifications of objects that approximate their ideal complete descriptions. Two kinds of informons must be distinguished.

Arbitrary (least defined) informons. These correspond to the so-called *arbitrary objects* introduced by universal quantifiers and by the negation of existentials.

[4] There is a structural analogy with Scott domains [15, 16], where elements are partially defined objects ordered by information content and $\perp$ denotes the least defined element. Here, informons form a partially ordered set under inclusion of properties. Distinct minimal informons carrying independent properties need not have a common refinement, although their intersection—and hence a meet—is always defined. Informational reasoning thus unfolds along directed subsets whose suprema correspond to limits of consistent information growth rather than to computational approximations.

Each of them represents an object of which no information is available besides the properties that are known to hold of all informons in the domain. We may denote the arbitrary informon associated to a given information state by $\perp_{\mathsf{o}}$[5]. In the limit, i.e., in a complete information state, $\perp_{\mathsf{o}}$ is the *intersection of all ideal informons* that live in its domain. This limit is itself approximated by "real" informons that live in the domains of partial information states.

At each stage of the inferential process, an arbitrary informon can therefore be identified with the bundle of properties representing the general information currently held about *all* (unknown) objects in the domain. If no general information is available, the bundle of properties initially associated to $\perp_{\mathsf{o}}$ is empty, but may expand at later stages. For example, if we are informed that the domain of discourse consists only of (Euclidean) triangles, the property of being a triangle holds of $\perp_{\mathsf{o}}$, but the property of being a right triangle does not. Later, we may discover new properties that hold of all informons in this domain. For example, that the sum of the interior angles of any triangle equals 180°. Each such discovery enlarges the bundle of properties belonging to $\perp_{\mathsf{o}}$. In this sense, learning and reasoning can refine the *information content carried by* $\perp_{\mathsf{o}}$ without altering its structural role as the least defined element of the ordering.

Note that by the informon inheritance principle, whatever property holds of an informon also holds of any refinement of it. Such arbitrary informons differ fundamentally from objects picked at random from some unknown domain: they are epistemic constructions, bundle of properties held by all the informons in the domain. Hence, while we may assume that an object picked at random either satisfies the property $T\,A(x)$ or satisfies $F\,A(x)$, we cannot make the same assumption for arbitrary informons. If $T\,A(x)$ holds of an arbitrary informon, it holds of all informons that may be introduced in the information state, and likewise for $F\,A(x)$. If we show that a property holds of an arbitrary triangle, it holds of all triangles; if we show that its negation holds of an arbitrary triangle, then no triangle has that property.

Reasoning with arbitrary informons therefore preserves maximal generality with respect to the domain of discourse, while yielding no object-specific information.

Specific informons. These are introduced by existential quantifiers and by the negation of universals. A specific informon embodies the assumption that the properties that hold of it, hold of some unknown object (ideal informon), though most of

[5] This usage parallels that adopted for the undefined truth value of a proposition in the informational semantics of DBBL (see [11]). In both cases, $\perp$ marks the least informative element of the corresponding domain—the starting point of informational growth.

its properties are not yet determined. Through successive inferences the information attached to it can be refined, producing progressively more defined informons. Such refinement mirrors the way witnesses are used in classical proof theory, but here it is viewed as a controlled expansion of the agent's information state. For example, in the domain of Euclidean triangles, the informon characterized by the property of having two equal sides may later be refined by adding the property that two of its angles are equal. Note that a specific informon is always a refinement of some arbitrary informon which is the bottom element of the information ordering. Hence, by the informon inheritance principle, any property that holds of an arbitrary informon holds also of any specific informon.

For future reference, the principal symbols employed in the informational semantics are summarized in Table 3.

Symbol	**Informal meaning**
$T\,A$	the information that the proposition A is true
$F\,A$	the information that the proposition A is false
$\bot$	undefined truth value (no information about A)
ι	a generic informon (bundle of properties, object–level information)
$\bot_{\mathsf{o}}$	arbitrary (least defined) informon
$\iota_1 \sqsubseteq \iota_2$	ι_1 approximates ι_2 (is equal or less defined than ι_2)

Table 3: Basic symbols for informational semantics.

What do we mean when we say that we *actually possess the information* that a sentence of the form $\forall x\, A$ or $\exists x\, A$ is true, or that it is false?

Let us start with the notion of *actually possessing* the information that $\forall x\, A$ is true. The answer cannot be that we literally possess the information that $A[x/a]$ is true for *all* the unknown objects of the domain. A more feasible and information-driven answer is the following: we are in the disposition to assent to *any* sentence of the form $A[x/a]$ where a stands for any ideal informon that is approximated by the real one currently associated to $\bot_{\mathsf{o}}$. Since any informon is a refinement of the arbitrary informon $\bot_{\mathsf{o}}$, being in a disposition to assent to $A(c)$ for *any* ideal informon c is tantamount to saying that $T\,A[x/a]$ holds for the arbitrary (least defined) informon $\bot_{\mathsf{o}}$.

Let us now turn to the notion of *actually possessing* the information that $\exists x\, Ax$ is true. This means that there is a specific ideal informon b that satisfies $T\,A(x)$. We may not yet know which further properties distinguish b from other, compatible ideal informons: the commitment is only that some (as yet largely undetermined) ideal informon satisfies $T\,A(x)$ as well as all the properties that hold of $\bot_{\mathsf{o}}$. Accordingly, we are in the disposition to assent to $A[x/b]$, where b stands for some ideal informon currently approximated by a real informon ι that represents the approximation of b in the current information state; subsequent inference may *refine* the informon associated to b by adding further properties.

A similar account applies to falsity. To *actually possess* the information that $\forall x\, Ax$ is false is to hold that there is a specific ideal informon that satisfies $F\,A(x)$. We are therefore in the disposition to dissent from $A[x/b]$, where b stands for some ideal informon currently approximated by the minimal informon ι such that $F\,A(x)$ holds of ι; as new information is acquired, the informon associated to b may be refined by adding properties that provide a better approximation. On the other hand, to actually possess the information that $\exists x A(x)$ is false means to be in a disposition to dissent from $A[x/a]$ where a names the arbitrary, least defined, informon and, therefore, from $A[x/b]$ for any specific ideal informon b.

A simple example will help clarify. Suppose we are informed that

$$\textit{Something is either a blick or a gleem.} \tag{1}$$

At this stage, we are in a disposition to assent to “b is a blick or a gleem” for some unkown object b about which this is the only information we possess. So the informon associated to b at this stage is just the minimal one whose only characterizing property is $T\,\mathit{blick}(x) \vee \mathit{gleem}(x)$. Next, we learn that

$$\textit{All blicks are plocks.} \tag{2}$$

Now, the property $T\,\mathit{blick}(x) \rightarrow \mathit{plock}(x)$ holds of $\bot_{\mathsf{o}}$ and hence also of the informon associated to b. Finally we learn that

$$\textit{All gleems are plocks.} \tag{3}$$

The current approximation of $\bot_{\mathsf{o}}$ thus has both implications in its bundle, and the informon associated with b is updated accordingly.

If no virtual information is allowed, the informon associated to b at depth 0 is stable. At depth 1, on the other hand, we can distinguish the cases $T\,\mathit{blick}(b)$ and $T\,\mathit{gleem}(b)$, each of which corresponds to a better approximation to the ideal informon b. Since in either case we infer $\mathit{plock}(b)$, the informon associated to b now supports also $T\,\mathit{plock}(x)$.

This example shows how the semantics of informational objects mirrors the growth of information brought about by additional premises and by logical inference. Moreover, it shows how the best approximations of the unknown objects in the domain that is available at depth $k > j$ can be more accurate than the best available at depth j.

4 Intelim and structural rules for FOL

In this section we propose intelim rules for quantifiers. These rules are formulated in accordance with their informational meaning explained in the previous section. We also introduce a version of RB that is adequate for quantificational reasoning.

Recall that in our framework a *parameter* is a syntactic symbol that identifies an ideal informon introduced in the course of inference. Parameters serve as persistent names for these ideal informons and may take the form of a functional term to track dependencies among them.

We use three sorts of parameters:

- **γ-parameters (arbitrary).** Stand for the same arbitrary (least defined) informon $\bot_{\mathsf{o}}$. They behave like free variables and carry no dependency arguments.

- **δ-parameters (specific).** Stand for specific ideal informons introduced by $T\,\exists$ or $F\,\forall$ elimination. A δ-parameter may carry a *dependency vector* of the currently active γ's, written $d[g_1, \ldots, g_k]$, as a bookkeeping device. The functional form of δ's will be used in connection with the γ-substitution rule (see below).

- **ε-parameters (neutral).** Introduced only by the branching rule RB. They may later be *declared arbitrary-for-$T\,A$* (or *arbitrary-for-$F\,A$*) in exactly one of the sibling subtrees generated by the branching; thereafter they behave as γ's in that subtree and as δ's in the sibling subtree. Declarations will be used in connection with the $T\,\forall$ and $F\,\exists$ introduction rules (see below).

Typographic convention. We distinguish parameter sorts by their initial letter (indices are used only when needed):

$$\underbrace{g, g_1, g_2, \ldots}_{\gamma\text{ (arbitrary)}} \qquad \underbrace{d, d_1, d_2, \ldots}_{\delta\text{ (specific)}} \qquad \underbrace{e, e_1, e_2, \ldots}_{\varepsilon\text{ (neutral)}}$$

Global discipline. (i) No parameter occurs in the *premises.* (ii) Declarations for ε's are *subtree-local.*

We use A^x_t to denote the result of substituting all occurrences of x in A with the parameter t. Conversely, we use A^t_x to denote the result of substituting all occurrences of t in A with the variable x.

$T\,\forall$ and $F\,\exists$ elimination.

$$\frac{T\,\forall x\, A}{T\, A^x_g}\ (T\,\forall E_\gamma) \qquad \frac{F\,\exists x\, A}{F\, A^x_g}\ (F\,\exists E_\gamma)$$

Side condition: g is a γ-parameter that does not occur in $A(x)$. It need not be globally fresh; the same g may be reused in other applications of $(\forall E_\gamma)$ provided it satisfies this local condition.

Although all γ-parameters represent the same arbitrary informon $\bot_{\mathsf{o}}$, they are treated syntactically as distinct placeholders. This allows simultaneous reasoning about several occurrences of an arbitrary object—for instance in $R(g_1, g_2)$—without attributing informational differences to the objects themselves. The differentiation among γ's is purely syntactic, reflecting distinct *uses* of the same informational constant rather than distinct entities. In this respect, γ-parameters play the role of *eigenvariables*: temporary names for an arbitrary ideal informon, semantically identical to all others but locally distinguished so that reasoning steps remain independent. The need for several γ-parameters becomes apparent when the rules that introduce them are applied in connection with the following substitution rule.

γ-Substitution.

$$\frac{T\, A(g)}{T\, A^g_t}\ (\gamma\text{-Subst}) \qquad \text{if } g \text{ is a } \gamma \text{ and } t \text{ is any term free for } g \text{ in } A.$$

Reading: γ's behave like free variables; once $T\, A(g)$ is obtained, g may be instantiated by any term t. Therefore, in $R(g_1, g_2)$ the two arbitrary parameters can be instantiated by (possibly) distinct terms.

In applications of the γ-substitution rule if a A contains a δ parameter d that depends on g, the substitution A^g_t replaces also the occurrences of g in the dependency vector associated with d. For example: in $T\,\forall x \forall y(\exists z R(x, z) \wedge \exists z R(y, z))$ we can apply $T\,\forall E_\gamma$ and $T\,\exists E_\delta$ to obtain $R(g_1, d_1[g_1]) \wedge R(g_2, d_2[g_2])$. If we then apply γ-Subst by replacing g_1 with t_1 and g_2 with t_2, the result is $R(t_1, d_1[t_1]) \wedge R(t_2, d_2[t_2])$. For brevity, γ-Subst can be applied simulateneously to different γ-parameters occurring in A, just as we did in the above example.

$T\,\forall$ and $F\,\exists$ introduction.

$$\frac{T\,A(g)}{T\,\forall x\,A^x_g}\,(T\,\forall I_\gamma) \qquad \frac{F\,A(g)}{F\,\exists x\,A^g_x}\,(F\,\exists I_\gamma)$$

Side condition: g is a γ-parameter.

These rules reflect the role of γ-parameters as arbitrary (least defined) informons. Whatever holds for a γ-paramenter holds also for any informon that can be introduced in the reasoning process.

When the truth of an existential or the falsity of a universal enters our information state, reasoning creates a new *specific informon* representing the (unknown) object whose existence is informationally asserted. In the syntax, this object is represented by a *new parameter.* Parameters are persistent identifiers of *ideal* informons: each parameter continues to denote the same ideal informon while its associated *real* informon—the current approximation available in the information state—is progressively refined through inference. When the *ideal informon* introduced by $T\,\exists A$ or $F\,\forall A$ depends on previously introduced arbitrary informons, its parameter takes the functional form $d[\vec{\gamma}]$, where the elements of $\vec{\gamma}$ are the γ-parameters occurring in A. The functional notation merely records informational dependencies; it has no semantic import. Introducing a term such as $d[\vec{\gamma}]$ to replace $T\,\exists A$ or $F\,\forall A$ should therefore not be read as postulating a function of $\vec{\gamma}$, but as introducing a new specific informon that depends on the terms that replace the γ's when the γ-substitution rule is applied.

$T\,\exists$ and $F\,\forall$ elimination.

$$\frac{T\,\exists x\,A}{T\,A^x_{d[\vec{\gamma}]}}\,(T\,\exists E_\delta) \qquad \frac{F\,\forall x\,A}{F\,A^x_{d[\vec{\gamma}]}}\,(F\,\forall E_\delta)$$

Side condition: $d(\vec{\gamma})$ is a fresh δ-parameter whose argument list consists exactly of the γ's active at the introduction site.

$T\,\exists$ and $F\,\forall$ introduction.

$$\frac{T\,A(t)}{T\,\exists x\,A^t_x}\,(T\,\exists I_\delta) \qquad \frac{F\,A(t)}{F\,\forall x\,A^t_x}\,(F\,\forall I_\delta)$$

Micro-example From $T\,\forall x\,(A(x) \to B(x))$ and $T\,\exists x\,A(x)$, infer $T\,\exists x\,B(x)$:

$$\begin{array}{ll}
T\,\forall x\,(A(x) \to B(x)) & \text{premise} \\
T\,\exists x\,A(x) & \text{premise} \\
\hline
T\,A(d) & T\,\exists E_{\delta} \\
T\,(A(g) \to B(g)) & T\,\forall E_{\gamma} \\
T\,(A(d) \to B(d)) & \gamma\text{-Subst} \\
T\,B(d) & T \to E \\
T\,\exists x\,B(x) & T\,\exists I_{\delta}
\end{array}$$

Remark 4.1. *We call a Q-*formula *any formula whose main logical operator is a quantifier. We say that Q-formula is* fully instantiated *in a branch r of an intelim tree if all its initial quantifiers have been eliminated (replaced by γ's and δ's) in r. For example: the formula $\forall x \exists y\, R(x,y)$ is fully instantiated in r if r contains $R(g,d)$ for some γ-parameter g and some δ-parameter d.*

If a Q-formula of the form $Q_1 x_1 \cdots Q_n x_n\, A(x_1, \ldots, x_n)$ is fully instantiated in a branch while the main operator of A is Boolean and some subformula of A is itself a Q-formula, that inner Q-formula may not be fully instantiated in that branch, depending on the upper bound on the propositional depth of the intelim tree. In the micro-example above, all Q-formulas are fully instantiated at depth 0. If we add the premise

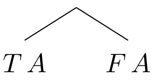
$$T\,\forall x \forall y\, \left(\exists z\,(R(x,y) \wedge R(y,z)) \;\vee\; \exists z\,(S(x,y) \wedge S(y,z))\right),$$

it becomes fully instantiated (without branching) as

$$T\;\exists z\,(R(g_1,g_2) \wedge R(g_2,z)) \;\vee\; \exists z\,(S(g_1,g_2) \wedge S(g_2,z)),$$

but the existential subformulas inside the disjunction cannot be fully instantiated at depth 0, because the disjunction may not be unpacked unless we are informed of he falsity of one disjunct. At depth 1, via RB, both existentials are fully instantiated in at least one branch. In general, there is always a propositional depth at which all the Q-formulas occurring in the premises become fully instantiated.

RB and neutral parameters The RB rule takes the same branching form as in the propositional case:



with the following provisos: (i) the formula A contains no γ-parameter; and (ii) at the split one may optionally introduce *fresh* ε-parameters as neutral names on the child branches; ε's have no arguments.

As for condition (i), just note that RB brings us to a virtual information state where A is decided. If A contains some γ-parameter, this would amount to assuming that A^g_x is always either true of all informons or false of all informons. Hence if a γ-parameter occurs in A one may reach incorrect conclusions as shown in Figure 3. What if, at depth > 0 we need to split on a formula that contains γ-parameters in order to reach deeper consequences of the assumptions? For example, from the following assumptions we can derive the conclusion $\exists x C(x)$[6] :

$$\begin{aligned}&T\,\forall x(A(x) \vee B(x))\\&T\,\forall x(A(x) \rightarrow C(x))\\&T\,\forall x(B(x) \rightarrow C(x)).\end{aligned}$$

Now, if we use $T\forall E_\gamma$, we obtain

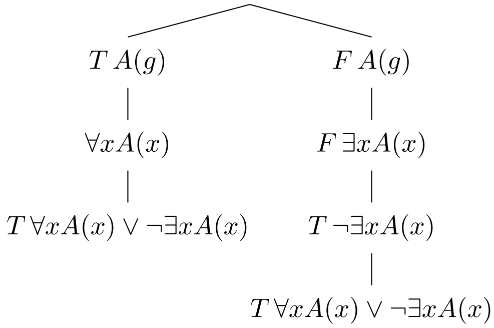

$T\,A(g)$
$F\,A(g)$
$\forall x A(x)$
$F\,\exists x A(x)$
$T\,\forall x A(x) \vee \neg\exists x A(x)$
$T\,\neg\exists x A(x)$
$T\,\forall x A(x) \vee \neg\exists x A(x)$

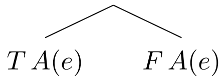
Figure 3: An incorrect depth-1 intelim tree (the RB-formula A contains a γ-paramenter g).

$$\begin{aligned}&T\,A(g) \vee B(g)\\&T\,A(g) \rightarrow C(g)\\&T\,B(g) \rightarrow C(g).\end{aligned}$$

At depth 0, with no use of virtual information, we cannot reach the any further conclusion. At depth 1 we can reason as follows: immagine a refinement of our

[6] We can also derive $\forall x C(x)$, but this case will be addressed later when we discuss "Critical quantifier introductions via neutral ε-parameters".

current information state where $A(x)$ is decided for some e (neutral ε-parameter). Then we can apply RB to the formula $A(e)$ and branch as follows:



We can therefore exploit γ-Subst to obtain the desidered conclusion as shown in Figure 4.

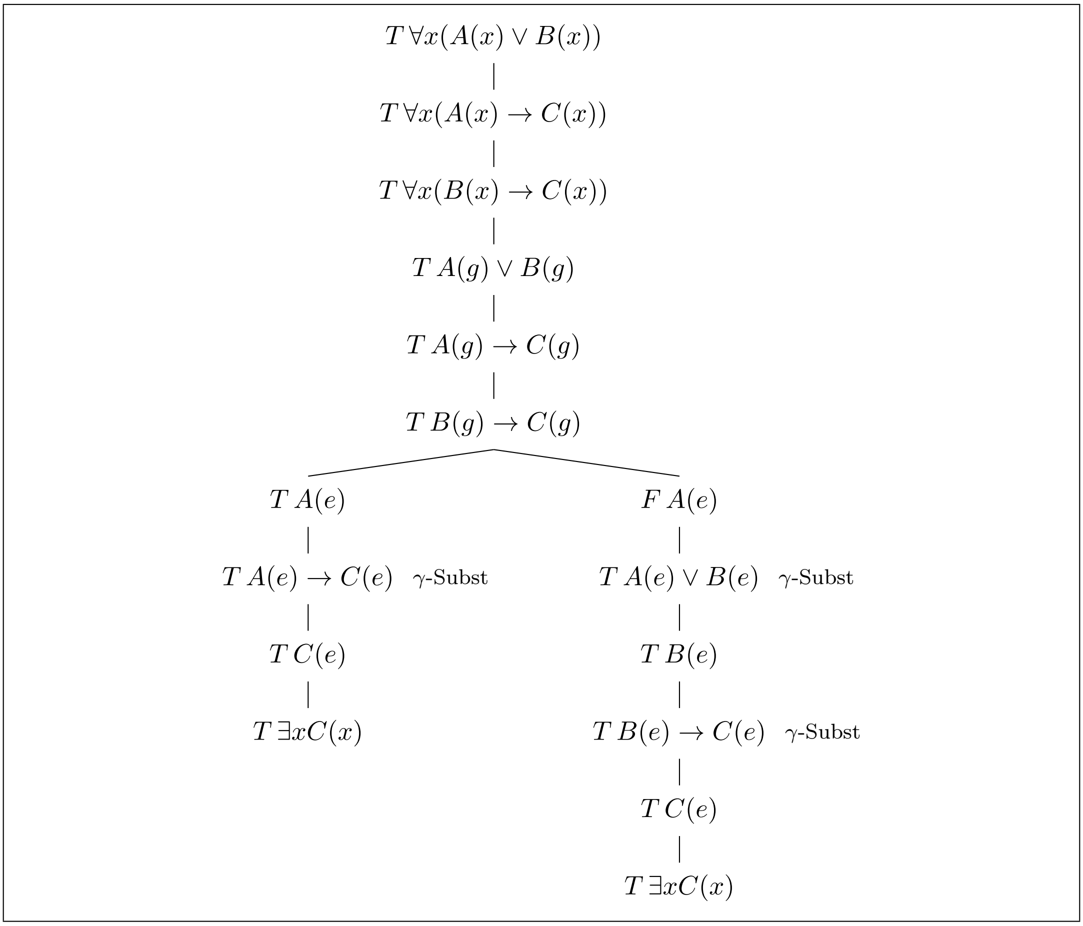
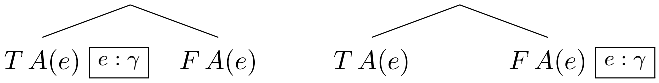


Figure 4: A correct depth-1 intelim tree.

As long as it stays neutral, an ε-parameter e represents the *minimal virtual informon* for which $A(x)$ is decided: along each child subtree, the current virtual information state contains either $T\,A(e)$ or $F\,A(e)$. No special informational role is fixed yet. Such a neutral ε may later be declared (as needed) to play the role of a γ-parameter on exactly one of the sibling subtrees generated by the branching; it then behaves like a δ on the other:



In the first case we say that e has been declared *arbitrary-for-$T\,A$*, and this licenses an application of $T\,\forall$-introduction. In the second, e has been declared *arbitrary-for-$F\,A$*, which licenses an application of $F\,\exists$-introduction. Figure 5 shows two examples of intelim trees with such declarations.

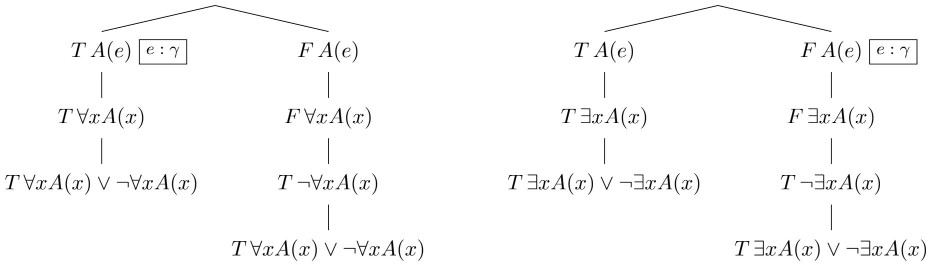

$T\,A(e)$ $\boxed{e:\gamma}$
$T\,\forall x A(x)$
$T\,\forall x A(x) \vee \neg\forall x A(x)$
$F\,A(e)$
$F\,\forall x A(x)$
$T\,\neg\forall x A(x)$
$T\,\forall x A(x) \vee \neg\forall x A(x)$
$T\,A(e)$
$T\,\exists x A(x)$
$T\,\exists x A(x) \vee \neg\exists x A(x)$
$F\,A(e)$ $\boxed{e:\gamma}$
$F\,\exists x A(x)$
$T\,\neg\exists x A(x)$
$T\,\exists x A(x) \vee \neg\exists x A(x)$

Figure 5: Two intelim trees with subtree-local declarations for ε-parameters.

Declaration discipline

- Declarations are subtree-local and cannot be made with contradictory polarities on sibling subtrees (the same e cannot be simultaneously arbitrary-for-$T\,A$ and arbitrary-for-$F\,A$ in sibling subtrees).
- Just as γ-parameters cannot occur in the formula to which RB is applied, an ε declared to be a γ in a subtree cannot occur in any application of RB *below* the node containing the declaration. Equivalently, RB can be applied to formulas containing only δ's or neutral ε's.

Critical quantifier introductions via neutral ε-parameters.

$$\frac{T\,B(e)}{T\,\forall x\,B^e_x}\;(T\,\forall I_\varepsilon) \qquad \frac{F\,B(e)}{F\,\exists x\,B^e_x}\;(F\,\forall I_\varepsilon)$$

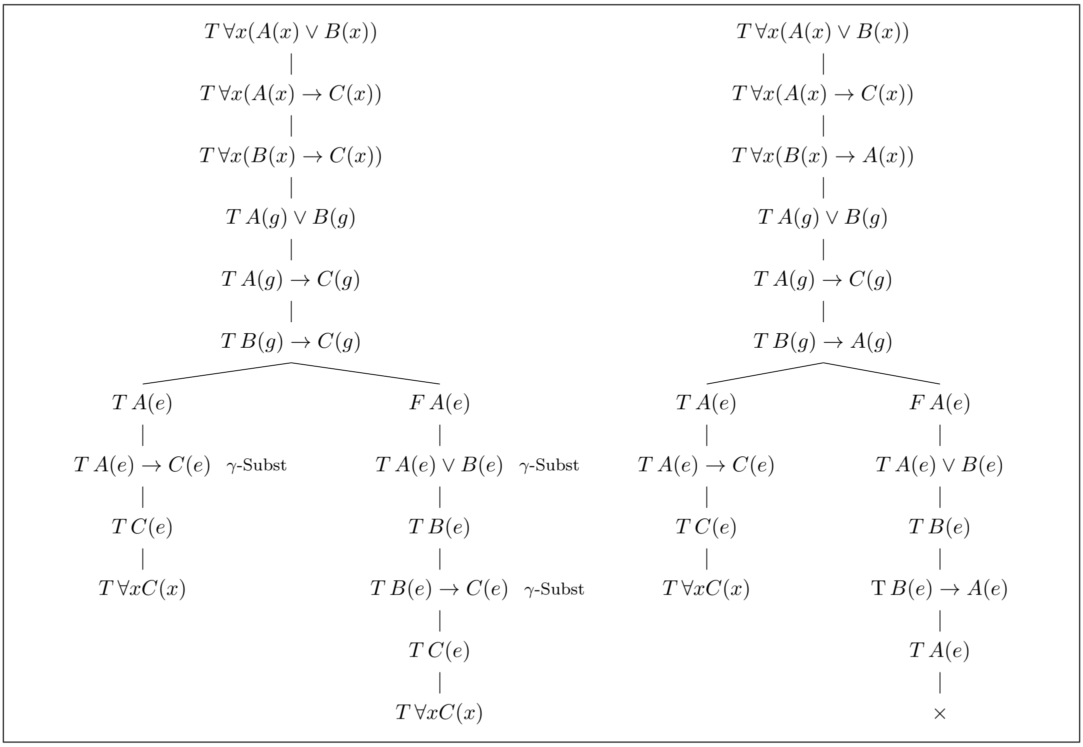

$T\,\forall x(A(x) \vee B(x))$
$T\,\forall x(A(x) \to C(x))$
$T\,\forall x(B(x) \to C(x))$
$T\,A(g) \vee B(g)$
$T\,A(g) \to C(g)$
$T\,B(g) \to C(g)$
$T\,A(e)$
$T\,A(e) \to C(e)$ γ-Subst
$T\,C(e)$
$T\,\forall x C(x)$
$F\,A(e)$
$T\,A(e) \vee B(e)$ γ-Subst
$T\,B(e)$
$T\,B(e) \to C(e)$ γ-Subst
$T\,C(e)$
$T\,\forall x C(x)$
$T\,\forall x(A(x) \vee B(x))$
$T\,\forall x(A(x) \to C(x))$
$T\,\forall x(B(x) \to A(x))$
$T\,A(g) \vee B(g)$
$T\,A(g) \to C(g)$
$T\,B(g) \to A(g)$
$T\,A(e)$
$T\,A(e) \to C(e)$
$T\,C(e)$
$T\,\forall x C(x)$
$F\,A(e)$
$T\,A(e) \vee B(e)$
$T\,B(e)$
$\mathrm{T}\,B(e) \to A(e)$
$T\,A(e)$
×

Figure 6: Examples of intelim trees with an application of $T\forall I_\varepsilon$.

Side condition: $T\,B(e)$ occurs on every open branch that contains e. Two examples are shown in Figure 6. In the righthand tree $C(e)$ one of the two branches is closed and so $C(e)$ still occurs in all *open* branches.

Remark 4.2 (Relation to Hilbert's ε)**.** *ε-parameters are reminiscent of Hilbert's ε-terms in postponing existential commitment via symbolic placeholders. In our signed setting, however, ε's are not term-forming operators: they are informational names introduced by RB that may remain neutral (minimal informons with either $T\,A(e)$ or $F\,A(e)$ per branch) or be declared arbitrary-for-$T\,A$ / arbitrary-for-$F\,A$ subtree-locally, thereby adopting γ-like / δ-like behaviour under the corresponding polarity.*

5 A bivariate measure of depth

The extension of depth-bounded reasoning to first-order logic requires a refinement of the notion of depth itself. Once quantifiers and parameters enter the picture, informational complexity no longer depends solely on the structure of veritons, but also on the number of new parameters introduced in an intelim tree. We therefore introduce a *bivariate* measure of depth that combines the propositional depth—governing the expansion of Boolean structure—with a measure of quantificational depth that records the generation of new parameters during instantiation. This allows us to compare inferences across levels of quantificational complexity while preserving the boundedness conditions that ensure tractable reasoning.

Formally, the quantificational component will be measured by counting the distinct parameters that occur in an intelim tree. The resulting bivariate index (p, q) thus captures, respectively, the depth of propositional reasoning and the minimum number of parameters required to obtain the conclusion. In what follows we shall examine how this measure stratifies first-order reasoning into a hierarchy of increasingly expressive but still finitely bounded fragments.

We shall also introduce a distinction, loosely inspired by [9] (see also [10]) between the quantificational depth required for a surface understanding of the premises and that required for deriving the conclusion. We may regard the first as measuring the *informational load of comprehension*—the minimal number of parameters a p-bounded agent must introduce to grasp what the premises say—whereas the second reflects the *informational cost of reasoning*, that is, the additional parameters that must be generated in order to reach a conclusion. The fact that these two measures need not coincide, and their interconnection with propositional depth captures an important intuition: logical inference can be *informationally creative* at a quantificational level, even when the propositional depth is bounded: it may increase quantificational depth, introducing new parameters that were not implicit in the initial representation of the premises. This divergence between surface understanding and deeper inference plays a central role in the interpretation of the bivariate hierarchy proposed below.

To make these ideas more precise, we shall define the bivariate depth of an intelim tree as an ordered pair (p, q), where p measures the maximal Boolean nesting of rules applied in the tree and q the number of distinct parameters required for instantiation at that propositional depth. The pair thus represents the minimal informational resources necessary for a bounded agent to construct a complete derivation from the given premises. By varying p and q we obtain a two-dimensional hierarchy of coherent depth-bounded approximations to full first-order logic, each capturing a stable level of informational complexity within which reasoning remains attainable within the given (or expected) resources.

Both at the propositional and the quantificational level, there are options about how to measure depth. For example, at the propositional level we have chosen to consider the maximum number of nested applications of RB that are required to reach the conclusion (i.e., the maximum number of virtual assumptions in a branch). However, for other purposes, the total number of RB applications in the whole tree might be more suitable (e.g. see [13]). What counts is that different measures of depth are polynomially related to each other. In this paper we shall therefore adopt the p-depth measure used in DBBL and propose a corresponding q-depth measure for quantificational complexity.

Quantificational depth (q-depth). For a given propositional depth p, replacing the quantifiers occurring in a formula by suitable parameters may already require a certain number of formulas to be fully instantiated at that depth[7]. This number represents the quantificational depth involved in merely *understanding* the premises at depth p. When reasoning proceeds from these premises to a conclusion, additional parameters may be introduced by applications of the quantifier rules or of RB, reflecting the *informational cost of inference.* We measure this cost by counting, for each branch of an intelim tree, the number of distinct parameters that occur in that branch *beyond* those required for the premises to be fully instantiated at the same propositional depth. For this purpose, let us refine the notion of *fully instantiated formula* sketched in Remark 4.1.

Fully instantiated formula. A signed formula is said to be *fully instantiated* in a branch r of an intelim tree if it satisfies the following recursive conditions:

- Any signed formula that contains no quantifiers is fully instantiated.
- A signed formula of the form $T\,\forall x\, A(x)$ (respectively $F\,\exists x\, A(x)$) is fully instantiated in r if (i) $T\,A(g)$ (respectively $F\,A(g)$) occurs in r for some γ-parameter g, and (ii) $A(g)$ is fully instantiated in r.
- A signed formula of the form $T\,\exists x\, A(x)$ (respectively $F\,\forall x\, A(x)$) is fully instantiated in r if (i) $T\,A(d)$ (respectively $F\,A(d)$) occurs in r for some δ-parameter d or neutral ε-parameter e, and (ii) $A(d)$ or $A(e)$ is fully instantiated in r.
- A signed formula of the form $s\,(A \circ B)$—where s is a sign and $\circ$ any binary connective—or $s\,\neg A$ is fully instantiated in r if every signed formula obtained from it by an application of an intelim rule to formulas already in r is itself fully instantiated in r.

A signed formula is *fully instantiated on a tree* if it is fully instantiated in every branch of that tree.

[7] Recall Remark 4.1 and the definition of “fully instantiated” just above it.

Formally, if $q_{\mathsf{prem}}(p)$ denotes the maximum number of distinct parameters *required* on a branch r for full instantiation of the premises at depth p, and $\mathsf{params}(r)$ the set of parameters occurring in branch r, the *quantificational depth* (or *q-depth*) of an intelim tree T is defined as:

$$q(T, p) = \max_{r \in \mathsf{branches}(T)} \big(|\mathsf{params}(r)| - q_{\mathsf{prem}}(p)\big).$$

A q-depth of zero does not exclude quantificational reasoning: it indicates that all inferences can be carried out without introducing new parameters beyond those already required for the premises to be fully instantiated. Positive values of q mark genuinely creative steps, where reasoning generates additional specific or arbitrary informons and thus increases quantificational depth relative to mere understanding. In the micro-example illustrating the intelim rules in the previous section the p-depth measure and the q-depth measure are both 0. The reader can verify that in the example discussed in Remark 4.1, $q(T, 0) = q(T, 1) = 0$; $q_{\mathsf{prem}}(0) = 2 \neq q_{\mathsf{prem}}(1) = 3$.

Examples for which $q(1)$ is positive, according to the proposed measure, are the ones in Figures 4 and 6. For the examples in Figure 5, $p = q = 1$: to reach the conclusion we need to introduce virtual information about an individual that is not implicitly mentioned in the premises, since there are no premises.

A more interesting example is illustrated in Figure 7, which shows a minimal intelim tree of depth 0 with premises P_1–P_3 and conclusion C. According to the proposed measure, its q-depth is equal to 2. A similar example is discussed in [7]. For the sake of readability, we omit the sequence of quantifier eliminations that make the premises fully instantiated in the (single) branch, as well as other trivial applications of intelim rules or of γ-Subst.

In the single branch we use the parameters $\{g_1, g_2, g_3, d_1, d_2, d_3\}$; since $q_{\mathsf{prem}}(0) = 4$ (the three γ's and d_1), the derivation introduces two additional δ-parameters, hence $q(T, 0) = 2$.

Let us now say that Γ *entails* A *at depth* (p, q) if there exists an intelim tree with premises $T\,\Gamma$ and conclusion $T\,A$ such that its bivariate complexity is bounded above by (p, q). Note that the application of the rules does not guarantee that a conclusion is obtained with an intelim tree of minimal complexity, but only that it is possible apply the rules by means of a tree of the given complexity. Proper complexity results based on this bivariate measure will be developed in future work.

6 Conclusions and future work

The framework presented in this paper extends depth-bounded reasoning from the propositional to the first-order level by grounding quantification in the informa-

$P_1: \quad T\,\forall x\,\exists y\,R_1(x,y)$

$P_2: \quad T\,\forall x\,\forall y\,\forall z\,\big((R_1(x,y) \wedge R_1(y,z)) \to R_2(x,z)\big)$

$P_3: \quad T\,\forall x\,\forall y\,\forall z\,\big((R_2(x,y) \wedge R_1(y,z)) \to R_3(x,z)\big)$

$\vdots$

$T\,R_1(g_1,d_1) \quad$ (from P_1 by $\exists E_\delta$, $d_1 = d[g_1]$)

$\vdots$

$T\,\big((R_1(g_1,g_2) \wedge R_1(g_2,g_3)) \to R_2(g_1,g_3)\big) \quad$ (from P_2 by $\forall E_\gamma$)

$T\,\big((R_2(g_1,g_2) \wedge R_1(g_2,g_3)) \to R_3(g_1,g_3)\big) \quad$ (from P_3 by $\forall E_\gamma$)

At this stage the premises are fully instantiated; $q_{\mathsf{prem}}(0) = 4$.

$\vdots$

$T\,R_1(d_1,d_2) \quad$ (from P_1 by $\exists E_\delta$, $d_2 = d[d_1]$; via γ-Subst on P_2)

$T\,R_2(g_1,d_2) \quad$ (from P_2 by $\to E$)

$T\,R_1(d_2,d_3) \quad$ (from P_1 by $\exists E_\delta$, $d_3 = d[d_2]$)

$T\,R_3(g_1,d_3) \quad$ (from P_3 by γ-Subst and $\to E$)

$T\,\exists y\,R_3(g_1,y) \quad (\exists I)$

$C: \quad T\,\forall x\,\exists y\,R_3(x,y) \quad$ ($\forall I_\gamma$ on g_1)

Figure 7: An example with bivariate complexity $(p,q) = (0,2)$.

tional semantics of *informons*. This shift replaces the classical view of quantifiers as ranging over fixed domains with an account of reasoning about partially specified objects—bundles of properties that evolve as information grows. The resulting proof theory preserves the inferential transparency of the propositional calculus while providing a direct representation of the informational commitments introduced by quantifiers.

The intelim rules developed in Section 4 show that universal and existential reasoning can be formulated without discharge, through explicit tracking of parameters representing ideal informons. The corresponding discipline of γ-, δ- and ε-parameters ensures that informational dependencies are maintained locally within branches, preserving soundness and avoiding spurious interactions between quantifier instances. In particular, the integration of the branching rule RB with neutral ε-parameters provides a uniform treatment of virtual information, allowing the same mechanism to govern both propositional alternatives and quantificational commitments.

Section 5 introduced a bivariate measure of depth (p,q) that jointly captures the

Boolean and quantificational dimensions of bounded reasoning. This measure offers a principled way to stratify first-order inference into finitely bounded fragments and to distinguish between the *informational load of comprehension*—the parameters needed to understand the premises—and the *informational cost of reasoning*—those additionally required to reach a conclusion. The divergence between these two aspects reflects a key philosophical insight: reasoning, even within fixed informational bounds, can be *informationally creative.*

The present study is intended as a conceptual foundation rather than a set of definitive technical results. Future work will aim to refine the bivariate hierarchy, to investigate how the two dimensions of depth interact in concrete examples, and to clarify the boundary between purely propositional and quantificational reasoning. Further exploration may also reveal how informational semantics can connect depth-bounded inference with broader questions about bounded rationality and the dynamics of understanding. In this sense, the framework proposed here is meant as a step toward a more unified view of reasoning as an evolving informational process.

References

[1] J.M. Crawford and D.W. Etherington. A non-deterministic semantics for tractable inference. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI '98)*, pages 286–291. AAAI Press, 1998.

[2] M. D'Agostino. Tractable depth-bounded logics and the problem of logical omniscience. In F. Montagna and H. Hosni, editors, *Probability, Uncertainty and Rationality*, CRM series, pages 245–275. Springer, 2010.

[3] M. D'Agostino. Analytic inference and the informational meaning of the logical operators. *Logique et Analyse*, 227:407–437, 2014.

[4] M. D'Agostino. An informational view of classical logic. *Theoretical Computer Science*, 606:79–97, 2015.

[5] M. D'Agostino, M. Finger, and D.M. Gabbay. Semantics and proof-theory of depth-bounded Boolean logics. *Theoretical Computer Science*, 480:43–68, 2013.

[6] M. D'Agostino and L. Floridi. The enduring scandal of deduction: Is propositional logic really uninformative? *Synthese*, 167(2):271–315, 2009.

[7] M. D'Agostino, C. Larese, and S. Modgil. Towards depth-bounded natural deduction for classical first-order logic. *Journal of Applied Logics – IfCoLog*, 8(2):423–451, 2021.

[8] D.M. Gabbay and J. Woods. The new logic. *Logic Journal of the IGPL*, 9(2):141–174, 2001.

[9] J. Hintikka. *Logic, Language Games and Information: Kantian Themes in the Philosophy of Logic.* Clarendon Press, Oxford, 1973.

[10] C. Larese. Hintikka's conception of syntheticity as the introduction of new individual. *Synthese*, 201, 2023.

[11] C. Larese M. D'Agostino, D.M. Gabbay and S. Modgil. *Depth-bounded Reasoning. Volume 1: Classical Propositional Logic.* Number 102 in Studies in Logic. College Publications, London, 2024.

[12] M. Mondadori. An improvement of Jeffrey's deductive trees. Annali dell'Università di Ferrara; Sez. III; Discussion paper 7, Università di Ferrara, 1989.

[13] P. Pardo. A modal view on resource-bounded propositional logics. *Studia Logica*, 110(4):1035–1080, 2022.

[14] W.V.O. Quine. *The Roots of Reference.* Open Court, 1974.

[15] Dana S. Scott. Outline of a mathematical theory of computation. In *Proceedings of the 4th Annual Princeton Conference on Information Sciences and Systems*, pages 169–176, Princeton, NJ, 1970. Princeton University. Reprinted in *Lecture Notes in Computer Science*, Vol. 188, Springer, 1985.

[16] Dana S. Scott. Domains for denotational semantics. In M. Nielsen and E. M. Schmidt, editors, *Automata, Languages and Programming*, volume 140 of *Lecture Notes in Computer Science*, pages 577–613. Springer, Berlin, Heidelberg, 1982.

Received October 2025

What's Logic?

Nachum Dershowitz
School of Computer Science and AI, Tel Aviv University
nachum@tau.ac.il

> Mathematics may be defined as the subject in which we never know what we are talking about nor whether what we are saying is true.
>
> Bertrand Russell

Abstract

Various suggestions as to what constitutes a "logic", the product of multiple discussions with Dov Gabbay, are summarized.

1 Introduction

For many years, Dov Gabbay was a frequent visitor and periodic speaker at the Advanced Seminar in Logic and Formal Methods held in the School of Computer Science of Tel Aviv University,[1] which met for two hours or so nearly every week during school terms. Regular participants included Arnon Avron, the late Yoram (Joram) Hirshfeld, Alex (Alexander) Rabinovich, the late Boaz (Boris) Trakhtenbrot, and myself. Colleagues from other Israeli institutions and visitors from abroad frequented and lectured at these meetings as well. Bob (Robert) Constable was a recurring attendee. Those of our graduate students who were working in related fields also took part.

Whenever Dov was on an extended stay in Israel, he would join us to present aspects of his latest research and results. The following list of titles of some of his talks highlights the breathtaking breadth of his interests and his manifold accomplishments:

[1] Now the School of Computer Science and AI.

- January 2005: What is a logical system, 2005?
- January 2007: A probabilistic approach to multiple choice voting.
- November 2008: Modal provability foundations for logical failure, with illustration in argumentation networks.
- May 2009: Talmudic logic.
- November 2009: Reactive intuitionistic models.
- February 2011: Equational approach to argumentation networks.
- January 2014: Avron's matrices and my equational approach.
- December 2014: Probabilistic argumentation theory.

Digressions were a prominent and indispensable feature of these meetings. Rarely would a speaker manage to complete his planned presentation in the allotted time.

The question of what constitutes a logic came up every now and then, especially when Dov—the world's preeminent authority on the vast gamut of logics—was present.

In what follows, I summarize our group's insights on this issue that were brought out in what we referred to as a "Logic Jamboree", held on May 4, 2011. Discussions continued into June of that year and resumed in June 2015, after Dov published another essay on the subject,[2] and after the numerous email conversations that ensued.

Of course, no agreed-upon conclusion was reached.

2 The Problem

There are many formal languages and deductive systems in the literature that are called "logics", some more logical than others. We asked ourselves: What is a logic—in its most general form? What are its constituents?

This is a subject that Dov had been pondering for ages. The abstract of his talk at our seminar back in 2005, entitled, "What is a logical system, 2005?" began as follows:

[2] Dov M. Gabbay, "What is a logical system? An evolutionary view: 1964–2014", in Jörg H. Siekmann, ed., *Handbook of the History of Logic*, vol. 9, North-Holland, pp. 41–132, 2014.

> The first paper under this title was published in 1994, in a book under the same title.[3] The idea is to assess the kinds of logical systems required by the landscape of applications of logic in computer science, artificial intelligence, language, law, common sense reasoning, etc., etc. The lecture will motivate and present the author's current view of the notion.

3 Background

Over the centuries, there have been very many descriptions of what is the essence of logic.

For Aristotle, the 4th-century-BCE founder of logical enterprise as an intellectual discipline, logic is said to be...

> ... the instrument (the "organon") by means of which we come to know anything.[4]

A plethora of other definitions may be found in the literature:
Logic may be defined as...

- ... as the science of thinking.[5]
- ... the science of the laws of thought considered as thought.[6]
- ... the science of reasoning.[7]
- ... the *Science of Deductive Thinking.*[8]
- ... the general science of reasoning considered in its most abstract sense, that is to say, as far as possible independently of any special subject of investigation.[9]

[3] Dov M. Gabbay, ed., *What is a Logical System?*, Clarendon Press, Oxford, 1994.

[4] Garth Kemerling, *Philosophy Pages*, `https://www.philosophypages.com/hy/2n.htm`.

[5] Georg Wilhelm Friedrich Hegel, *Wissenschaft der Logik (Science of Logic)*, 1812, §33.

[6] Edward Tagart, *Remarks on Mathematical or Demonstrative Reasoning: Its Connexion with Logic; and Its Application to Science, Physical and Metaphysical, with Reference to Some Recent Publications*, John Green, London, 1837, quoting a draft entry of the *Encyclopedia Britannica.*

[7] John Stuart Mill, *A System of Logic, Ratiocinative and Inductive: Being a Connected View of the Principles of Evidence, and Methods of Scientific Investigation*, 1843, §2.

[8] William Dexter Wilson, *An Elementary Treatise on Logic: Including Pt. I. Analysis*, 1856.

[9] Hugh McColl, "Symbolical Reasoning", *Mind: A quarterly review of psychology and philosophy* V, 1880.

- … the science of the regulative principles of thought, that is, the science of the axioms and laws to which thought must conform in order that it may be valid.[10]
- … the *Science of the Laws of Thought*, or *The Science which directs the operations of the intellect in its knowledge of Truth*, or *The Science which is concerned with the observance of due order in our intellectual operations.*[11]
- … the science which directs the operations of the mind in the attainment of truth. [12]
- … the science of the principles and conditions of correct thinking.[13]
- … the systematic knowledge of the necessary forms of thinking; it has nothing to do with the things we think about.[14]
- … the science of evaluating human reasoning[15]
- … the organized body of knowledge, or science, that evaluates arguments.[16]
- … any precise notation for expressing statements that can be judged true or false.[17]

Averroes in the 12th century averred the following:[18]

> Since it has been determined that the Law makes it obligatory to reflect upon existing things by means of the intellect and to consider them; and consideration is nothing more than inferring and drawing out the unknown from the known; and this is syllogistic reasoning or by means of syllogistic reasoning; therefore, it is obligatory that we go about reflecting upon the existing things by means of intellectual syllogistic

[10] Prasamma Kumar Ray, *A Text-book of Deductive Logic: For the use of students*, 1892.

[11] Richard Frederick Clarke, *Logic*. Longmans, Green & Co., London, 3rd ed. 1895.

[12] George Hayward Joyce, *Principles of Logic*, p. 1, 1908.

[13] Roy W. Sellars , *The Essentials of Logic*, 1917.

[14] Helge Lundholm, “Logical-aesthetic cognition: The formal appeal of art”, in H. Lundholm, *The Aesthetic Sentiment: A criticism and an original excursion*, Sci-Art Publishers, pp. 87–106, 1941.

[15] Bangs L. Tapscott, *Elementary Applied Symbolic Logic*, Prentice Hall, NJ, 1976.

[16] Patrick J. Hurley, *Logic & Critical Thinking*, Wadsworth, Australia, 1982.

[17] John Sowa, “Fads and fallacies about logic”, *IEEE Intelligent Systems*, pp. 84–87, 2007.

[18] *The Book of the Decisive Treatise, Determining the Connection Between the Law and Wisdom*, trans. Charles E. Butterworth.

> reasoning. And it is evident that this manner of reflection the Law calls for and urges is the most complete kind of reflection by means of the most complete kind of syllogistic reasoning and is the one called "demonstration."

Immanuel Kant wrote in his *Critique of Pure Reason*:[19]

> [Logic] deals with concepts, judgments, and inferences, corresponding exactly to the functions and order of those powers of the mind, which are comprehended under the broad designation of understanding in general.... If the understanding in general is explained as the faculty of rules, then the power of judgment is the faculty of subsuming under rules, i.e., of determining whether something stands under a given rule or not.

Gottlob Frege:[20]

> I did not wish to present an abstract logic in formulas, but to express a content through written symbols in a more precise and perspicuous way than is possible with words.... I wished to produce, not a mere calculus ratiocinator, but a *lingua characteristica* in the Leibnizian sense. I wish to blend together the few symbols which I introduce and the symbols already available in mathematics to form a single formula language.... Existing symbols correspond to the word-stems of language; while the symbols I add to them are comparable to the suffixes and form words that logically interrelate the contents embedded in the stems.

Wilfrid Hodges:[21]

> The word 'logic' ... is ambiguous. In its first meaning, a logic is a collection of closely related artificial languages. There are certain languages called first-order languages, and together they form first-order logic. In the same spirit, there are several closely related languages called modal languages, and together they form modal logic. Likewise second-order logic, deontic logic and so forth. In its second but older meaning, logic is the study of the rules of sound argument.

[19] "Transcendental Analytic", Book II, *The Analytic of Principles*, A131.

[20] "Über den Zweck der Begriffsschrift", Jenaische Zeitschrift für Naturwissenschaft, 16, 1–10, 1882. English translation by T. W. Bynumin [Frege 1972, 90–91].

[21] "Classical Logic I: First-Order Logic", in Lou Goble, ed., *The Blackwell Guide to Philosophical Logic*, Wiley-Blackwell, Malden, MA, p. 9, 2001.

Lutz Straßburger:[22]

> The ... problem is to find the "least common denominator" for a definition of logic. The reason is that that there is no generally accepted consensus under logicians about the question what a logic actually is. Not only is the model theoretician's understanding of a logic ("a logic is something that has a syntax and a semantics") different from the proof theoretician's understanding ("a logic is a deductive system that has the cut elimination property"), we also see in other areas of research various different notions of "logic", which are all tailored for a particular application.

Straßburger went on to propose the following formal definition:

> **Definition.** A *logic* $\mathcal{L} = (\mathcal{A}_{\mathcal{L}}, \Rightarrow_{\mathcal{L}})$ is a set $\mathcal{A}_{\mathcal{L}}$ of formulæ, together with a binary relation $\Rightarrow_{\mathcal{L}} \subseteq \mathcal{A}_{\mathcal{L}} \times \mathcal{A}_{\mathcal{L}}$, called the *consequence relation*, that is reflexive and transitive.
>
> In other words, a logic is simply a preorder.

This attempt to define a "universal" logic was deemed by most us to be way too general.

4 Dov Gabbay

Dov developed his comprehensive notion of logic over many years. Here is a sampling of his thoughts.

4.1 1990

We do not mean "Logic" as it is now. We mean "Logic", as it will be, as a result of the interaction with computing. It covers the new stage of the evolution in logic. It is the new logic we are thinking of.[23]

4.2 1994

I am a logic, each one of us is a logic.[24]

[22] "What is a Logic, and What is a Proof?", in Jean-Yves Beziau, ed., *Logica Universalis: Towards a General Theory of Logic*, 2nd ed., Birkhäuser, Basel, pp. 135–173, 2007.

[23] Dov M. Gabbay, "Editorial", *Journal of Logic and Computation* 10, pp. 1–2, 1990.

[24] Dov M. Gabbay, "What is a Logical System?", in *Studies in Logic and Computation Series*, Oxford University Press, Oxford, pp. 179–216, 1994.

4.3 2005

We understand a logic to be a formalized idealization of a type of agent.[25]

We ... argue that a logic is not just a consequence relation (this notion is now marginal), but more like a perpetual multilevel reasoning system with mechanisms, actions, errors, and input and output points, more reminiscent of an operating system perpetually working and evolving.[26]

4.4 2015

A logical system is a pair

(set X of well-founded formulæ, a methodology to generate X).[27]

4.5 Formalization

A logic L consists of the following:

- An algebra A.
- A family of vertex-labelled directed graphs G with finite indegree.
- An interpretation I of vertex labels mapping labels in G to operators in A.
- The algebra and graphs must be such that for any graph in G there is always at least one solution to the set of equations induced by I, assigning consistent domain values to all the vertices.

5 Bob Constable

The following is a lightly edited abstract for Bob's talk for the Jamboree. "I", then, in this section, is Bob.

Logic is one of the oldest intellectual disciplines; it provides the *deductive method* for the sciences and all rational discourse. It is central to the philosophy of knowledge and draws on understanding human cognition. After more than 2360 years it still inspires innovations and probes more deeply into its core concerns, which

[25] Dov M. Gabbay, J. Woods, The Reach of Abduction: *A Practical Logic of Cognitive Systems*, vol. 2, North-Holland, Amsterdam, 2005.

[26] Abstract for talk, January 2005.

[27] Email of June 2015. Arnon pointed out that a "logical system" should not be conflated with "logic".

include providing and investigating modes of thought used to organize the evidential structure of knowledge. In the 21st century this involves learning to create knowledge from vast amounts of *digital evidence* as well as providing the designs for computer systems that assist us in reasoning about "digital knowledge." At the core of this activity is the desire to construct convincing evidence for declarative statements (assertions). Logic raises questions such as:

- What assertions are logically equivalent because they are justified by the same evidence?
- What conclusions indisputably follow from given assumptions (hypotheses)?
- What utterances are meaningful as knowledge claims?
- And so forth.

Natural language conveys evidence and knowledge, and the study of its grammars reveals universal means of structuring arguments. Aristotle's fundamental works on logic, the *Organon* (approximately 350 BCE), are grounded in grammar and mark an ancient origin of logic as a discipline. Aristotle also introduced the notion of a particular logic (of syllogisms) with specific *rules for demonstrating* that an assertion follows from premises. Logic was a key component of the study of *rhetoric.*

The logical mode of thought is essential to mathematics and indeed is required to convey nearly all pure mathematical thought; thus logic can be seen as *universal pure mathematics.* Euclid's *Elements* is the canonical example of logic in service of mathematics, and the canonical example of a logic. Amazingly, this logic (circa 306 BCE) remains to this day an active topic of logical investigation, as is clear from topics in this Advanced Seminar on Logic and Formal Methods. Logical systems are studied mathematically (at least since Leibnitz and especially with Boole), creating a subject called *mathematical logic*, now required of most computer science majors.

I consider the development of logic after the critical year 1907 when Russell wrote his *The Principles of Mathematics*,[28] introducing *type theory*, and Brouwer[29] proposed an intuitive computational basis for logic as an aspect of mathematical thought, an approach now called *intuitionism.* Russell worked to reduce mathematics to logic, and Brouwer tried to show that logic expressed general mathematical constructions. I argue that Russell and Brouwer were each fundamentally and deeply right, although mainstream mathematical logic does not yet reflect this.

[28] Bertrand Russell, *The Principles of Mathematics*, Cambridge University Press, Cambridge, 1908.

[29] See "On the foundations of mathematics", pp. 11–98, in A. Heyting, ed., *L. E. J. Brouwer. Collected Works*, vol. 1. North-Holland, Amsterdam, 1975.

(See my short paper, "The Triumph of Types: *Principia Mathematica's* Impact on Computer Science,"[30] presented at the 100th anniversary celebration of the publication of *Principia Mathematica* of Whitehead and Russell.[31] We see most clearly where both authors are right from certain results in logic that gave birth to computer science and from the subsequent deep interactions between logic and computer science—explaining the curricular requirement mentioned above.

Although many of Brouwer's radical insights are thoroughly studied and well understood in both logic and computer science, one of his "logical" insights has been soundly rejected, including by my colleagues and me. That insight is his belief that there is no fundamental separation between what are called *object logics* and *meta-logics*. Brouwer relied on this insight to prove his principle of *Bar Induction*.[32] I now think Brouwer was right on this point as well and hope to incorporate his approach more fully into the logic I have studied most deeply and helped create, Computational Type Theory (CTT).[33,34]

This theory already incorporates proof terms into its logic as well as the evaluation of terms. However, CTT is also supported by an implemented meta-language (ML) that automates reasoning using *tactics*. I also use this specific CTT logic to illustrate why there are comprehensive logics such as *Principia Mathematica* and its offspring, such as Higher-Order Logic (HOL),[35] CTT, and the *Calculus of Inductive Constructions* (CIC).[36] I explain why some of these including HOL, CIC, and CTT are *implemented as interactive theorem proving systems* (e.g. by HOL, Coq, and Nuprl, respectively) and how Brouwer's insights could make these implemented logics even more useful. (Brouwer's ideas are especially potent for Nuprl because it is also a *Logical Programming Environment* in the spirit of early Lisp machines, and

[30] `https://hdl.handle.net/1813/28696`.

[31] A. N. Whitehead and B. Russell, *Principia Mathematica*, volumes 1–3, 2nd ed., Cambridge University Press, Cambridge, 1925–1927.

[32] S. C. Kleene and R. E. Vesley, *Foundations of Intuitionistic Mathematics*, North-Holland, Amsterdam, 1965.

[33] Robert L. Constable, Stuart F. Allen, H. M. Bromley, W. R. Cleaveland, J. F. Cremer, R. W. Harper, Douglas J. Howe, T. B. Knoblock, N. P. Mendler, P. Panangaden, James T. Sasaki, and Scott F. Smith, *Implementing Mathematics with the Nuprl Proof Development System*, Prentice-Hall, NJ, 1986.

[34] Stuart Allen, Mark Bickford, Robert Constable, Richard Eaton, Christoph Kreitz, Lori Lorigo, and Evan Moran. "Innovations in computational type theory using Nuprl", *Journal of Applied Logic* 4(4), pp. 428–469, 2006.

[35] Michael Gordon and Tom Melham, *Introduction to HOL: A Theorem Proving Environment for Higher-Order Logic*, Cambridge University Press, Cambridge, 1993.

[36] Christine Paulin-Mohring, "Inductive definitions in the system Coq; rules and properties", in J. F. Groote M. Bezem, ed., *Typed Lambda Calculi and Applications*, Lecture Notes in Computer Science, Springer-Verlag, Berlin, 1993.

Lisp embodies Brouwer's notions quite well.)

Why does mainstream logic adhere to the object logic vs. meta-logic distinction? One of the key reason's is *Tarski's Theorem*[37]—that for all sufficiently expressive formal logics, certainly for all comprehensive logics, the notion of truth cannot be defined in the logic. Truth can be defined in a sufficiently strong meta-logic. For instance, the truth of first-order arithmetic cannot be defined in that theory, but it can be defined in higher-order logic or in set theory. Some logicians think that Tarski's Theorem is more consequential even than Gödel's Theorem. I point out that it is sometimes very informative and sometimes not. It is not so informative for set theory and type theory and thus might be overvalued as a design issue. (It is easy to model ZFC set theory in ZFC plus κ, one inaccessible cardinal. In contrast, modeling CTT with one universe using partial equivalence relations (PER models)[38] rather than CTT with two universes is very informative.) I also point out that CTT has progressively incorporated more and more meta-logic and speculate about the limit of this trend.

6 Arnon Avron

In turn, Arnon posited the following principles:

- A (classic) logic L consists of a language of formulas and a Tarskian (reflexive, transitive, monotonic) consequence relation $\vdash$ between (finite) sets of formulas and a single formula.
- It must satisfy an instantiation property: if $P \vdash q$, for formula q and set of formulas P, then $P\sigma \vdash q\sigma$ for any "substitution instance" $q\sigma$ of q and corresponding instances $P\sigma$ of the formulas in P.
- It is also desirable to formulate an appropriate notion of "equality" or "equivalence" of logics. One wouldn't want too many classical propositional logics.

Yoram Hirshfeld insisted that there must also be some informal notion of truth.

[37] Alfred Tarski, "The Concept of Truth in Formalized Languages", pp. 152–278, in *Logic, Semantics, Meta-Mathematics*, Clarendon Press, Oxford, 1956.

[38] Stuart F. Allen, "A Non-type-theoretic Definition of Martin-Löf's Types", in *Proceedings of the 2nd IEEE Symposium on Logic in Computer Science*, pp. 215–224, June 1987.

7 Nachum Dershowitz

I began with the standard idea that where there is logic, there is a universe of *(possible, partial) worlds*, typically presented formally as collections of formulæ. There is a binary *inference* relation $\vdash$ between worlds, so that $u \vdash v$ means that world v is a consequence of world u. And there is a truth function τ, mapping worlds to a set of truth values, one of whose values is designated *true*. These notions must satisfy the soundness requirement that $\tau(u) \rightarrow \tau(v)$ whenever $u \vdash v$ for worlds u, v.

Then I tried to generalize and refine those notions.

7.1 Syntax: Inference

Inference is central to logic, regardless of any particular means of composing formulæ and structuring arguments.

An *inference system* consists of

- a set $\mathbb{A}$ of objects, which we call *presentations*, as we have in mind sets (or bags, or sequences) of formulæ, or of sequents, or the like, plus
- a binary relation $\Vdash \subseteq \mathbb{A} \times \mathbb{A}$ over presentations, which should be thought of as a single *inference step*.

The following notions follow:

- A *proof* is an inference sequence $A_0 \Vdash A_1 \Vdash \cdots \Vdash A_n$, where $A_i \in \mathbb{A}$.
- Let $\mathbb{P} = \{A_0 \Vdash A_1 \Vdash \cdots \Vdash A_n : A_i \in \mathbb{A}\}$ be all proofs of the system in question.
- The reflexive-transitive closure $\vdash$ of $\Vdash$ is the (Tarskian) *consequence relation*.
- The *theory* $Th(A)$ of a presentation $A \in \mathbb{A}$ is the set of all its consequences:

$$Th(A) \quad = \quad \{B : A \vdash B\}$$

- The *support* $Sp(B)$ of a presentation $B \in \mathbb{A}$ is its inverse:

$$Sp(B) \quad = \quad \{A : B \in Th(A)\}$$

- The *theory* $Th(\mathcal{A})$ of a *set* $\mathcal{A} \subseteq \mathbb{A}$ of presentations is the union of the theories:

$$Th(\mathcal{A}) \quad = \quad \bigcup_{A \in \mathcal{A}} Th(A)$$

It follows that

$$\begin{aligned} A \Vdash B &\quad \text{implies} \quad Sp(A) \subseteq Sp(B) \\ A \Vdash B &\quad \text{implies} \quad Th(B) \subseteq Th(A) \end{aligned}$$

Theories are *deductively closed* in the sense that

$$Th(Th(\mathcal{A})) \quad = \quad Th(\mathcal{A})$$

for all $\mathcal{A} \subseteq \mathbb{A}$.

It may be useful to refer to the beginning and end.

- The first element of a proof is its *premise*, and the last, its *conclusion*:

$$\begin{aligned} Pre(A_0 \Vdash A_1 \Vdash \cdots \Vdash A_n) &\quad = \quad A_0 \\ Con(A_0 \Vdash A_1 \Vdash \cdots \Vdash A_n) &\quad = \quad A_n \end{aligned}$$

We have that

$$Pre(p) \quad \vdash \quad Con(p)$$

for all $p \in \mathbb{P}$.

At this juncture, what we have is not any different than a transition system, whose states are the presentations, whose transitions are inferences, and whose proofs are (finite) computations.

7.2 Semantics: Satisfaction

An inference system without an accompanying notion of truth is just a computational system.[39,40] In particular, logical inference should preserve truth.[41]

A *logic* consists of

- an inference system $\Vdash$,
- a set of *models* $\mathbb{U}$, which serve to provide semantics for presentations, and
- a *satisfaction relation*, $\vDash \subseteq \mathbb{U} \times \mathbb{A}$, connecting them.

[39] Pace Straßburger. See the end of Section 3.

[40] Turing in his famous 1936 paper noted that theorem proving is an instance of nondeterministic computation.

[41] The centrality of models for the notion of logic was stressed by Alex Rabinovich.

We require *(strong) soundness* of any logic:

$$A \Vdash B \quad \text{and} \quad M \vDash A \quad \text{imply} \quad M \vDash B$$

for all $A, B \in \mathbb{A}$ and $M \in \mathbb{U}$. If B is a consequence of A, then whenever and wherever A is true, so is B.

Satisfaction can be extended to sets of models $\mathcal{M} \subseteq \mathbb{U}$: $\mathcal{M} \vDash A$ iff $M \vDash A$ for all $M \in \mathcal{M}$.

The following notions follow:

- *Modeling* $\mathfrak{M}(\cdot) : \mathbb{A} \to \wp(\mathbb{U})$, gives the set (class) of models of a presentation:

$$\mathfrak{M}(A) \quad = \quad \{M : M \vDash A\}$$

- A presentation A is *valid* if $\mathfrak{M}(A) = \mathbb{U}$, that is, if it holds in all models.
- A presentation A is *unsatisfiable* if $\mathfrak{M}(A) = \varnothing$, that is, if it holds in none.
- *Entailment* $\vDash \subseteq \mathbb{A} \times \mathbb{A}$:

$$A \vDash B \quad \text{iff} \quad \mathfrak{M}(A) \subseteq \mathfrak{M}(B)$$

Put this way, soundness is expressed as follows:

$$A \Vdash B \quad \text{implies} \quad A \vDash B$$

(Strong) Completeness is the opposite:

$$A \vDash B \quad \text{implies} \quad A \vdash B$$

We have, for all $A \in \mathbb{A}$, $M \in \mathbb{U}$, $\mathcal{M} \subseteq \mathbb{U}$,

$$\begin{aligned} M \vDash A \quad &\text{iff} \quad M \in \mathfrak{M}(A) \\ \mathcal{M} \vDash A \quad &\text{iff} \quad \mathcal{M} \subseteq \mathfrak{M}(A) \end{aligned}$$

One might have a distinguished *null* presentation, which is modeled by every model. Let's denote it (there may be more than one) $\bigcirc$. There also might be an *inconsistent* presentation, $\Box$, which is modeled by nothing. In symbols:

$$\begin{aligned} \mathfrak{M}(\bigcirc) \quad &= \quad \mathbb{U} \\ \mathfrak{M}(\Box) \quad &= \quad \varnothing \end{aligned}$$

We can refer to the theory $Th(\bigcirc)$ of the null presentation(s) as the system's *tautologies*. We can refer to the support $Sp(\square)$ of any inconsistent presentation as the system's *contradictions*.[42]

As would be expected, every tautology is valid and every contradiction is unsatisfiable.

Omitting the null presentation ($A = \bigcirc$) from the above notations, we may define *weak soundness* as

$$\Vdash B \quad \text{implies} \quad \vDash B$$

—tautologies are valid. Analogously, *weak completeness* is

$$\vDash B \quad \text{implies} \quad \vdash B$$

One possible semantics is this: $\mathbb{U} = \wp(\mathbb{A})$ and $\mathfrak{M}(A) = Sp(A) \smallsetminus Sp(\square)$. It is sound, since

$$A \Vdash B \quad \text{and} \quad M \in Sp(A) \smallsetminus Sp(\square) \quad \text{imply} \quad M \in Sp(B) \smallsetminus Sp(\square)$$

7.3 Abstractness

An inference system is "abstract" if its inferences are generic.[43]

A function $\sigma : \mathbb{A} \to \mathbb{A}$ (written here in postfix) is a *substitution* if

$$\mathfrak{M}(A\sigma) \quad \subseteq \quad \mathfrak{M}(A)$$

for all $A \in \mathbb{A}$. Let Σ be some set of substitutions for a given inference system. Each $A\sigma$ is an *instance* of A.

A sound inference system is *abstract* (with respect to Σ) if

$$A \Vdash B \quad \text{implies} \quad A\sigma \vDash B\sigma$$

for all presentations $A, B \in \mathbb{A}$ and all substitutions $\sigma \in \Sigma$.

One trivial way to achieve abstractness would be for inference to be "schematic":

$$A \Vdash B \quad \text{implies} \quad A\sigma \vdash B\sigma$$

for all $A, B \in \mathbb{A}$ and $\sigma \in \Sigma$.

[42] Or *oxymorons*.

[43] The importance of abstractness for the notion of logic was stressed by Arnon. See Section 6.

A *sentence*[44] is a presentation A such that

$$A\sigma \quad = \quad A$$

for all substitutions $\sigma \in \Sigma$. Let $\widehat{\mathbb{A}}$ denote all sentences. Let

$$A^* \quad = \quad \{A\sigma : \sigma \in \Sigma, A\sigma \in \widehat{\mathbb{A}}\}$$

for $A \in \mathbb{A}$, be all its sentential instances.

Let the *value* $[\![A]\!]$ of sentence $A \in \widehat{\mathbb{A}}$ be the models of its theory

$$[\![A]\!] \quad = \quad \bigcup_{B \in Th(A)} \mathfrak{M}(B)$$

We can think of

$$\mathbb{V} \quad = \quad \{[\![A]\!] : A \in \widehat{\mathbb{A}}\}$$

as the logic's *truth values*, or—more likely—truth values can be (representations of) a particular partitioning of $\mathbb{V}$.

Verum and *falsum* would be the values in $\mathbb{V}$ that contain the models $\mathbb{U}$ and $\varnothing$, respectively.

More generally, the *meaning* $[\![A]\!] \subseteq \mathbb{V}$ of a presentation A could be its possible values under instantiation:

$$[\![A]\!] \quad = \quad \{[\![A']\!] : A' \in A^*\}$$

Finally, I suggested to call a logic *cogent* if it is both sound and abstract.

> The rest is interpretation; go study and finish.
>
> Hillel,
> *Babylonian Talmud*, Shabbat 31a.

[44] Maybe it ought to be called a *paragraph*, since we are talking about a whole collection of formulæ.

8 Conclusion

Dov Gabbay's intermittent participation in the seminar meetings of Tel Aviv University's Laboratory for Logic and Language was for me and my colleagues a constant source of illumination.

> We are like someone in a very dark night over whom lightning flashes time and time again. Among us there is one for whom the lightning flashes time and time again, so that he is always, as it were, in unceasing light.
>
> Maimonides

Received September 2025

Logic in Times of Big Data

Marcelo Finger*
Department of Computer Science, IME
University of São Paulo
Rua do Matão, 1010
05508 – São Paulo – SP – Brazil
mfinger@ime.usp.br

Abstract

Big data is all over us these days. In computational terms, logics are complex, at times undecidable, so how can it tackle the challenges of Big Data. In this shor piece in honor of Dov Gabbay, I discuss the weak points of employing logic, just to argue that logic should be used for what logic does best. Namely, proving properties of system, in particular the neural networks that are used to process big data application. We then discuss how Łukasiewicz Infinitely-valued Logic can be used for that purpose, and provide a brief presentation of the topic.

1 Introduction

I start this paper, as it is suitable in such occasions in which we pay tribute to a remarkable person, with a personal note. I have been a student of Dov since 1989, when he was my MSc dissertation supervisor at Imperial College, and later in my PhD in temporal logics. At the time we entertained the mathematical basis for the notion of *changing the past*, considering it actually a rather commonplace activity. Since then, I have traveled the world of logics, trailing Dov's footsteps. In this journey, I have dealt with combination of logics, substructural and subclassical logics, probabilistic logics and, more recently, fuzzy logics in the form of Łukasiewicz

This work was carried out at the Center for Artificial Intelligence (C4AI-USP), with support by the São Paulo Research Foundation (FAPESP) [grant #2019/07665-4] and by the IBM Corporation. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001.

[*] Partly supported by CNPq grant PQ 302963/2022-7, Fapesp projects 2023/00488-5 (SPIRA-BM) and 2024/19144-7.

Infinitely-valued Logic. As we will see, this latest step is related with the topic of addressing the emergence of Big Data.

When I finish my PhD, I immediately started as a researcher assistant in a joint project in computational linguistics and logic held by professors Ruth Kempson and Dov Gabbay. Being the middleman between a logician and a linguist had the unforeseen effect of putting me in the track of Big Data. Upon returning to Brazil, I was involved in the development of several annotated corpora of Brazilian and European Portuguese, and even then the techniques for dealing with that kind of data were not logic based. So I started to be involved (actually, enveloped would be a more descriptive term) in Machine Learning.

Several decades later, we were in the middle of the COVID-19 pandemic, locked down in our houses and learning to work and live through remote interaction. The pandemic was particularly brutal in Brazil, and I decided to join the research work on ways to deal with pulmonary diseases, recognizing by non intrusive, even remote methods. With Dr. Ester C. Sabino and Anna S. S. Levin, the SPIRA project was started with the aim of providing a system for early-detection of respiratory insufficiency via artificially intelligent audio analysis. It was the definitive embrace of Big Data for me.[1]

2 The Role of Logic in Big Data

It has been argued that "Logic is a modeling tool that is also explainable". But it is hard to agree with such statement without considering several points. First, if logic is explainable then we must admit that this does not mean that it is easy to understand by other people. That is, we may codify existing knowledge in logic, but such knowledge is not easily transfered to other people's minds.

To illustrate that, consider the topic on axiomatic set theory. There are dozens of published books on the subject, each one with hundreds of pages. But when one concentrate all axioms in a single place, it rarely takes more than a couple of pages, enumerating something between ten or fifteen axioms that rarely take more than a single line of symbols to express. On the one hand, this may reveal the synthetic expressivity of symbolic logic. But on the other hand, this also reveals the difficulty of translating this expressivity in simple and clear words. Admittedly, some axioms are more immediate to explain, but how many page does it take to explain the Axiom of Choice, in its many equivalent formulations?

Furthermore, it is the case that, when formulas are automatically generated,

[1] It became rapidly clear how hard it was to collect patient data; thus the realization of the hollowness of the term *big data* was also a consequence of this embrace.

they may be encoded in files with sizes of of several megabytes. And while neural networks are considered opaque boxes of encrypted knowledge, I see no reason to consider a megabyte sized file of "explainable" propositional logic formulas to be any less opaque.

And then one has to consider the computational complexity of Logics. It is a fact that most logical formalisms presented in the literature are either intractable of non-computable. The synthetic expressivity of symbolic logic is the counterpart of its computational complexity, in which to decide an inference normally takes an exponential time on the size of the input formulas, or even longer.

There has been a huge advance in modern technology to compute SAT problems with millions of variables in a few minutes, or even less, a feat that was considered unthinkable when I was a PhD student. But no one can claim to understand the "meaning" of the 20 Mega bytes input file for that task.

Leaving the propositional case, we must consider the fact that first order logic is *not* a good tool for modeling the real world. Logic was quite successful as a tool to model mathematical objects, but that does not transfer immediately to representations of the real world. To see the problems, consider the simple formula representing the statement (possibly false) that *all swans are white*:

$$\forall x(swan(x) \to white(x))$$

In the usual semantics of first order logic, *swan* and *white* are sets of domain elements. However, in the real world, those predicates are much fuzzier. For instance, the notion of a swan is quite hard to define, simply from its anatomical and physiological properties; if we move to the genetics of the definition, it is doubtful that a full characterization based on the presence or absence of specific bases exists for swans, and even if it exists, the cost of collecting its DNA and having analyzed is not negligible, so that the majority of the birds attributed to the class of swans have never been certified. The fuzziness of *white* is even more pronounced; the perception of colors varies from person to person and, more importantly, from culture to culture. So the meaning of *white* may vary from place to place, and from time to time. This fact is so pronounced that manufacturers of paint have developed what is usually called in the industry as a "reference white".

We must also mention that the notion of *all* is also fuzzy in natural language, and rarely corresponds to the set-theoretical interpretation of all elements in a set; natural language speakers normally disregard simple and rare exceptions, while in set theory a single counterexample is enough to falsify a ∀-statement.

Modern neuroscience theories tend to argue by means of experiments that there are no universal abstractions, and that all human categorizations are culture dependable, and thus time and location dependable as well. In particular, this includes all categories that are considered to be human sentiments. In fact, what I consider as happiness may be quite different from what my grandfather considered as happiness as would be expected, according to that view, from someone living in a different country in a different century.

My point here is not to trash logic completely, quite the opposite. By calling the attention to the activities that logic is not a suitable tool, the goal is to stress that Logic should be used for what logic is good at. For what logic is unsurpassable. Even when dealing with Big Data.

One such task for which logic has no substitute is formally proving properties of systems.

3 Successes of Computational Logic

Formal logic is associated with negative results, the most prominent of which is Goedel's Incompleteness Theorems. However, seeing those results as negative is an anachronistic bias. It stems from a period in which one could entertain that mathematics had no limits, in which the notions of Truth and Provability could be confused. Today, Goedel's results should be seen as a (positive) proof of existence of non-recursive functions, which served as basis for the theoretical and practical development of modern computer science.

Another fundamental result that was conceived in the heart of formal logic is the NP-completeness of SAT. Here the *negativity* of the result is even less clear, as this result put Complexity Theory in the heart of theoretical computer science. It is also a positive existential result, for it provided the first instance of a problem that was complete for a given complexity class. It is also the basis for the most important open problem in computer science, the $P \stackrel{?}{=} NP$ problem. But it also opened a view to our ignorance about important and consequential questions, that are not only computational in nature.

In the first years after the $P \stackrel{?}{=} NP$ was posed by Cook in 1971, it was believed that solving NP-complete problems was absolutely impractical, that any formula containing more than a handful of variables would take forever to be solved. And here lies a great success of computational logic.

First, it has been empirically shown that most SAT problems are quite easy in practice, which triggered the development of very clever and very efficient algorithms for SAT solving. Today we can solve SAT problems with millions of variables sometimes in less than a minute using a regular portable computing device. And this opened the possibility of using SAT problems as a form of "assembly language" into which several other NP-complete problems can be converted and solved.

However, the direct translation of problems into SAT may not be the most efficient form of solving NP problems whose translation into SAT may generate too large formulas even for modern SAT-solvers. Inspired by the success of SAT solving and the availability of free and open SAT solvers, the search for algorithms for several other logic-based NP-complete problems has since experienced a crescendo of interest. We have today quite efficient solvers for problems such as maximum satisfiability (MaxSAT), minimum satisfiability (MinSAT), probabilistic satisfiability (PSAT), among many others. And even NP-complete non-classical logics, such as Łukasiewicz infinitely-valued Logic ($Ł_\infty$), and more complex logics such as modal and temporal logics, are now being explored by computational tools.

Even though these achievements are substantial in the face of NP-hardness, they do not provide a clear path for logic to face Big Data.

4 Logic Meets Big Data

Jan Łukasiewicz was a polish logician that, in 1920, created a three-valued logic to deal with Aristotle's notion of *future contingents*, that is, temporal statements about the future which, at the present time, cannot have their truth value assertained. For instance, the sentence *tomorrow it will rain* is neither true nor false today; the truth value $\frac{1}{2}$ was proposed for that situation in the logic $Ł_3$. From three truth values, it was generalized to any number n of truth values, thus generating the family of logics $Ł_n$, parameterized by n. Later in 1930, Łukasiewicz with Tarski generalized $Ł_n$ to countably many truth values, giving origin to $Ł_\infty$, which is the logic we focus here.[2]

4.1 Łukasiewicz Infinitely-Valued Logic

The basic language $\mathcal{L}$ of Łukasiewicz Infinitely-valued Logic ($Ł_\infty$) comprehends the formulas built from a countable set of propositional variables $\mathbb{P}$ and the disjunction ($\oplus$) and negation ($\neg$) operators. Define a *valuation* as a function $v : \mathcal{L} \to [0, 1]$, such that, for $\varphi, \psi \in \mathcal{L}$:

$$v(\varphi \oplus \psi) = \min(1, v(\varphi) + v(\psi)); \tag{1}$$

$$v(\neg\varphi) = 1 - v(\varphi). \tag{2}$$

[2] The presentation in this section was jointly developed Sandro M. Preto; for proofs and furthehr detail, please refer to [13].

One may just give an assignment $v_{\mathbb{P}}$ which maps propositional variables to a value in the interval $[0,1]$ and extend this mapping to a valuation by obeying (1) and (2); such extension is uniquely determined by the assignment $v_{\mathbb{P}}$. We denote by $\mathbf{Val}$ the set of all valuations, that is the *semantics* of Ł$_\infty$, by $\mathrm{Var}(\Phi)$ the set of all propositional variables occurring in the formulas $\varphi \in \Phi$ and by $\mathbf{X}_n$ the set of propositional variables $\{X_1, \ldots, X_n\} \subset \mathbb{P}$. A formula φ is *satisfiable* if there exists $v \in \mathbf{Val}$ such that $v(\varphi) = 1$; otherwise it is *unsatisfiable*. A set of formulas Φ is satisfiable if there exists $v \in \mathbf{Val}$ such that $v(\varphi) = 1$, for all $\varphi \in \Phi$. We denote by $\mathbf{Val}_\Phi$ the set of all valuations $v \in \mathbf{Val}$ that satisfy a set of formulas Φ; we call such a restricted set of valuations a *semantics modulo satisfiability*. A set of formulas Φ *entails* a formula φ if $v(\varphi) = 1$, for all $v \in \mathbf{Val}_\Phi$; in this case we write $\Phi \models \varphi$, we say that formulas in Φ are the *premises* and formula φ is the *conclusion*. From disjunction and negation we derive the following operators:

Conjunction:	$\varphi \odot \psi =_{\text{def}} \neg(\neg\varphi \oplus \neg\psi)$	$v(\varphi \odot \psi) = \max(0, v(\varphi) + v(\psi) - 1)$
Implication:	$\varphi \to \psi =_{\text{def}} \neg\varphi \oplus \psi$	$v(\varphi \to \psi) = \min(1, 1 - v(\varphi) + v(\psi))$
Maximum:	$\varphi \vee \psi =_{\text{def}} \neg(\neg\varphi \oplus \psi) \oplus \psi$	$v(\varphi \vee \psi) = \max(v(\varphi), v(\psi))$
Minimum:	$\varphi \wedge \psi =_{\text{def}} \neg(\neg\varphi \vee \neg\psi)$	$v(\varphi \wedge \psi) = \min(v(\varphi), v(\psi))$
Bi-implication:	$\varphi \leftrightarrow \psi =_{\text{def}} (\varphi \to \psi) \wedge (\psi \to \varphi)$	$v(\varphi \leftrightarrow \psi) = 1 - \lvert v(\varphi) - v(\psi)\rvert$

Note that $v(\varphi \to \psi) = 1$ iff $v(\varphi) \leq v(\psi)$; similarly, $v(\varphi \leftrightarrow \psi) = 1$ iff $v(\varphi) = v(\psi)$. Let X be a propositional variable, then $v(X \odot \neg X) = 0$, for any $v \in \mathbf{Val}$; we define the constant $\mathbf{0}$ by $X \odot \neg X$. We also define $0\varphi =_{\text{def}} \mathbf{0}$, $1\varphi =_{\text{def}} \varphi$ and $n\varphi =_{\text{def}} \varphi \oplus \cdots \oplus \varphi$, n times, for $n \in \mathbb{N} \setminus \{0,1\}$; and $\bigoplus_{i \in \varnothing} \varphi_i =_{\text{def}} \mathbf{0}$.

A *rational McNaughton function* $f : [0,1]^n \to [0,1]$ is a function that satisfies the following conditions:

- f is continuous with respect to the usual topology of $[0,1]$ as an interval of the real number line;
- There are linear polynomials $p_1, \ldots, p_m$ over $[0,1]^n$ with rational coefficients such that, for each point $\mathbf{x} \in [0,1]^n$, there is an index $i \in \{1, \ldots, m\}$ with $f(\mathbf{x}) = p_i(\mathbf{x})$. Polynomials $p_1, \ldots, p_m$ are the *linear pieces* of f.

Writing a rational McNaughton function f we would have

$$f = \begin{cases} a_{1,1}x_1 + \cdots a_{1,n}x_n + b_1, & (x_1, \ldots, x_n) \in I_1 \\ \vdots & \vdots \\ a_{m,1}x_1 + \cdots a_{m,n}x_n + b_m, & (x_1, \ldots, x_n) \in I_m \end{cases} \tag{3}$$

where I_i's are n-dimensional polyhedra that partition the cube $[0,1]^n$ and $a_{i,j}, b_i \in \mathbb{Q}$.

A *McNaughton function* is a particular case of rational McNaughton function whose linear pieces coefficients are restricted to integer values. It is also represented by (3), where $a_{i,j}, b_i \in \mathbb{Z}$. In the traditional setting of function representation by logical formulas, the formulas of Ł$_\infty$ represent exactly the McNaughton functions [9, 10]. This is stated by following by the result.

Theorem 4.1 ([9])**.** *f is a McNaughton function iff there is a Ł$_\infty$-formula φ_f over variables $X_1, \ldots, X_n$ such that, for every valuation v,*

$$f(v(X_1), \ldots, v(X_n)) = v(\varphi_f)$$

4.2 Rational McNaughton Functions

As seen, Ł$_\infty$-formulas capture exactly McNaughton functions. However, if one wants to approximate any continuous function such as, for instance, a Neural Network, one should aim as representing rational McNaughton functions. In fact, that class of functions provide a Weierstrass-like approximation of continuous functions, as follows.

Theorem 4.2 ([2])**.** *Let $f : [0,1]^n \to [0,1]$ be a continuous function and let $\varepsilon > 0$. Then there is a rational McNaughton function $\tilde{f} : [0,1]^n \to [0,1]$ such that $|f(\mathbf{x}) - \tilde{f}(\mathbf{x})| < \varepsilon$, for all $\mathbf{x} \in [0,1]^n$.*

Rational McNaughton functions can be represented in many logics:

- Logic *Ł*Π½ by [3];
- Logic ∃*Ł* by [1];
- Rational Lukasiewicz Logic by [6].

However, those logics are quite complex, belonging to complexity classes above NP, while Ł$_\infty$is NP-complete. Furthermore, no known algorithm exists to compute rational McNaughton function representation in those logics.

We have developed a new representation of functions, called *representation modulo satisfiability* to enhance the representational capacity of Ł$_\infty$ in order to algorithmically encompass rational McNaughton functions [4, 11, 12].

Definition 4.3. *Let φ be a formula and let Φ be a set of formulas. We say that a set of propositional variables $\mathbf{X}_n$* determines φ modulo Φ-satisfiable *if:*

- *For any $\langle x_1, \ldots, x_n \rangle \in [0,1]^n$, there exists at least one valuation $v \in \mathbf{Val}_\Phi$, such that $v(X_j) = x_j$, for $j = 1, \ldots, n$;*

- *For any pair of valuations $v, v' \in \mathbf{Val}_\Phi$ such that $v(X_j) = v'(X_j)$, for $j = 1, \ldots, n$, we have that $v(\varphi) = v'(\varphi)$ — i.e. valuations in $\mathbf{Val}_\Phi$ are truth-functional on variables in $\mathbf{X}_n$.*

Definition 4.4. *Let $f : [0,1]^n \to [0,1]$ be a function and $\langle \varphi, \Phi \rangle$ be a pair where φ is a formula and Φ is a set of formulas. We say that φ* represents f modulo Φ-satisfiable *or that $\langle \varphi, \Phi \rangle$* represents f (in the system Ł$_\infty$-MODSAT) *if:*

- *$\mathbf{X}_n$ determines φ modulo Φ-satisfiable;*

- *$f(v(X_1), \ldots, v(X_n)) = v(\varphi)$, for $v \in \mathbf{Val}_\Phi$.*

For instance, a function that takes constant value $\frac{1}{b}$, with $b \in \mathbb{N}^*$, may be represented by the pair

$$\langle \varphi, \Phi \rangle = \Big\langle \; Z_{\frac{1}{b}}, \;\; \Big\{ Z_{\frac{1}{b}} \leftrightarrow \neg(b-1) Z_{\frac{1}{b}} \Big\} \; \Big\rangle, \tag{4}$$

where formula φ has only one propositional variable — denoted by $Z_{1/b}$ — and set Φ is a singleton comprehending formula $Z_{1/b} \leftrightarrow \neg(d-1) Z_{1/b}$ — denoted by $\varphi_{1/b}$. In fact, for any valuation $v \in \mathbf{Val}_{\varphi_{1/b}} \neq \varnothing$, we have that $v(Z_{1/b}) = \frac{1}{b}$. Moreover, function that takes constant value $\frac{a}{b}$, with $a \in \mathbb{N}$, may be represented by the pair $\langle a\varphi, \Phi \rangle$.

The expressions as $n\psi$, with $n \in \mathbb{N} \backslash \{0, 1\}$, have exponential length on the binary representation of n, but this can be turned into a polynomial representation by means of a representation modulo satisfiability; see [13]. In fact there is a polynomial time algorithm to bring any rational McNaughton function f (of special format) into a pair $\langle \varphi_f, \Phi_f \rangle$ that represents it.

Theorem 4.5 ([13])**.** *There is a polynomial algorithm that inputs a rational McNaughton function f encoded in a special format, called* closed regional format*, and outputs a Ł$_\infty$-MODSAT representation as a pair $\langle \varphi_f, \Phi_f \rangle$.*

4.3 Representation of Neural Networks

Neural networks are computational models that aim to generalize a pattern found in a dataset from which they are modeled; for our purposes, we identify neural networks with the computable functions they determine. For instance, given a dataset with observations on the weather conditions of a day together with the information of whether in the next day it rains or not, we are able to define — or *train*, in the jargon of the field —, based on these data, a neural network that, given the weather conditions of today, predicts whether it will rain tomorrow; such defining procedure is done by means of the so-called learning algorithms [7]. A neural network, depending on its architecture, may be seen either as a piecewise linear function or as a continuous function that can be approximated by one [8].

To exemplify the representing capacity given by Theorem 4.5, we focus on *rational McNaughton neural networks (RMcN^3)*, that are neural networks[3] which are exactly rational McNaughton functions, as defined in [14]. Let $f : [0,1]^n \to [0,1]$ be a RMcN^3 that, according to an input $\mathbf{x} \in [0,1]^n$, induces prediction **Yes**, if $f(\mathbf{x}) \geq 0.5$, and **No**, if $f(\mathbf{x}) < 0.5$ — for instance, for answering the question of whether it will rain tomorrow —; $f(\mathbf{x})$ may be interpreted as the probability of the answer being **Yes**. We call *binary classification RMcN^3* such a composition $\sigma \circ f$ of a RMcN^3 f with the *step function* $\sigma : [0,1] \to \{\mathbf{Yes}, \mathbf{No}\}$ given by

$$\sigma(x) = \begin{cases} \mathbf{Yes}, & \text{if } x \geq 0.5 \\ \mathbf{No}, & \text{if } x < 0.5 \end{cases}$$

4.4 Reachability and Robustness Analysis of Neural Networks

We next investigate ways to formally analyze reachability and robustness of rational McNaughton neural networks via properties of Ł$_\infty$.

The *reachability* of a given state in the context of RMcN^3 may be seen as the problem of determining if a neural network $f : [0,1]^n \to [0,1]$ reaches a specific probability $\pi = \frac{a}{b} \in [0,1] \cap \mathbb{Q}$, that is determining whether there is some $\mathbf{x} \in [0,1]^n$ for which $f(\mathbf{x}) = \pi$, or as the problem of determining if f reaches at least probability $\pi = \frac{a}{b} \in [0,1] \cap \mathbb{Q}$, that is determining whether there is some $\mathbf{x} \in [0,1]^n$ for which $f(\mathbf{x}) \geq \pi$. Such properties may be modeled in terms of the satisfiability of a Ł$_\infty$-formula. Let $\langle \varphi, \Phi \rangle$ be a representation of f in Ł$_\infty$-MODSAT; we claim that f reaches probability $\pi = \frac{a}{b}$ if, and only if, formula

$$\left(\bigwedge \Phi\right) \ \wedge \ \varphi_{\frac{1}{b}} \ \wedge \ aZ_{\frac{1}{b}} \leftrightarrow \varphi \tag{5}$$

is satisfiable and that f reaches at least probability $\pi = \frac{a}{b}$ if, and only if, formula

$$\left(\bigwedge \Phi\right) \ \wedge \ \varphi_{\frac{1}{b}} \ \wedge \ aZ_{\frac{1}{b}} \to \varphi \tag{6}$$

is satisfiable.

[3] In terms of neural network architecture, we are restricted here to *feed-foeward* networks.

Theorem 4.6 ([13])**.** *Let $f : [0,1]^n \to [0,1]$ be a RMcN^3 and $\langle \varphi, \Phi \rangle$ be its representation in Ł$_\infty$-MODSAT. Then, f reaches (at least) probability $\pi = \frac{a}{b} \in [0,1] \cap \mathbb{Q}$ if, and only if, formula* (5) *(formula* (6)*) is satisfiable.*

Corollary 4.7 ([13])**.** *The problem of deciding if a RMcN^3 given by a rational McNaughton function in regional format reaches (at least) probability $\pi = \frac{a}{b} \in [0,1] \cap \mathbb{Q}$ is in NP.*

Furthermore, it is possible to determine, according to a precision $\delta = 2^{-k}$, the maximum probability Π which a RMcN^3 reaches by means of a binary search through the possible values in the binary representation of Π.

Theorem 4.8. *Given a precision $\delta > 0$, maximum probability Π which a RMcN^3 reaches may be computed with $\lceil |\log \delta| \rceil + 1$ checks of satisfiability of formulas of the form* (6)*.*

The second propeerty we may analyze using this type of formalism is robustness. Neural networks in general are said to be *robust* if they maintain their predictions even when inputs suffer small perturbations. The literature provides examples of deep neural networks (that are not necessarily rational McNaughton functions) which perform well at the task of image recognition and, nonetheless, are susceptible to *adversarial examples*, that is, images with small perturbations which were originally classified properly and, still, had the prediction changed after the perturbation [15].

We model such concept of robustness by an instance of entailment in Ł$_\infty$. Let $\langle \varphi, \Phi \rangle$ be a representation of f in Ł$_\infty$-MODSAT for which $\mathbf{X}_n$ determines φ modulo Φ-satisfiable; for each propositional variable $X \in \mathrm{Var}(\varphi) \cup \mathrm{Var}(\Phi)$, we introduce a new variable X' and for each propositional variable $X_i \in \mathbf{X}_n$, we introduce a new variable P_i. Define the pair $\langle \varphi', \Phi' \rangle$, where all occurrences of variables X in φ or Φ are replaced by X' in order to obtain φ' and Φ'. We claim that robustness of $\sigma \circ f$ with respect to perturbation limit $\varepsilon = \frac{\alpha}{\beta}$ and to probability $\pi = \frac{a}{b}$ is equivalent to the validity of the entailment

$$\begin{aligned} \Phi,\ \Phi',\ \varphi_{\frac{1}{\beta}},\ \varphi_{\frac{1}{b}},\ \varphi_{\frac{1}{2}}, & \\ P_1 \to \alpha Z_{\frac{1}{\beta}},\ \ldots,\ P_n \to \alpha Z_{\frac{1}{\beta}}, & \\ (X_1' \leftrightarrow X_1 \oplus P_1) \vee (X_1' \leftrightarrow \neg(X_1 \to P_1)), & \\ \ldots, & \\ (X_n' \leftrightarrow X_n \oplus P_n) \vee (X_n' \leftrightarrow \neg(X_n \to P_n)), & \\ aZ_{\frac{1}{b}} \to \varphi & \models Z_{\frac{1}{2}} \to \varphi'. \end{aligned} \tag{7}$$

Theorem 4.9. *Let $f : [0,1]^n \to [0,1]$ be a RMcN^3 and $\langle \varphi, \Phi \rangle$ be its representation in Ł$_\infty$-MODSAT from which $\langle \varphi', \Phi' \rangle$ is defined as in previous discussion. Then, binary classification RMcN^3 $\sigma \circ f$ is robust with respect to $\varepsilon = \frac{\alpha}{\beta} \in \mathbb{Q}$ and $\pi = \frac{a}{b} \in [0,1] \cap \mathbb{Q}$ if, and only if,* (7) *holds.*

Corollary 4.10. *The problem of deciding if a binary classification RMcN^3 given by a rational McNaughton function in regional format is robust with respect to $\varepsilon = \frac{\alpha}{\beta} \in \mathbb{Q}$ and $\pi = \frac{a}{b} \in [0,1] \cap \mathbb{Q}$ is in coNP.*

Again, we can determine, according to a precision $\delta = 2^{-k}$, the maximum perturbation limit E for which a binary classification RMcN^3 remains robust with respect to a fixed probability π by means of a binary search through the possible values in the binary representation of E.

However, as it should be clear from the previous results, the verification of reachability and robustness is, respectively, NP-hard and coNP-hard. This intractability is a consequence of the complexity of deciding properties of Ł$_\infty$, namely satisfiability and entailment.

5 Conclusions and Future Works

Logics should be used for things in which logics excel. Proving properties of systems is one such area, and the modeling of neural networks is one such potential area. Since classical logics cannot model continuous functions in an efficient way, we proposed the use of rational McNaughton functions as a way of approximating any continuous function; in particular, they can also approximate neural networks. That class of functions may be represented in non-classical logic Ł$_\infty$, where the proof of properties such as reachability and robustness lies “only” on the NP/coNP complexity classes.

Future work should investigate ways of making such approach more efficient. Approximations of Ł$_\infty$-inference have been proposed in [5], and better Ł$_\infty$-solvers are currently under investigation. Other classes of neural networks also need to be explored going beyond simple feed-forward networks. Looking for better algorithms for representing neural networks directly, without extracting first its approximated rational McNaughton function, is also a possible area of research. Finally, we should also investigate a larger class of properties beyond reachability that can be tackled by this proposed neural network Ł$_\infty$-interpretability approach.

References

[1] Stefano Aguzzoli and Daniele Mundici. *Weierstrass Approximation Theorem and Łukasiewicz Formulas with one Quantified Variable*, pages 315–335. Physica-Verlag HD, Heidelberg, 2003.

[2] P Amato and M Porto. An algorithm for the automatic generation of a logical formula representing a control law. *Neural Network World*, 10(5):777–786, 2000.

[3] Francesc Esteva, Lluís Godo, and Franco Montagna. The $ł\Pi$ and $ł\Pi\frac{1}{2}$ logics: two complete fuzzy systems joining Łukasiewicz and product logics. *Archive for Mathematical Logic*, 40(1):39–67, Jan 2001.

[4] Marcelo Finger and Sandro Preto. Probably partially true: Satisfiability for Łukasiewicz infinitely-valued probabilistic logic and related topics. *Journal of Automated Reasoning*, 64(7):1269–1286, 2020.

[5] Marcelo Finger and Sandro Preto. Polyhedral semantics and the tractable approximation of Łukasiewicz infinitely-valued logic. *Journal of Logic and Computation*, 35(5):exad059, 10 2023.

[6] B Gerla. Rational Łukasiewicz logic and DMV-algebras. *Neural Network World*, 11(6):579–594, 2001.

[7] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[8] Moshe Leshno, Vladimir Ya. Lin, Allan Pinkus, and Shimon Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, 6(6):861–867, 1993.

[9] R. McNaughton. A theorem about infinite-valued sentential logic. *Journal of Symbolic Logic*, 16:1–13, 1951.

[10] Daniele Mundici. A constructive proof of McNaughton's theorem in infinite-valued logic. *The Journal of Symbolic Logic*, 59(2):596–602, 1994.

[11] Sandro Preto and Marcelo Finger. An efficient algorithm for representing piecewise linear functions into logic. *Electronic Notes in Theoretical Computer Science*, 351:167–186, 2020. Proceedings of LSFA 2020, the 15th International Workshop on Logical and Semantic Frameworks, with Applications (LSFA 2020).

[12] Sandro Preto and Marcelo Finger. Efficient representation of piecewise linear functions into łukasiewicz logic modulo satisfiability. *Mathematical Structures in Computer Science*, 32(9):1119–1144, 2022.

[13] Sandro Preto and Marcelo Finger. Proving properties of binary classification neural networks via Łukasiewicz logic. *Logic Journal of the IGPL*, 31(5):805–821, 06 2022.

[14] Sandro Preto and Marcelo Finger. Regional, lattice and logical representations of neural networks. In Cynthia Kop and Helida Salles Santos, editors, Proceedings of the 19th International Workshop on *Logical and Semantic Frameworks, with Applications,* Goiânia, Brazil, 18th-20th September 2024, volume 421 of *Electronic Proceedings in Theoretical Computer Science*, pages 64–79. Open Publishing Association, 2025.

[15] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

Received September 2025

Is Mathematics Empirical?

Michael Gabbay
Dept. of Philosophy, Cambridge
mg639@cam.ac.uk

1

There is a reasonably well known, but underdiscussed paradox that is sometimes referred to as the *diagonal paradox*. The paradox, presented graphically in Figure 1, is that an apparently reasonable argument with limits yields the conclusion that the length of the diagonal of a unit square is 2 rather than $\sqrt{2}$. We can present the argument with some degree of mathematical rigour. Consider, for each n, a staircase function in the range $0 \leq x \leq 1$ given by the equation:

$$(ny - \lceil nx \rceil)(nx - \lfloor ny \rfloor) = 0. \tag{1}$$

All the points (x, y) where the the ceiling (rounded up) of $n \times y$ is equal to the floor (rounded down) of $n \times x$.

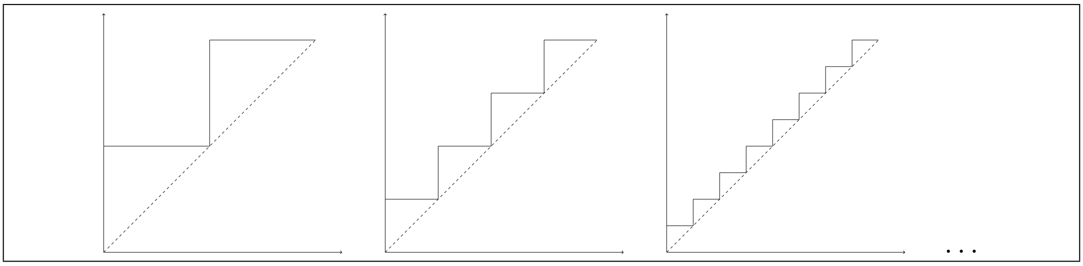


Figure 1

As n increases, the number of 'stairs' increases and their height and width decreases. Although, given n and x there may be many y that satisfy (1), the range of y tends to x following a standard definition of a limit:

$$\forall x \forall \epsilon \exists n \forall m_{\geq n} \forall y \big((my - \lceil mx \rceil)(mx - \lfloor my \rfloor) = 0 \rightarrow |x - y| < \epsilon\big)$$

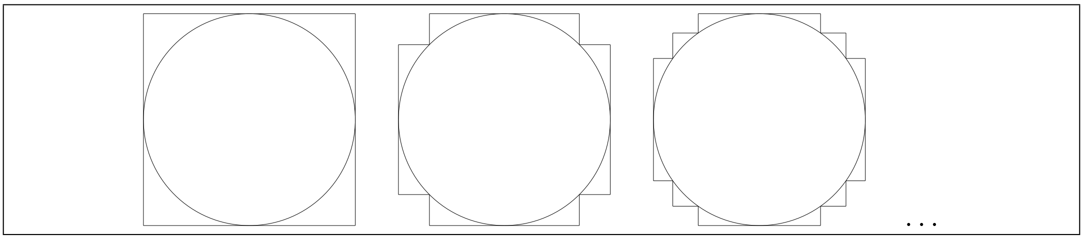


Figure 2

That is, at any point on the line $y = x$ and any arbitrary small amount ϵ there is an n such that the staircase is within ϵ of that point. This seems to imply that for a given x, as n tends to infinity, the solutions for (1) tend to y, and thus the limit of the solutions to 1 as n tends to infinity is given by the equation:

$$y = x \tag{2}$$

But for each n, the sum of the vertical and horizontal components of the line is always 2: in every case the line goes up 1 and across 1. Thus the limit of the lengths of the solutions to (1) as n increases is also 2, and the argument concludes that the length of the solution to (2), i.e. the diagonal of the unit square, is 2 rather than $\sqrt{2}$. The point thus far is that the key concepts in the presentation of this paradox can be made perfectly mathematically rigorous. Let us call this (both the argument and its conclusion) the *staircase argument.*

A similar argument can be given that the circumference of a unit circle is 8 rather than 2π, see Figure 2. Analytically, the equation for this family of 'curves', for $-1 \leq x \leq 1$, is:

$$\Big(ny^2 - \lceil n(1-x^2)\rceil\Big)\Big(n(1-x^2) - \lfloor ny^2 \rfloor\Big) = 0. \tag{3}$$

Again, as n increases, the staircase becomes arbitrarily close to the line $y = \sqrt{1-x^2}$: a unit semicircle. But now, similarly to the case of the diagonal of a unit square, we have that the length of the semicircle (from -1 to 1) is 4. Therefore a whole unit circle has length 8.

2

An accurate, but unsatisfactory, solution to the paradox is to point out that it involves a fallacy: it assumes that a property of each element of a limiting sequence is retained by the limit. This is easily shown not to be the case, for example each of the initial segments of natural numbers is a finite sequence, but their limit is infinite. By itself this does not fully solve the problem as apparently identical reasoning is used in another, accepted and utterly fundamental piece of mathematical reasoning.

Consider the area under the curve $y = \sqrt{1-x^2}$. We can show that it is $\pi/2$ by evaluating the simple integral:

$$\int_{-1}^{1} \sqrt{1-x^2}\, dx \tag{4}$$

which gives us a value of $^{\pi}/_{2}$.

But consider the principles of the calculus behind this. Integration is founded on the idea that the area of the curve can be divided into ever smaller rectangles, Figure 3, as the rectangles get thinner and thinner the sum of their areas approaches the desired area. This assumes that the upper perimeter of these rectangles, i.e. the 'staircase' formed by the tops of the rectangles, approaches the original curve. But now the key question is this: *if we know that taking the limit of the length of this staircase leads to false conclusions, how are we sure that the limit of its enclosed area doesn't either*?

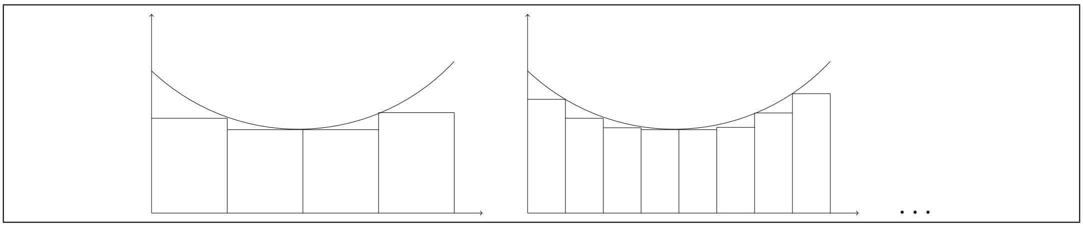


Figure 3

In fact, the situation is worse, for more properly, integration is founded on the idea that the rectangles used to approximate the area of the curve may be either just too small (beneath the curve) or just to large (above the curve). A function is integrable if the limit of the area approximations from below (using rectangles that underestimate the curve) is the same as the limit from above (using rectangles that overestimate it). This more subtle definition covers cases of non-continuous functions and effectively acknowledges that sometimes, approximating as a limit of staircase of rectangles doesn't produce the correct results.

The point is not to cast doubt on whether the arc-length of a unit circle is 2π, or whether the length of the diagonal of a square is $\sqrt{2}$. The point is to identify whether there are *mathematical* grounds for accepting reasoning based on limits of staircase approximations when considering the area under a curve, at the same time as rejecting such reasoning when considering the length of a curve.

Notice that statement the area under the unit semicircle, i.e. that the integral 4 evaluates to $\pi/2$, is unitless, and purely mathematical. The integral is not presented in terms of any unit, such as m^2, and evaluates to the unitless $^{\pi}/_{2}$.

The original argument then returns: on what basis do we know that the property of the elements of a particular sequence doesn't carry over to the limit? After all, a certain construction apparently succeeds for area but fails for length, on what basis do we determine this. The argument concludes that the basis can only be an empirical one: from our knowledge of real-world lines and areas we know that the fundamental principles of calculus yield correct results, but attempting to apply them to length as in 1 does not; we know what the lengths and areas of/under curves should be, and calculus stands on that basis. Therefore a significant amount of core mathematics is fundamentally empirical. Let us refer to this as the *main argument* of the paper.

3

A natural meta-mathematical response to the main argument is simply to say that the staircase argument for length is inconsistent whereas the apparently similar argument for integrals is not. Moreover, we can separate the two types of reasoning by their subject matter: one is about length the other is about area, two related but separate properties of 2-dimensional space. The fact that one method of taking limits in the case of length yields inconsistency, it may be argued, is not in itself proof that similar methods are unsuitable for area. It may be true that we accept integral calculus because it seems consistent does not make it empirical any more than using ZF set theory is empirical because, in our experience, it appears consistent and useful. The argument is then that the *theory of area* we get from the staircase argument is consistent, but the *theory of length* is inconsistent. We can therefore reject the application of the staircase argument to length on the basis of meta-mathematical considerations of consistency.

This flaw in this response is that it is not in fact inconsistent to accept the conclusion of the staircase argument: that the length of the diagonal of a square is, in fact, 2. A model that supports this conclusion can be seen in the mathematics of vectors. If ‘half-line’ is interpreted as a sequence of vectors from a starting point (i.e. a sequence of steps), and ‘length’ is interpreted as the sum of the absolute values of the horizontal and vertical components, then we a perfectly consistent notion of ‘line’ and ‘length’. In this case ‘length’ is a somewhat useless measure of a line, but nevertheless a consistent one. Neither is such a mathematics of lines completely irrelevant: it is the mathematics of pixelated drawings of lines.[1] The problem with the conclusion of the staircase argument is not that it is inconsistent as a mathematics of length, *it is simply wrong*, and we know that it is wrong by examining lines in reality: an empirical process. Moreover, even though we know it is wrong for length, we still employ it for area, not simply because of consistency (remember consistency is not enough for us to accept the staircase argument for length) but because the resulting mathematics is proved *right*: again by our empirical knowledge of area.

4

The fact that the conclusion of the staircase argument may be part of a consistent theory of two dimensional space suggests another method of blocking responding to it. Perhaps it rests on a confusion between correct mathematics and correctly *applied* mathematics. The thought is that the staircase argument merely introduces an alternative theory of length, and the empirical aspect does not justify it, merely its application. It could then be argued that the familiar mathematics of length and area is not *justified* by experience, rather it is its *application* to reality that is justified by experience. There is nothing controversial about this, the applicability of a theory may stand on the basis of experience, but that does not make the theory itself empirical.

But this response depends on an incorrect analysis of the subject matter of calculus. Take for example the function $f(x) = x^2$: that function has arc-lengths between any two values of x; those arc-lengths are unitless, simply numbers, they are not properties of actual curves in space. We can use f to model some aspects of reality, but f, and its arc-lengths, are not physical properties. Now the arc-length of f between 0 and 1 is approximately 1.48, but according to the staircase argument it is 2, that is simply wrong. The point is made clearer by noting that, in the case of each iteration of the staircase, the mathematical notion of arc-length used is the normal one, and is used correctly. The *length* of each staircase is indeed the sum of the *lengths* vertical and horizontal components, and here 'length' has its usual

[1] A more useful measure of length would be the roots of the sums of the squares of horizontal and vertical components of each vector (i.e. step) constituting a 'line'. Or, more in the spirit of the model, a pair (ϵ, r) where ϵ, the 'resolution', is a number smaller than the absolute value of any horizontal or vertical vector, and r is the sum of the squares of horizontal and vertical components divided by ϵ.

At the limit of the staircase construction, the first measure above, involving roots, corresponds to the correct length of the line. But how do we identify it as correct? By showing it corresponds to the results of our empirical methods for measuring length.

abstract mathematical meaning. The word ‘length’ has the same meaning when the argument concludes that the *length* of the limit of the staircases is 2. This is not a matter of misapplying a different theory of length to reality. The conclusion of the staircase argument relates to the standard mathematical abstract of length, and this conclusion is rejected, but a construction just like it is accepted for area, the mathematical concept for area.

To reiterate. If we say that the arc length of a function $g(x)$ between arguments a and b is, say, 2, then the truth conditions of that are that the ratio of that length to the length of a constant function (e.g. x=0) between 0 and 1, is $2:1$. The staircase argument will invariably conclude that the ratio is $b-a+g(b)-g(a):1$ which we know to be wrong independently of whether these ratios are applied to any physical measure.

5

A more promising line of argument notes that the foundations of calculus need not rest on rectangles, but trapezia. Rectangles, we may say, are a convenient simplification while the area under the curve should properly be characterised as the limit of increasingly thin trapezia. In this case not only can the fundamental theorems still be shown, but the straight-line approximation to the arc-length derived from the tops of the trapezia (what might be called a *linear spline interpolation*) tends to the correct limit. The question of why the staircase (rectangle) method works for area but not arc-length can now be sidelined as a mathematical amusement: it’s not the right way of thinking about it anyway.

Indeed, the ancient argument that the area of a circle is given by the formula πr^2 famously relies on rearranging ever numerous sectors of a circle into a shape that tends to a rectangle. As the sectors get smaller, the sides become flatter, with their limit being a rectangle with sides r and πr.

But for this observation to replace the staircase argument it must be shown not merely as an apparently more sensible alternative or else the main point stands. Why should we regard using trapezia as correct over rectangles beyond the empirical observation that it makes everything work? The answer must be that the construction of the staircase argument does actually yield the curve it is supposed to, whereas approximating the curve by the tops of trapezia, call it the *gradient construction*, does.

But the limit of the staircase construction is exactly the same set of points as the limit of the gradient construction, so something else must make the difference. As the name of the construction suggests, the difference is that the staircase construction is increasingly jagged, whereas the staircase construction is increasingly smooth. A line is smooth, so the limit of the gradient construction is the line whereas the limit of the staircase construction is not. But this is insufficient for we have already noted that we cannot automatically conclude that the limit of a sequence has a property (in the case being jagged) just because every initial segment of it is: perhaps a limit of jagged lines can be smooth. And what does it mean to be "smooth" anyway? There is an obvious, pointwise, answer to this: a line should be pointwise differentiable; the limit of the staircase construction is an infinite set of points with no gradients; the limit of the gradient construction is the same point, but each one with a fixed gradient. A point may become fixed in the staircase construction, but its gradient will oscillate between 0 and undefined, in the gradient construction points get fixed easily and their gradients tend to a limit.

But there is a serious consequence to this reasoning: we must argue that a line is not merely a collection of points, but one that is differentiable. Actually, this is not enough, we must say that a line must differentiable to the *correct* derivative: examples exist of apparent lines which are differentiable, but to the wrong derivative. An infamous example is the Cantor Function,[2] which is a kind of staircase function but only ever differentiates to zero. The stronger mathematical condition is that a line (or curve) is not merely a collection of points, but is an *absolutely continuous* function. The condition of 'absolute' continuity is effectively the condition that the methods behind the Fundamental Theorem of Calculus apply.

Applying this back to the original question – which asked why the fundamental methods of ever better staircase approximations to a curve works for integration but not for arc-length – the response is then essentially that a line is a function to which the fundamental theorem of calculus can be applied: it can be differentiated; and it equals the integral of its derivative (plus a constant).

But this leaves us with a highly formalist notion of the fundamental concepts of mathematics. The term 'line' is only properly understood within the context of the mathematical theory of calculus. But the theory of calculus is intelligible only with a prior understanding of the notion of a line – for example, the justification of the method of differentiation requires the notion of *lines* which ever better approximate the gradient of a tangent *line.* The concept of a line and the associated mathematics must therefore come together, as a unified formal and conceptual theory. But on what basis does this theory stand or fall? On the basis that when we apply it to reality, it works. And mathematics (or at least calculus) is again fundamentally empirical.

[2] O. Dovgoshey et. al., "The Cantor function", *Expositiones Mathematicae*, Volume 24, Issue 1.

6

The issue, unsurprisingly, relates to the behaviour of limits of infinite sequences. The fact remains that an infinite series is only ever experienced by us theoretically. We either find a real entity, or state, that we can describe as the result of an infinite process (like is done in Zeno's paradoxes of motion); or we may identify a finite series and consider its hypothetical limit to infinity. The mathematical techniques that have been developed over centuries to carry out this theory are justified in part, by their application to reality. This empirical justification, I have argue here, is not merely the verification that an independently grounded mathematical theory is applicable to reality (arc-length and area); but is in fact part of the validation of the mathematics itself. Mathematics, or at least Calculus, therefore has an inextricable empirical core.

Received September 2025

Arithmetising logic: polynomial semantics of FOL

Murdoch J. Gabbay[*]
Heriot-Watt University, Scotland, UK
www.gabbay.org.uk

Abstract

We propose a compositional shallow translation from an impredicative first-order logic with equality, into polynomials; that is, we give a direct, compositional, arithmetic notion of first-order logic validity.

Arithmetisation means translating validity of some assertion ϕ of a model M, into some property R_M of a polynomial $P(\phi, M)$, such that (to a high degree of certainty) ϕ is valid in M if and only if R_M holds of $P(\phi, M)$. Arithmetisation is widely used in cryptography, because many cryptographic protocols work by exploiting properties of finite fields to prove assertions like "R holds of a polynomial P" — a typical property R is 'has roots in $1, \ldots, k_M$', where k_M is determined by the cardinality of M.

Meanwhile, first-order logic is a simple but expressive language for specifying and reasoning about computation. In particular, it is easily powerful enough to express inductive definitions, Turing-complete computation, and correctness properties thereof. We can think of first-order logic as a high-level, virtual-machine-independent model of reasoning and computation.

Techniques already exist to arithmetise computation by coding it in an abstract machine whose computation has been arithmetised by hand; i.e. by a *deep embedding* in some preconstructed monolithic cryptographic artefact. What distinguishes our translation is its shallowness, compositionality, and uniform treatment of logical connectives: it translates logical connectives directly to operations on polynomials, *without* a deep encoding in an abstract machine.

The translation is deceptively straightforward: truth maps to 0, false maps to any strictly positive number, conjunction and universal quantification map to $+$, and disjunction and existential quantification map to $*$. Yet, we shall see that the outcome is surprisingly powerful and expressive.

Thanks to three anonymous referees and to several colleagues for their kind attention to this work.

[*] I dedicate this paper to my father Dov Gabbay on the occasion of his 80th birthday. You have found logic in everything and inspired many others to do the same. Thank you and Happy Birthday.

Vol. 12 No. 6 2025
Journal of Applied Logics — IfCoLog Journal of Logics and their Applications

1 Introduction

We take a step towards unifying mathematically structured programming (by which we mean *'computation that is specified using logic'*) with cryptographically verifiable computation (meaning *'executions that can be cryptographically verified'*). We do this by giving

1. a *compositional, shallow translation* of first-order logic to polynomials (Figure 2),
2. a sound and complete translation of logical validity (i.e. predicates being true) to corresponding polynomial equalities (Theorems 2.4.4 and 4.2.7), and
3. an outline of what it means to give *succinct proofs* (in the cryptographic sense) of the resulting polynomial equalities.

Schemes with this effect exist for von Neumann style abstract machines (a brief but clear survey is in [17]), for logic circuits [9, 18], and even for LISP [2] (reference list not exhaustive).[1] Yet first-order logic has as good a claim as any to be a universal language of mathematics — and in particular it is a commonly and successfully used as a high-level language for specifying computation. To our knowledge, a compositional semantics for FOL has not yet been given in polynomials.

We summarise what this means and why it matters. This summary is aimed not only at cryptographers but also at readers from a logic and computation background, to provide some context:

Boolean circuits contrasted with arithmetic circuits

Logic uses Boolean circuits: combinations of *and*, *or*, and *not* operating on variables that each hold one bit of information: *true* or *false*. Because *and* and *or* distribute over one another, and because *not* satisfies de Morgan dualities, Boolean circuits have a normal form called a *disjunctive normal form*.

Cryptography uses arithmetic circuits:[2] combinations of $+$, $*$, and $x \in \mathbb{Z}$ operating on variables that hold numbers (either from a finite field or possibly just integers). Because $*$ distributes over $+$ (i.e. $x * (y + z) = x * y + x * z$) arithmetic circuits also have a normal form: *polynomials*, as familiar e.g. from maths in school.

Polynomials are very expressive: they can represent functions (by interpolation [25, Chapter 9][3]), vectors of integers (by evaluating at $\{1, \ldots, n\}$ where n is the

[1] ...and it would be impossible to even try, because progress in verifiable computation is currently so rapid.

[2] Not an exclusive claim: cryptography also uses Boolean circuits. But what interests us here is the use of arithmetic ones.

[3] A compact but clear overview is some online course notes [8, Subsection 5.1] (direct link).

length of the vector), multisets (by identification with their multiset of roots), and they can be composed. Polynomials over integers can be evaluated to numbers, and this has unexpected implications including the Schwartz-Zippel lemma [28, Lemma 3.3, Subsection 3.4] (see also a clear historical discussion [19]).

Propositions in disjunctive normal form are not quite as expressive as polynomials, but they have excellent properties: distributivity of *and* over *or* and of *or* over *and*; de Morgan duality between *true* and *false*; and bitwise number representation which are the basis of computer chips;[4] and an extremely well-developed theory which includes logic (including first-order, higher-order, and matching logics), declarative programming languages, type-checkers, model-checkers, SAT solvers, formal verification tools, and moving on from there to dependent type theories, modal logics (like TLA+), model theory, and more.

Mathematically structured programming contrasted with cryptography

Mathematically structured programming (**MSP**) is an approach to programming, based on logic, that tries to minimise the distance between a program and its specification, such that (ideally) the specification *is* the program and is directly executable, without requiring the programmer to code up a realisation in a lower-level programming language.[5]

Benefits include (arguably) easier verification, and greater productivity, because the code is kept close to its logical intention. Disadvantages include lower efficiency: it can sometimes be harder to write high-level code that runs quickly on *native silicon*, by which I mean that the high-level code compiles to efficient low-level instruction sequences suitable for execution on physical (usually silicon-based) computer processing chips. In polynomials it may be that a higher-level MSP-based approach is actually more efficient.[6]

Cryptography has historically been about secure and verifiable communication (think: encryption and error-correcting codes); but with the rise of distributed sys-

[4] There are two distinct notions of ‘number’ in play here: *integers*, which relate to arithmetic circuits; and *bitwise integers*, which are a distinct thing and relate to Boolean circuits.

[5] Of course, all programming languages do this to some extent (an XKCD comic strip (permalink) makes a pithy, and entertaining, comment on this), but MSP takes this abstraction to its logical (pun intended) extreme. Aside from the exceptionally high levels of abstraction, MSP is also characterised by something else: using specification languages based on mathematical logic and its semantics.

[6] It is debatable whether the standard instruction-set based architectures that have evolved for efficiency on silicon are also an optimal compilation target in a world made out of polynomials. A remarkable article in The Register [24] provides further context and is well worth reading (see the discussion of Dylan). See also an experience report [12, Section 6] on a (successful) use of functional programming in financial markets.

tems, cloud computing, and layer 2 blockchains, the focus of interest in cryptographic techniques has shifted and broadened to include *secure and verifiable computation*. Thus we see modern interest in applications such as succinct proofs of executions, fully homomorphic encryption, multiparty computation, zero knowledge computation, garbled circuits, and more.

The Unique Selling Point

First-order logic is a simple yet powerful logic which has as good a claim as any logic to be a universal (or at least a canonical) language of mathematics. Thanks to this simplicity and power, it is heavily used in mathematically structured programming. Thus, giving a compositional shallow mapping of first-order logic predicates to polynomials has as good a claim as any to being a canonical simple and high-level, and yet potentially very practical, treatment of verifiable computation.

Note that our mapping encodes *semantics and validity*, not *syntax and derivability*. In particular, it would be mistaken to assume that this paper works by encoding the syntax of FOL and the structure of well-formed derivation trees into arithmetic circuits. This paper takes a semantic approach that is arguably simpler; see Remark 6.3.3 for more discussion, with an example.

We will see how we can write specifications in first-order logic, compile them to polynomials, and evaluate them there. The translation has low overhead, and it bridges the gap between the abstractness of logic and having direct access to the underlying polynomials. Because our translation is very compositional, we will be able to compose simple specifications to make more complex ones. We hope this will inform both concrete applications as well as new research in mathematical logic.

Who should read this paper

I hope that practitioners will be interested in the ideas in this paper and think about basing practical systems on them. For such readers, the use of logic in MSP style may be new, but we will explain it clearly and with examples. I would argue that this application of logic is not particularly difficult — this is not rocket-science grade logic, it is just propositions and some easy quantifiers — and this is worth looking into because it will yield dividends in simplicity, and offers a promise of reductions in engineering effort and risk. Shorter, simpler, and safer: *pick three!*

The reader with a logic background should also find plenty of food for thought and future research. There is a new polynomial-based denotation, potential for the reader to develop their own tailored logics based on the same ideas (classical, non-classical, and resource-sensitive), and there are some interesting things going on in our applications having to do with use of reflection, truth-modalities, and more (see Remark 6.1.1). The use of cryptography, if this is unfamiliar to the reader, is clearly signposted and clearly explained, and even if the reader prefers to just focus on the logic, this paper should still be entirely clear.

2 Polynomial semantics for FOL

2.1 Some background on polynomials

We recall some standard definitions and notations. We start off very simply:

Notation 2.1.1. We will be interested in $\mathbb{Z}$ (integers), $\mathbb{N}$ (nonnegative whole numbers), and $\mathbb{Q}$ (rationals).

We may use subscripts to indicate ranges, writing e.g. $\mathbb{N}_{\geq 0}$ (which is just $\mathbb{N}$), $\mathbb{N}_{\geq 1}$ (strictly positive whole numbers), $\mathbb{Q}_{\geq 0}$, and so on.

For $n \geq 0$ we may write

$$1..n = \{1, \ldots, n\} \subseteq \mathbb{N}.$$

If $n = 0$ then by convention $1..n = \varnothing$.

Note in particular that $\mathbb{N} = \{0, 1, 2, \ldots\}$; as is standard in computer science, we start counting at 0, not 1. However, in keeping with standard usage in mathematics, we will start indexing matrix rows and columns from 1, not 0.

Definition 2.1.2. For the purposes of this paper, a **rational univariate polynomial** (or just **polynomial** for short) is an element $P \in \mathbb{Q}[X]$, where X is a **variable**.

Thus, $P \in \mathbb{Q}[X]$ is a formal sum of powers of X. For example 0, $1 + X$, and X^2 are all polynomials.

Polynomials can be added and multiplied in the usual way. For example, $(1 + X) * (1 - X) = 1 - X^2$.

Remark 2.1.3. Definition 2.1.2 might seem simple, but for clarity we note a distinction between *expressions denoting polynomials*, and polynomials.

For example: $(1 + X) * (1 - X)$ is a polynomial expression. It denotes the polynomial $1 - X^2$, but is not identical to it.

So given $P, Q \in \mathbb{Q}[X]$, when we write $P * Q$ the reader should understand this polynomial expression to mean 'that polynomial in $\mathbb{Q}[X]$ obtained by performing a polynomial multiplication of P and Q', unless stated otherwise.

Polynomial interpolation in Definition 2.1.4 describes a standard method for using a polynomial to express a vector [25, Chapter 9].[7]

[7] The Wikipedia page and this overview are excellent and very accessible introductions.

Definition 2.1.4 (Polynomial interpolation). Suppose $ln \geq 0$ and $(a_1, \dots, a_{ln}) \in \mathbb{Z}^{ln}$ is a ln-tuple of integers.[8]

1. Say that a polynomial $P \in \mathbb{Q}[X]$ **interpolates** $(a_1, \dots, a_{ln})$ when $P(i) = a_i$ for each $i \in 1..ln$.
2. Recall that every ln-tuple of integers $(a_1, \dots, a_{ln})$ can be interpolated by a unique **polynomial interpolant**
$$\mathit{itplnt}(a_1, \dots, a_{ln}) \in \mathbb{Q}[X]$$
of degree ln-1 (at most).[9] Thus, $\mathit{itplnt}(a_1, \dots, a_{ln})(i) = a_i$ for $i \in 1..ln$.

Notation 2.1.5 (Instantiation). Suppose $P, Q \in \mathbb{Q}[X]$ are polynomials. Write $P(Q)$ for the polynomial obtained by **instantiating** X to Q in P.[10] For example if $P = X + 1$ and $Q = 2 * X$ then

$$P(Q) = 2 * X + 1.$$

In the case that Q is just a number $x \in \mathbb{Q}$, then we may identify this with the number obtained by instantiating X to x and evaluating the result using arithmetic. For example:

$$(X * (X + 1))(6) \;=\; 6 * (6 + 1) \;=\; 42.$$

It will always be clear what is intended.

2.2 The syntax of our logic (with examples)

Remark 2.2.1. Our goal in this paper is to define a logic, give it a semantics in $\mathbb{Q}[X]$, and explore the properties of this semantics. We will:

1. define the formal syntax of terms and predicates in Definition 2.2.3;
2. discuss some intuitions in Remark 2.2.4;
3. define the polynomial semantics for our formal syntax in Definition 2.3.2;
4. discuss the semantics in Remark 2.3.5; and
5. prove a soundness and completeness result in Theorem 2.4.4.

[8] A tuple of rationals would do just as well, but we will only interpolate tuples of integers.

[9] For example, 1-tuple (a) can be interpolated by $a * X^0$, which is a zero degree polynomial. By convention, we interpolate the 0-tuple () with 0 and consider this to have degree -1. This edge case will not arise in our applications.

[10] The reader might have seen this called *composition* or *evaluation*. I prefer *instantiation* because this is descriptive: X in P is a formal symbol and we instantiate it to Q.

$$\begin{aligned} t ::= {} & X \mid q \mid t+t \mid t-t \mid t*t \mid \mathsf{len}(\mathtt{C}) \mid \mathsf{reify}(\phi) \quad (q\in\mathbb{Q},\ \mathtt{C}\in\mathsf{MatrixVar}) \\ & \mid \mathtt{C}_i(t) \qquad (i \in 1..\mathit{arity}(\mathtt{C})) \end{aligned}$$

$$\begin{aligned} \phi,\psi ::= {} & \top \mid \bot \mid t=t \mid \phi \land \phi \mid \phi \lor \phi \mid \forall_{\mathtt{C}} X.\phi \mid \exists X_{\mathtt{C}}.\phi \\ & \mid t<\mathtt{C}_i \mid t>\mathtt{C}_i \qquad (i \in 1..\mathit{arity}(\mathtt{C})) \end{aligned}$$

The grammars above are in BNF style, such that ϕ to the right of $::=$ represents any predicate. In particular, $\phi\land\phi$ is not a typo; it denotes a conjunction of two (possibly non-equal) predicates. Similarly the two t in $t=t$ denote two (possibly non-equal) terms. $1..\mathit{arity}(\mathtt{C})$ denotes the set $\{1, 2, \dots, \mathit{arity}(\mathtt{C})\} \subseteq \mathbb{N}$.

Figure 1: Syntax of terms and predicates (Definition 2.2.3(3))

Remark 2.2.2. Note that in the interpretation we are about to build in Figure 2, 0 represents truth and any non-zero value represents false; furthermore $\land$ and $\forall$ (conjunction and universal quantification) map to $+$ (addition), and $\lor$ and $\exists$ (disjunction and existential quantification) map to $*$.

If the reader is used to thinking of truth as 1 and false as 0, as is traditional in logic circuits, then be careful: everything below will be flipped with respect to what you initially expect. Our motivation for representing truth by 0 is that it gives us Theorem 2.4.4.[11] More on this in Remarks 2.3.5 and 2.4.5.

Definition 2.2.3 (Syntax of terms and predicates).

1. Fix one **index variable symbol** X.
2. Fix a set of **matrix variable symbols** $\mathtt{C}, \mathtt{D} \in \mathsf{MatrixVar}$. To each matrix variable symbol we associate a fixed but arbitrary **arity** $\mathit{arity}(\mathtt{C}) \in \mathbb{N}_{\geq 1}$.
3. Define **terms** and **predicates** inductively as in Figure 1. In that Figure:
   (a) i ranges over whole numbers between 1 and the arity of $\mathtt{C}$ inclusive, or in symbols: $i \in 1..\mathit{arity}(\mathtt{C})$.
   (b) This is just formal syntax; we give it meaning later in Figure 2. In particular: $\mathtt{C}_i$, $\mathtt{C}_i(t)$, $\mathsf{len}(\mathtt{C})$, and $\mathsf{reify}(\phi)$ are just formal syntax.

Remark 2.2.4. We give some intuitive discussion of the syntax in Definition 2.2.3:

[11] Representing truth with 0 and non-truth with nonzero does have some pedigree. In a paper from 1943 [14] on page 51 just after equation 17, Kleene writes *"a representing function π of P, the value of which is to be 0, 1, or undefined according as the value of P is true, false, or undefined"*. More recently, in the `sysexists.h` file (credited to Eric Allman in 1980) which describes status codes for system programs, 0 represents successful termination and nonzero values represent various kinds of failure.

1. The index variable symbol X is the (unique) variable of the polynomial semantics.[12] Terms and predicates will map to $\mathbb{Q}[X]$ (polynomials over X), as per Figure 2.
2. Intuitively a matrix variable symbol $\mathtt{C} \in \mathsf{MatrixVar}$ ranges over all integer matrices of size $arity(\mathtt{C}) \times ln$, for all $ln \in \mathbb{N}_{\geq 1}$.
   Using interpolation (Definition 2.1.4) we can also think of $\mathtt{C}$ as ranging over elements of $\mathbb{Q}[X]^{arity(\mathtt{C})} \times \mathbb{N}_{\geq 1}$, where the first component is an $arity(\mathtt{C})$-tuple of polynomials over $\mathbb{Q}[X]$, and the second component is the $n \geq 1$ that tells us how to reconstruct the matrix by considering the values of those polynomials on the set $1..n$.
3. Terms let us build elements of $\mathbb{Q}[X]$.
   The intuition of $\mathtt{C}_i(t)$ is an interpolating polynomial for the ith row of $\mathtt{C}$, applied to t (more on this below). The intuition of $\mathsf{len}(\mathtt{C})$ is the number of columns in $\mathtt{C}$ (its *length*). The intuition of $\mathsf{reify}(\phi)$ is to treat ϕ as a number (this makes the logic *impredicative*; we fold predicates back down into terms).
4. Predicates are based on first-order logic. There is equality (of terms) but no negation (because negation is expressible; see Subsection 4.2.2).
   Intuitively, quantification $\forall_{\mathtt{C}} X$ quantifies over X ranging over whole numbers from 1 to the number of columns in the matrix denoted by $\mathtt{C}$; so if $\mathtt{C}$ denotes an $arity(\mathtt{C}) \times ln$ matrix, then $\forall_{\mathtt{C}} X.\phi$ checks $\phi(1), \ldots, \phi(ln)$, where $\phi(x)$ is the value of (some polynomial denotation which we have not yet constructed of) ϕ for X instantiated to $x \in 1..ln$.
5. The intuition of the range checks $t < \mathtt{C}_i$ and $t > \mathtt{C}_i$ is to statically check that elements of the ith row of $\mathtt{C}$ takes values greater than or less than t respectively. We use this later to statically check for out-of-bounds pointer errors (see the discussion in Examples 3.2.8 and 3.2.9), and we make further use of this in Section 4.

EXAMPLE 2.2.5. We give some examples of the syntax in action (even if we have not yet assigned it any formal meaning, we can illustrate how it is supposed to be used). Assume three matrix variable symbols:

1. $\mathtt{neg}$ with $arity(\mathtt{neg}) = 2$.
2. $\mathtt{add}$ with $arity(\mathtt{add}) = 3$.
3. $\mathtt{pos}$ with arity $arity(\mathtt{pos}) = 1$.

Then we can write some easy predicates:

1. $\forall_{\mathtt{neg}} X.\mathtt{neg}_2(X) = (\text{-}1) * \mathtt{neg}_1(X)$.

[12] A multivariate generalisation of the ideas in this paper is possible, and may be helpful for efficiency. However, in terms of expressivity just the one X will suffice for our needs.

2. $\forall_{\mathtt{add}} X.\mathtt{add}_3(X) = \mathtt{add}_1(X) + \mathtt{add}_2(X)$.
3. $0 < \mathtt{pos}_1$.

Intuitively, we are supposed to read these predicates as follows:

1. `neg` is (= denotes) a matrix with two rows, and the Xth entry in row 2 is (as per the predicate) supposed to be the negation of the Xth entry in row 1, like this:
$$\begin{pmatrix} 0 & 1 & 2 \\ 0 & \text{-}1 & \text{-}2 \end{pmatrix}$$
2. `add` is a matrix with three rows. Entries in row 3 are (as per the predicate) supposed to be the sum of the entries in rows 1 and 2, like this:
$$\begin{pmatrix} 0 & 1 & 2 \\ 2 & 1 & 0 \\ 2 & 2 & 2 \end{pmatrix}$$
3. `pos` is a matrix with just one row, which should consist entirely of strictly positive numbers, like this:
$$\begin{pmatrix} 3 & 2 & 1 \end{pmatrix}$$

In terms of intuition, this is all straightforward.[13]

We do not have a notion of validity yet, so all of the above is just intuition. Once we have defined validity, we will come back to this; see Example 2.3.4.

Remark 2.2.6. We make a few comments on some simple (non)redundancies in the syntaxes of terms and predicates:

1. Subtraction in terms is redundant: we can just express $t - t'$ as $t + (\text{-}1) * t'$.
2. $\top$ and $\bot$ are redundant: we can just express $\top$ as $0 = 0$, and $\bot$ as $1 = 0$.
3. If we admit a unary predicate-former $\mathtt{isZero}(t)$ with semantics $[\![\mathtt{isZero}(t)]\!]_\varsigma = [\![t]\!]_\varsigma^2$, then $\top$, $\bot$, and $t - t'$ become expressible as
$$\top = \mathtt{isZero}(0), \qquad \bot = \mathtt{isZero}(1), \quad \text{and} \quad (t = t') = \mathtt{isZero}(t - t').$$
4. Negation does not appear explicitly in the syntax or semantics of Figures 1 and 2. Surprisingly, negation is expressible in the rest of the logic: see Subsection 4.2.2.

See also Remark 2.4.6.

[13] In fact, it will turn out that `pos` is more interesting than one might expect. We will see more of it later in Subsection 4.1.1.

Notation 2.2.7. We set up some convenient abbreviations, writing:

$$\begin{array}{lll} \mathtt{C}_i < t & \text{for} & t > \mathtt{C}_i \\ t \leq \mathtt{C}_i & \text{for} & t - 1 < \mathtt{C}_i \\ \mathtt{C}_i \leq t & \text{for} & \mathtt{C}_i < t + 1 \\ t < \mathtt{C}_i < t' & \text{for} & (t < \mathtt{C}_i) \land (\mathtt{C}_i < t') \\ t \leq \mathtt{C}_i \leq t' & \text{for} & (t - 1 < \mathtt{C}_i) \land (\mathtt{C}_i < t + 1). \end{array}$$

Note that this is meaningful because when we build our semantics, our notion of interpretation in Definition 2.3.2(1) will evaluate matrix variable symbols $\mathtt{C}$ to *integer* matrices.

There is not much to this otherwise: we could also just as well take the notation above as primitive in our syntax from Figure 1, and interpret $\leq$ (a binary predicate symbol in our formal syntax) using $\leq$ (the binary predicate 'less than or equals' on numbers, which returns a truth-value) in Figure 2, alongside interpreting $<$ (the predicate symbol) using $<$ (the predicate 'strictly less than').

2.3 The semantics (with examples)

We are now ready to define a semantics in polynomials for the syntax from Definition 2.2.3.

Some notation will be useful for Figure 2 and Definition 2.3.2:

Notation 2.3.1 (Some basic matrix syntax). Suppose M is a matrix. Then:

1. We may write $len(M)$ for the number of columns in M, and we may call this the **length** of M.
2. We may write M_i for the ith row in M, which is a vector of length $len(M)$.
3. As usual, we may write $M_{i,j}$ for the i,j-th element in M; i is the row number and j is the column number.

Definition 2.3.2 (Polynomial semantics).

1. An **interpretation** ς assigns to each matrix variable symbol $\mathtt{C} \in \mathsf{MatrixVar}$ with arity $arity(\mathtt{C}) \in \mathbb{N}_{\geq 1}$ an $arity(\mathtt{C}) \times n$ integer matrix (= having elements from $\mathbb{Z}$).
   Using interpolation (Definition 2.1.4) the reader can also think of $\varsigma(\mathtt{C})$ as a vector of $arity(\mathtt{C})$ many polynomials in X, along with a number n encoding the information that we care about the values of these polynomials on $1..n$.
2. Given an interpretation ς, define **polynomial semantics**

$$[\![t]\!]_\varsigma \in \mathbb{Q}[X] \quad \text{and} \quad [\![\phi]\!]_\varsigma \in \mathbb{Q}[X]$$

$$
\begin{array}{rlrl}
[\![X]\!]_\varsigma &= X & [\![q]\!]_\varsigma &= q \qquad (q \in \mathbb{Q}) \\
[\![t+t']\!]_\varsigma &= [\![t]\!]_\varsigma + [\![t']\!]_\varsigma & [\![t-t']\!]_\varsigma &= [\![t]\!]_\varsigma - [\![t']\!]_\varsigma \\
[\![t*t']\!]_\varsigma &= [\![t]\!]_\varsigma * [\![t']\!]_\varsigma & [\![\mathtt{C}_i(t)]\!]_\varsigma &= \mathit{itplnt}(\varsigma(\mathtt{C})_i)([\![t]\!]_\varsigma) \\
[\![\mathsf{len}(\mathtt{C})]\!]_\varsigma &= \mathit{len}(\varsigma(\mathtt{C})) & [\![\mathsf{reify}(\phi)]\!]_\varsigma &= [\![\phi]\!]_\varsigma
\end{array}
$$

$$
\begin{array}{rlrl}
[\![\mathsf{T}]\!]_\varsigma &= 0 & [\![\bot]\!]_\varsigma &= 1 \\
[\![t = t']\!]_\varsigma &= ([\![t]\!]_\varsigma - [\![t']\!]_\varsigma)^2 & & \\
[\![\phi \wedge \phi']\!]_\varsigma &= [\![\phi]\!]_\varsigma + [\![\phi']\!]_\varsigma & [\![\phi \vee \phi']\!]_\varsigma &= [\![\phi]\!]_\varsigma * [\![\phi']\!]_\varsigma \\
[\![\forall_{\mathtt{C}} X.\phi]\!]_\varsigma &= \sum_{x \in 1..\mathit{len}(\varsigma(\mathtt{C}))} [\![\phi]\!]_\varsigma(x) & &
\end{array}
$$

$$[\![\exists_{\mathtt{C}} X.\phi]\!]_\varsigma = \prod_{x \in 1..\mathit{len}(\varsigma(\mathtt{C}))} [\![\phi]\!]_\varsigma(x)$$

$$[\![t > \mathtt{C}_i]\!]_\varsigma = \begin{cases} 0 & \text{if } \forall j \in 1..\mathit{len}(\varsigma(\mathtt{C})).([\![t]\!]_\varsigma > \varsigma(\mathtt{C})_{i,j}) \\ 1 & \text{otherwise} \end{cases}$$

$$[\![t < \mathtt{C}_i]\!]_\varsigma = \begin{cases} 0 & \text{if } \forall y \in 1..\mathit{len}(\varsigma(\mathtt{C})).([\![t]\!]_\varsigma < \varsigma(\mathtt{C})_{i,j}) \\ 1 & \text{otherwise} \end{cases}$$

$$x \vDash_\varsigma \phi \iff [\![\phi]\!]_\varsigma(x) = 0$$

Negation is expressible using the rest of the logic: see Subsection 4.2.2.

Figure 2: Polynomial semantics for terms and predicates (Definition 2.3.2)

for terms and predicates as in Figure 2. See Remark 2.3.5 below for exposition on the notation and design.

If ς is irrelevant (e.g. because no matrix variable symbols are mentioned), we may drop the subscript ς and write $[\![t]\!]_\varsigma$ just as $[\![t]\!]$, or $[\![\phi]\!]_\varsigma$ just as $[\![\phi]\!]$. For example, we may write $[\![2=0]\!] = 4$.

3. Given $x \in \mathbb{Q}$, define a **judgement** by

$$x \vDash_\varsigma \phi \quad \text{means} \quad [\![\phi]\!]_\varsigma(x) = 0,$$

as per Figure 2. Judgements return truth-values ('true' or 'false').[14]

4. If x or ς is irrelevant in $x \vDash_\varsigma \phi$ (e.g. because ϕ is fully-quantified, or mentions no matrix variable symbols) then we may omit it, writing

$$\vDash_\varsigma \phi \quad \text{or} \quad x \vDash \phi \quad \text{or even just} \quad \vDash \phi.$$

When $x \vDash_\varsigma \phi$ we will call ϕ **valid** (in the context of x and ς). If x and/or ς is irrelevant, we may just call ϕ **valid**.

[14] So if ϕ is a predicate and $x \in \mathbb{Q}$ and ς is an interpretation then: $[\![\phi]\!]_\varsigma$ is a polynomial; $[\![\phi]\!]_\varsigma(x)$ is a rational number; and $x \vDash_\varsigma \phi$ is a truth-value.

Remark 2.3.3. Definition 2.3.2 is written in the language of logicians. It is straightforward to map this to language that may be more familiar to cryptographers:

1. The predicate ϕ corresponds to what in a cryptographic proof-scheme is called the *instance*. This defines the problem to be solved and is known to the *prover* and the *validator*.
2. The interpretation ς, which maps matrix variable symbols to matrices — equivalently: to interpolating polynomials — corresponds to the *witness*. This information is known to the prover, but not necessarily to the validator.
3. The validator and prover agree on ϕ, and — on provision of a cryptographic proof-scheme — the prover succinctly or in zero knowledge would prove (in the cryptographic sense) that it knows a witness ς that makes the instance valid.[15]

The reader can find overviews in [26] and [28]. The choice (and if necessary, the design) of proof-schemes is not monolithic, and may depend on the structure of ϕ — e.g. DNF (disjunctive normal form), Horn clause, etc — and on the intended use-case (succinctness, zero knowledge, rational polynomials vs. polynomials over finite fields, etc). For different choices, different proof-schemes making different trade-offs may be appropriate.

Example 2.3.4. We now use Figure 2 to translate the predicates from Example 2.2.5 into polynomials. Recall that we assumed three matrix variable symbols with associated predicates — since we now have a notion of validity, we will call these *validity predicates* — as follows:

1. $\mathit{arity}(\mathtt{neg}) = 2$ and validity predicate $\chi_{\mathtt{neg}} = \forall_{\mathtt{neg}} X.\mathtt{neg}_2(X) = (\text{-}1) * \mathtt{neg}_1(X)$.
2. $\mathit{arity}(\mathtt{add}) = 3$ and validity predicate $\chi_{\mathtt{add}} = \forall_{\mathtt{add}} X.\mathtt{add}_3(X) = \mathtt{add}_1(X) + \mathtt{add}_2(X)$.
3. $\mathit{arity}(\mathtt{pos}) = 1$ and validity predicate $0 < \mathtt{pos}_1$.

We intend these to model *negation*, *addition*, and the property of a number being *positive*.

[15] This use of the word witness is consistent with the notion of witness to an existential quantification in logic. Our notion of validity $\vDash_\varsigma \phi$ is agnostic to where ϕ and ς come from, but in our intended application ϕ will be universally quantified (supplied by the validator) and ς will be existentially quantified (supplied by the prover).

Or, to put things more abstractly: we intend validity judgements be used such that a validator fixes some desired property, and a prover constructs a witnessing model to that property, and proves that it has done so, without necessarily revealing the model to the validator.

There is a subtle possibility for confusion here: in cryptography some details of the witness may be fixed by the validator, e.g. “Find me a model with at least 10 elements” or “Let the input to the program be 42”. Such is the power of logic, that this information can be adjoined to ϕ.

Given an interpretation ς, the translations of these validity predicates are polynomials

$$[\![\chi_{\mathtt{neg}}]\!]_\varsigma,\ [\![\chi_{\mathtt{add}}]\!]_\varsigma,\ [\![\chi_{\mathtt{pos}}]\!]_\varsigma\ \in \mathbb{Q}[X]$$

as per Figure 2 as follows:

$$\begin{aligned}
[\![\chi_{\mathtt{neg}}]\!]_\varsigma &= \textstyle\sum_{x\in 1..len(\varsigma(\mathtt{neg}))}(\mathit{itplnt}(\varsigma(\mathtt{neg})_2)(x) - (\text{-}1) * \mathit{itplnt}(\varsigma(\mathtt{neg})_1)(x))^2 \\
[\![\chi_{\mathtt{add}}]\!]_\varsigma &= \textstyle\sum_{x\in 1..len(\varsigma(\mathtt{add}))}\left(\begin{array}{l}\mathit{itplnt}(\varsigma(\mathtt{add})_3)(x)\ - \\ (\mathit{itplnt}(\varsigma(\mathtt{add})_1)(x)+\mathit{itplnt}(\varsigma(\mathtt{add})_2)(x))\end{array}\right)^2 \\
[\![\chi_{\mathtt{pos}}]\!]_\varsigma &= \begin{cases}0 & \text{if every entry in } \varsigma(\mathtt{pos}) \text{ is} > 0\\ 1 & \text{if some entry in } \varsigma(\mathtt{pos}) \text{ is} \leq 0\end{cases}
\end{aligned}$$

Let us unpack some of the notation above. We consider just the first clause (the second and third are no different): `neg` is a matrix variable symbol (a symbol in our syntax from Figure 1); $\varsigma(\mathtt{neg})$ is a matrix (as per Definition 2.3.2(1)); and $\mathit{itplnt}(\varsigma(\mathtt{neg})_1)$ is a polynomial interpolating the first row of that matrix at 1, 2, ..., $len(\varsigma(\mathtt{neg}))$ where $len(\varsigma(\mathtt{neg}))$ is simply the number of columns in that matrix.

Note that none of these polynomials have X free in them, so in fact they are just numbers. Thus, following Definition 2.3.2(4) the predicates are *valid* in the context of ς when that number is 0, and they are *invalid* when that number is strictly greater than 0.

The reader can check that the predicates are valid of ς, precisely when ς satisfies the conditions that we would intuitively expect — as spelled out in Example 2.2.5 — based on reading the predicates. This observation will be formalised in our soundness and completeness result in Theorem 2.4.4.

Remark 2.3.5. Some comments on Figure 2:

1. $\top$ maps to $[\![\top]\!]_\varsigma = 0$ and $\bot$ maps to $[\![\bot]\!]_\varsigma = 1$. Conjunction and universal quantification map to addition, and disjunction and existential quantification map to multiplication (cf. also Remark 2.4.5). Consequently we can say:

   In our denotation, 'zero' means 'true' and 'nonzero' means 'false'.

   This might be confusing because usually (e.g. in logic gates) 1 means 'true' and (just) 0 means 'false', but setting things up as we do gives us Theorem 2.4.4.

2. The semantics maps terms and predicates to polynomials over X. So for example, the clause $[\![X]\!]_\varsigma = X$ says that the X denotes itself.

3. Similarly the clause $[\![q]\!]_\varsigma = q$ says that $q \in \mathbb{Q}$ denotes itself.

4. The clause $[\![\mathtt{C}_i(t)]\!]_\varsigma = \mathit{itplnt}(\varsigma(\mathtt{C})_i)([\![t]\!]_\varsigma)$ implicitly assumes $i \in 1..\mathit{arity}(\mathtt{C})$, since as per Definition 2.2.3(3a) and Notation 2.3.1(2) it would make no sense to talk about $\mathtt{C}_i$ or $\varsigma(\mathtt{C})_i$ otherwise.

5. We break down the clause $[\![\mathtt{C}_i(t)]\!]_\varsigma = \mathit{itplnt}(\varsigma(\mathtt{C})_i)([\![t]\!]_\varsigma)$, as follows:

   - $\varsigma(\mathtt{C})$ is an integer matrix with $\mathit{arity}(\mathtt{C})$ rows and $\mathit{len}(\varsigma(\mathtt{C}))$ columns. For brevity we may write
   $$C = \varsigma(\mathtt{C}).$$
   Note that the number of rows in C *must* be $\mathit{arity}(\mathtt{C})$; but the number of columns in C is determined by the choice of ς.
   - As per Notation 2.3.1(2), C_i denotes the ith row of C, which is a vector of $\mathit{len}(C)$ integers.
   - $\mathit{itplnt}(C_i) = P$ is a polynomial in X that interpolates the vector C_i at values $1, \dots, \mathit{len}(C)$, as per Definition 2.1.4.
   - $P([\![t]\!]_\varsigma)$ is P instantiated at $[\![t]\!]_\varsigma$, as per Notation 2.1.5.

6. Reading the previous point, the reader might ask why we bother with matrices at all. Why not just let $\varsigma(\mathtt{C})_i$ *be* a polynomial?

   Because a matrix and its interpolant are not the same thing, if we care about bounds (which we do); a vector has a *length*, and a polynomial *a priori* does not. For example, the same polynomial X interpolates many distinct vectors, including (1), $(1, 2)$ and $(1, 2, 3)$.[16]

Remark 2.3.6. Some comments on Definition 2.3.2:

1. Recall that $\varsigma(\mathtt{C})$ in Definition 2.3.2(1) is an integer matrix C with $\mathit{arity}(\mathtt{C})$ rows and n columns. The number of rows in C is determined by $\mathit{arity}(\mathtt{C})$, but its number of columns depends on ς.

2. Interpretations ς map matrix variable symbols to integer matrices. They do not instantiate the (single) variable symbol X. Furthermore, although we call $\varsigma(\mathtt{C})$ a matrix, we use it as a two-dimensional array (no matrix multiplication, for example). If every time we write 'integer matrix' the reader reads 'two-dimensional array with integer entries', then no harm will come of it.

[16] Obviously, in an implementation we care about efficiency so we might (or might not) prefer the rows of our matrix to be delivered as a vector of interpolating polynomials, along with an intended length. We can choose how best to represent our data. But, what that representation *represents*, is a matrix.

3. In Definition 2.3.2(2), $[\![\phi]\!]_\varsigma$ is a polynomial in X — it is not a truth-value. For example,
$$[\![X = 2 * X]\!]_\varsigma = X^2$$
(there are no matrix variable symbols here so the choice of ς does not matter).

4. In Definition 2.3.2(3), $x \vDash_\varsigma \phi$ is a truth-value — it is not a polynomial. For example,
$$0 \vDash_\varsigma X = 2 * X \ \text{ is true, and } \ 1 \vDash_\varsigma X = 2 * X \ \text{is false.}$$

5. The denotation of universal quantification $\forall$ is a summation, like the denotation of $\wedge$. However, $\forall$ cannot be emulated in the logic by repeated $\wedge$, because the length of the summation depends on the number of columns in the matrix $\varsigma(\mathtt{C})$, which varies depending on the choice of ς. So universal quantification has its own distinct identity.

6. A design path which we do not explore in this paper is that our denotation is compatible with notions of 'logical implication' $P \Longrightarrow Q$ and 'logical equivalence' $P \Longleftrightarrow Q$ on polynomials $P, Q \in \mathbb{Q}[X]$, such that
$$\begin{array}{lll} P \Longrightarrow Q & \text{when} \ \ \forall x{\in}\mathbb{Q}.P(x) = 0 \Longrightarrow Q(x) = 0 & \quad \text{and} \\ P \Longleftrightarrow Q & \text{when} \ \ \forall x{\in}\mathbb{Q}.P(x) = 0 \Longleftrightarrow Q(x) = 0. & \end{array}$$
Note that $P \Longrightarrow Q$ does not quite mean the same thing as $P \mid Q$ (P divides Q), because the multiplicities of the roots in P may differ from those in Q. Similarly, $P \Longleftrightarrow Q$ does not quite mean the same thing as $P = Q$. For example:
$$(X - 1) \Longleftrightarrow (X - 1) * (X^2 + 1) \quad \text{and} \quad (X - 1) \Longleftrightarrow (X - 1)^2.$$
We do not explore these notions of implication and equivalence explicitly in what follows, but they are implicit in much of the maths; exploring this explicitly is future work.

   For now, it seems to be more useful to explore a technically different design path and consider a *complementation operation* $\sim\!\phi$; see Notation 4.2.6 and Theorem 4.2.7.

2.4 Soundness and completeness

Definition 2.4.1. Call a polynomial $P \in \mathbb{Q}[X]$ **nonnegative** when $P(x) \geq 0$ for every $x \in \mathbb{Q}$.

Lemma 2.4.2. *Suppose ϕ is a predicate and ς is an interpretation. Then $[\![\phi]\!]_\varsigma$ is a nonnegative polynomial (Definition 2.4.1).*

Proof. By a routine induction, following Figure 2:

- $[\![\top]\!]_\varsigma = 0$ and $[\![\bot]\!]_\varsigma = 1$. We note that 0 and 1 are nonnegative.
- $[\![t = t']\!]_\varsigma = ([\![t]\!]_\varsigma - [\![t']\!]_\varsigma)^2$. It is a fact of arithmetic that this is a nonnegative polynomial, because it is a square.[17]
- $[\![\phi \wedge \phi']\!]_\varsigma = [\![\phi]\!]_\varsigma + [\![\phi']\!]_\varsigma$ and $[\![\phi \vee \phi']\!]_\varsigma = [\![\phi]\!]_\varsigma * [\![\phi']\!]_\varsigma$. We observe that the sum and product of two (by inductive hypothesis) nonnegative polynomials, is nonnegative.
- As per Figure 2, $[\![\forall_{\mathtt{C}} X.\phi]\!] = \sum_{x\in 1..len(\varsigma(\mathtt{C}))} [\![\phi]\!]_\varsigma$. We observe that this is a sum of (by inductive hypothesis) nonnegative polynomials, and so it is also nonnegative. Similarly for $\exists_{\mathtt{C}} X.\phi$.
- $[\![t < \mathtt{C}_i]\!]_\varsigma$ and $[\![\mathtt{C}_i < t]\!]_\varsigma$ are equal to 0 or 1, and both are nonnegative. □

Remark 2.4.3. Lemma 2.4.2 is interesting because it suggests that we can identify the nonnegative polynomials in $\mathbb{Q}[X]$ as having *logical content*, in the sense that these can be the denotations of predicates; in contrast to the class of all polynomials in $\mathbb{Q}[X]$, which can be the denotation of terms but not necessarily of predicates.

Theorem 2.4.4 (Soundness & completeness). *Suppose ϕ and ϕ' are predicates, and t and t' are terms, and ς is an interpretation and $x \in \mathbb{Q}$. Then:*

1. $x \vDash_\varsigma \top$ *and* $x \nvDash_\varsigma \bot$
2. $x \vDash_\varsigma t = t'$ *if and only if* $[\![t]\!]_\varsigma(x) = [\![t']\!]_\varsigma(x)$.
3. $x \vDash_\varsigma \phi \wedge \phi'$ *if and only if* $x \vDash_\varsigma \phi \;\wedge\; x \vDash_\varsigma \phi'$.
4. $x \vDash_\varsigma \phi \vee \phi'$ *if and only if* $x \vDash_\varsigma \phi \;\vee\; x \vDash_\varsigma \phi'$.
5. $x \vDash_\varsigma \forall_{\mathtt{C}} X.\phi$ *if and only if* $x' \vDash_\varsigma \phi$ *for every* $x' \in 1..len(\varsigma(\mathtt{C}))$.
6. $x \vDash_\varsigma \exists_{\mathtt{C}} X.\phi$ *if and only if* $x' \vDash_\varsigma \phi$ *for some* $x' \in 1..len(\varsigma(\mathtt{C}))$.
7. $x \vDash_\varsigma t < \mathtt{C}_i$ *if and only if every entry in the ith row of $\varsigma(\mathtt{C})$ is strictly greater than* $[\![t]\!]_\varsigma$.[18]
8. $x \vDash_\varsigma \mathtt{C}_i < t$ *if and only if every entry in the ith row of $\varsigma(\mathtt{C})$ is strictly less than* $[\![t]\!]_\varsigma$.

Proof. We consider each part in turn, unpacking Figure 2. The argument is just by routine arithmetic:

[17] Indeed, the square is there precisely to guarantee nonnegativity.

[18] Note that $[\![t]\!]_\varsigma$ is just a number, because t is a term that is assumed to not mention X.

1. $x \vDash \top$ when $0 = 0$, which is true. $x \vDash \bot$ when $1 = 0$, which is false.
2. $x \vDash t = t'$ when $(t - t')^2(x) = 0$. By elementary facts of arithmetic, this is if and only if $t(x) = t'(x)$.
3. $x \vDash \phi \wedge \phi'$ when $[\![\phi]\!](x) + [\![\phi']\!](x) = 0$. Using Lemma 2.4.2, this holds if and only if $[\![\phi]\!] = 0$ and $[\![\phi']\!] = 0$, and thus if and only if $x \vDash \phi \wedge x \vDash \phi'$.
4. $x \vDash \phi \vee \phi'$ when $[\![\phi]\!] * [\![\phi']\!] = 0$. This holds and only if $[\![\phi]\!] = 0$ or $[\![\phi']\!] = 0$.
5. As for $\wedge$, the sum $\sum_{x \in 1..len(\varsigma(\mathtt{C}))} [\![\phi]\!]_\varsigma$ is zero if and only if all of its component summands are zero.
6. The product $\prod_{x \in 1..len(\varsigma(\mathtt{C}))} [\![\phi]\!]_\varsigma$ is zero if and only if one of its component summands is zero.
7. By Figure 2, $[\![t < \mathtt{C}_i]\!]_\varsigma$ means precisely that t is strictly less than every entry in the ith row of $\varsigma(\mathtt{C})$. Similarly for $\mathtt{C}_i < t$.

   We will also use $t \leq \mathtt{C}_i$ and $\mathtt{C}_i \leq t$ as per Notation 2.2.7. Because $\varsigma(\mathtt{C})$ is an integer matrix, $[\![t \leq \mathtt{C}_i]\!]_\varsigma$ means precisely that t is less than or equal to every entry in the ith row of $\varsigma(\mathtt{C})$, and similarly for $\mathtt{C}_i \leq t$. □

Remark 2.4.5. Theorem 2.4.4 is important — a logical judgement is valid precisely when its polynomial translation is — but mathematically it is elementary. It just dresses up the fact that in $\mathbb{Q}$,

- a sum of two nonnegative numbers is zero if and only if both numbers are zero, and
- a product of two nonnegative numbers is zero if and only if at least one of the numbers is zero.

But of course, this is just what we need to precisely interpret logical conjunction / universal quantification, and logical disjunction / existential quantification, respectively.

What is absent from Theorem 2.4.4, because it is absent from our syntax in Figure 1, is *negation* — there is no explicit predicate-former $\neg\phi$. This is because it can be expressed using the rest of the logic; we will see later in Notation 4.2.6 and Theorem 4.2.7 that our logic is *complemented*. That is: given ϕ, we can write a complement predicate $\sim\!\phi$ that is valid if and only if ϕ is not valid.

But note also that our examples will show that even the negation-free fragment of FOL is very expressive (expressive enough to e.g. encode a Turing-complete model of computation; see Subsection 3.9).

For now, we just note that Theorem 2.4.4 is more powerful than it looks, and it will turn out that it can indeed lay claim to expressing full first-order logic with negation.

Remark 2.4.6 is in a sense a continuation of Remark 2.2.6:

Remark 2.4.6 (Dualities).

1. $\exists$ is a natural de Morgan dual to $\forall$ (as usual). We will use $\forall$ all the time in our practical examples below, but not make much use of $\exists$. It costs nothing to include it anyway.
2. reify maps from nonnegative polynomials to all polynomials just by being the identity. This is in a sense dual to the 'comparison with zero' ($t=0$), which maps from all polynomials to nonnegative polynomials by squaring. We *will* use this, most notably in our complementation operation from Notation 4.2.6.

3 Expressing functions

3.1 Simpler is better

Remark 3.1.1. We have our logic and its denotation. We are now ready to use it to specify functions.

The format of the results in this section is generally consistent:

1. We specify some (well-known) mathematical function, such as exponentiation.
2. We write a predicate in our formal syntax from Definition 2.2.3(3). This will correspond in some fairly natural way to the specification.
3. We state a soundness lemma, whose proof is usually elementary from Theorem 2.4.4, that the semantics of the predicate as per Figure 2 accurately captures the specification.

This can be a bit fiddly, because concretely manipulating large-ish expressions by hand can be fiddly, but mathematically it is straightforward. There is *supposed* to be a close match between the mathematical intuition, the formal syntax, and the semantics, so that the soundness proofs become straightforward. Our framework is designed to impose a minimal overhead, above and beyond any complexities that may be inherent in what is being expressed.

So for example, if the reader

- looks at Definition 3.2.1 (mathematical definition of exponentiation), and then
- looks at Figure 3 (formal predicate in our logic), and finally perhaps
- looks at Example 3.2.11 (unpacking the polynomial to which the formal predicate compiles)

and finds that it all looks rather straightforward, then that simplicity is a feature, not a bug.

3.2 Expressing a power function (with worked examples)

We start with a simple example: how to express a power function (exponentiation) a^b for $a, b \in \mathbb{N}_{\geq 0}$.

DEFINITION 3.2.1. Recall the standard inductive definition of a^b for $a, b \in \mathbb{N}_{\geq 0}$:

$$\begin{array}{lrcl} \text{base case} & \mathit{pow}(a, 0) & = & 1 \\ \text{inductive step} & \mathit{pow}(a, b{+}1) & = & a * \mathit{pow}(a, b) \end{array}$$

REMARK 3.2.2. The reader may know that this definition is not particularly efficient. In Remark 3.2.14 we mention the efficient definition; but for the purposes of an initial example, Definition 3.2.1 will serve us very well.

Note an elementary point that definitions like Definition 3.2.1 implicitly equate *computation* with *derivation*, because when we try to compute a value — $\mathit{pow}(2, 2)$, say — we end up with a simple derivation-tree as follows:

$$\dfrac{\dfrac{\dfrac{}{\mathit{pow}(2,0) = 1}\,\text{base case}}{\mathit{pow}(2,1) = 2}\,\text{inductive step}}{\mathit{pow}(2,2) = 4}\,\text{inductive step}$$

This derivation-tree, simple as it is, illustrates the close link between logic and computation.

The schema in Definition 3.2.4 and Figure 3 is typical of how we will work, so we spell it out now:

NOTATION 3.2.3.

1. For each function or relation of interest we assume a matrix variable symbol $\mathtt{C}$.
2. For each matrix variable symbol $\mathtt{C}$, we write out a **validity predicate** $\chi_{\mathtt{C}}$, corresponding to the function or relation which we wish to specify.
3. We say that an interpretation ς **satisfies** this validity predicate when $\vDash_\varsigma \chi_{\mathtt{C}}$.

Recall that the $\vDash_\varsigma \chi_{\mathtt{C}}$ notation is from Definition 2.3.2(4); it means $x \vDash_\varsigma \chi_{\mathtt{C}}$ and so $[\![\chi_{\mathtt{C}}]\!](x) = 0$ for some x (the choice of which will not matter, because X will be quantified in $\chi_{\mathtt{C}}$, as we shall see).

$$\begin{array}{llllll} \mathtt{pow}_1(t) & \text{sugars to} & \mathsf{in}_1(t) & \mathtt{pow}_2(t) & \text{sugars to} & \mathsf{in}_2(t) \\ \mathtt{pow}_3(t) & \text{sugars to} & \mathsf{out}(t) & \mathtt{pow}_4(t) & \text{sugars to} & \mathsf{rec}(t) \end{array}$$

$$\begin{aligned} \mathsf{baseCase}(X) &= \; (\mathsf{in}_2(X) = 0) \land (\mathsf{out}(X) = 1) \\ \mathsf{indStep}(X) &= \; \mathsf{in}_1(X) = \mathsf{in}_1(\mathsf{rec}(X)) \land \\ & \mathsf{in}_2(X) = \mathsf{in}_2(\mathsf{rec}(X)){+}1 \land \\ & \mathsf{out}(X) = \mathsf{in}_1(X) * \mathsf{out}(\mathsf{rec}(X)) \\ \chi_{\mathtt{pow}} &= \; 1 \leq \mathsf{rec} \leq \mathsf{len}(\mathtt{pow}) \;\land\; \forall_{\mathtt{pow}} X.(\mathsf{baseCase}(X) \;\lor\; \mathsf{indStep}(X)) \end{aligned}$$

Figure 3: Expressing a power function (exponentiation) a^b (Definitions 3.2.1 and 3.2.4)

Definition 3.2.4. Assume a matrix variable symbol $\mathtt{pow}$ with arity $\mathit{arity}(\mathtt{pow}) = 4$. Thus, the interpretation $\varsigma(\mathtt{pow})$ is a $4 \times n$ integer matrix, for $n \geq 1$.[19] We will encode an assertion that $\mathtt{pow}$ encodes a (partial) power function, such that:

1. Row 1 (the values of $\mathtt{pow}_1(X)$) stores input values a.
2. Row 2 (the values of $\mathtt{pow}_2(X)$) stores input values b.
3. Row 3 (the values of $\mathtt{pow}_3(X)$) stores output values a^b.
4. Row 4 (the values of $\mathtt{pow}_4(X)$) stores a pointer (specifically: a column index) to a column in the $\mathtt{pow}$ matrix that represents the recursive call to *pow* in clause 2 of Definition 3.2.1 above.

In Figure 3 we

1. define syntactic sugar,
2. use it to express the clauses from Definition 3.2.1, and
3. write a *validity predicate* $\chi_{\mathtt{pow}}$, to express that $\mathtt{pow}$ encodes a partial power function.

Examples of matrices that do and do not satisfy the validity predicate $\chi_{\mathtt{pow}}$ are in Examples 3.2.8 and 3.2.9.

Definition 3.2.5. Suppose $\mathit{ln} \in \mathbb{N}_{\geq 1}$. Say that a $4 \times \mathit{ln}$ matrix M **encodes a partial power function** when

$$\forall x \in 1..\mathit{ln}.\ M(3,x) = M(1,x)^{M(2,x)}.$$

[19] *Pedant point:* Calling $\varsigma(\mathtt{pow})$ an 'interpretation' might be a little misleading, so let us unpack the layers. ς is an interpretation, as per Definition 2.3.2(1). $\mathtt{pow}$ is a syntactic symbol which we call a matrix variable symbol. $\varsigma(\mathtt{pow})$ is the matrix value (a *denotation*) which the interpretation ς assigns to the syntactic matrix variable symbol $\mathtt{pow}$.

In words, M encodes a partial power function when for each column, the third entry of that column is equal to the first entry raised to the power of the second.

Lemma 3.2.6. *Suppose ς is an interpretation (Definition 2.3.2(1)). Then*

- *if $\vDash_{\varsigma} \chi_{\mathtt{pow}}$*
- *then $\varsigma(\mathtt{pow})$ encodes a partial partial power function (Definition 3.2.5).*

Proof. We examine the clauses in Figure 3 and using Theorem 2.4.4 we check that they correctly encode the definition in Definition 3.2.1.

We also need to check well-foundedness — that the pointers in the matrix do not point in a circle — but this follows from the treatment of the second input argument (named b in Definition 3.2.1). An overall discussion of this point is in Remark 3.9.8. □

Remark 3.2.7. The significance of Lemma 3.2.6 is not that the power function can be interpolated. *Any* function $f : \mathbb{N} \to \mathbb{N}$, however fast it goes to infinity, coincides with some polynomial on any given $\{1, \dots, \mathit{ln}\}$; just use the interpolating polynomial $\mathit{itplnt}(f(1), \dots, f(\mathit{ln}))$ from Definition 2.1.4. The significance of Lemma 3.2.6 is the predicate characterisation of valid computations of the power function, uniformly and independently of ς.

Example 3.2.8. We illustrate some matrices that pass the validity check $\vDash_{\varsigma} \chi_{\mathtt{pow}}$ and so by Lemma 3.2.6 encode a partial power function:

$$\begin{pmatrix} 2 & 2 & 2 \\ 0 & 1 & 2 \\ 1 & 2 & 4 \\ 1 & 1 & 2 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 2 & 2 & 2 \\ 2 & 1 & 0 \\ 4 & 2 & 1 \\ 2 & 3 & 3 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 2 & 2 & 2 & 2 \\ 0 & 2 & 1 & 0 \\ 1 & 4 & 2 & 1 \\ 2 & 3 & 1 & 1 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 3 & 2 & 3 & 2 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 3 & 2 \\ 1 & 1 & 1 & 2 \end{pmatrix}$$

Note how all of these matrices can be viewed as concrete implementations of the derivation-tree illustrated in Remark 3.2.2:

1. The leftmost matrix encodes the base case of a derivation that $2^2 = 4$ in column 1, the second step in column 2, and the third and final step of the derivation in column 3. The fourth entry in columns 2 and 3 contain a number which is a pointer to the previous step/column in the derivation; the fourth entry in column 1 also contains a number, but it is not used.
2. The columns do not need to appear in any particular order, so long as the pointers match up; the next matrix illustrates this (by putting the columns in the reverse order).
3. Columns can also be duplicated; the third matrix illustrates this.

4. The fourth (rightmost) matrix includes information from two computations; that $2^1 = 2$ and $3^1 = 3$.

Example 3.2.9. The converse implication for Lemma 3.2.6 does not hold: it is possible for $\varsigma(\mathtt{pow})$ to encode a partial power function yet $\neg(\vDash_\varsigma \chi_{\mathtt{pow}})$. Take for example

$$\varsigma(\mathtt{pow}) = \begin{pmatrix} 2 \\ 2 \\ 4 \\ 2 \end{pmatrix}$$

This one-column matrix encodes a partial power function because $2^2 = 4$, but it fails the validity predicate because the entry in the fourth row does not point to a column showing that $2^1 = 2$, so this fails the last line of $\mathsf{indStep}$ in Figure 3. We can think of it as corresponding to an *incomplete* derivation-tree:

$$\dfrac{????}{pow(2,2) = 4}\,\text{inductive step}$$

We consider another example:

$$\varsigma(\mathtt{pow}) = \begin{pmatrix} 2 & 2 & 2 \\ 2 & 1 & 0 \\ 4 & 2 & 1 \\ 2 & 3 & 0 \end{pmatrix}$$

This matrix encodes a partial power function because $2^2 = 4$ and $2^1 = 2$ and $2^0 = 1$. Furthermore, the entry in the fourth row of the first column *does* point to a column showing that $2^1 = 2$, that the fourth row of the second column does point to a column showing that $2^0 = 1$. However, this fails the range check just for a technical reason: the entry in the fourth row of the third column is 0, which is not in the range 1 to 3.

Remark 3.2.10. In *constructive logic*, something is only deemed 'true' when we have a concrete witness for its truth [29].[20]

The way our encoding of logic works, as illustrated in Examples 3.2.8 and 3.2.9, feels constructive in this sense. It might help such a reader to think of matrices as follows: the matrices are used to encode deduplicated derivation-trees. Columns correspond to nodes, and we include index pointers to link these nodes together.

[20] For example, in classical logic $P \Longrightarrow Q$ is equivalent to $\neg P \vee Q$. Constructively, $P \Longrightarrow Q$ is a concrete method for turning a witness to the truth of P, into a witness to the truth of Q.

Example 3.2.11. We take a moment to unpack the concrete polynomial that Figure 3 expresses.

Assume we have some interpretation ς and write $pow = \varsigma(\mathtt{pow})$. Recall from Definition 2.3.2(1) that this is a $4 \times n$ matrix for some $n \geq 1$. Mirroring the sugar in Figure 3, we write:

$$\begin{aligned} in_1 &= itplnt(pow_1) \\ in_2 &= itplnt(pow_2) \\ out &= itplnt(pow_3) \\ rec &= itplnt(pow_4) \end{aligned}$$

Assume that every entry in pow_4 is a valid index of a column in pow, so that pow passes the range check and

$$[\![0 < \mathtt{pow}_4 < len(\mathtt{pow}) + 1]\!]_\varsigma = 0$$

— meaning that this is 'true'.

Then $[\![\chi_{\mathtt{pow}}]\!]_\varsigma$ from Figure 3 expands as follows — where, as per Figure 2, $\mathsf{\wedge}$ maps to $+$, $\mathsf{\vee}$ maps to $*$, $\mathsf{\forall}$ maps to $\sum$, and $t = t'$ maps to $(t - t')^2$:

$$\begin{aligned} \chi_{\mathtt{pow}} = {} & 1 \leq \mathtt{pow}_4 \leq \mathsf{len}(\mathtt{pow}) \;\mathsf{\wedge} \\ & \mathsf{\forall}_{\mathtt{pow}} X.(\mathsf{baseCase}(X) \;\; \mathsf{\vee} \;\; \mathsf{indStep}(X)) \end{aligned}$$

$$\begin{aligned} [\![\chi_{\mathtt{pow}}]\!]_\varsigma = 0 \; + \; \textstyle\sum_{1 \leq x \leq n} \; \Big(& (in_1(x)^2 + (out(x) - 1)^2) \; * \\ & ((in_1(x) - in_1(rec(x)))^2 + \\ & (in_2(x) - (in_2(rec(x)) + 1))^2 + \\ & (out(x) - in_1(x) * out(rec(x)))^2) \Big) \end{aligned}$$

If this evaluates to zero, then pow really does encode a partial power function.[21]

Remark 3.2.12 (Analysis of polynomial degree). We can make some comments on the degree of the polynomial in Example 3.2.11:

1. Range checks correspond to constant numbers (0 if true, 1 if false). This makes range checks 'free', in the sense that they contribute nothing to the degree of the polynomial.
2. $\mathsf{\wedge}$ and $\mathsf{\forall}$ corresponds to $+$. This means that conjunction and universal quantification are 'free', in the sense that they do not increase the degree of a polynomial beyond the degrees of the components.
3. $\mathsf{\vee}$ corresponds to $*$. This is linearly expensive, in the sense that the degree is the sum of the degrees of the components.

[21] We consider how to check this using the Schwartz-Zippel lemma later, in Lemma 5.1.2.

4. Polynomial instantiation, as triggered e.g. by the recursive call $out(rec(x))$ above, is multiplicatively expensive: it multiplies degrees of the polynomials involved. For example, if we set $P(X) = X^2$ and $Q(X) = X^3$ then $P(Q)(X) = (X^3)^2 = X^6$.
5. The length of the matrix is linearly expensive in the degree of the interpolating polynomial: as per Definition 2.1.4(2), to interpolate a vector of length n requires a polynomial of degree at most n-1.
6. The implementation is optimised for simplicity, not efficiency. We will come to that shortly.

For brevity write $n = len(pow)$; so this is the number of columns in pow and is (one plus) the degree of the interpolants in_1, in_2, out, and rec. Checking the polynomial, we see that its degree is $O(n^4)$, coming from the component $(in_1 * out(rec))^2$ on the far right-hand side, and in particular from the polynomial instantiation $out(rec)$. Now for pow as specified in Definition 3.2.1, n (the number of columns) is equal to b, so the *degree order* of this specification, by which we mean the degree of the polynomials it produces, is $O(b^4)$. We improve this bound in Remark 3.2.14.

Remark 3.2.13. Just because a polynomial has high degree does not necessarily make it complicated nor expensive to compute. For instance, X^{100} had degree 100, but 10^{100} is not particularly expensive to compute, and the polynomial has only one root up to multiplicities, so in the logical sense mentioned in Remark 2.3.6(6) it is equivalent to X^2.[22]

Degree order as a complexity metric is its own thing.

Remark 3.2.14. Continuing Remark 3.2.12, we can also try to keep n small, which amounts to minimising the number of calls to pow. As is well-known, a more efficient specification of a^b is this:

$$\begin{aligned} pow(a, 0) &= 1 \\ pow(a, 2 * b) &= pow(a, b) * pow(a, b) \\ pow(a, 2 * b + 1) &= a * pow(a, b) * pow(a, b) \end{aligned}$$

Logically, this leads to smaller derivations, and algorithmically it yields a runtime that is logarithmic in b rather than linear (parametrically over the complexity of our multiplication algorithm).

[22] It might be possible to efficiently 'renormalise' our polynomials to eliminate repeated roots, using formal derivatives and polynomial long division. Indeed, what really matters to us is not the degree of the polynomial itself, but how many distinct rational roots it has. But for now, just looking a the degree of the polynomial will suffice.

$$\begin{aligned}\chi_{\mathtt{sqrt}} = \quad & 0 \leq \mathtt{sqrt}_2 \wedge \\ & \forall_{\mathtt{sqrt}} X.\mathtt{sqrt}_2(X) * \mathtt{sqrt}_2(X) = \mathtt{sqrt}_1(X)\end{aligned}$$

Figure 4: Expressing a positive square root function $\sqrt{a}$ (Definitions 3.3.1 and 3.3.2)

We could translate this into our logic as we did in Definition 3.2.4, and this would yield an implementation of degree order $O(log(b)^4)$. More on efficiency in Remark 3.4.6(2) and Subsection 6.2.

3.3 Expressing a square root function

The square root $\sqrt{a}$ is a simple example which nicely illustrates that validity predicates are *assertions*, not computations. In particular, we can verify that a matrix represents a partial square root function; just square the claimed root (see Definition 3.3.3). We do not have to design a square root algorithm ourselves. This is an elementary point, which may be familiar to readers familiar with succinct proofs, but we spell out the details:

Definition 3.3.1. Recall that for $a \in \mathbb{Q}_{\geq 0}$, the **(positive) square root function** $\sqrt{a}$ is specified by

$$(\sqrt{a})^2 = a \wedge \sqrt{a} \geq 0.$$

Definition 3.3.2. Assume a matrix variable symbol $\mathtt{sqrt}$ with arity $\mathit{arity}(\mathtt{sqrt}) = 2$. In Figure 4 we use our logic to express a validity predicate $\chi_{\mathtt{sqrt}}$ that $\mathtt{sqrt}$ encodes a partial positive square root function, such that:

1. Row 1 (values of $\mathtt{sqrt}_1(X)$) stores input values a.
2. Row 2 (values of $\mathtt{sqrt}_2(X)$) stores output values $\sqrt{a} \geq 0$.

Definition 3.3.3. Suppose $\mathit{ln} \in \mathbb{N}_{\geq 1}$. Say that a $2 \times \mathit{ln}$ matrix M **encodes a partial positive square root function** when

$$\forall x \in 1..\mathit{ln}.\ (0 \leq M(2,x) \wedge M(2,x)^2 = M(1,x)).$$

Lemma 3.3.4. *Suppose ς is an interpretation (Definition 2.3.2(1)). Then the following are equivalent:*

1. $\vDash_{\varsigma} \chi_{\mathtt{sqrt}}$.
2. *$\varsigma(\mathtt{sqrt})$ encodes a partial positive square root function, as per Definition 3.3.3.*

Proof. We examine $\chi_{\mathtt{sqrt}}$ in Figure 4 and see that it correctly encodes the definition in Definition 3.5.1. □

Remark 3.3.5. There are no pointers involved in Lemma 3.3.4, so we get a logical equivalence. Contrast with Lemma 3.2.6 and Examples 3.2.8 and 3.2.9 for `pow`.

Remark 3.3.6. The point made by Definition 3.3.3, Figure 4, and Lemma 3.3.4 is not that positive square roots are particularly interesting.[23] What matters is that actually *computing* $\sqrt{a}$ is done by whoever provides ς (i.e. the 'prover'). Intuitively, our logic is about *verification*. Thus, we assume a ς is provided, and we verify that it satisfies whatever validity predicates have been claimed of it.

3.4 Expressing Fibonacci

The Fibonacci function is a classic example in a certain kind of undergraduate course. We consider it in our system. It is a simple example, though nontrivial, and as we discuss in Remark 3.4.6 it raises some useful points:

Definition 3.4.1. Recall that the Fibonacci sequence $\mathit{fib}(a)$ for $a \in \mathbb{N}_{\geq 0}$ is defined by:

$$\begin{array}{lr} \text{base case 0} & \mathit{fib}(0) = 0 \\ \text{base case 1} & \mathit{fib}(1) = 1 \\ \text{inductive step} & \mathit{fib}(a+2) = \mathit{fib}(a) + \mathit{fib}(a{+}1) \end{array}$$

Definition 3.4.2. Assume a matrix variable symbol `fib` with arity $\mathit{arity}(\texttt{fib}) = 3$. We can think of this as a three-row matrix.

We will use our logic to express a validity predicate $\chi_{\texttt{fib}}$ that `fib` encodes the Fibonacci function, such that:

1. Row 1 (values of $\texttt{fib}_1(X)$) stores input values a.
2. Row 2 (values of $\texttt{fib}_2(X)$) stores output values $\mathit{fib}(a)$.
3. Row 3 (values of $\texttt{fib}_3(X)$) stores an index of a column in `fib`, corresponding to the first recursive call to fib in clause 3 of Definition 3.4.1.
4. Row 4 (values of $\texttt{fib}_3(X)$) stores an index of a column in `fib`, corresponding to the second recursive call to fib in clause 3 of Definition 3.4.1.

Definition 3.4.3. Suppose $\mathit{ln} \in \mathbb{N}_{\geq 1}$. Say that a $4 \times \mathit{ln}$ matrix M **encodes a partial Fibonacci function** when

$$\forall x \in 1..\mathit{ln}.\ M(2, x) = \mathit{fib}(M(1, x)).$$

Definition 3.4.4. In Figure 5 we

1. define syntactic sugar,

[23] They are, but not in ways that matter to us here.

$$\begin{array}{llll} \mathtt{fib}_1(t) \text{ sugars to } \mathsf{in}(t) & & \mathtt{fib}_2(t) \text{ sugars to } \mathsf{out}(t) \\ \mathtt{fib}_3(t) \text{ sugars to } \mathsf{rec}_1(t) & & \mathtt{fib}_4(t) \text{ sugars to } \mathsf{rec}_2(t) \end{array}$$

$$\begin{array}{rl} \mathsf{baseCase0}(X) = & (\mathsf{in}(X) = 0) \land (\mathsf{out}(X) = 1) \\ \mathsf{baseCase1}(X) = & (\mathsf{in}(X) = 1) \land (\mathsf{out}(X) = 1) \\ \mathsf{indStep}(X) = & \mathsf{in}(X) = \mathsf{in}(\mathsf{rec}_1(X)) + 2 \quad \land \\ & \mathsf{in}(\mathsf{rec}_2(X)) = \mathsf{in}(\mathsf{rec}_1(X)) + 1 \quad \land \\ & \mathsf{out}(X) = \mathsf{out}(\mathsf{rec}_1(X)) + \mathsf{out}(\mathsf{rec}_2(X)) \\ \chi_{\mathtt{fib}} = & 1 \leq \mathsf{rec}_1 \leq \mathsf{len}(\mathtt{fib}) \ \land \ 1 \leq \mathsf{rec}_2 \leq \mathsf{len}(\mathtt{fib}) \land \\ & \forall_{\mathtt{fib}} X.(\mathsf{baseCase0}(X) \ \lor \ \mathsf{baseCase1}(X) \ \lor \ \mathsf{indStep}(X)) \end{array}$$

Figure 5: Expressing the Fibonacci function *fib*(a) (Definitions 3.4.1 and 3.4.4)

2. use it to express the clauses from Definition 3.4.1, and
3. write a validity predicate $\chi_{\mathtt{fib}}$ to express that $\mathtt{fib}$ encodes a partial Fibonacci function.

LEMMA 3.4.5. *Suppose ς is an interpretation (Definition 2.3.2(1)). Then*

1. *if* $\models_\varsigma \chi_{\mathtt{fib}}$,
2. *then* $\varsigma(\mathtt{fib})$ *encodes a partial Fibonacci function, as per Definition 3.4.3.*

Proof. We examine the clauses in Figure 5 and see that they correctly encode the definition in Definition 3.4.1. The range checks $1 \leq \mathtt{fib}_3 \leq \mathsf{len}(\mathtt{fib})$ and $1 \leq \mathtt{fib}_4 \leq \mathsf{len}(\mathtt{fib})$ are there to exclude out-of-bounds pointer errors. □

REMARK 3.4.6. There are a few interesting observations to make about Definition 3.4.1 and Figure 5.

1. It is not the case that $\varsigma(\mathtt{fib})$ has to contain the entries of the Fibonacci sequence *in order*. Each column has to contain a value a, the number *fib*(a), and (if $a \notin \{0, 1\}$) it contains pointers to two columns representing 'recursive calls'. Where those recursive calls are located inside $\varsigma(\mathtt{fib})$ does not really matter.
2. We put 'recursive calls' in scare quotes because these are not recursive calls to a computation.[24] Definition 3.4.1 is famous in undergraduate courses because, if translated directly to code, it is very inefficient; each call to *fib* creates two sub-calls, thus giving the program exponential complexity.

[24] Well ... it depends on your point of view. They are recursive calls, but in a computation that is a verification of another computation that we assume has already been made.

> A standard solution is to memoise the computation; store the results of previous calls and replace them with lookups if the function is called again on the same input.
> Figure 5 is already *naturally memoised*. Indeed, it is nothing more than a lookup-table. As we noted in Remark 3.3.6, our logic is *verifying* (not computing!) a ς.[25]
> So the complexity of the program implicit in Definition 3.4.1 is actually linear (not exponential), because our semantics is in some sense inherently memoised.

Remark 3.4.7 (Fast Fibonacci and algebraic extensions of $\mathbb{Q}$). Optimised methods for computing Fibonacci numbers exist. See for instance a well-written online overview [22] and an algorithmic treatment [7] which uses the Binet formula — a paper on this topic exists [10], however, the webpage includes significant additional optimisations.

This is interesting because The Binet formula uses $\sqrt{5}$, which is irrational and so is not in $\mathbb{Q}$, which is the field we use in this paper. So what this example illustrates is that $\mathbb{Q}$ may not be enough: we may wish to talk about roots of polynomials that are not rationals, such as $x^2 = 5$, and if so, we might need to consider algebraic extensions, such as $\mathbb{Q}(\sqrt{5})$. Technically, this should not be a problem.

3.5 Expressing Ackermann's function

Ackermann's function is another classic example. We check how it comes out in our framework:

Definition 3.5.1. Recall Ackermann's function $Ack(a, b)$ for $a, b \in \mathbb{N}_{\geq 0}$, defined by:

$$\begin{array}{lrcl} \text{base case} & Ack(0, b) & = & b + 1 \\ \text{inductive step 0} & Ack(a + 1, 0) & = & Ack(a, 1) \\ \text{inductive step 1} & Ack(a + 1, b + 1) & = & Ack(a, Ack(a + 1, b)) \end{array}$$

Definition 3.5.2. Assume a matrix variable symbol $\mathtt{Ack}$ with arity $arity(\mathtt{Ack}) = 5$. We can think of this as a five-row matrix.

We will use our logic to express a validity predicate $\chi_{\mathtt{Ack}}$ that $\mathtt{Ack}$ encodes a (partial) Ackermann function, such that:

1. Row 1 (values of $\mathtt{Ack}_1(X)$) stores input values a.

[25] An interpretation ς could contain repeated columns, corresponding to repeated calls to the same function on the same inputs, but this would be an inefficient prover; the verifier is still efficient on that (needlessly complex) input. Example 3.2.8 includes some basic examples with duplicated columns; they could be removed, but beyond padding out the matrix they do no harm.

$$\begin{array}{llllll}
\mathtt{Ack}_1(t) & \text{sugars to} & \mathsf{in}_1(t) & \mathtt{Ack}_2(t) & \text{sugars to} & \mathsf{in}_2(t)\\
\mathtt{Ack}_3(t) & \text{sugars to} & \mathsf{out}(t) & \mathtt{Ack}_4(t) & \text{sugars to} & \mathsf{rec}_1(t)\\
\mathtt{Ack}_5(t) & \text{sugars to} & \mathsf{rec}_2(t) & & &
\end{array}$$

$$\begin{array}{rl}
\mathsf{baseCase}(X) = & (\mathsf{in}_1(X) = 0) \land (\mathsf{out}(X) = \mathsf{in}_2(X) + 1)\\
\mathsf{indStep0}(X) = & \begin{array}{rcll} \mathsf{in}_1(X) &=& \mathsf{in}_1(\mathsf{rec}_1(X)) + 1 & \land\\ \mathsf{in}_2(X) &=& 0 & \land\\ \mathsf{out}(X) &=& \mathsf{out}(\mathsf{rec}_1(X)) & \end{array}\\
\mathsf{indStep1}(X) = & \begin{array}{rcll} \mathsf{in}_1(X) &=& \mathsf{in}_1(\mathsf{rec}_2(X)) + 1 & \land\\ \mathsf{in}_2(X) &=& \mathsf{in}_2(\mathsf{rec}_1(X)) + 1 & \land\\ \mathsf{out}(X) &=& \mathsf{out}(\mathsf{rec}_2(X)) & \land\\ \mathsf{in}_1(\mathsf{rec}_1(X)) &=& \mathsf{in}_1(\mathsf{rec}_2(X)) + 1 & \land\\ \mathsf{in}_2(\mathsf{rec}_2(X)) &=& \mathsf{out}(\mathsf{rec}_1(X)) & \end{array}\\
\chi_{\mathtt{Ack}} = & 1 \leq \mathsf{rec}_1 \leq \mathsf{len}(\mathtt{Ack}) \ \land \ 1 \leq \mathsf{rec}_2 \leq \mathsf{len}(\mathtt{Ack}) \ \land\\
& \quad \forall_{\mathtt{Ack}} X.(\mathsf{baseCase}(X) \ \lor \ \mathsf{indStep0}(X) \ \lor \ \mathsf{indStep1}(X))
\end{array}$$

Figure 6: Expressing Ackermann's function $Ack(a, b)$ (Definitions 3.5.1 and 3.5.4)

2. Row 2 (values of $\mathtt{Ack}_2(X)$) stores input values b.
3. Row 3 (values of $\mathtt{Ack}_3(X)$) stores output values $Ack(a, b)$.
4. Row 4 (values of $\mathtt{Ack}_4(X)$) stores an index of a column in the $\mathtt{Ack}$ matrix, corresponding to the first recursive call to Ack in clauses 2 and 3 of Definition 3.5.1.
5. Row 5 (the values of $\mathtt{Ack}_5(X)$) stores an index of another column in the $\mathtt{Ack}$ matrix, corresponding to the second recursive call to Ack in clause 3 of Definition 3.5.1.

DEFINITION 3.5.3. Suppose $ln \in \mathbb{N}_{\geq 1}$. Say that a $5 \times ln$ matrix M **encodes a partial Ackermann's function** when

$$\forall x \in 1..ln.M(3, x) = Ack(M(1, x), M(2, x)).$$

DEFINITION 3.5.4. In Figure 6 we

1. define syntactic sugar,
2. use it to express the clauses from Definition 3.5.1, and
3. write a validity predicate $\chi_{\mathtt{Ack}}$ to express that $\mathtt{Ack}$ encodes a partial Ackermann function.

LEMMA 3.5.5. *Suppose ς is an interpretation (Definition 2.3.2(1)). Then*

1. *if* $\vDash_\varsigma \chi_{\texttt{Ack}}$
2. *then* $\varsigma(\texttt{Ack})$ *encodes a partial Ackermann function, as per Definition 3.5.3.*

Proof. We examine the three clauses in Figure 6 and see that they correctly encode the definition in Definition 3.5.1. The range checks $1 \leq \texttt{Ack}_4 \leq \mathsf{len}(\texttt{Ack})$ and $1 \leq \texttt{Ack}_5 \leq \mathsf{len}(\texttt{Ack})$ are there to exclude out-of-bounds pointer errors. □

3.6 Bitwise representation

Unsigned 64-bit integers — the familiar `uint64` datatype — are a standard datatype of practical interest in computing. Bitwise integers are the native notion of number in many programming languages.[26] For example, C uses 64-bit or 32-bit integers, and Ethereum uses 256-bit integers. It is straightforward to represent them:

Definition 3.6.1. We express that t is equal to zero or one (i.e. that t represents a single bit of binary data) using a little syntactic sugar:

$$\mathsf{isBinary}(t) \quad \text{is sugar for} \quad t = 0 \lor t = 1.$$

Unpacking Figure 2, $\mathsf{isBinary}(t)$ corresponds to a polynomial $t^2 * (t-1)^2$.

Definition 3.6.2. Assume a matrix variable symbol `uint64` with $\mathit{arity}(\texttt{uint64}) = 65$, which we can think of as representing a 65-row matrix.

1. Row 1 should store a whole number $0 \leq n < 2^{64}$, and
2. rows 2 to 65 should store its binary expansion (with smallest bit first).

Validity of `uint64` is simply

$$\begin{aligned}\chi_{\texttt{uint64}} \;=\; & \forall_{\texttt{uint64}} X.(\texttt{uint64}_1(X) = \textstyle\sum_{i\in 0..63} 2^i * \texttt{uint64}_{i+2}(X) \land \\ & \qquad\qquad \mathsf{isBinary}(\texttt{uint64}_2(X)) \land \ldots \land \mathsf{isBinary}(\texttt{uint64}_{65}(X)))\end{aligned}$$

This just asserts that for each column in `uint64`, elements 2 to 65 of that vector do indeed store bits representing the binary representation of element 1.

Lemma 3.6.3. *Suppose ς is an interpretation (Definition 2.3.2(1)). Then the following are equivalent:*

1. $\vDash_\varsigma \chi_{\texttt{uint64}}$ *for* $\chi_{\texttt{uint64}}$ *as defined in Definition 3.6.2.*
2. $\varsigma(\texttt{uint64})$ *encodes a list of bitwise expansions, by which we mean precisely that*

$$\forall 1 \leq x \leq \mathit{len}(\varsigma(\texttt{uint64})).\ \varsigma(\texttt{uint64})_1(x) = \sum_{i\in 0..63} 2^i * \varsigma(\texttt{uint64})_{i+2}(x).$$

[26] Indeed, in the underlying silicon — in the low-level instructions that computer chips actually execute — 64-bit integers are typically the native notion of *data*.

Proof. We examine $\chi_{\texttt{uint64}}$ in Definition 3.6.2 and using Theorem 2.4.4 and basic arithmetic see that this is precisely what it encodes. Note that there are no range checks, pointers, or recursive calls involved; this is just straightforward arithmetic. Note also that 2^i is in the predicate above is just a number (not a power of X), so has degree zero in X. □

REMARK 3.6.4. We get a decimal representation just by generalising isBinary to isDecimal:

$$\mathsf{isDecimal}(t) \quad \text{is sugar for} \quad t=0 \vee t=1 \vee \ldots \vee t=9.$$

and use a matrix variable symbol $\texttt{uint64}^{\text{base }10}$ with validity predicate

$$\begin{aligned}\mathsf{clause}(X) \;&=\; (\texttt{uint64}^{\text{base }10}_1(X) = \textstyle\sum_{i\in 0..63} 10^i * \texttt{uint64}^{\text{base }10}_{i+2}(X) \wedge \\ &\quad \mathsf{isDecimal}(\texttt{uint64}^{\text{base }10}_2(X)) \wedge \ldots \wedge \mathsf{isDecimal}(\texttt{uint64}^{\text{base }10}_{65}(X))).\end{aligned}$$

We can also nest representations and for example represent a number in the range 0 to $(2^{64})^{64}$ by taking its 64-digit expansion base 2^{64}.

3.7 Range check

REMARK 3.7.1. The clause for the range check $<$ in Figure 2 is simple — just check that the relevant row in the matrix satisfies the required bound, and return 0 if it is satisfied and 1 if it is not.

This in itself does not increase expressivity: the range check can be compiled down to the language without $<$ (at some cost); see Remark 5.1.3(4). It could also be computed direct on silicon, as we note in Subsection 5.2.[27]

In particular, we see from our examples — such as those in Figures 3 and 6 — that we typically want to prove range checks that have an *upper and lower* bound, having this form:

$$1 \leq \mathsf{C}_i \leq \mathsf{len}(\mathsf{C}).$$

We can translate predicates of this form into ones that do not mention $<$, as follows:

DEFINITION 3.7.2. Suppose $l, r \in \mathbb{Z}$ and $l \leq r$. Then define a predicate $\mathsf{range}_{l,r}(X)$ by

$$\mathsf{range}_{l,r}(X) = (X = l) \vee (X = l{+}1) \vee \ldots \vee (X = r{-}1) \vee (X = r).$$

REMARK 3.7.3. Range checks are special cases of *set membership* assertions. Succinct (and zero knowledge) set membership schemes are a field of research in their

[27] This would leak a bit of information, but if all we are leaking is pointers then that might not matter, depending on the use case.

own right ([5] contains a recent survey). So: the schema in Definition 3.7.2 may not be optimal and in a practical implementation of these ideas we might wish to improve it further, but for proof-of-concept Definition 3.7.2 will serve our purposes, and it is simple. Lemma 3.7.4 says that it does what we need it to do:

Lemma 3.7.4. *Suppose ς is an interpretation. Then the following are equivalent:*

1. $\vDash_\varsigma 1 \leq \mathsf{C}_i \leq \mathsf{len}(\mathsf{C})$.
2. $\vDash_\varsigma \forall X_{\mathsf{C}}.\mathsf{range}_{1,\mathsf{len}(\mathsf{C})}(\mathsf{C}_i(X))$.

Proof. $\varsigma(\mathsf{C})$ is by Definition 2.3.2(1) an integer matrix, and the polynomial $[\![\mathsf{range}_{l,r}]\!]$ by construction in Definition 3.7.2 and Figure 2 has zeroes precisely at the integers between l and r inclusive, which by Notation 2.2.7 and Figure 2 is precisely what $1 \leq \mathsf{C}_i \leq \mathsf{len}(\mathsf{C})$ checks. □

Remark 3.7.5. If $\varsigma(\mathsf{C})$ is a very long matrix then $[\![\mathsf{range}_{1,\mathsf{len}(\mathsf{C})}(\mathsf{C}_i(X))]\!]$ might be a large (high-degree) polynomial. But, in the context of the *succinct proofs* (see discussion in Subsection 5.1) this may be is a worthwhile trade-off.

Remark 3.7.6. We can also exploit `uint64` from Subsection 3.6 to do a simple range check. Assume a matrix variable symbol `range64` with $\mathit{arity}(\mathtt{range64}) = 2$, and with validity condition

$$\mathsf{clause}(X) \;=\; 1{\leq}\mathtt{range64}_2{\leq}\mathsf{len}(\mathtt{uint64}) \wedge (\mathtt{range64}_1(X){=}\mathtt{uint64}_1(\mathtt{range64}_2(X))).$$

Intuitively, we can think of the above as follows:

1. $\mathtt{range64}_1(X)$ stores some number n.
2. $\mathtt{range64}_2(X)$ stores a pointer to a proof in `uint64` that n has a 64-bit representation.
3. The pointers in $\mathtt{range64}_2$ must be in-bound for the addressable range of `uint64`; i.e. they must be numbers between 1 and $\mathsf{len}(\mathtt{uint64})$.

The above can only happen if $0 \leq n < 2^{64}$, so this all amounts to doing range checks.

3.8 Iteration

Definition 3.8.1. Recall that if $f : \mathbb{Z} \to \mathbb{Z}$ is a function and $a \in \mathbb{N}_{\geq 0}$, then the **$a$-fold iteration** $\mathit{iter}_f(a) : \mathbb{Z} \to \mathbb{Z}$ of f is the function that inputs $b \in \mathbb{Z}$ and returns $(\overbrace{f \circ \cdots \circ f}^{a \text{ times}})(b)$.

This can be defined by:

$$\begin{array}{ll} \text{base case} & \mathit{iter}_f(0)(b) = b \\ \text{inductive step} & \mathit{iter}_f(a+1)(b) = f(\mathit{iter}_f(a)(b)) \end{array}$$

$$\begin{array}{lll@{\qquad}lll}
\mathtt{iter}_{f,1}(t) & \text{sugars to} & \mathsf{in}_1(t) & \mathtt{iter}_{f,2}(t) & \text{sugars to} & \mathsf{in}_2(t) \\
\mathtt{iter}_{f,3}(t) & \text{sugars to} & \mathsf{out}(t) & \mathtt{iter}_{f,4}(t) & \text{sugars to} & \mathsf{rec}(t) \\
\mathtt{iter}_{f,5}(t) & \text{sugars to} & \mathsf{callf}(t) & & &
\end{array}$$

$$\begin{array}{rl}
\mathsf{baseCase}(X) = & (\mathsf{in}_1(X) = 0) \wedge (\mathsf{out}(X) = \mathsf{in}_2(X)) \\
\mathsf{indStep}(X) = & \begin{array}{rl} \mathsf{in}_1(X) = & \mathsf{in}_1(\mathsf{rec}(X)) + 1 \wedge \\ \mathsf{in}_2(X) = & \mathsf{in}_2(\mathsf{rec}(X)) \wedge \\ \mathsf{out}(X) = & \mathtt{f}_2(\mathsf{callf}(X)) \wedge \\ \mathtt{f}_1(\mathsf{callf}(X)) = & \mathsf{out}(\mathsf{rec}(X)) \end{array} \\
\chi_{\mathtt{iter}_f} = & 1 \leq \mathsf{rec} \leq \mathsf{len}(\mathtt{iter}) \ \wedge \ 1 \leq \mathsf{callf} \leq \mathsf{len}(\mathtt{f}) \ \wedge \\
& \quad \forall_{\mathtt{iter}_f} X.(\mathsf{baseCase}(X) \ \vee \ \mathsf{indStep}(X))
\end{array}$$

Figure 7: Expressing $iter_f(a)(b)$; a-fold application of f to b (Definitions 3.8.1 and 3.8.2)

DEFINITION 3.8.2. Assume a matrix variable symbol $\mathtt{f}$, with arity $arity(\mathtt{f}) = 2$, along with a validity predicate $\chi_{\mathtt{f}}$ which expresses that $\mathtt{f}_3(X) = f(\mathtt{f}_1(X), \mathtt{f}_2(X))$. We can think of this intuitively as a matrix of columns all having the form $(a, b, f(a, b))$, for various values of a and b.

There is no obstacle to having more rows, as we did for instance in Figure 6; we take three just for simplicity.

We now assume a binary propositional constant $\mathtt{iter}_f$ with arity $arity(\mathtt{iter}_f) = 5$. In Figure 7 we use our logic to express that $\mathtt{iter}_f$ encodes an a-fold iteration of f on an input value b, such that:

1. Row 1 (values of $\mathtt{iter}_{f,1}(X)$) stores the number of iterations a.
2. Row 2 (values of $\mathtt{iter}_{f,2}(X)$) stores input values b.
3. Row 3 (values of $\mathtt{iter}_{f,3}(X)$) stores output values $iter_f(a)(b)$.
4. Row 4 (values of $\mathtt{iter}_{f,4}(X)$) stores pointers to recursive calls to $\mathtt{iter}_f$, as required in clause 2 of Definition 3.8.1 above.
5. Row 5 (values of $\mathtt{iter}_f(5, X)$) stores pointers to calls to $\mathtt{f}$, corresponding to the call to f in clause 2 of Definition 3.8.1 above.

DEFINITION 3.8.3. Suppose $f : \mathbb{Z} \to \mathbb{Z}$ is a unary function and suppose $ln \in \mathbb{N}_{\geq 1}$. Say that a $5 \times ln$ matrix M **encodes a partial iteration of** f when

$$\forall x \in 1..ln.\ M(3, x) = f^{M(1,x)}(M(2, x)).$$

LEMMA 3.8.4. *Suppose ς is an interpretation (Definition 2.3.2(1)). Then*

1. *if* $\vDash_{\varsigma}$ *iterfValid*
2. *then* $\varsigma(\mathsf{iterf})$ *encodes a partial iteration of* f*, as per Definition 3.8.3.*

Proof. We examine the clauses in Figure 7 and using Theorem 2.4.4 see that they do indeed encode the definition in Definition 3.8.1. The bound checks $1 \leq \mathtt{iter}_4 \leq \mathsf{len}(\mathtt{iter})$ and $1 \leq \mathtt{iter}_5 \leq \mathsf{len}(\mathtt{f})$ exclude out-of-bounds pointer errors. □

Remark 3.8.5. We see no inherent barrier to mutual inductive definitions. We just declare several matrix variable symbols and engineer pointers between them for the (mutual) inductive calls. It may be necessary to add a counter to prevent looping; see Subsection 3.9.

3.9 SK combinators

Remark 3.9.1. The SK combinator system forms a Turing-complete model of computation; see [6, 15] or [4, Chapter 7]).[28] Combinators are also a quite common compile target for modern high-level programming languages.[29] So to specify combinators is interesting in itself, and also it shows how our logic can specify the valid execution traces of a realistic and relevant combinatory virtual machine.

Definition 3.9.2.

1. **Combinator expressions** are a formal syntax defined by:

$$t ::= S \mid K \mid tt.$$

So a combinator is either an S, a K, or an **application** of a combinator term to a combinator term.

2. An evaluation relation for combinators is as follows:

$$\frac{}{(Ks)t \overset{*}{\to} s}\,(\mathbf{Kred}) \qquad \frac{}{((Ss)t)u \overset{*}{\to} (st)(su)}\,(\mathbf{Sred})$$

$$\frac{}{t \overset{*}{\to} t}\,(\mathbf{Id}) \qquad \frac{s \overset{*}{\to} s' \quad t \overset{*}{\to} t'}{st \overset{*}{\to} s't'}\,(\mathbf{Par}) \qquad \frac{s \overset{*}{\to} t \quad t \overset{*}{\to} u}{s \overset{*}{\to} u}\,(\mathbf{Tran})$$

The ‘red’ in (**Kred**) and (**Sred**) stands for ‘reduce’; intuitively these represent a single computation step. (**Id**) is the identity (no computation); (**Par**)

[28] A quite clear treatment is available online [23]. A compilation of λ-terms to SK combinator terms is at `https://en.wikipedia.org/wiki/Combinatory_logic#Conversion_of_a_lambda_term_to_an_equivalent_combinatorial_term`

[29] One nice recent example is *Nock*: `https://nock.is/intro.html`.

is parallel computation; and (**Tran**) (for *transitivity*) chains computations sequentially. The relation $\overset{*}{\to}$ is intuitively a multistep computation relation. When $t \overset{*}{\to} t'$ holds, we may say that t' is a (possibly trivial) **reduct** of t.

EXAMPLE 3.9.3. We give a very simple example of a $\overset{*}{\to}$ derivation, which is valid for t any combinator term:

$$\dfrac{\dfrac{\dfrac{}{t \overset{*}{\to} t}\,(\mathbf{Id})}{(Kt)(Kt) \overset{*}{\to} t}\,(\mathbf{Kred})}{(SKKt) \overset{*}{\to} t}\,(\mathbf{Sred})$$

This example shows that SKK behaves as an identity element, i.e. it codes the identity function that inputs a value t and outputs that same value t.

The computation above is simple, of course, but it is nontrivial, and using (**Par**) and (**Tran**) we can parallelise and chain this and other computations. Larger examples include Boolean logic, arithmetic, and fixpoint operators (like the famous Y combinator). So the example above, while small, is canonical.

We now consider how to encode $\overset{*}{\to}$ in our logic. We use a trick to make our encoding more compact:

DEFINITION 3.9.4.

1. The **Cantor pairing function** is a well-known way of bijecting $\mathbb{N}_{\geq 0} \times \mathbb{N}_{\geq 0}$ with $\mathbb{N}_{\geq 0}$. It has the characteristic that it can be expressed as a polynomial over $\mathbb{Q}$ (indeed, this is the only quadratic polynomial pairing function) [21]:[30]

$$\langle x, y \rangle' = \frac{(x+y)*(x+y+1)}{2} + x = \frac{x^2 + 3*x + 2*x*y + y + y^2}{2}. \qquad (1)$$

2. The pairing function $\langle x, y \rangle'$ above is not quite what we need, because we want to reserve 0 to represent the combinator S and 1 to represent the combinator K. Accordingly, we adjust it with a two-element offset:

$$\langle x, y \rangle = \langle x, y \rangle' + 2.$$

[30] Clear and accessible accounts of the Cantor pairing function are also on Wikipedia at `https://en.wikipedia.org/wiki/Pairing_function#Cantor_pairing_function` and `https://en.wikipedia.org/wiki/Fueter%E2%80%93P%C3%B3lya_theorem`.

3. We then define a *Gödel encoding*[31] from the syntax of combinator expressions (Definition 3.9.2(1)) to $\mathbb{N}$, which is an injection defined by:

$$gdl(S) = 0 \quad gdl(K) = 1 \quad gdl(tt') = \langle gdl(t), gdl(t')\rangle.$$

   It is routine to prove that this injection is actually a bijection between combinator expressions and the nonnegative natural numbers $\mathbb{N}_{\geq 0}$.

Remark 3.9.5. In Equation (1) in Definition 3.9.4, we divide by 2. If we want a polynomial with coefficients over $\mathbb{Z}$ we could just as well use $\langle x, y\rangle' * 2$, and retain integer coefficients throughout. We would lose bijectivity of the pairing function, but in the context of this prototype that would be fine too.

Definition 3.9.6. Assume a matrix variable symbol `evl` with arity $arity(\texttt{evl}) = 8$, which we can think of as storing an 8-row matrix. In Figure 8, we

1. define syntactic sugar,
2. use it to express the clauses from Definition 3.9.2, and
3. write a validity predicate χ_{SK} to express that `evl` encodes a partial combinator reduction relation.

Remark 3.9.7. We read through the rows in `evl` one by one:

1. Row 1 represents the left-hand (unreduced) term.
2. Row 2 represents the right-hand (reduced) term.
3. Row 3 represents a subterm s of the term on the left-hand side.
4. Row 4 represents a subterm t of the term on the left-hand side.
5. Row 5 represents a subterm u of the term on the left-hand side (see rule (**Sred**)).
6. Rows 6 and 7 represent pointers to proof-obligations above the line (= inductive function-calls), as required in (**Par**) and (**Tran**).
7. Row 8 represents an abstract inductive quantity which increments with each derivation step. This prevents a derivation with cycles, and thus ensures that the underlying derivation is acyclic.[32]

Remark 3.9.8. Remark 3.9.7(7) describes an abstract counter to ensure acyclicity of the derivation trees represented by the matrices. The reader might wonder why we did not provision a similar abstract inductive quantity to avoid cycles in our matrices

[31] By which we mean an injective map from the syntax of a language to natural numbers. Such maps are so called because Gödel used one such in his 1931 paper on incompleteness results.

[32] Coinductive derivation systems (whose derivation-trees may be cyclic) do exist; but the evaluation relation for combinators is inductively defined.

$$\begin{array}{ll}
0 \text{ sugars to } \mathsf{S} & 1 \text{ sugars to } \mathsf{K} \\
\mathtt{evl}_1(t) \text{ sugars to } \mathsf{lhs}(t) & \mathtt{evl}_2(t) \text{ sugars to } \mathsf{rhs}(t) \\
\mathtt{evl}_3(t) \text{ sugars to } \mathsf{s}(t) & \mathtt{evl}_4(t) \text{ sugars to } \mathsf{t}(t) \\
\mathtt{evl}_5(t) \text{ sugars to } \mathsf{u}(t) & \\
\mathtt{evl}_6(t) \text{ sugars to } \mathsf{rec1}(t) & \mathtt{evl}_7(t) \text{ sugars to } \mathsf{rec2}(t) \\
\mathtt{evl}_8(t) \text{ sugars to } \mathsf{count}(t) &
\end{array}$$

$$\begin{array}{rl}
\mathsf{Kred}(X) = & \mathsf{lhs}(X) = \langle\langle \mathsf{K}, \mathsf{rhs}(X)\rangle, \mathsf{t}(X)\rangle \\
\mathsf{Sred}(X) = & \mathsf{lhs}(X) = \langle\langle\langle \mathsf{S}, \mathsf{s}(X)\rangle, \mathsf{t}(X)\rangle, \mathsf{u}(X)\rangle \wedge \\
 & \mathsf{rhs}(X) = \langle\langle \mathsf{s}(X), \mathsf{t}(X)\rangle, \langle \mathsf{s}(X), \mathsf{u}(X)\rangle\rangle \\
\mathsf{Id}(X) = & \mathsf{lhs}(X) = \mathsf{rhs}(X) \\
\mathsf{Par}(X) = & \mathsf{lhs}(X) = \langle \mathsf{lhs}(\mathsf{rec1}(X)), \mathsf{lhs}(\mathsf{rec2}(X))\rangle \wedge \\
 & \mathsf{rhs}(X) = \langle \mathsf{rhs}(\mathsf{rec1}(X)), \mathsf{rhs}(\mathsf{rec2}(X))\rangle \wedge \\
 & \mathsf{count}(X) = \mathsf{count}(\mathsf{rec1}(X)){+}1 \wedge \\
 & \mathsf{count}(X) = \mathsf{count}(\mathsf{rec2}(X)){+}1 \\
\mathsf{Tran}(X) = & \mathsf{lhs}(X) = \mathsf{lhs}(\mathsf{rec1}(X)) \wedge \\
 & \mathsf{rhs}(X) = \mathsf{rhs}(\mathsf{rec2}(X)) \wedge \\
 & \mathsf{rhs}(\mathsf{rec1}(X)) = \mathsf{lhs}(\mathsf{rec2}(X)) \wedge \\
 & \mathsf{count}(X) = \mathsf{count}(\mathsf{rec1}(X)){+}1 \wedge \\
 & \mathsf{count}(X) = \mathsf{count}(\mathsf{rec2}(X)){+}1 \\
\chi_{SK} = & 1 \leq \mathsf{rec1} \leq \mathsf{len}(\mathtt{evl}) \wedge \\
 & 1 \leq \mathsf{rec2} \leq \mathsf{len}(\mathtt{evl}) \wedge \\
 & \forall_{\mathtt{evl}} X.(\mathsf{Kred}(X) \vee \mathsf{Sred}(X) \vee \\
 & \qquad \mathsf{Id}(X) \vee \mathsf{Par}(X) \vee \mathsf{Tran}(X))
\end{array}$$

Figure 8: Expressing evaluation of combinator expressions (Definitions 3.9.2 and 3.9.6)

for exponentiation, Fibonacci, Ackermann's function, or iteration. Iteration already has a counter inherent to its design, to count the number of times to apply the function. Exponentiation, Fibonacci, and Ackermann's are inherently inductive on their numerical inputs; an explicit count entry would be harmless, but for those functions it would be redundant. Combinators are a structurally more complex definition and we need an explicit measure of derivation depth.

Definition 3.9.9. Suppose $\mathit{ln} \in \mathbb{N}_{\geq 1}$. Say that an $8 \times \mathit{ln}$ matrix M **encodes a**

multistep combinator reduction relation when

$$\forall x \in 1..ln.\ gdl^{-1}(M(1,x)) \xrightarrow{*} gdl^{-1}(M(2,x)).$$

Above, the $\xrightarrow{*}$ relation is defined in Definition 3.9.2. The *gdl* function bijects combinator expressions with nonnegative natural numbers (Definition 3.9.4(3)).

Lemma 3.9.10. *Suppose ς is an interpretation (Definition 2.3.2(1)). Then*

1. *if* $\vDash_\varsigma \chi_{SK}$,
2. *then* $\varsigma(\texttt{evl})$ *encodes a partial combinator reduction relation, as per Definition 3.9.9.*

Proof. We examine the clauses in Figure 8 and using Theorem 2.4.4 see that they do indeed encode the definition in Definition 3.9.2. This is a little bit fiddly because of the Gödel encoding of the combinator syntax, and the fact that there are four derivation rules, and one of them has three subderivations, so it is a challenge to not make a typo or miss out a bit of a rule — but conceptually this is a straightforward translation from one logical syntax into another. □

Remark 3.9.11. We use the Cantor pairing function to build a simple Gödel encoding that packs a combinator term into a single number. We might prefer instead to store a combinator term as a list or as a tree of symbols. This is possible: it would add complexity to the representation above and thus to its validity predicate, but would not change the fundamental nature of what we are doing.

4 From range check to general recursion

4.1 Simple functions obtainable directly from the range check

Remark 4.1.1. A curious feature of our syntax in Figure 1 is the range check predicate $<$.

Our logic does not have a 'runtime' and a 'compile time' as such, but $<$ has the flavour of a static 'compile time' check, in the sense that (for example) given ς, $[\![t < \texttt{C}_i]\!]_\varsigma$ computes a number rather than a polynomial in X.[33]

In the examples above, we use range check to check for out-of-bounds pointer errors (see Examples 3.2.8 and 3.2.9 for concrete examples). But what is the range check really doing, and what is its full power? In this section we will examine this operator in more detail. We start by using the range check to specify some simple predicates and functions:

[33] So we can still say that our semantics is polynomial, even though $t < \texttt{C}_i$ does not *look* like a polynomial expression: it evaluates to a number $[\![t < \texttt{C}_i]\!]_\varsigma$, and a number is trivially a polynomial.

4.1.1 Strictly positive numbers pos

Definition 4.1.2 (Strictly positive numbers). Assume a matrix variable symbol $\mathtt{pos}$ with

$$\mathit{arity}(\mathtt{pos}) = 1 \quad \text{and validity predicate} \quad \chi_{\mathtt{pos}}(X) = (0 < \mathtt{pos}_1).$$

Lemma 4.1.3. *Suppose ς is an interpretation. Then the following are equivalent:*

1. $\vDash_{\varsigma} \chi_{\mathtt{pos}}$.
2. $\varsigma(\mathtt{pos})$ *is a 1-row matrix (i.e. a vector) of strictly positive integers in* $\mathbb{N}_{>0}$.

Proof. This is exactly what the clause for range check in Figure 2 expresses, for $0 < \mathtt{pos}_1$. □

Remark 4.1.4. We introduce a strict inequality $<$ rather than an inequality $\leq$ in Figure 1 so that we can express being *strictly positive* $\mathtt{pos}$. If we took $\leq$ as primitive, it would be harder to express $\mathtt{pos}$. This is because our logic does not include negation as primitive, so we cannot just express $\mathtt{pos}$ from a combination of $\leq$ and $\neq$.

4.1.2 The sign function *sign* and its corollary functions

Definition 4.1.5. We define functions

$$\begin{aligned} \mathit{sign} &: \mathbb{Q} \to \{\text{-}1, 0, 1\} \subseteq \mathbb{Q} \\ \mathit{isZero} &: \mathbb{Q} \to \{0, 1\} \\ \mathit{neg} &: \mathbb{Q} \to \{0, 1\} \\ \mathit{neq} &: (\mathbb{Q} \times \mathbb{Q}) \to \{0, 1\} \\ \mathit{leq} &: (\mathbb{Q} \times \mathbb{Q}) \to \{0, 1\} \end{aligned}$$

such that for $a, b \in \mathbb{Q}$:

$$\mathit{sign}(a) = \begin{cases} 0 & \text{if } a = 0 \\ 1 & \text{if } a > 0 \\ \text{-}1 & \text{if } a < 0 \end{cases}$$

$$\mathit{isZero}(a) = \begin{cases} 0 & \text{if } a = 0 \\ 1 & \text{if } a \neq 0 \end{cases} \qquad \mathit{neg}(a) = \begin{cases} 1 & \text{if } a = 0 \\ 0 & \text{if } a \neq 0 \end{cases}$$

$$\mathit{neq}(a, b) = \begin{cases} 0 & \text{if } a \neq b \\ 1 & \text{if } a = b \end{cases} \qquad \mathit{leq}(a, b) = \begin{cases} 0 & \text{if } a \leq b \\ 1 & \text{if } a \not\leq b \end{cases}$$

Remark 4.1.6. *sign* is the fundamental entity in Definition 4.1.5, and *isZero*, *neg*, *neq*, and *leq* follow from it, in the sense that we can derive everything using polynomial expressions using *sign*:

$$\begin{aligned} \mathit{isZero}(a) &= \mathit{sign}(a)^2 \\ \mathit{neg}(a) &= 1 - \mathit{isZero}(a) = 1 - \mathit{sign}(a)^2 \\ \mathit{neq}(a,b) &= \mathit{neg}(a-b) = 1 - (\mathit{sign}(a-b))^2 \\ \mathit{leq}(a,b) &= \mathit{sign}(a-b) * (\mathit{sign}(a-b) - 1)/2 \end{aligned}$$

We can use `pos` from Definition 4.1.2 to express *sign*, *isZero*, *neg*, *neq*, and *leq* from Definition 4.1.5:

Definition 4.1.7. Assume matrix variable symbols `sign`, `isZero`, and `neg` such that

$$\mathit{arity}(\texttt{sign}) = \mathit{arity}(\texttt{isZero}) = \mathit{arity}(\texttt{neg}) = 3.$$

1. Sugar $\texttt{sign}_1(X)$ to $\mathsf{sign.in}(X)$. This stores an input value a.
2. Sugar $\texttt{sign}_2(X)$ to $\mathsf{sign.out}(X)$. This stores the corresponding output value $\mathit{sign}(a)$.
3. Sugar $\texttt{sign}_3(X)$ to $\mathsf{sign.ptr}(X)$. This stores a pointer to a column in $\texttt{pos}_1$ (Definition 4.1.2) holding the absolute value of a.

We use similar sugar for `isZero` and `neg`. The validity predicates for `sign`, `isZero` and `neg` (using `pos`) are:

$$\begin{aligned} \chi_{\texttt{sign}}(X) = 1 \leq \mathsf{sign.ptr} \leq \mathsf{len}(\mathsf{pos})\ \land \\ \forall_{\texttt{sign}} X.\big(\ & (\mathsf{sign.in}(X) = 0 \land \mathsf{sign.out}(X) = 0)\ \lor \\ & (\mathsf{sign.in}(X) = \mathsf{pos}_1(\mathsf{sign.ptr}(X)) \land \mathsf{sign.out}(X) = 1)\ \lor \\ & (((\text{-}1) * \mathsf{sign.in}(X)) = \mathsf{pos}_1(\mathsf{sign.ptr}(X)) \land \mathsf{sign.out}(X) = \text{-}1)\big) \end{aligned}$$

$$\begin{aligned} \chi_{\texttt{isZero}}(X) = 1 \leq \mathsf{isZero.ptr} \leq \mathsf{len}(\mathsf{pos})\ \land \\ \forall_{\texttt{isZero}} X.\big(\ & (\mathsf{isZero.in}(X) = 0 \land \mathsf{isZero.out}(X) = 0)\ \lor \\ & (\mathsf{isZero.in}(X)^2 = \mathsf{pos}_1(\mathsf{isZero.ptr}(X)) \land \mathsf{isZero.out}(X) = 1)\big) \end{aligned}$$

$$\begin{aligned} \chi_{\texttt{neg}}(X) = 1 \leq \mathsf{neg.ptr} \leq \mathsf{len}(\mathsf{pos})\ \land \\ \forall_{\texttt{neg}} X.\big(\ & (\mathsf{neg.in}(X) = 0 \land \mathsf{neg.out}(X) = 1)\ \lor \\ & (\mathsf{neg.in}(X)^2 = \mathsf{pos}_1(\mathsf{neg.ptr}(X)) \land \mathsf{neg.out}(X) = 0)\big) \end{aligned}$$

Assume matrix variable symbols `neq`, and `leq` such that

$$\mathit{arity}(\texttt{neq}) = \mathit{arity}(\texttt{leq}) = 4.$$

1. Sugar $\texttt{neq}_1(X)$ to $\texttt{neq.in}_1(X)$. This stores an input value a.
2. Sugar $\texttt{neq}_2(X)$ to $\texttt{neq.in}_2(X)$. This stores an input value b.

3. Sugar $\mathtt{neq}_3(X)$ to $\mathsf{neq.out}(X)$. This stores the corresponding output value $neq(a, b)$.
4. Sugar $\mathtt{neq}_4(X)$ to $\mathsf{neq.ptr}(X)$. This stores a pointer to a column in $\mathtt{pos}_1$ (Definition 4.1.2).

We use similar sugar for `leq`. The validity predicates for `neq` and `leq` (using `pos`) are:

$$\begin{aligned}\chi_{\mathtt{neq}}(X) = 1 \leq \mathsf{neq.ptr} \leq \mathsf{len}(\mathtt{pos}) \wedge \\ \forall_{\mathtt{neq}} X.(\ &((\mathtt{neq.in}_1(X) \text{-} \mathtt{neq.in}_2(X)) = 0 \wedge \mathsf{neq.out}(X) = 1) \vee \\ &((\mathtt{neq.in}_1(X) \text{-} \mathtt{neq.in}_2(X))^2 = \mathtt{pos}_1(\mathsf{neq.ptr}(X)) \wedge \\ &\quad \mathsf{neq.out}(X) = 0))\end{aligned}$$

$$\begin{aligned}\chi_{\mathtt{leq}}(X) = 1 \leq \mathsf{leq.ptr} \leq \mathsf{len}(\mathtt{pos}) \wedge \\ \forall_{\mathtt{leq}} X.(\ &((\mathsf{leq.a}(X) \text{-} \mathsf{leq.b}(X)) = 0 \wedge \mathsf{leq.out}(X) = 0) \vee \\ &((\mathsf{leq.b}(X) \text{-} \mathsf{leq.a}(X)) = \mathtt{pos}_1(\mathsf{leq.ptr}(X)) \wedge \mathsf{leq.out}(X) = 0) \vee \\ &((\mathsf{leq.a}(X) \text{-} \mathsf{leq.b}(X)) = \mathtt{pos}_1(\mathsf{leq.ptr}(X)) \wedge \mathsf{leq.out}(X) = 1))\end{aligned}$$

Lemma 4.1.8. *Suppose $f \in \{sign, isZero, neg, neq, leq\}$ is one of the functions from Definition 4.1.5. Then the corresponding validity predicate $\chi_{\mathtt{f}}$ from Definition 4.1.7 correctly encodes f, in the sense that*

- *for each corresponding $\mathtt{f} \in \{\mathtt{sign}, \mathtt{isZero}, \mathtt{neg}, \mathtt{neq}, \mathtt{leq}\}$ and for every interpretation ς,*
- *if $\vDash_\varsigma \chi_{\mathtt{pos}}$ (Definition 4.1.2) and $\vDash_\varsigma \chi_{\mathtt{f}}$ then*
- *for every $x \in 1..len(\varsigma(\mathtt{f}))$ we have*

$$\begin{aligned}\varsigma(\mathtt{sign})(2, x) &= sign(\varsigma(\mathtt{sign})(1, x)) \\ \varsigma(\mathtt{isZero})(2, x) &= isZero(\varsigma(\mathtt{isZero}(1, x))) \\ \varsigma(\mathtt{neg})(2, x) &= neg(\varsigma(\mathtt{neg})(1, x)) \\ \varsigma(\mathtt{neq})(3, x) &= neq(\varsigma(\mathtt{neq})(1, x), \varsigma(\mathtt{neq})(2, x)) \\ \varsigma(\mathtt{leq})(3, x) &= leq(\varsigma(\mathtt{leq})(1, x), \varsigma(\mathtt{leq})(2, x))\end{aligned}$$

Proof. The functions in Definition 4.1.5 are simple enough, and so are the validity predicates in Definition 4.1.7. We check the correspondence and use Theorem 2.4.4. □

Remark 4.1.9.

1. There is some apparent overlap between *isZero* from Definition 4.1.5 and the 'is equal to zero' function $\lambda x.[\![X = 0]\!](x) = \lambda x.x^2$ which we can build from the syntax and denotation in Figures 1 and 2.

But these are different functions: with *isZero*, the output is either 0 or 1, whereas with $\lambda x.x^2$ the output may be any nonnegative value. For example:

$$\mathit{isZero}(2) = 1 \quad \text{whereas} \quad [\![2 = 0]\!] = 4,$$

as per Figure 2.

Knowing that the output is precisely equal to 0 or 1 is a stronger property, and it matters because (for example) we can then easily get the 'negation' predicate (mapping 0 to 1 and any nonzero value to 0) as $1 - \mathit{isZero}(a)$. We cannot do this with $\lambda x.[\![X = 0]\!](x) = \lambda x.x^2$.

2. *neg* is short for *negation*. It maps 0 (corresponding to 'truth') to 1, and any nonzero value (corresponding to 'false') to 0.

   This is a negation, and we have:

   (a) We have *excluded middle*, since $a * \mathit{neg}(a) = 0$ for every $a \in \mathbb{Q}$.[34]

   (b) $a + \mathit{neg}(a) \neq 0$ for every $a \in \mathbb{Q}$. This equals a if $a \neq 0$, and equals 1 if $a = 0$.[35]

   (c) Double negation $\mathit{neg}(\mathit{neg}(a))$ maps 0 to 0 and any nonzero element to 1. If we think of a truth-value $a \in \mathbb{Q}_{\geq 0}$ as being either 0 or a nonzero 'error value', then $\mathit{neg}(\mathit{neg}(a))$ throws away the precise error value and just retains the information whether a was true or false.

   neg is not unique with the properties above; e.g. neg' mapping a to $2 * \mathit{neg}(a)$ has similar properties. This is as expected and it just reflects that there are many false (= nonzero) values.

4.1.3 Back from pos to the range check

Remark 4.1.10 (From *pos* back to $<$). We saw above how, in the presence of the rest of the logic, the range checks $t < \mathtt{C}_i$ and $\mathtt{C}_i < t$ can express predicates for strictly positive numbers (Definition 4.1.2), and a *sign* function (Definition 4.1.7).

We take a moment to note that we can also go in the other direction, from *pos* to range checks. Let us for this Remark take a *pos* predicate as primitive in our syntax.

We can express the range check

$$t < \mathtt{C}_i \quad \text{as} \quad \forall_{\mathtt{C}} X.\mathit{pos}(\mathtt{C}_i(X) - t),$$

[34] Excluded middle is the property 'ϕ-or-not-ϕ', which when filtered through our polynomial semantics from Figure 2 renders as $a * \mathit{neg}(a) = 0$.

[35] This corresponds to 'ϕ-and-not-ϕ', which we expect to be false, i.e. nonzero.

and

$$t > \mathtt{C}_i \quad \text{as} \quad \forall_{\mathtt{C}} X.pos(t - \mathtt{C}_i(X)).$$

Thus in some sense, $<$, *pos*, and also *sign* are all doing the same thing.

Remark 4.1.11. Continuing Remark 4.1.10, $<$ is also clearly *different* from *pos* and *sign*, because $t < \mathtt{C}_i$ is variable-free and so has the flavour of a static check on the interpretation ς, whereas $pos(x)$ and $sign(x)$ have the flavour of being predicates.

Yet, in the context of the rest of the system they have the same expressivity. What deeper structure is being manifested here?

We propose that at a deeper level, what these constructs do is introduce the power of a *knowledge modality.* This is a modal operator that internalises some notion of necessitation, truth, knowledge, or provability inside the logic itself [30] — that something is not just true or false, but also known to be such, proved to be such, or necessarily such, etc.

A feature of the notion of 'false' in our logic is that it is modelled by being nonzero, and there are infinitely many nonzero numbers so we cannot explicitly enumerate over all the possible ways of being false within our logic.[36] What is significant about $<$, *pos*, and *sign* is not so much what they detect (though this is of course relevant); what really matters is that they map the truth or falsity of what they measure, to a finite set of values ($\{0,1\}$ or $\{0,1,\text{-}1\}$), and this is where the effect of a knowledge modality comes from.

What makes $<$ special, and better to use as a primitive than *pos* and *sign*, is that it still has a polynomial semantics — because (as we mentioned in Remark 4.1.1) $[\![t < \mathtt{C}_i]\!]_\varsigma$ returns a number, which is a polynomial. More on this in future work.

4.2 Arbitrary functions in terms and complementation in predicates

Remark 4.2.1. The term language in Figure 1 does not accommodate arbitrary functions $f : \mathbb{Q}^n \to \mathbb{Q}$. This is by design, since Figure 2 presents a *polynomial* semantics. But it would be nice if we could have our cake and eat it too: use arbitrary functions in our terms, and *still* get the goodness of polynomial expressions.

This is impossible in the general case, but we do not need the general case: a simple polynomial interpolation is sufficient. So, we will use a matrix variable

[36] This would not be solved by switching to a finite field. In practical terms, any finite field large enough to be cryptographically secure is also large enough to be practically unenumerable. This is why cryptographic assurances work! Thus, modelling 'true' by a single value and 'false' by many values is not a bug of our semantics. Far from it; it reflects an essential feature of the cryptographic notion of proof.

symbol $\mathtt{f}$, with a validity predicate that expresses that $\mathtt{f}$ interpolates the desired function f.

Similarly, the predicate language in Figure 1 does not have a negation $\neg\phi$. We can easily (in retrospect) interpret truth as zero and falsity as non-zero and $\wedge$ as $+$ and $\vee$ as $*$, because 0, $+$, and $*$ are naturally polynomial. Negation is different: no low-degree polynomial maps 0 to a non-zero number and any non-zero number to 0.[37] But it turns out that we do not need to map *any* non-zero number to 0: we will use a matrix variable symbol $\mathtt{neg}$, with a validity predicate that expresses that $\mathtt{neg}$ interpolates negation as required.

4.2.1 Syntax with arbitrary f and compilation to syntax without

Consider the following scenario:

1. We have some function $f : \mathbb{Q} \to \mathbb{Q}$. (For simplicity we assume f is unary, but functions with more arguments can also be accommodated.)
2. We have represented f in our logic with
   - a matrix variable symbol $\mathtt{f}$ of arity $\mathit{arity}(\mathtt{f}) \geq 2$, and
   - a validity predicate $\chi_{\mathtt{f}}$, such that $\mathtt{f}_1$ represents input values, and $\mathtt{f}_2$ represents output values. In symbols, we write:

   $$\vDash_\varsigma \chi_{\mathtt{f}} \quad \Longrightarrow \quad \forall x \in 1..\mathit{len}(\varsigma(\mathtt{f})).\ \mathtt{f}_2(x) = f(\mathtt{f}_1(x)).$$

   Other rows may be present in $\mathtt{f}$ (see for example our implementation of *sign* in Definition 4.1.7, which requires a third row of data). This is fine and will not affect our reasoning.
3. We have another matrix variable symbol $\mathtt{P}$ with $\mathit{arity}(\mathtt{P})$, and suppose $\mathtt{P}$ has a validity predicate $\chi_{\mathtt{P}}$ that uses an **f-enriched term syntax**, which enriches the syntax from Figure 1 such that if t is an f-enriched term then $f(t)$ is an f-enriched term.

   There are no other restrictions on $\chi_{\mathtt{P}}$; it is just some predicate, but in the f-enriched syntax.

   The denotation of the f-enriched term $f(t)$, is f applied to the denotation of t. That is, we enrich Figure 2 with a clause

   $$[\![f(t)]\!]_\varsigma(x) = f([\![t]\!]_\varsigma(x)). \tag{2}$$

We want to transform $\chi_{\mathtt{P}}$ into another validity predicate $\chi'_{\mathtt{P}}$ such that

[37] In a finite field with n elements, this would require a polynomial of degree n. In an infinite field, it is impossible.

- $\chi'_{\mathtt{P}}$ means the same thing as $\chi_{\mathtt{P}}$ (in a sense we will make formal in Definition 4.2.3 and Proposition 4.2.4) but
- $\chi'_{\mathtt{P}}$ is in the unenriched syntax, and it will replace calls to f with uses of the matrix variable symbol $\mathtt{f}$.

Specifically, in Definition 4.2.3 we will show how to transform

- a matrix variable symbol $\mathtt{P}$ with arity $\mathit{arity}(\mathtt{P})$ and validity predicate $\chi_{\mathtt{P}}$ that contains some innermost instance of $f(t)$,[38] into
- a matrix variable symbol $\mathtt{P}'$ with arity $\mathit{arity}(\mathtt{P}') = \mathit{arity}(\mathtt{P}){+}1$ and a validity predicate $\chi_{\mathtt{P}'}$ that does not contain this instance of $f(t)$.

Since syntax is finite, we can eliminate all instances of f from our validity predicate by repeating this transformation.

REMARK 4.2.2. The transformation outlined in Definition 4.2.3 is straightforward: we replace calls to f with pointers to a matrix that stores a polynomial interpolation with the same values at those calls. A polynomial can approximate a function at any finite number of points, so we know this can work; the rest is just a matter of getting the technicalities right.

DEFINITION 4.2.3 (The transformation). For clarity, we sugar $\mathtt{f}_1(t)$ to $\mathsf{f.in}(t)$ and $\mathtt{f}_2(t)$ to $\mathsf{f.out}(t)$.

1. Looking at the syntax in Figure 1, we see that the instance of $f(t)$ must occur inside some subpredicate $\phi = (t'{=}t'')$ where $f(t)$ appears in t' or t''. We obtain $\chi_{\mathtt{P}'}$ from $\chi_{\mathtt{P}}$ by replacing ϕ in $\chi_{\mathtt{P}}$ with
$$(\mathsf{f.in}(\mathsf{P}'_{n+1}(X)) = t) \ \wedge\ \phi[f(t) \mapsto \mathsf{f.out}(\mathsf{P}'_{n+1}(X))],$$
where $\phi[f(t) \mapsto \mathsf{f.out}(\mathsf{P}'_{n+1}(X))]$ denotes the predicate obtained by replacing any instances of $f(t)$ in ϕ (of which we assumed there is at least one) with $\mathsf{f.out}(\mathsf{P}'_{n+1}(X))$.
Using Theorem 2.4.4 we see that $\chi_{\mathtt{P}'}$ expresses (subject to range checks which we will add below) that the value in $\mathsf{P}'_{n+1}(X)$ points to a column in $\mathtt{f}$ that contains $(t, f(t))$.
2. We repeat the transformation above until there are no mentions of f left. The result is a matrix variable symbol $\mathtt{P}^{(n)}$ of arity $\mathit{arity}(\mathtt{P}^{(n)}) = \mathit{arity}(\mathtt{P}) + n$ (where n is the number of times we applied our transformation), and with validity predicate $\chi_{\mathtt{P}^{(n)}}$.

[38] 'Innermost' means that t does not itself contain some subterm of the form $f(t')$.

3. Now we just need to wrap this in range checks to prevent out-of-bounds pointer errors in the extra rows, so we set:

$$\chi'_{\mathsf{P}} = \chi_{\mathsf{P}^{(n)}} \wedge (1 \leq \mathsf{P}^{(n)}_{i+1} \leq \mathsf{len}(\mathtt{f})) \wedge \ldots \wedge (1 \leq \mathsf{P}^{(n)}_{i+n} \leq \mathsf{len}(\mathtt{f})).$$

Proposition 4.2.4. *Suppose ς is an interpretation such that $\vDash_{\varsigma} \chi_{\mathtt{f}}$. Then the following are equivalent:*

1. *$\vDash_{\varsigma} \chi_{\mathsf{P}}$ in the enriched syntax and denotation described above — in which $f(t)$ is a term if t is a term, and $[\![f(t)]\!]_{\varsigma}(x) = f([\![t]\!]_{\varsigma}(x))$ as per equation 2.*
2. *$\vDash_{\varsigma} \chi'_{\mathsf{P}}$ in the unenriched semantics.*

Proof. The argument is already in Definition 4.2.3: calls to f are replaced by range-checked pointers to columns in $\mathtt{f}$. We have assumed that $\vDash_{\varsigma} \chi_{\mathtt{f}}$, and we have assumed that this implies that $\varsigma(\mathtt{f})_2(x) = f(\varsigma(\mathtt{f})_1(x))$ for every column in $\varsigma(\mathtt{f})$, so that a call to f in the enriched syntax is precisely equivalent to a lookup in $\varsigma(\mathtt{f})$ of a column starting with the denotation of that term. □

Remark 4.2.5. Each step of the transformation in Definition 4.2.3(1) removes one innermost instance of $f(t)$ and increases the arity of the matrix variable symbol by 1 (modulo optimisations, e.g if $f(t)$ appears more than once). So we can say that, roughly speaking,

> each function-call to f costs an extra row in the validity matrix.

This is fine because it does not affect the width of the matrices, and so does not increase the degree of the interpolating polynomials.[39]

Overall, the process described above is not much different from what we do in ordinary mathematics, when we define e.g. Kuratowski pairs as $(x, y) = \{\{x\}, \{x, y\}\}$ and then proceed to use (x, y) directly; or when we define and use functions in any modern programming language and expect that these may be inlined, optimised, or compiled to a lower-level representation. In principle everything will get unwound to some more primitive syntax, so we know we are safe, but the higher-level syntax is more usable.

In a practical implementation of this logic, we might expect to provide functions as primitive, and let the compiler translate them to a lower-level representation as described above.

[39] Tall but narrow matrices are cheap; see Remark 6.2.1(2). Thus, a transformation that increases arity (and thus the height of matrices) but does not affect their width, is free, to a first approximation.

4.2.2 An application: predicate complementation

We conclude by bringing together what we have learned, to express negation as a macro. In more technical terminology: we will show that our logic is *complemented.*

Recall from Figures 1 and 2 that our syntax and semantics do not include predicate negation $\neg\phi$. This is because we have one 'true' truth-value 0, and many 'false' truth-values $\mathbb{Q}\setminus\{0\}$.

However, in the presence of the rest of the logic, the *range-check* $<$ allows us to implement negation as a macro operation. We saw a strong indication of this in *neg* and `neg` from Subsection 4.1.2. This negation acts at the level of terms, but using the ideas in this Subsection we can fold it back into our logic.

Notation 4.2.6 and Theorem 4.2.7 are written in a syntax enriched with *neg*, as per Subsection 4.2.1:

Notation 4.2.6. Suppose ϕ is a predicate. Then we define $\boldsymbol{\sim}\phi$ the **complement** of ϕ by[40]

$$\boldsymbol{\sim}\phi \quad \text{as shorthand for} \quad \mathit{neg}(\mathsf{reify}(\phi)) = 0.$$

As described in Proposition 4.2.4, $\boldsymbol{\sim}\phi$ can be compiled down to an unenriched syntax that assumes a matrix variable symbol `neg` of arity 3 and a validity predicate $\chi_{\mathtt{neg}}$ as per Definition 4.1.7. This licenses us to work directly with *neg*-enriched syntax, and we obtain Theorem 4.2.7:

Theorem 4.2.7. *Suppose ς is an interpretation and $x\in\mathbb{Q}$ and ϕ is a predicate. Then*

$$x \vDash_\varsigma \boldsymbol{\sim}\phi \quad \textit{if and only if} \quad x \nvDash_\varsigma \phi.$$

Proof. By chasing definitions. The fact that this works is in itself a mathematical result that needs to be checked, so we give full details:

$$\begin{array}{rll}
x \vDash_\varsigma \boldsymbol{\sim}\phi \iff & [\![\boldsymbol{\sim}\phi]\!]_\varsigma(x) = 0 & \text{Figure 2} \\
\iff & [\![\mathit{neg}(\mathsf{reify}(\phi)) = 0]\!]_\varsigma(x) = 0 & \text{Notation 4.2.6} \\
\iff & ([\![\mathit{neg}(\mathsf{reify}(\phi))]\!]_\varsigma - 0)^2(x) = 0 & \text{Figure 2} \\
\iff & [\![\mathit{neg}(\mathsf{reify}(\phi))]\!]_\varsigma(x) = 0 & \text{Fact} \\
\iff & \mathit{neg}([\![\mathsf{reify}(\phi)]\!]_\varsigma(x)) = 0 & \text{Equation 2 from Subsection 4.2.1} \\
\iff & [\![\mathsf{reify}(\phi)]\!]_\varsigma(x) \neq 0 & \text{Definition 4.1.5} \\
\iff & [\![\phi]\!]_\varsigma(x) \neq 0 & \text{Figure 2} \\
\iff & x \nvDash_\varsigma \phi & \text{Figure 2}
\end{array}$$

The use of equation 2 from Subsection 4.2.1 above (i.e. treating *neg* as a primitive in our syntax) is justified by Proposition 4.2.4. □

[40] This implementation is not necessarily optimal, but it is simple.

Theorems 4.2.7 and 2.4.4 taken together prove that our logic can translate the expressive power of full first-order logic — at least! — into polynomials. Each step towards this is arguably elementary, but the overall result seems surprising and powerful.

4.3 General recursion

We show how to implement a minimisation function, in the sense of general recursive functions [11], in our logic.[41] The construction below is rather fiddly, but this is worth doing because it demonstrates that our logic is expressive enough to verify general recursion.

We already had a hint of this, because we specified Ackermann's function in Subsection 3.5 and this is known as a canonical function that is recursive but not primitive recursive. But here, we isolate and implement a minimisation function in and of itself.

Recall the (standard) definition:

Definition 4.3.1. Suppose $f : (\mathbb{N}_{\geq 1} \times \mathbb{N}^k) \to \mathbb{N}$.[42] Then define the **minimisation** $\mu(f)$ by

$$\mu(f) : \mathbb{N}^k \to \mathbb{N}$$
$$\mu(f)(x_1, \ldots, x_k) = min\{n \in \mathbb{N}_{\geq 1} \mid f(n, x_1, \ldots, x_k) = 0\}.$$

In words: $\mu(f)$ maps $(x_1, \ldots, x_k) \in \mathbb{N}^k$ to the least zero value of $\lambda n.f(n, x_1, \ldots, x_k)$.

If no such zero value exists, so that $\{n \in \mathbb{N}_{\geq 1} \mid f(n, x_1, \ldots, x_k) = 0\} = \varnothing$, then the value of $\mu(f)(x_1, \ldots, x_k)$ is not specified.[43]

Remark 4.3.2. We can think of μ as generalising a *while* loop: given some parameters $(x_1, \ldots, x_k) \in \mathbb{N}^k$, μ searches $n \in \mathbb{N}_{\geq 1}$ and continues while the value of $f(n, x_1, \ldots, x_k) \neq 0$. If $f(n, x_1, \ldots, x_k) = 0$, then μ stops and returns that (first) value of n.

Now we give a slightly lower-level version of Definition 4.3.1. It is still abstract, but it is closer to what we would implement in our logic.

Notation 4.3.3. Suppose $U \subseteq \mathbb{N}_{\geq 1} \times \mathbb{N}$ and $b \in \mathbb{N}$. Then define

$$U_b = \{a \in \mathbb{N}_{\geq 1} \mid (a, b) \in U\}.$$

[41] The Wikipedia page [31] gives a clear and accessible presentation.

[42] We started counting at 1 in the first argument for the technical reason that we will be most interested in applying this to identify columns in matrices, which we count from 1, not 0. There is no deeper mathematical significance here.

[43] . . . or undefined.

If U_b is an initial segment of $\mathbb{N}_{\geq 1}$ for every $b \in \mathbb{N}$, then call U **initial in its first component**.[44]

Recall that *sign* is defined in Definition 4.1.5, and note that if we restrict *sign* to $\mathbb{N}$ then it maps to $\{0, 1\}$.

Definition 4.3.4. Assume we are given the following data:

1. A function $f : (\mathbb{N}_{\geq 1} \times \mathbb{N}) \to \mathbb{N}$ (i.e. for simplicity we have set $k = 1$ in Definition 4.3.1).
2. A $U \subseteq \mathbb{N}_{\geq 1} \times \mathbb{N}$ that is initial in its first component (Notation 4.3.3).

We specify a pair of functions

$$\mathit{min}_{f,U}, \mathit{acc}_{f,U} : U \to \mathbb{N}$$

($\mathit{min}_{f,U}$ is short for 'minimal point' and $\mathit{acc}_{f,U}$ is short for 'accumulator') as follows:

$$\begin{aligned}
& 1 \leq \mathit{min}_{f,U}(a, b) \\
a{\neq}1 \Longrightarrow\ & \mathit{min}_{f,U}(a, b) = \mathit{min}_{f,U}(a\text{-}1, b) \\
& f(\mathit{min}_{f,U}(a, b), b) \leq f(a, b) \\
a{=}1 \Longrightarrow\ & \mathit{acc}_{f,U}(a, b) = \mathit{sign}(f(a, b) - f(\mathit{min}_{f,U}(a, b), b)) \\
a{\neq}1 \Longrightarrow\ & \mathit{acc}_{f,U}(a, b) = \mathit{acc}_{f,U}(a\text{-}1, b) * \mathit{sign}(f(a, b){-}f(\mathit{min}_{f,U}(a, b), b)) \\
\mathit{min}_{f,U}(a, b){\neq}1 \Longrightarrow\ & \mathit{acc}_{f,U}(\mathit{min}_{f,U}(a, b)\text{-}1, b) = 1
\end{aligned}$$

We make sense of Definition 4.3.4 using a lemma:

Lemma 4.3.5. *Continuing the notation of Definition 4.3.4, we have the following properties, for each $b \in \mathbb{N}$:*

1. *$\lambda a{\in}U_b.\mathit{min}_{f,U}(a, b)$ is a constant function into $\mathbb{N}_{\geq 1}$.*
   Write $\mu_{f,U}(b) \in \mathbb{N}_{\geq 1}$ for its unique value.
2. *$f(\mu_{f,U}(b), b) \leq f(a, b)$ for every $a \in U_b$.*
   Thus, $f(\mu_{f,U}(b), b)$ is a lower bound for $\{f(a, b) \mid a \in U_b\}$.
3. *Furthermore, since $f(\mu_{f,U}(b), b) = f(\mu_{f,U}(b), b)$, also $f(\mu_{f,U}(b))$ is a* greatest *lower bound for this set. Thus,*

   $$\mu_{f,U}(b) \in \mathbb{N}_{\geq 1} \textit{ is the index of a minimal value of } \lambda a{\in}U_b.f(n, b).$$
4. *$\mathit{acc}_{f,U}(a, b) = \prod_{i \in 1..a} \mathit{sign}(f(a, b) - \mu_{f,U}(b))$.*
   We observe by arithmetic that this is equal to 1 or 0, and $\mathit{acc}_{f,U}(a, b)$ is equal to 0 precisely when $f(i, b) = f(\mu_{f,U}(b), b)$ for some $i \in 1..a$.

[44] An initial segment $I \subseteq \mathbb{N}_{\geq 1}$ is such that if $a \in I$ and $a > 1$ then $a - 1 \in I$. Note that $\varnothing \subseteq \mathbb{N}_{\geq 1}$ is initial.

$$\begin{array}{rll@{\qquad}rll}
\mathtt{mu}_{f,1}(X) & \text{sugars to} & \mathsf{ptr.min}(X) & \mathtt{mu}_{f,2}(X) & \text{sugars to} & \mathsf{ptr.f}(X) \\
\mathtt{f}_1(\mathsf{ptr.f}(X)) & \text{sugars to} & \mathsf{a}(X) & & & \\
\mathtt{f}_2(\mathsf{ptr.f}(X)) & \text{sugars to} & \mathsf{b}(X) & \mathtt{f}_3(\mathsf{ptr.f}(X)) & \text{sugars to} & \mathsf{f.ab}(X) \\
\mathtt{mu}_{f,3}(X) & \text{sugars to} & \mathsf{acc}(X) & \mathtt{mu}_{f,4}(X) & \text{sugars to} & \mathsf{ptr.prev}(X)
\end{array}$$

Write $t \neq t'$ as shorthand for $\mathit{sign}(t-t')^2 = 1$. Write $t \leq t'$ as shorthand for $\mathit{sign}(t-t'+1) = 1$.

$$\begin{array}{ll}
\chi_{\mu_f} = & 1 \leq \mathsf{ptr.min} \leq \mathsf{len}(\mathtt{mu}) \wedge 1 \leq \mathsf{ptr.prev} \leq \mathsf{len}(\mathtt{mu}) \wedge 1 \leq \mathsf{ptr.f} \leq \mathsf{len}(\mathtt{f}) \wedge \\
& \forall_{\mathtt{mu}} X.\Big(\ (\mathsf{a}(X) = 1 \ \vee \ \mathsf{ptr.min}(\mathsf{ptr.prev}(X)) = \mathsf{ptr.min}(X)) \wedge \\
& \qquad (\mathsf{f.ab}(\mathsf{ptr.min}(X)) \leq \mathsf{f.ab}(X)) \wedge \\
& \qquad \left(\begin{array}{l}
(\mathsf{a}(X) = 1 \wedge \ \mathsf{acc}(X) = \mathit{sign}(\mathsf{f.ab}(X) - \mathsf{f.ab}(\mathsf{ptr.min}(X)))) \vee \\
(\mathsf{a}(X) \neq 1 \wedge \ \mathsf{acc}(X) = \mathit{sign}(\mathsf{f.ab}(X) - \\
\qquad\qquad\qquad \mathsf{f.ab}(\mathsf{ptr.min}(X))) * \mathsf{acc}(\mathsf{ptr.prev}(X)))
\end{array} \right) \wedge \\
& \qquad (\mathsf{a}(\mathsf{ptr.min}(X)) = 1 \ \vee \ \mathsf{acc}(\mathsf{ptr.prev}(X)) = 1) \wedge \\
& \qquad (\mathsf{a}(X) = 1 \ \vee \ \mathsf{a}(\mathsf{ptr.prev}(X)) = \mathsf{a}(X) - 1) \wedge \\
& \qquad (\mathsf{a}(X) = 1 \ \vee \ \mathsf{b}(\mathsf{ptr.prev}(X)) = \mathsf{b}(X))\Big)
\end{array}$$

Figure 9: Expressing a minimisation operator μ (Definition 4.3.6)

5. *If $\mu_{f,U}(b) > 1$ then $acc_{f,U}(\mu_{f,U}(b)-1, b) = 1$, which means that $\mu_{f,U}(b) \lneq f(a,b)$ for every $1 \leq a < \mu_{f,U}(b)$. Thus,*

   $\mu_{f,U}(b)$ is also least *such that $\lambda a{\in}U_b.f(a,b)$ attains its minimal value.*

Proof. Direct from the construction, as described in the statement of this Lemma. □

We can now implement minimisation in our logic, just by translating Definition 4.3.4 into it:

Definition 4.3.6. Suppose we are given a matrix variable symbol $\mathtt{f}$ whose first three rows $\mathtt{f}_1$, $\mathtt{f}_2$, and $\mathtt{f}_3$ are intended to represent the two input values a and b and one output value $f(a,b)$ of a binary function $f : (\mathbb{N}_{\geq 1} \times \mathbb{N}) \to \mathbb{N}$.

Assume a matrix variable symbol $\mathtt{mu}_f$ with arity $\mathit{arity}(\mathtt{mu}) = 4$. In Figure 9, we define syntactic sugar and use it to express a validity predicate χ_{μ_f}, based on Definition 4.3.4.

REMARK 4.3.7. Note that in the predicate χ_{μ_f} in Figure 9, we use the function *sign* from Definition 4.1.5 directly in terms. Its implementation in matrix semantics is in Definition 4.1.7.

We can do this because as per Subsection 4.2, the function call can be compiled down to the purely polynomial term language. This is not a problem and our presentation in Figure 9 is more readable as a result.

A simple and natural bit of terminology will be useful in a moment:

DEFINITION 4.3.8. Suppose that:

1. $f : (\mathbb{N}_{\geq 1} \times \mathbb{N}) \to \mathbb{N}$.
2. $\mathtt{f}$ with arity $\mathit{arity}(\mathtt{f}) \geq 3$ is a matrix variable symbol.
3. ς is an interpretation.

Then say that ς **interprets** f when

$$\varsigma(\mathtt{f})_3(x) = f(\varsigma(\mathtt{f})_1(x), \varsigma(\mathtt{f})_2(x))$$

for every $x \in 1..\mathit{len}(\varsigma(\mathtt{f}))$.

PROPOSITION 4.3.9. *Suppose $f : (\mathbb{N}_{\geq 1} \times \mathbb{N}) \to \mathbb{N}$, and $\mathtt{f}$ with arity $\mathit{arity}(\mathtt{f}) \geq 3$ is a matrix variable symbol, and suppose ς is an interpretation that interprets f (Definition 4.3.8). Then*

- *if $\vDash_\varsigma \chi_{\mu_f}$ (Figure 9),*
- *then $[\![\mathsf{ptr.min}(X)]\!]_\varsigma(x) = \mu_{f,U}([\![\mathsf{b}(X)]\!]_\varsigma(x))$,*

where we define $U \subseteq \mathbb{N}_{\geq 1} \times \mathbb{N}$ by

$$U = \{[\![\mathsf{a}(X)]\!]_\varsigma(x), [\![\mathsf{b}(X)]\!]_\varsigma(x) \mid x \in 1..\mathit{len}(\varsigma(\mathtt{mu}))\}$$

(this is just a fancy way of setting U to be the set of (a, b) pairs over which $\varsigma(\mathtt{mu})$ minimises).

Proof. We examine Figure 9 and see through careful inspection that it translates Definition 4.3.4 into our logic: $\varsigma(\mathtt{mu}_f)_1$ corresponds to $\mathit{min}_{f,U}$ and $\varsigma(\mathtt{mu}_f)_2$ corresponds to $\mathit{acc}_{f,U}$. □

REMARK 4.3.10. Proposition 4.3.9 does not describe the full minimisation outlined in Definition 4.3.1, but this is only because minimisation implicitly contains an unbounded search, whereas an interpretation ς delivers *finite* matrices.

This is reasonable, because it is the prover's job to run a program and — after a finite time — it must stop and present the ς it has generated. If the prover does not terminate then no ς is generated and there is nothing for the verifier to do. So in our context, minimisation means finding minimal values within a given run.

Thus Proposition 4.3.9 does the reasonable thing, which is to tell us that if we are given some $b \in \mathbb{N}$ and $a \in \mathbb{N}_{\geq 1}$ and a ς that satisfies the validation predicates and for which $\varsigma(\mathtt{mu})$ contains a column which references $f(a, b)$ — if all the above holds, then $\varsigma(\mathtt{mu})$ correctly identifies the first index $i \in 1..a$ to yield the lowest value of $f(i, b)$.

4.4 Logic gates

We conclude with a small detour: how to encode a circuit of logic gates in our framework.

In one sense, considering logic gates is a waste: we already have predicates and terms, which are higher-level and (as we have seen) compile to polynomials. Yet it is still an interesting exercise to spell out how this might be done.

We will only implement a *nand* gate; this is known to be a *universal logic gate* from which all others may be derived (see [16] for a clear presentation), and it will also be completely clear from the example of *nand* below how other logic gates could be directly implemented.

Recall from Remark 2.3.5(1) that we model 0 as 'true' and 1 as 'false', so the bits are flipped. *nand* should return 1 (false) if both of its inputs are 0 (true), and 0 (true) otherwise.

Definition 4.4.1. It is straightforward to encode NAND as a simple polynomial:

$$\mathit{nand}(a, b) = (1 - a) * (1 - b).$$

Now we want to connect our individual logic gates into circuits.

We could try to nest our gates as functions, but we see from Remark 3.2.12(4) that this would result in polynomials of high degree; a more efficient method is available.

Definition 4.4.2. Assume a matrix variable symbol $\mathtt{nand}$ with arity $\mathit{arity}(\mathtt{nand}) = 5$. In Figure 10, we define syntactic sugar and use it to express a validity predicate $\chi_{\mathtt{nand}}$, based on Definition 4.4.1.

Remark 4.4.3. We read through this carefully. Suppose we are given an interpretation ς and consider column number x in $\varsigma(\mathtt{nand})$, which is a 5-tuple of elements $(a, b, r, p_1, p_2) \in \mathbb{Z}^5$:

1. a represents the first argument to the NAND gate. This must be equal to either 0 or 1.[45]

[45] We impose this condition explicitly in Figure 10 as $\mathsf{nand.in}_1(X) = 0 \lor \mathsf{nand.in}_1(X) = 1$ and

$$
\begin{array}{llll}
\mathtt{nand}_1(X) \text{ sugars to } \mathsf{nand.in}_1(X) & & \mathtt{nand}_2(X) \text{ sugars to } \mathsf{nand.in}_2(X) \\
\mathtt{nand}_3(X) \text{ sugars to } \mathsf{nand.out}(X) & & \\
\mathtt{nand}_4(X) \text{ sugars to } \mathsf{in}_1\mathsf{.ptr}(X) & & \mathtt{nand}_5(X) \text{ sugars to } \mathsf{in}_2\mathsf{.ptr}(X)
\end{array}
$$

$$
\begin{aligned}
\chi_{\mathtt{nand}} = \;& 0 \leq \mathsf{in}_1\mathsf{.ptr} \leq \mathsf{len}(\mathtt{nand}) \;\wedge\; 0 \leq \mathsf{in}_2\mathsf{.ptr} \leq \mathsf{len}(\mathtt{nand}) \;\wedge \\
& \forall_{\mathtt{nand}} X.(\; (\mathsf{nand.out}(X) = (1 - \mathsf{nand.in}_1(X)) * (1 - \mathsf{nand.in}_2(X))) \;\wedge \\
& \qquad (\mathsf{in}_1\mathsf{.ptr}(X) = 0 \;\vee\; \mathsf{nand.out}(\mathsf{in}_1\mathsf{.ptr}(X)) = \mathsf{nand.in}_1(X)) \;\wedge \\
& \qquad (\mathsf{in}_2\mathsf{.ptr}(X) = 0 \;\vee\; \mathsf{nand.out}(\mathsf{in}_2\mathsf{.ptr}(X)) = \mathsf{nand.in}_2(X)) \;\wedge \\
& \qquad (\mathsf{nand.in}_1(X) = 0 \vee \mathsf{nand.in}_1(X) = 1) \;\wedge \\
& \qquad (\mathsf{nand.in}_2(X) = 0 \vee \mathsf{nand.in}_2(X) = 1))
\end{aligned}
$$

Figure 10: Expressing a circuit of NAND gates (Definition 4.4.2)

2. b represents the second argument to the NAND gate.
3. r represents the result. Note that $\chi_{\mathtt{nand}}$ insists that $r = \mathit{nand}(a, b)$.
4. p_1 represents *either* a null pointer 0, or it is a pointer to the output of some other gate. Note that $\chi_{\mathtt{nand}}$ insists that the value of that output is equal to a; thus we 'wire' the output to the first input of this gate.
5. Similarly for p_2.

Visibly this implements a circuit of NAND gates, where input wires are ones such that p_1 (or p_2) is zero, and output wires are outputs that are not pointed to.[46] A more explicit encoding would also be possible, but this will do.

Remark 4.4.4.

1. Circuits with multiple types of gate (AND, OR, NOT) could be encoded, either as distinct matrix variable symbols with their own validity predicates, or by generalising Figure 10 directly with an additional row of data to describe what type of gate is stored.
2. The degree of the polynomial $[\![\chi_{\mathtt{nand}}]\!]_{\varsigma}$ scales as $O(n^2)$, where n is $\mathit{len}(\varsigma(\mathtt{nand}))$. Our framework is not optimised for the special case of representing a circuit of simple logic gates, but if we specialise anyway then this does not obviously incur any serious overhead.

similarly for $\mathsf{nand.in}_2$; however, we could also insist on $0 \leq \mathsf{nand.in}_1 \leq 1$. Because an interpretation ς returns an *integer* matrix, this would suffice. These two methods are however subtly different: if we care about zero knowledge then it may be that range checked values get exposed to the validator. See point 2 of the discussion in Subsection 5.2.

[46] Note that in this system wires can be duplicated; the output of a gate can be wired to zero, one, or many outputs.

5 Efficiency: succinctness and built-in functions

5.1 Schwartz-Zippel

We give the reader a flavour of how the polynomial semantics can be applied.

Recall the arithmetisation of valid SK-combinator reduction $\chi_{\texttt{evl}}$ from Subsection 3.9, and for simplicity write it in the form $\forall_{\texttt{evl}} X.\phi$ for suitable ϕ. Suppose a *prover* wants to convince a *verifier* that $\vDash_\varsigma \forall_{\texttt{evl}} X.\phi$, where the first column in $\varsigma(\texttt{evl})$ encodes some evaluation of a large and complicated SK-combinator term. For brevity write $n = \mathit{len}(\varsigma(\texttt{evl}))$. The verifier could check that $[\![\phi[X{:=}i]]\!]_\varsigma = 0$ for every $i \in 1..n$, but that could be quite slow if n is large. Instead, the prover can compute a factorisation $[\![\phi]\!]_\varsigma = h * \mathit{zeroes}_n$, where

$$\mathit{zeroes}_n = \prod_{i \in 1..n} (X\text{-}i) \in \mathbb{Q}[X], \tag{3}$$

and $h \in \mathbb{Q}[X]$ is the result of a long division $[\![\phi]\!]_\varsigma / \mathit{zeroes}_n$, which has zero remainder because we assumed that $[\![\phi]\!]_\varsigma$ has roots at $1..n$.

By the Schwartz-Zippel lemma, polynomial equality and evaluation at a random point are equivalent, to a high degree of probability (see [28, Lemma 3.3, Subsection 3.4] or [19]):[47]

Lemma 5.1.1 (Schwartz-Zippel). *Suppose $P \in \mathbb{F}[X]$ is a nonzero polynomial of degree d, over a field $\mathbb{F}$, where $\mathbb{F}$ is much larger than d. ($\mathbb{F}$ could be a finite field, but it could also be $\mathbb{Q}$.)*

Suppose $S {\subseteq_{\mathit{fin}}} \mathbb{F}$ is a finite subset of $\mathbb{F}$ and write $\#S$ for the cardinality of (number of elements in) S. Then:

1. *For a randomly chosen $x \in S$, the probability of $P(x)$ being zero is at most $d/\#S$.*
   Thus as $\#S$ becomes very large relative to d, $P(x) \neq 0$ with a very high probability.[48]
2. *As a corollary, if $P, Q \in \mathbb{F}[X]$ then with a high degree of probability for large S the following are equivalent:*

[47] The Schwartz-Zippel lemma is actually a more general statement about multivariate polynomials over integral domains. Lemma 5.1.1 is the special case of a univariate polynomial over $\mathbb{Q}$. Even so, what is written captures the spirit and character of the general result.

[48] We can quantify this with some concrete numbers. A reasonable value for d is 10^9 (a gigabyte is roughly 10^9 bytes), and in one commercial system `https://docs.starkware.co/starkex/crypto/stark-curve.html` (permalink) $\mathbb{F}$ is finite with size roughly 2^{251} so we can just set $S = \mathbb{F}$. Then we can compute $2^{251}/10^9 \approx 4 * 10^{66}$. For comparison, there are only about 10^{57} protons and neutrons in the solar system.

- *$P = Q$ (P and Q are identical polynomials).*
- *$P(x) = Q(x)$ at a randomly-chosen $x \in S$ (P and Q evaluate to the same value at a random point).*

Proof. P has at most d roots. If $\#S/d$ is large, x is probably not one of them. The corollary follows taking $P - Q$. □

Lemma 5.1.2.

1. *If $[\![\phi]\!]_\varsigma(q) = h(q) * zeroes_n(q)$ for a random $q \in \mathbb{Q}$ then $[\![\phi]\!]_\varsigma = h * zeroes_n$ to a high degree of probability.*
2. *As a corollary, to check $\vDash_\varsigma \forall_{\mathtt{evl}} X.\phi$ it suffices to pick a random $q \in \mathbb{Q}$ and check the equality above, for $n = len(\mathtt{evl})$.*

Proof. We just apply Lemma 5.1.1 to $[\![\phi]\!]_\varsigma$ and $h_{\mathtt{C},\phi} * zeroes_{len(\varsigma(\mathtt{C}))}$ in $\mathbb{Q}[X]$. The corollary follows by the discussion above. □

Remark 5.1.3. The proof-scheme above is representative of a style of cryptographic proof, in the sense that many cryptographic proof-schemes work by the verifier sampling polynomials at randomly-chosen points.

Note that on its own, this scheme is too simplistic to be practical. A nice elementary exposition of how to build from Schwartz-Zippel to something more practical is online [1]. Here, we will put things in context:

1. The scheme above is not *succinct*: the prover has to communicate all of $\varsigma(\mathtt{evl})$ to the verifier. If this is large (e.g. gigabytes or terabytes) then that puts strain on the communication channel and on the verifier's memory. A succinct proof-scheme can attain near certainty about validity, *without* being bottlenecked by the network or the verifier's power.
2. The scheme above is not *zero-knowledge*. The prover reveals $\varsigma(\mathtt{evl})$ to the verifier; it might want to keep it secret. A zero knowledge proof-scheme can attain near certainty without revealing the witnessing data in ς to the verifier.
3. The computations above use the field of rationals $\mathbb{Q}$, so there is no bound on the possible magnitude of the individual numbers involved. Practical proof-schemes usually use a finite field $\mathbb{F}$.
4. The range-check primitive $t < \mathtt{C}$ is neither succinct nor zero-knowledge, at least if the prover just sends $\varsigma(\mathtt{C})$ to the verifier to check the range. Range-check schemes are well-studied, and are a special case of succinct (and zero knowledge) set membership schemes. This is a field of research in is own right ([5] contains a recent survey).

It remains to refine the FOL semantics to generate polynomials of a suitable form for efficiently interfacing with cryptographic methods, and / or create a suitable cryptographic scheme of our own; see the discussion of future work in Remark 6.1.1. It would also be helpful to educate the wider (non-logic) community on techniques from logic and computation, such as induction, derivation-trees, the distinction between syntax and semantics, the concept of computation existing independently of a virtual machine, and logical specification of correctness properties as a separate thing from execution of the code itself.[49]

5.2 Built-in functions

Having touched on notions of efficiency in the previous Subsection, we now make a few observations about making things run quickly on native silicon (i.e. directly on the underlying processor hardware). In this case, we may prefer to check some validity properties directly:

1. We gave a polynomial circuit to check that a matrix expresses bitwise expansion of 64-bit unsigned integers in Subsection 3.6. But bitwise operations may be rapid on underlying hardware — not least because this is how numbers are represented anyway. In practice, rather than check a validity predicate, a

[49] Let's make this concrete by listing some misconceptions about logic that I have encountered. *Misconception:* 'Computation' means 'execution of a Turing machine'. Thus if it doesn't run on a virtual machine (with memory and a program counter etc), it's not computation. As a further corollary, all data is token strings (not 'representable as'; *is*). *Misconception:* λ-calculus, logic programming, and denotational semantics do not exist. *Misconception:* Derivation-trees. Is this an execution trace? Why does it branch; is it non-deterministic? What virtual machine does it execute on? *Misconception:* Logic is undecidable. Thus *every* predicate is undecidable and logic cannot be used to say anything useful in practice. *Misconception:* Logic is not Turing-complete and thus cannot be used to specify computation. *Misconception:* Induction makes no sense, since the inductive hypothesis assumes what you want to prove. *Misconception:* Logic is nice, but it has many predicates and they are all different. Is there a *uniform* language to specify things? (The person is asking for a programming language.) *Misconception:* 'Prove' means 'test', as in 'unit test'. 'Logic' means 'rules', as in 'business logic'. *Misconception:* Algorithms are only thing that really matters. Programming is just connecting algorithms together in the right order, which should be straightforward. *Misconception:* The only assertion relevant to programming is 'computation has been executed according to the rules of the virtual machine'. Furthermore, if this is shown then it means the program is correct.

The list goes on. All this is to note that barriers to practical realisation of verifiable computation via a shallow compositional embedding of logic in polynomials, are cultural and educational, not just technical and mathematical. Even if the mathematics in this paper were perfect and sufficient (and it's not), even that alone would not suffice. *Pro tip:* don't say you have a 'logic'; say *'framework'* instead.

verifier might recognise that this is a bitwise expansion operation and prefer to just traverse the matrix and check its correctness directly.

In a monolithic system, where everything has to run in an abstract machine built in arithmetic circuits, this might be a bit problematic because it would involve stepping outside the abstraction provided. The system we describe in this paper is quite modular, so this may be less of a problem.

2. The range check primitive $<$ in Figure 1 can be checked rapidly in silicon.

   If we are just using $<$ to range check that pointers are in bound for the matrices they point to, then information leakage would be minimal: all a validator would have is a set of pointers, but without knowing how the prover has laid out its data in its matrices, this does not impart much useful information. For extra security, the numbers could always be obfuscated.

   Concretely, we would do this by providing `pos` from Definition 4.1.2 as a built-in matrix variable symbol of infinite length — calls to the validity check of `pos` would just get passed directly to the underlying silicon. As per Remark 4.1.10, just providing this would provide all the power of $<$.

So we see that if we just care about speed, then efficient checking in silicon for a small set of special built-in validity predicates, like range checks and bitwise operations, should be fine. If we want a full zero-knowledge treatment, such that the verifier never learns the precise witnesses used, this would not be possible.

But even here, we might (depending on circumstances) be happy to accept a small amount of information leakage, if this buys us enough efficiency. In particular, range checks for pointers could be done outside of a zero knowledge wrapper and information leakage may be acceptable, if all the verifier has is a bunch of pointers but no information about what they point to or how they connect. This seems plausible, but working through how to do it in practice is future work.

6 Conclusions and further work

6.1 Future work and other observations

Remark 6.1.1. There are plenty of directions to go from here. For example:

1. *Implementation.* We have set up a shallow translation from logic to polynomials, and from logical judgements to polynomial equality judgements. An obvious next step is to engineer a practical cryptographic proving system based on these ideas.

   This is current work and we will do no more here than sketch one high-level design choice: which polynomial ring to map to.

In this paper we use rational polynomials $\mathbb{Q}[X]$, but cryptographic methods often (though not universally) use polynomials over finite fields; e.g. $\mathbb{F}[X]$ where $\mathbb{F}$ is a finite field. If we switch to polynomials over a finite field then the character of the semantics changes in several ways. Most immediately, the mapping $[\![\phi \wedge \phi']\!] = [\![\phi]\!] + [\![\phi']\!]$ is compromised, because in a finite field the sum of two nonzero numbers is not necessarily nonzero. Various possibilities exist to overcome this, including: we set $[\![\phi \wedge \phi']\!] = gcd([\![\phi]\!], [\![\phi']\!])$ (greatest common divisor); or we set $[\![\phi \wedge \phi']\!] = Z * [\![\phi]\!] + Z' * [\![\phi']\!]$ where Z and Z' are indeterminates (variable symbols, similar to the X in Definition 2.2.3) representing randomly-chosen numbers.

However, it may also be that rational polynomials are practical *as is*. Very recent work (announced after this material was drafted) argues for cryptographic proof-schemes over rational polynomials and proposes a system that can work with them directly [13]. The pace of change in this field is rapid; perhaps cryptography is growing towards this material as quickly as this material is growing towards cryptography.

2. *HOL and other more expressive logics.* We have considered a first-order logic in this paper. What about other logics; for example a higher-order logic?

   We see no barrier to this. Indeed in a certain sense we have already taken the key step towards this when we considered arbitrary functions in Subsection 4.2. As we note in Remark 4.2.1, the model-theoretic power of the logic — HOL vs FOL, for example — is not so important here, because we are only interested in behaviour at finitely many points so we can interpolate it with a polynomial. We consider FOL in this paper because it is a simple system and so useful for proof-of-logical-concept, but the techniques should generalise.

   For a precise and technical expression of a related point, see Remark 6.1.2, which notes that the logic in this paper is FOL, but it is not *just* FOL.

3. *Polynomial-inspired logical systems.* In the discussion up to now, we have considered a preexisting logic (FOL) and given it a polynomial semantics. But the ideas could flow in the other direction as well, from semantics to logic, and we can ask: what logics does the polynomial semantics suggest?

   For a start, we can consider constructs inspired by polynomial division. Define $P \setminus Q$ to be P with any roots it shares with Q divided out. This would model non-implication $\phi \not\Rightarrow \psi$ (which can also be written as $\phi \wedge \neg\psi$). See also next point.

4. *Resource-sensitive logics.* Following the previous point, a polynomial can be viewed as a *multisets* of roots. Thus it is natural to finesse our logic to make it resource-sensitive, in the sense of linear logic or logics with subexponentials.

The validity judgement of our logic is clean and simple

$$x \vDash_\varsigma \phi \Longleftrightarrow [\![\phi]\!]_\varsigma(x) = 0,$$

as per Figure 2. It just says that x is a root of $[\![\phi]\!]_\varsigma$. But polynomials, being multisets of roots, suggest other validity judgements. For example, it is natural to consider

$$\vDash_\varsigma (\Phi \vdash \psi) \Longleftrightarrow (\sum_{\phi\in\Phi} [\![\phi]\!]_\varsigma) \mid [\![\psi]\!]_\varsigma$$

corresponding to an assertion that Φ implies ψ.[50] This notion of validity has some element of resource-sensitivity, because divisibility includes root multiplicity.[51] The design space of logics here seems interesting and merits investigation.

5. *Multivariate polynomials.* Our polynomials are univariate, and correspondingly our first-order logic has just one variable symbol X (and arbitrarily many matrix variable symbols).
   It would be natural to generalise to the multivariate case — thus permitting X, but also X' and Y, and so on, in Figure 1. This would naturally give a translation to multivariate first-order logic, just by setting $[\![X']\!] = X'$ for each variable symbols in Figure 2. Nothing in the maths in Figure 2 would be affected.
   This might be convenient and useful for applying multivariate arithmetisation techniques, but we do not think it would increase expressivity. It seems likely that we could emulate the multivariate case using Skolem functions (see next point) and (functions representing) ϵ-terms [3]. Looking into this generalisation is future work.
6. *Skolem functions.* We include an existential quantifier $\exists$ in our syntax and semantics, but in practice a more efficient technique might be to use *Skolem functions* [3]. A Skolem function is just a way to replace an existential quantifier with a function symbol; e.g. $\forall x.\exists y.x = y$ becomes $\exists f.\forall x.x = f(x)$. The f can then be compiled down to a matrix variable symbol in our base syntax, using the techniques in Subsection 4.2.
   Thus mathematically, one way to view our matrix variable symbols is as Skolem symbols.
7. *Tree-structured data.* Tree-structured data is used pervasively in declarative programming. In light of this paper, it would be interesting to give labelled

[50] Let's read that out: if x is a root of $[\![\phi]\!]_\varsigma$ for every $\phi \in \Phi$ then it is a root of $[\![\psi]\!]_\varsigma$.

[51] See also Remark 3.2.12 which comments on the efficiencies of being able to remove repeated roots.

trees additive and multiplicative structure sufficient to do polynomial arithmetic directly over trees (something along these lines has been done [27]) so that, instead of working as we do in this paper with a ring of polynomials over rationals ($\mathbb{Q}[X]$), we could work with a ring of polynomials over trees: $\mathsf{Tree}[X]$. Integers can already emulate trees via a Gödel encoding, but if we had polynomials with coefficients that already *are* trees, then we might be able to avoid some encoding and decoding. This is speculative but it is an interesting motivation for future work.

Remark 6.1.2. How powerful is our logic, really? In fact, extremely powerful!

The syntax in Figure 1 has a term-former reify that maps predicates back down to terms. This reflects the fact that in our semantics, terms map to polynomials, and so do predicates (predicates map to *non-negative* polynomials, but a non-negative polynomial is still just a polynomial).

This is unusual and it makes the logic impredicative. We have called our logic 'first-order logic' — and this was not untrue since it comes with a specific intended model in which the variable X ranges over numbers. But what that left unsaid at the start, for simplicity, was that our logic *also* has an in-built mapping between predicates and terms; with reify going from predicates to terms, and $t{=}0$ going from terms to predicates (cf. Remark 2.4.6).[52]

So this is FOL, but not *just* FOL, and the full expressivity of the syntax in Figure 1, with its intended semantics in Figure 2 is a topic for future research.

6.2 More on efficiency

We discussed efficiency and succinctness in Remarks 3.2.12 and 3.4.6(2). We make a few more general observations, based on the following question:

How big do our polynomials get?

Remark 6.2.1.

1. When we succinctly check a polynomial equality

$$[\![\phi]\!]_{\varsigma}(q) = h_{\mathsf{C},\phi}(q) * \mathit{zeroes}_{\mathit{len}(\varsigma(\mathsf{C}))}(q),$$

   (*zeroes* is defined in equation (3) in Subsection 5.1) lower-degree polynomials are better because *a)* they are easier to evaluate and *b)* they have fewer roots (so that a succinct check is less likely to give a false positive).

[52] This is *not* a Gödel encoding! A Gödel encoding would inject raw predicate syntax into numbers. reify behaves more like a type casing function, mapping predicate semantics into term semantics — as a no-op, since both are just polynomials.

So we can ask what contributes to the degree of the polynomial that is $[\![\phi]\!]_{\varsigma}$. We discussed this in the context of a simple but paradigmatic example in Remark 3.2.12.

One important point is that $\land$ maps to addition, which does not increase degree — we can say: *conjunction is free.* Thus, succinctly checking a conjunction of validity predicates is no more expensive than checking the most expensive (= highest degree) clause.

Thus, a program consisting of a large number of matrix variable symbols with relatively simple, low-degree validity predicates, is better than a program with a smaller number of matrix variable symbols with more complex validity predicates. This fits in nicely with a modular programming philosophy, where we try to factor our system into a many relatively simple parts. The reader will know that factoring a system into smaller pieces, where possible, is generally accepted as good practice *anyway* — but here it has concrete mathematical implications: we get lower degree polynomials.

2. Having matrices with lots of rows is cheap, because rows do not directly increase the degree of polynomials. Long interpolation vectors cost, because (as we noted in Definition 2.1.4(2) and Remark 3.2.12(5)) a vector of length n requires in general a degree n-1 interpolating polynomial.

   So we want to design our validation predicates to create tall but narrow matrices.

3. So our question becomes:

   > *What leads to wide matrices?*

   Broadly speaking, this is bounded by how many *distinct* function calls are made overall ('distinct', because the system is naturally memoised; see Remark 3.4.6(2)).

   Counting such calls is a familiar task; for example, exponentiation a^b can be computed in $O(ln(b))$ distinct calls (as discussed in Remark 3.2.14). So after all this discussion, our conclusion is fairly straightforward:

   (a) computation and control flow are free, and
   (b) the cost of verification scales with the size of the *longest deduplicated derivation chain.*

   Broadly speaking, this corresponds to the inherent complexity of proving something, so we can say that our translation is efficient in the sense that it converts a logical problem into an arithmetic one of roughly equal (and thus not much larger) complexity.

6.3 Other comments

Remark 6.3.1 (Logic programming). Logic programming can be understood as the engineering of predicates and derivation rules to make the search for derivation-trees computationally efficient (an old but to-the-point research survey is in [20]).

Our examples in Section 3 show how we can encode computation as derivation-trees, as is standard in mathematically structured programming. A so-called *validity predicate* $\chi_{\mathtt{C}}$ expresses when a matrix $\varsigma(\mathtt{C})$ expresses a valid derivation-tree, and thus a valid computation. The reader can think of $\varsigma(\mathtt{C})$ as being a kind of 'generalised computation trace' (but remember this is very general, and in particular derivation trees need not be linear).

Given that this paper uses derivation-trees, and so does logic programming, is there a connection? Yes, and no.

For us in this paper, *finding* a valid derivation-tree / providing a suitable ς, is for the prover. Many methods for generating derivation-trees exist — including logic-programming systems, but also rewrite systems, or just by some suitable evaluation strategy. What is true is that logic programming is one way to efficiently construct derivation-trees, and our semantics consumes them, so there is a natural match here.

Remark 6.3.2 (Why matrices?). Why do we need matrices variable symbols in Figure 2? Why not just encode all our information into natural numbers, as we do e.g. when we use a Cantor pairing function to Gödel encode combinator terms in Definition 3.9.4? Why not just do this or something like it throughout?

We could try, but a key issue is our use of *pointers*. The reader can see in the examples in Section 3 how we use entries in our matrices to point to columns in matrices (we make an effort to really spell this out in Examples 3.2.8 and 3.2.9). If we Gödel encode all structure in natural numbers, then this would be harder to do because we would have to unpack those pointers from the encoding before using them. This is not necessarily impossible, but it would add layers of complexity which for now we prefer to do without.

To put this another way: we use derivation-trees a lot in our semantics, and trees are graphs, and graphs are labelled nodes with edges. There is a natural structural map from here to matrices, and thence to interpolating polynomials and succinct proofs in the cryptographic sense. Other design decisions may be possible, but this paper follows a(n in retrospect) natural structural path through the mathematics.

Remark 6.3.3 (FOL truth vs. FOL derivability). This paper maps FOL judgements directly to polynomial equalities (Figure 2), such that valid (true) judgements map to valid (true) equalities.

Another approach to encoding FOL is to map its syntax and derivation-trees to polynomials with a circuit that identifies the (encodings of) well-formed derivation trees. That is also a good approach, but it encodes *derivability*, not *validity*.

To see how these differ, consider the FOL predicate 0=0. This is valid, since 0 *is* equal to 0; and it is derivable by the equality right-derivation rule. As per Figure 2 $[\![0=0]\!]_\varsigma = 0$, which represents 'true'. Thus, $[\![0=0]\!]_\varsigma$ is true because it literally *is* true; indeed no reasoning or circuitry is required in this case; it is just a fact! An arithmetisation of FOL that uses derivability would encode a (simple) derivation tree like this

$$\frac{}{\vdash 0=0}\,(=\mathbf{R})$$

along with a circuit expressing that this is well-formed.[53] It makes sense, but it is a different approach.

It is the difference between asserting '$\forall x.\exists y.x=y$ is true in some finite model' and 'I have a valid derivation-tree of $\forall x.\exists y.x=y$'. The former is a (true) assertion about concrete behaviour in finite models (finite, because ς is unbounded, but finite), whereas the latter is an assertion about derivability.

For computation, finite models are what matters most, so our design choice is natural. We labour this point because derivation-trees still feature in this paper but they arise in the spirit of being generalised computation traces; the embedding of the FOL logic remains semantic and direct as discussed above. An 'infinite models' version of the ideas in this paper, possibly working through explicit consideration of FOL derivation-trees, is possible future work.

REMARK 6.3.4 (Rapid coprocessors). We could use the ideas in this paper for fast and efficient prototyping of a correct-by-construction *coprocessor* for an existing system: to create some function, specified and arithmetised using the techniques in this paper, for the specific purpose of being called from some other zero-knowledge abstract machine.

We have seen how the compositional nature of our semantics lends itself — by design — to specifying functional programs, which require no external state

[53] Note that this would require us to think about how to represent the syntax of FOL propositions and the syntax of FOL derivation-trees, in polynomials. This is possible but it's just an extra layer of complexity.

We can turn this around: the relative straightforwardness of the representation in this paper is a feature, not a bug, and it is no accident but the result of specific design decisions. Of course we hope there will be scope to make things *even simpler* and more direct, and if so this is future work.

and which are correct by construction.[54] Functional components are modular and compositional, and therefore portable and well-suited to being plugged together.

This compositionality could be useful and may mean that the barrier to extracting practical value from these ideas may be relatively low.

Furthermore, pure speed is not the only metric of success: shorter development time, greater modularity, readability, maintainability, portability, and amenability to formal verification of correctness, are also valid criteria, amongst others. High-level specifications tend to perform well by these metrics, so a compositional shallow translation directly to polynomials is interesting.

6.4 Final remarks

The fundamental substrate of cryptography is *polynomials*, whereas the fundamental substrate of logic and computation is formal logic and Boolean semantics. First-order logic is a simple but powerful logic and in this paper we have given it a polynomial-based notion of validity.

Specifically, we map predicates to polynomials, and FOL validity judgements to polynomial equalities. At a high level, the mapping is simple:

1. Truth maps to zero, and falsity maps to one.
2. $\wedge$ and $\forall$ map to $+$, and $\vee$ and $\exists$ map to $*$ (see Figure 2 and the discussion in Remark 3.2.12).

Then simple facts of arithmetic — notably that a sum of two nonnegative numbers is zero if and only if both numbers are zero (Lemma 2.4.2) — give us a soundness and completeness result (Theorem 2.4.4).

It is perhaps surprising that from this elementary beginning, we manage to pull out so much logical and computational content.

Using matrix variable symbols and some simple logic, we specify a library of exemplar functions (Sections 3 and 4), including the Turing-complete SK-combinator system.

In the judgement $\vDash_\varsigma \phi$ for ϕ a closed predicate in Definition 2.3.2(4):

- The predicate ϕ corresponds to what cryptographers would call the *instance*. This is publicly known data that determines the problem to be solved.
- The interpretation ς of the matrix variable symbols (ς maps them to matrices, and thus to polynomials via interpolation) corresponds to what cryptographers would call a *witness*. This information is known to the prover but not necessarily to anyone else.

[54] Correct by construction with respect to the specification. If the specification is buggy then the program constructed from it will be too. But it will faithfully satisfy the (incorrect) specification.

The point of cryptographically verified computation is to devise ways for the prover, given a ϕ, to efficiently (the technical term is *succinctly*) and perhaps confidentially (in *zero knowledge*) prove knowledge of a witness ς that makes ϕ valid.

Note that the interpretation ς may have arbitrary size; the size of the interpretation ς and the size of the validity predicate ϕ are orthogonal parameters. Even a simple and compact validity predicate (like the one in Figure 8) can express the correctness of an arbitrarily complex computation trace as encoded in ς.[55] Furthermore, computations correspond to derivations (in the style of mathematically structured programming), which are trees: they are not restricted to being a linear trace. Finally, we note that the range check primitive acts in the context of our logic as a truth modality (see Remark 4.1.11) and using this we can encode even more interesting structure (see Section 4).

Our mapping of logical assertions to polynomial equalities (Definition 2.3.2(3)) is clean, compositional, and modular, and overheads seem to be pleasingly low. We get a nice mapping from derivation-trees to matrices, and modularity in the specification corresponds to relatively lower degree polynomials. Practical application is plausible both in and of itself, and as a coprocessor for other systems.

Because this system is based on logic, implementations are likely to be amenable to powerful techniques from computer science, including algebraic and logic-based optimisation techniques, formal verification, correctness-by-construction, and type systems. Although not everything in this paper is elementary, it is noteworthy that the complexities that arise come from *the applications*, and not from the translation of the logic, which is simple and direct. In other words: our logic and its polynomial translation do not seem to make a thing any harder than it inherently *already is*. For instance: a $\wedge$ turns into a $+$ and a $\vee$ turns into $*$, thus one connective = one arithmetic operation. It is hard to imagine a more transparent translation!

Finally, logics other than first-order logic exist, so implicit in the act of mapping of *this* logic to polynomials is that we might do the same for *others* — and conversely, there is a possibility for logicians to create interesting new logics inspired by polynomial semantics like the one we present here.

This paper spans two fields with long mathematical traditions: cryptography and formal logic. Cryptography has historically been about encryption — but thanks to the rise of distributed systems, cloud computing, and blockchains, the focus of interest in the field has come nearer to what logicians and programmers do. Thus cryptography is increasingly interested in, and also interesting to, the theory of logic

[55] This seems to be a point of confusion, so I will repeat it: a big *computation trace* does not necessarily mean that the *notion of validity of that trace* is particularly complex. Furthermore, the validity predicate ϕ only needs to be compiled to a polynomial once, and then we can check validity of as many interpretations that claim to validate ϕ, as we like.

and computation. I hope this paper can promote further collaboration and transfer of ideas between these fields.

References

[1] Ittai Abraham and Alin Tomescu. A simple and succinct zero knowledge proof. Tutorial blog post at https://decentralizedthoughts.github.io/2020-12-08-a-simple-and-succinct-zero-knowledge-proof/ (permalink), December 2020.

[2] Nada Amin, John Burnham, François Garillot, Rosario Gennaro, Chhi'mèd Künzang, Daniel Rogozin, and Cameron Wong. LURK: Lambda, the ultimate recursive knowledge. Cryptology ePrint Archive, Paper 2023/369, 2023.

[3] Jeremy Avigad and Richard Zach. The Epsilon Calculus. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Fall 2020 edition, 2020.

[4] Henk P. Barendregt. *The Lambda Calculus: its Syntax and Semantics (revised ed.)*, volume 103 of *Studies in Logic and the Foundations of Mathematics*. North-Holland, 1984.

[5] Daniel Benarroch, Matteo Campanelli, Dario Fiore, Kobi Gurkan, and Dimitris Kolonelos. Zero-knowledge proofs for set membership: efficient, succinct, modular. *Designs, Codes and Cryptography*, 91(11):3457–3525, Nov 2023.

[6] Katalin Bimbó. Combinatory Logic. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Winter 2020 edition, 2020.

[7] Manuel Capel. The fastest way to compute the Fibonacci sequence. Blog post at https://mancap314.github.io/fastest-fibonacci.html (permalink), August 2020.

[8] Jeffrey R. Chasnov. *Numerical methods*. Hong Kong University of Science and Technology, 9 2025. Course notes available online at https://batch.libretexts.org/print/Letter/Finished/math-96025/Full.pdf.

[9] George Danezis, Cédric Fournet, Jens Groth, and Markulf Kohlweiss. Square span programs with applications to succinct NIZK arguments. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology – ASIACRYPT 2014*, pages 532–550, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

[10] Ali Dasdan. Twelve simple algorithms to compute Fibonacci numbers, 2018. Available at https://arxiv.org/abs/1803.07199.

[11] Walter Dean and Alberto Naibo. Recursive Functions. In Edward N. Zalta and Uri Nodelman, editors, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Summer 2024 edition, 2024. https://plato.stanford.edu/entries/recursive-functions/ (permalink).

[12] Atze Dijkstra, José Pedro Magalhães, and Pierre Néron. Functional programming in financial markets (experience report). In *Proceedings of the International Conference*

on Functional Programming 2024 (ICFP 2024), volume 8, pages 234 – 248, New York, NY, USA, August 2024. Association for Computing Machinery. Article number 244, available at https://doi.org/10.1145/3674633.

[13] Albert Garreta, Hendrik Waldner, Katerina Hristova, and Luca Dall'Ava. Zinc: Succinct arguments with small arithmetization overheads from IOPs of proximity to the integers. Cryptology ePrint Archive, Paper 2025/316, 2025. Available at https://eprint.iacr.org/2025/316.

[14] Stephen C. Kleene. Recursive predicates and quantifiers. *Transactions of the American Mathematical Society*, 53:41–73, 1943. Available online at https://www.ams.org/journals/tran/1943-053-01/S0002-9947-1943-0007371-8/S0002-9947-1943-0007371-8.pdf (permalink).

[15] Jan-Willem Klop, Vincent van Oostrom, and Femke van Raamsdonk. Combinatory reduction systems, introduction and survey. *Theoretical Computer Science*, 121:279–308, 1993.

[16] Tony R. Kuphaldt. Gate universality. In *Lessons in Electric Circuits*. All about circuits, Accessed 2024. Free online textbook, available at https://www.allaboutcircuits.com/textbook/digital/chpt-3/gate-universality/ (permalink).

[17] LambdaClass blog. Our highly subjective view on the history of zero-knowledge proofs. https://blog.lambdaclass.com/our-highly-subjective-view-on-the-history-of-zero-knowledge-proofs/ (permalink, February 2024. Accessed: 2025 08 16.

[18] Xi Lin, Heyang Cao, Feng-Hao Liu, Zhedong Wang, and Mingsheng Wang. Shorter ZK-SNARKs from square span programs over ideal lattices. *Cybersecurity*, 7(1):33, Mar 2024.

[19] Richard Lipton. The curious history of the Schwartz-Zippel lemma. Online blog post available at https://rjlipton.com/2009/11/30/the-curious-history-of-the-schwartz-zippel-lemma/ (permalink), November 2009.

[20] Dale Miller, Gopalan Nadathur, Frank Pfenning, and Andre Scedrov. Uniform proofs as a foundation for logic programming. *Annals of Pure and Applied Logic*, 51(1):125–157, 1991. Available at https://doi.org/10.1016/0168-0072(91)90068-W.

[21] Melvyn B. Nathanson. Cantor Polynomials and the Fueter-Pólya Theorem. *The American Mathematical Monthly*, 123(10):1001–1012, 2016. Available at https://doi.org/10.4169/amer.math.monthly.123.10.1001.

[22] Nayuki. Fast Fibonacci algorithms. Blog post at https://www.nayuki.io/page/fast-fibonacci-algorithms (permalink), January 2023.

[23] Mike O'Donnell. The SKI combinator calculus: a universal formal system. Online course notes (permalink), May 2021. Accessed: 2025 08 16.

[24] Liam Proven. War of the workstations: How the lowest bidders shaped today's tech landscape. Online article at https://www.theregister.com/2023/12/25/the_war_of_the_workstations/ (permalink), December 2023.

[25] Abner J. Salgado and Steven M. Wise. Polynomial interpolation. In *Classical Numerical Analysis: A Comprehensive Course*, page 231–265. Cambridge University Press, 2022.

[26] Berry Schoenmakers. Σ-protocols. In Henk C. A. van Tilborg and Sushil Jajodia, editors, *Encyclopedia of Cryptography and Security*, pages 1206–1208. Springer US, Boston, MA, 2011.

[27] Paul Tarau. Computing with catalan families, generically. In Marco Gavanelli and John Reppy, editors, *Practical Aspects of Declarative Languages*, pages 117–134, Cham, 2016. Springer International Publishing.

[28] Justin Thaler. *Proofs, Arguments, and Zero-Knowledge*. Number 4:2–4 in Foundations and Trends in Privacy and Security. Now publishers, December 2022. Available online at https://people.cs.georgetown.edu/jthaler/ProofsArgsAndZK.pdf (permalink).

[29] Dirk van Dalen. Intuitionistic logic. In D.M. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic, 2nd Edition, Volume 5*, pages 1–114. Kluwer, 2002.

[30] Rineke (L.C.) Verbrugge. Provability Logic. In Edward N. Zalta and Uri Nodelman, editors, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Summer 2024 edition, 2024.

[31] Wikipedia. General recursive function. http://en.wikipedia.org/w/index.php?title=General%20recursive%20function&oldid=1303228064, 2025. Accessed 16 August 2025.

Received August 2025

AI beyond ChatGPT

Artur d'Avila Garcez
City St George's, University of London

Abstract

More than two years since the release of GPT4 and following the recent underwhelming rollout of GPT5, the debate around the risks of AI has cooled off. Leading figures continue to disagree on what needs to be done: some claim that Big Tech is best placed to take care of AI safety, others argue in favor of open source, and others still call for immediate regulation of AI and social media. As society contemplates the impact of AI on everyday life — with hundreds of millions of users of large language models taking part in what feels like the world's largest experiment - the technological innovations that led to GPT5 receive less and less attention. But the technology is central to the study of the risks and opportunities of AI. The opacity surrounding AI technology contributes to fears of an upcoming AI bubble burst. Without a clear understanding of the technology and Big Tech's use of data, regulatory efforts are in the dark. In this opinion article in honor of Dov Gabbay's 80th birthday, I seek to refocus attention on the achievements and limitations of AI technology. I argue that the emerging field of neurosymbolic AI can address the problems of current AI: lack of fairness, reliability, safety and energy efficiency. Geoffrey Hinton suggested recently that one should *treat AI like mothers who are wired to want the best for their (less intelligent) babies*, the users of AI. The field of neurosymbolic AI has been concerned for many years about how to *wire* prior knowledge into neural networks. I will review progress on neurosymbolic AI towards achieving fairness, reliability and safety via such wiring of knowledge within a broader AI accountability ecosystem. I will also point out how the application of the neurosymbolic cycle, translating neural networks into logic and vice-versa, enables efficiency as an alternative to scaling-up. On a personal note, I thank Dov for allowing me to pursue my then unorthodox ideas in neurosymbolic AI when I was his PhD student more than 25 years ago. Dov would always offer the best advice on the role of logic in AI, ask the difficult question of the benefit that neural networks would bring to logic and, most importantly, be open to new ideas from adjacent fields and new research directions, a trait that many AI researchers require today.

I thank Chris Percy, Simo Dragicevic, Luis Lamb and Moshe Vardi for valuable discussions on accountability in AI and for their comments on earlier versions of this article.

1 Five days of chaos

The New York Times, now suing GPT5's owner OpenAI for copyright violation, best summarized the drama that unfolded at OpenAI after the release of GPT4 as *five days of chaos*[1]. An OpenAI board member had written an article stating "OpenAI has also drawn criticism for many other safety and ethics issues related to the launches of ChatGPT and GPT4, including regarding copyright issues, labor conditions for data annotators, and the susceptibility of their products to jailbreaks that allow users to bypass safety controls" [3]. Following the departure of key technical talent, some of whom went on to create an AI safety company, little seems to have changed in OpenAI's governance or accountability approach. ChatGPT's lack of reliability, fairness and data efficiency has been noted many times since [12], with perhaps the most prominent example being the case of the law professor falsely accused of sexual harassment. I will argue that this reliability problem cannot possibly be solved *case by case* or *after the event*, although after seeing so many cases we seem to have become desensitized to the problem. There must be a better way of achieving AI that can offer certain guarantees to model alignment with a lower financial and human cost. I will emphasize the need for an accountability ecosystem as proposed in [10]. I will focus on how technology can be leveraged to promote accountability in AI. A lot of the technological claims from two years ago hinged on RLHF (Reinforcement Learning with Human Feedback). We now know that RLHF is both unethical, exposing data labellers to the worst of the internet, and too costly. Some of the competition to ChatGPT such as DeepSeek adopted a different approach based on *distillation*, something that I will discuss later and that is closer to the neurosymbolic approach.

2 Risk and Accountability in AI

Large Language Models (LLMs), including OpenAI's ChatGPT, Meta's Llama and Google's Gemini, can all be viewed as very large computer programs. The programs are learned from examples: inputs and their desired outputs mapped to vector representations (embeddings) given very large amounts of data scraped from the internet. The learned program function f is capable of producing coherent sentences of text in response to human interaction (prompts). LLMs produce answers — right or wrong — to any given question. They show reasoning capability and can offer explanations to the answers and, most impressive of all, can produce code in

[1] `https://www.nytimes.com/2023/11/22/technology/how-sam-altman-returned-openai.html`

a chosen programming language given prompts in natural language. LLMs are a great engineering achievement, are excellent at text summarization and language translation. They may help improve productivity of anyone who is diligent and sufficiently knowledgeable to check when the LLM might have made a mistake, and yet they have great potential to deceive all those who are not diligent or sufficiently knowledgeable. As an auto-regressive model, LLMs do everything they do by doing only one thing: predicting the probability of the next word token x_{t+1} in a sequence of words, given the current tokens x_t at time t, where $x_{t+1} = f(x_t)$. Having made a choice of the next token with the highest probability, LLMs will apply the same program function f again and again, recursively, to build a sequence of many tokens.

Should tech companies have been allowed to release LLMs worldwide to hundreds of millions of users and collect their data without any external scrutiny? There are various technical and non-technical reasons why current AI, using very large-scale transformer neural networks to learn the function f, may not be deployed in this way: lack of trust or fairness, reliability issues and public safety. Fixing reliability issues case-by-case with RLHF has proven to be too costly, financially and in human terms, as already pointed out. A commonly-adopted risk mitigation strategy became known as the *human-in-the-loop* approach: making sure that a human is ultimately responsible for any decision making. In such cases, the AI system is seen only as an assistant to the humans who are supposed to be in control. However, the situation is more nuanced. Simply apportioning blame or liability to humans does not solve the problem. It is necessary to empower the user of AI, the data scientist and the domain expert, to interpret the answers, ask *what-if* questions, and if necessary intervene in the AI system. Having very large and opaque LLMs doesn't empower the user. Here, the technology of explainable AI has an important role as we try to make sense of large neural networks, as discussed later.

Consider an LLM's ability to produce code. If ChatGPT was allowed to work, not as a stand-alone computer program function f, but in a loop whereby the code that is generated can be executed automatically, data is collected from the execution, and function f is updated to seek to improve the code given the data, one can see straight away how such self-improving LLM with decision making autonomy - termed agentic AI - may pose a serious risk to current computer systems. Recent experiments indicated that at present this loop of automatic synthetic data generation can create the opposite effect, self-impairing, producing a degradation in performance [11]. Agentic AI, therefore, will require guardrails once it is allowed to take action on your behalf, even if that action is something as simple as organizing a holiday, paying for the air tickets and deciding on the sightseeing tour schedule. The idea of a self-improving loop blurs the distinction that exists currently between model training and test-time compute. Up until now, the separation between training and run-time has been a very useful tool in the engineering of large AI models.

3 AI challenges: reasoning, efficiency, fairness

When LLMs make stuff up such as referring to non-existing articles in the above-mentioned case of the law professor, they are said to hallucinate. AI will require systems that never hallucinate, that reason reliably, that can handle novelty and treat exceptions well given less data. This is very different from the current *scale-is-all-you-need* LLM approach. Two years on, LLMs continue to hallucinate even with the cost of post-hoc model alignment with RLHF skyrocketing as performance seems to plateau on benchmarks. In the case of so-called reasoning models, claimed to "think before it answers" or to be capable of "truly general reasoning", little is known about their inner workings. Let's assume that such models are a kind of "GPT-Go": a generative pre-trained transformer to which test-time compute is incorporated as a search process in the style of Google DeepMind's earlier Alpha-Go system. The search uses "Chain of Thought" (CoT) prompting: generation of synthetic data using the transformer itself in a chain that breaks down a prompt into sub-prompts. It is reasonable to expect an increase in test-time compute to improve reasoning performance because reasoning tasks are typically solved by breaking down a problem into sub-problems. In fact, neurosymbolic approaches have shown improvement in reasoning performance when learning is combined with search-based reasoning. In [7], knowledge distillation is used to build a search tree from a trained neural network, with the search tree used to carry out formal reasoning at test time. The problem with the "GPT-Go" approach is the lack of reliability of the synthetic data generation, known as the curse of recursion [11], and the combinatorial nature of the CoT input, best described as *infinite uses of finite means* (a finite dictionary giving rise to infinite possible texts), that is, the well-known problem that small changes in input may produce diverging outputs due to the inevitable accumulation of errors in the calculations of a neural network. As a result, CoT may solve one reasoning task today, only to fail at an analogous reasoning task tomorrow. This is illustrated by the examples provided in [6] showing that a mere change in naming convention can affect reasoning performance dramatically. What we see in practice is that eliminating so-called hallucinations is very hard, if not impossible, and it takes only one bad hallucination to destroy trust forever.

In neurosymbolic AI, based on a long tradition of combining neural networks with logic, the goal is to ascribe meaning to neural computation by offering a formal semantics to neural networks. Reasoning is provably sound rather than benchmark dependent. Instead of merely adjusting the inputs of the network (the CoT approach), the neurosymbolic approach designs the network architecture or the training loss function based on symbolic knowledge that is either learned or already known. The intended results are more confidence in the outputs of the network and a parsimonious learning that can benefit from the combination of data and knowledge in a more modular network.

Neurosymbolic AI integrates learning and reasoning in model development by following a development cycle known as the neurosymbolic cycle: (i) extract symbolic knowledge descriptions from partially trained networks, (ii) reason formally about what has been learned, (iii) compress the network by instilling consolidated knowledge back into the network before further training with data. Reasoning in neurosymbolic AI follows the tradition of knowledge representation in AI, founded on logic and formal definitions of a semantics for deep learning [9], rather than based on informal evaluations of reasoning capabilities. Evaluating neural networks with respect to formally-defined, sound or approximate reasoning allows for the much needed controlling of the accumulation of errors. The use of distillation is a step in the direction of neurosymbolic AI in that distillation is a form of knowledge extraction to obtain network compression (steps (i) and (iii) of the neurosymbolic cycle). The use of test-time compute is also a step in the direction of neurosymbolic AI because reasoning is, in essence, implemented in a computer by means of a search process (step (ii) of the neurosymbolic cycle). However, there are many forms of knowledge representation and reasoning to be mapped onto neural networks (analogy, modal, epistemic and higher-order logic, temporal, normative, abductive and abstract reasoning [2]) and distillation without explicit knowledge, that is, without a symbolic description and semantics, is limited in what it can offer to reliability, explanation, knowledge reuse and transfer learning. Broadly speaking, neurosymbolic AI is tasked with the theory, algorithms and tools capable of establishing a principled understanding of the correspondences that exist between neural and symbolic representations. Ultimately, the intended outcome of neurosymbolic AI is to achieve safety via verifiable descriptions of network modules, energy efficiency via parsimonious learning from data and knowledge, fairness by imposing requirements specified in logic, and trust by empowering the users of AI with the help of explainability.

4 Desiderata for neurosymbolic AI

Addressing the challenges of data efficiency, safety, fairness and, ultimately, trust in AI will require breakthroughs in neurosymbolic AI. Neural computation has shown with deep learning that neural networks must be the substrate of AI, the foundation layer upon which AI is implemented. But, each of the above problems with deep learning have been stubbornly difficult to fix despite a huge investment. Already in the mid 1990's and early 2000's, the importance of neural computation as the substrate of AI was obvious to a small group of researchers advocating neurosymbolic AI. The value of symbol manipulation and abstract reasoning offered by symbolic logic was also obvious to that small group. It could be argued, however, that neurosymbolic AI starts together with connectionism itself, with McCulloch and Pitts's 1943 paper *A Logical Calculus of the Ideas Immanent in Nervous Activity*, which influenced John Von Neumann's 1952 *Lectures on Probabilistic Logics and the Synthesis of Reliable Organisms from Unreliable Components*, both indicating that the separation between distributed vector representations (network embeddings) and localist symbolic representations (logic) did not exist at the time. Even Alan Turing's 1948 *Intelligent Machinery* introduced a type of neural network called a B-type machine. It was not until after the term *Artificial Intelligence* was coined by John McCarthy, admittedly for the sake of securing funding[2] ahead of the now famous Dartmouth Workshop in 1956, that the field separated into two: symbolic AI and connectionism (or neural networks). This has slowed down progress in AI as the research community went in two separate ways with their own conferences, journals and associations. Following the huge success and subsequent decline of symbolic AI (the first wave of AI) and the huge success of deep learning since 2015 (the second wave of AI) with its now obvious limitations on fairness, reliability, safety and energy efficiency, neurosymbolic AI is finally regarded by many as *the third wave of AI* [1].

Neurosymbolic AI uses knowledge extraction to explain and, if needed, intervene in an AI system. The goal is to apply the neurosymbolic cycle to control the learning process in ways that can offer correctness or fairness guarantees to the neural network, producing a more compact and hopefully more efficient network in the process. The 2024 Chemistry Nobel prize-winning AI system *AlphaFold* from Google DeepMind is arguably the greatest achievement of AI to date. AI systems of this kind (in the case of *AlphaFold*, a system for protein structure prediction) hold the promise to cure many diseases. They are application-specific systems also known as *narrow AI* compared to LLMs, which are intended to be generalist. From particle physics to drug synthesis, energy efficiency and novel materials, AI is being adopted as the new language of scientific discovery. However, failure to provide a description capable of conveying a deeper sense of understanding of the solutions being offered by AI can be very unsatisfactory. In a closed environment, such as a board game, large-scale simulation may be sufficient to get a neural network to learn to reason. In open-ended situations, the reasoning task is much harder than data-driven reasoning

[2] See the Lighthill debate on Artificial Intelligence: `https://www.youtube.com/watch?v=03p2CADwGF8`.

by similarity, requiring at least for now the use of explicit descriptions. An explicit description is one that can be manipulated by asking the question *what might happen if I were to make this change?*, without making the change. Hence, such description is required to be amenable to symbolic manipulation. AI will soon require systems that adapt to novelty from only a few examples, that check their understanding, that can multi-task and reuse knowledge to improve data efficiency and that can reason in sophisticated ways using first-order and higher-order logic. Adapting to novelty requires an ability to create abstract, simple representations (whether this happens in the brain or in the mind) but also to change representation from time to time [4]. Change of representation allows looking at a problem from a different angle to obtain new insight given an analogous situation. This forms the core challenge of the latest research in neurosymbolic AI: (i) extraction of relevant descriptions from complex, very large networks at an adequate level of abstraction for the application at hand, (ii) sound reasoning and learning with various representations: spatial, temporal, abstract, relational and multimodal [2], (iii) data and knowledge reuse with modularity to extrapolate efficiently to multiple tasks in different domains of application. The next decade of research in neurosymbolic AI should make key strides towards addressing the challenge.

5 Avoiding a race to the bottom

Now that the AI race is on, influential leaders have been arguing for more investment in safety research or regulation of the AI industry. It should be obvious that worldwide regulation is not achievable in the current geopolitical climate [5]. An alternative to regulation has been put forward in [10]: digital technology itself, as part of an adequate AI accountability ecosystem, can offer a new path to a more parsimonious AI where neural models are more modular, trained with fewer data and validated symbolically during multiple stages of model development. This is quite different from what the EU AI Act has achieved. Regulation without accountability increases risks by encouraging weak competitiveness.

A proper accountability ecosystem can avoid a race to the bottom by mapping out general principles into implementation of industry processes using mechanisms such as internal auditing, external accreditation, investigative journalism, a risk-based regulation approach and market shaping. In [10], as early as 2021, it was stated: “at present the ecosystem is unbalanced, which can be seen in the failures of certain mechanisms that have been attempted by leading technology companies. By taking an ecosystem perspective, we can identify certain elements that need developing for the system as a whole to function effectively. Corporate governance mechanisms such as standardized processes and internal audit frameworks, leading to potential external accreditation, need to be made to work together in ways that go beyond regulatory requirements, especially in technologies' early period of evolution and deployment when regulation lags practice." Key stakeholders in this process include corporate actors, market counterparts, academia, civil society and government.

An AI system to predict harm from online gambling was used as a case study in [10] because of the high regulatory focus, divergent regulatory perspectives worldwide and longstanding debates over ethical dilemmas in gambling. Two key elements of the accountability ecosystem were discussed: (i) interventions to reduce bias and (ii) increased transparency via model explainability. The benefits of having an industry-specific accountability process were illustrated to the extent that it can be documented, reviewed, benchmarked, challenged and improved upon, both to build trust that the underlying ethical principle is being taken seriously and to identify specific areas to do more. Results were drawn from the risk profiling of gambling behavior: symbolic knowledge was extracted from a neural network predicting problem gambling and the explainable AI (XAI) technology was evaluated on indirect gender bias and algorithmic fairness metrics. It was argued that effective regulation requires accountability, the adoption of a risk-based approach, and the definition of a risk-mitigation strategy informed by objective metrics, such as fidelity of knowledge extraction when mapping neural and symbolic processes with the use of XAI [13].

6 Conclusion

I argued that achieving accountability in AI with data efficiency, fairness and safety will require a new approach based on neurosymbolic AI. Although I am convinced that neurosymbolic AI can make AI fairer, safer and more energy efficient, it is difficult to see how the use of technology alone could solve the problem of misinformation at scale polluting the internet with unreliable AI-generated data. Although the widespread adoption of AI technology should be celebrated in its potential to increase productivity, current GPT-style AI will also magnify errors as users become complacent. Malicious users have been spreading misinformation at a fast pace while rushed regulation without accountability has failed to protect the public. In particular, lessons were not learned from the failures to protect children from harm in the context of social media.[3] As humanity has to adapt on-the-fly to *the big LLM*

[3] See the US Senate hearing on protecting children online: `https://www.commerce.senate.gov/2021/10/protectingkidsonline:testimonyfromafacebookwhistleblower`

experiment, AI-based errors and misinformation seem to have to be accepted as a "fact of life". Controlling the large-scale spreading of misinformation is no longer in the command of any social media platform and the cost of checking for misinformation is increasing considerably. At the heart of the problem is a business model that has caused turmoil in corporate media, where software is perceived to be free, when in fact it is paid for with data. Alternatives are needed to this business model while concerns around freedom of speech, the inadequacy of current incentives and other moral dilemmas prevent companies and governments from making changes.

One idea is to treat data as fiduciary money, the idea of *data banks*, allowing users to pay for bits consumed in exchange for privacy, while paying users for bits produced, the term *bits* being used here as defined in Shannon's information theory. For this to work, payment systems would have to scale-up to allow for micro-payments to take place at close-to-zero transaction cost, while also addressing the problem of unique digital identity. Given a choice, people may prefer, differently from the subscription model, to *pay-as-you-go* for bits of add-free information. As AI moves from the academic labs into everyday life, new ideas and ways of doing the things that we take for granted will need to be debated, decided upon and implemented quickly. Mechanisms of technology-enabled local direct democracy [8] may help communities manage this huge transformation.

AI is not only changing the world of employment, but also education. Learning at schools and universities will need to change to instill a culture of critical and creative thinking, of learning from the history of science, and to cultivate the values of scientific inquiry, promoting skeptical interrogation based on sound principles of uncertainty quantification. AI will need to be taught at schools with the goal of creating a more discerning and informed society. Leaders, decision makers and domain experts should probably also learn the basics of AI. It has taken society many years to learn to distinguish genuine and malicious websites. Learning whether or not to trust the output of LLMs is much harder and will require the help of technological advancements such as explainable and neurosymbolic AI, but also a new economic and social contract, empowering local democracy, requiring fast policy decisions in a very fast-changing world.

References

[1] Artur d'Avila Garcez and Luís C. Lamb. Neurosymbolic AI: the 3rd wave. *Artif. Intell. Rev.*, 56(11):12387–12406, 2023.

[2] Artur d'Avila Garcez, Luis C. Lamb, and Dov M. Gabbay. *Neural-Symbolic Cognitive Reasoning.* Springer, 2008.

[3] Andrew Imbrie, Owen Daniels, and Helen Toner. Decoding intentions. `https://cset.georgetown.edu/publication/decoding-intentions/`, October 2023. Center for Security and Emerging Technology [Online; accessed 20-Jan-2025].

[4] Daniel Kahneman. *Thinking, fast and slow.* Farrar, Straus and Giroux, New York, 2011.

[5] Chris Miller. *Chip War: The Fight for the World's Most Critical Technology.* New York, Scribner, 2022.

[6] Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. GSM-symbolic: Understanding the limitations of mathematical reasoning in large language models, 2024.

[7] Kwun Ho Ngan, James Phelan, Esma Mansouri-Benssassi, Joe Townsend, and Artur S. d'Avila Garcez. Closing the neural-symbolic cycle: Knowledge extraction, user intervention and distillation from convolutional neural networks. In *Proceedings of the 17th International Workshop on Neural-Symbolic Learning and Reasoning, Siena, Italy, July 3-5, 2023*, volume 3432 of *CEUR Workshop Proceedings*, pages 19–43. CEUR-WS.org, 2023.

[8] Ritesh Noothigattu, Neil Gaikwad, Edmond Awad, Sohan Dsouza, Iyad Rahwan, Pradeep Ravikumar, and Ariel Procaccia. A voting-based system for ethical decision making. In *Proc. AAAI Conference on Artificial Intelligence, (AAAI-18)*, pages 1587–1594. AAAI Press, 2018.

[9] Simon Odense and Artur d'Avila Garcez. A semantic framework for neurosymbolic computation. *Artif. Intell.*, 340:104273, 2025.

[10] Chris Percy, Simo Dragicevic, Sanjoy Sarkar, and Artur d'Avila Garcez. Accountability in AI: From principles to industry-specific accreditation. *AI Commun.*, 34(3):181–196, January 2021.

[11] Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Yarin Gal, Nicolas Papernot, and Ross J. Anderson. The curse of recursion: Training on generated data makes models forget. *CoRR*, abs/2305.17493, 2023.

[12] Benedikt J. Wagner and Artur d'Avlia Garcez. A neurosymbolic approach to AI alignment. *Neurosymbolic AI journal*, 1:1–12, August 2024. IOS Press.

[13] Adam White and Artur S. d'Avila Garcez. Measurable counterfactual local explanations for any classifier. In Giuseppe De Giacomo et al., editor, *ECAI 2020 - 24th European Conference on Artificial Intelligence, 29 Aug - 8 Sep 2020, Santiago de Compostela, Spain*, volume 325 of *Frontiers in Artificial Intelligence and Applications*, pages 2529–2535. IOS Press, 2020.

Received September 2025

Combining Tableaux for Combinations of Logics
I. Generic Fibring of Tableaux

Valentin Goranko
Department of Philosophy, Stockholm University
valentin.goranko@philosophy.su.se

Dedicated to Dov Gabbay, from whom I have learned so much,
on the occasion of his 80th anniversary.

Abstract

This is the first paper in a series exploring the following generic question for a given type of combination of logics:

Given tableaux systems for two logics and a combination of these logics of a given type, how to combine systematically and uniformly the tableaux systems for component logics into a tableaux system for the combined logic by preserving important properties of the components?

Here I consider the case of general fibring of logics, introduced by Dov Gabbay. Starting with the basic cases of propositional merger and simple nesting of logics, I present natural generic versions of fibring of tableaux for these constructions, illustrate them with some examples, and establish respective results on preservation of soundness, completeness, and termination from the component tableaux to the combined tableau. Then I extend the combined tableau construction to the general case of fibring by iterating the basic cases and mention some potential applications.

1 Introduction

Combinations of logics have been considered since the emergence of composite non-classical logical systems began in the mid/late 1900s and several types of combinations of logical systems have been explored since the late 1980s, but that research area was properly shaped and systematically developed in the pioneering works of Dov Gabbay in the early 1990s [14, 15, 16], where the first general results in that area were published (cf also [1, 17], and references therein). Since then research on various combinations of logics has flourished, witnessed by the series of proceedings of the biennial symposium on Frontiers of Combining Systems (FroCoS) held since 1996 and, inter alia, by the handbook chapter on combining modal logics [22], representative of the then current state of affairs in that area.

The most general types of combinations of logical systems studied so far are *fusions*, *fibrings* and *products* [16, 13]. There are also several more specific types of combining logical systems, including *multi-modal*, *synchronizing*, *temporalising*, *epistemising*, *multi-agenting*, and *first-order extensions of propositional logics.*

This is the first paper in an intended series exploring the following general question for a given type of combination of logics:

> *Given tableaux systems for two logics and a combination of these logics of the given type, how to combine systematically and uniformly the tableaux systems for component logics into a tableaux system for the combined logic by preserving important properties of the components?*

There are two main approaches to combining tableaux:

1. Building directly a tableaux system for the combined logic.
2. Combining the existing component tableaux into a new tableau-like system of relatively autonomous and possibly interacting tableau components.

Both approaches have pros and cons, but I argue that the latter one, which I explore here, is superior, for several reasons:

- it is easier to construct and to prove preservation results, by following a general methodology rather than ad hoc techniques;
- it is modular and more flexible;
- it can also produce more efficient tableaux systems.

There have been many studies and results on *combining axiomatic systems*, and also many on *tableaux for combined logical systems*, yet *very few* (to our knowledge) on *combining tableaux* for logical systems. Except for the pioneering paper [2] and related two chapters in [16], that topic seems surprisingly under-explored. To begin with, here is a quote from the 2003 book [13]:

> *Thus, we need a spectrum of methodologies providing us with means of combining homogeneously given logics (say, tableau systems) and perhaps meta-methodologies for combining methodologies. These challenging problems are far beyond the scope of this book; many of them are still open for investigation.*

Not much progress has been made, or at least not much has been published since then, on combining tableaux. In particular:

- the Handbook of Tableaux Methods [6] essentially does not mention combinations of logics.
- the chapter on "Combining Modal Logics" in the Handbook of Modal Logic [4] only says that the question will not be discussed there.
- Respectively, the chapter on "Modal Proof Theory" in [4] only mentions some important cases of combined modal logics (mostly, epistemic) but not the general question of combining tableaux.
- few sporadically appearing papers related to the problem (though, including some important early ones) have appeared in either of the proceedings series, of TABLEAUX and of FroCos.

But, is the problem of combining tableaux for combinations of logics relevant and important at all? I argue that it is, because many increasingly expressive and complex logical systems are being designed from components for which tableaux systems have already been provided. And, the series of papers initiated here endeavours to demonstrate that much more can (and should) be done on that problem. Furthermore, the methodology for combining tableaux essentially extends the very notion of a tableau system by naturally brining modularity into such systems for the combined logics.

Here I consider the case of *fibring of logics* [15, 16]. Starting with some basic cases, I gradually outline a general methodology for *fibring tableaux*, i.e., compositionally constructing tableaux for the natural and generic version of fibring of logics with unconstrained fibrings of their languages and semantics and obtain general results on preservation of soundness, completeness, and termination from the component tableaux to the combined tableau. I focus here on combining by fibring of modal logics (in a broad sense) and of modal logics with FOL.

The paper is organised as follows: In Section 2 I provide preliminaries on terminology, notation and some technical concepts related to combining logics and tableaux. In Section 3 I gradually present the construction of fibring of tableaux, starting with the basic cases of propositional merger, and then simple nesting of logics and their tableaux. I then show how the full fibring construction is iteratively reducible to these. In Section 4 I mention some specific applications of fibring of tableaux, for first-order modal logics and for temporal epistemic logics, and then I conclude with an outline of the main points of the present work and on the planned future work.

2 Preliminaries

The reader is assumed to have some background on classical logic and on propositional modal and temporal logics, e.g., within the relevant chapters respectively of [19] for classical logic, [3] for normal modal logics, and [18] for temporal logics. Still, for the reader's convenience I include here a concise summary of basic terminology and notation related to logics and tableaux that will be used in the paper.

2.1 Logics, generically

The logical systems that I consider here are generically specified by:

- a *formal language*, with inductively defined set of *formulae* in it;
- a class of abstract *models*, defining **contexts of evaluation**, that include all entities needed to evaluate the truth of the formulae of the logic. For instance, for classical propositional logic these are *truth assignments*; for modal logic they are *pointed Kripke models*, i.e. pairs (Kripke model, a world in it); for classical FO logic – pairs (FO structure, variable assignment in it), etc.
- *formal semantics*, defining *truth of formulae* at contexts of evaluation;
- eventually, *satisfiability* of formulae, as truth at some context of evaluation, and *validity*, as truth at all contexts of evaluation.

I will focus here modal (in broad sense) and first-order logics.

2.2 Combining logical languages

Consider logical languages $\mathcal{L}_1$ and $\mathcal{L}_2$, usually with shared boolean connectives, and possibly also shared atomic propositions, plus some "private" logical connectives for each. The **fully combined language**, hereafter denoted by $\mathcal{L}_1 \uplus \mathcal{L}_2$, is based on the union of the vocabularies of $\mathcal{L}_1$ and $\mathcal{L}_2$, where the formulae are defined by applying all operators (shared and private) of both languages without restrictions:

$$\phi = \text{...the clauses for } \phi \text{ in } \mathcal{L}_1\text{...} \mid \text{...the clauses for } \phi \text{ in } \mathcal{L}_2\text{...}$$

Some natural fragments can be defined by restrictions on that definition, e.g. *propositional merger of* $\mathcal{L}_1$ *and* $\mathcal{L}_2$ (cf. Section 3.3) and *nesting of* $\mathcal{L}_2$ *in* $\mathcal{L}_1$ (cf. Section 3.4). Every such fragment of $\phi \in \mathcal{L}_1 \uplus \mathcal{L}_2$ defines a combined language, which will be generically denoted by $\mathcal{L}_1 + \mathcal{L}_2$.

2.3 Approximations of formulae

Given logical languages $\mathcal{L}_1$ and $\mathcal{L}_2$ with shared boolean connectives, by $\mathcal{L}_{1\setminus 2}$ I denote the set of *private* logical operators of $\mathcal{L}_1$, i.e., those in $\mathcal{L}_1$ but not in $\mathcal{L}_2$. The set $\mathcal{L}_{2\setminus 1}$ is defined likewise. We need a few technical concepts, defined below.

Definition 2.1. Given a formula ϕ in the combined language $\mathcal{L}_1 \uplus \mathcal{L}_2$:

1. An occurrence of a subformula ψ in ϕ is **1-external in** ϕ if it is not strictly in the scope of any operator from $\mathcal{L}_{1\setminus 2}$.
2. The formula ϕ is **monolithic**, if it is atomic or its main connective is non-boolean.
3. A subformula ψ in ϕ is **1-prime in** ϕ if it is monolithic, non-atomic, 1-external in ϕ, i.e., it has a main connective from $\mathcal{L}_{1\setminus 2}$, and is a maximal subformula of ϕ with these properties.
4. The **2-external** and **2-prime** occurrences of subformulae in ϕ are defined likewise, by swapping the indices 1 and 2 above.
5. The **1-approximation** of ϕ is a formula $\alpha_1(\phi) \in \mathcal{L}_1$ obtained by replacing each 2-prime subformula ψ in ϕ with a new (not occurring in ϕ) special propositional variable a_ψ associated with it.
6. The **reversal of 1-approximation** of ϕ is α_1^-, defined by $\alpha_1^-(\alpha_1(\phi)) = \phi$.

   The **2-approximation of** ϕ, $\alpha_2(\phi) \in \mathcal{L}_2$ and $\alpha_2^-(\cdot)$ are defined likewise.
7. A **signed formula** is one prefixed by T or F. The signed formula Tϕ means that ϕ is to be rendered true, whereas Fϕ means that ϕ is to be rendered false.

For example, let L_1 be a normal modal logic with primitive modal operators $\Box$ and $\Diamond$ and let L_2 be a temporal logic with the Priorian temporal operators H, P, G, F as primitives. Consider the following formula of $L_1 \circ L_2$:

$$\phi = (\mathsf{H}((\Box p \to p) \to \mathsf{H}q) \wedge \Box\neg\mathsf{G}q) \to (\Box(\neg p \vee \mathsf{G}\Diamond p) \vee \neg\mathsf{H}p)$$

Then:

- the subformula $(\Box p \to p) \to \mathsf{H}q$ is 1-external but not 2-external in ϕ, whereas $\neg p \vee \mathsf{G}\Diamond p$ is 2-external but not 1-external in ϕ.

- The subformulae $p, q, \mathsf{H}q, \mathsf{G}\Diamond p, \mathsf{H}((\Box p \to p) \to \mathsf{H}q)$, and $\Box(\neg p \vee \mathsf{G}\Diamond p)$ are monolithic, while $\neg \mathsf{G}q$ and $\neg p \vee \Box\Diamond p$ are not.

- The subformulae $\psi_1 = \mathsf{H}((\Box p \to p) \to \mathsf{H}q), \psi_2 = \mathsf{G}q, \psi_3 = \mathsf{G}\Diamond p, \psi_4 = \mathsf{H}p$ are 2-prime, while $\mathsf{H}q$ and $\neg\mathsf{G}q$ are not. Respectively, $\psi_5 = \Box p$, $\psi_6 = \Box\neg\mathsf{G}q$, and $\psi_7 = \Box(\neg p \vee \mathsf{G}\Diamond p)$ are 1-prime, while $\Diamond p$ and $\mathsf{G}\Diamond p$ are not.

- $\alpha_1(\phi) = (\mathsf{a}_{\psi_1} \wedge \Box\neg\mathsf{a}_{\psi_2}) \to (\Box(\neg p \vee \mathsf{a}_{\psi_3}) \vee \neg\mathsf{a}_{\psi_4})$;
  $\alpha_2(\phi) = (\mathsf{H}((\mathsf{a}_{\psi_5} \to p) \to \mathsf{H}q) \wedge \mathsf{a}_{\psi_6}) \to (\mathsf{a}_{\psi_7} \vee \neg\mathsf{H}p)$.

These notions are illustrated further, in examples 3.3.3, 3.6.1, and 3.6.2.

2.4 Closure of formulae

We fix a generic logical language $\mathcal{L}$ and only consider formulae in it. The **closure of a formula** ϕ is a (usually finite) set $cl(\phi)$ of **component formulae** that are relevant to computing the truth of ϕ. Often the closure consists of all subformulae of ϕ and their negations. These are not all necessary, nor always sufficient[1], but this is not essential here. Also, the notion of a closure must be suitably modified for first-order languages, but these will not be treated in any detail here, so I omit that definition.

Now, for any set of formulae Γ, I define

$$cl(\Gamma) := \bigcup\{cl(\phi) \mid \phi \in \Gamma\}$$

2.5 Generic semantic tableaux: assumptions, terminology, notation

The reader is assumed familiar with basics of tableaux systems e.g. within [12], or [11], or [20]. Still, for self-containment I include here a compendium of assumptions, terminology, and notation related to tableaux that will be used further in the series of papers started with the present one.

To begin with, by a **(Semantic) Tableaux** I mean a *rule-based procedure for systematic step-by-step search of a model of a given logic L satisfying a given input*

[1] The subformulae are not sufficient for instance when ϕ contains fixpoint operators that are computed recursively. E.g., the temporal operator "sometime in the future" $\mathsf{F}p$ unfolds as $p \vee \mathsf{XF}p$, so $\mathsf{XF}p$ has to be added to $cl(\mathsf{F}p)$. Likewise, the common knowledge operator $\mathsf{C}_A p$ unfolds as $p \wedge \mathsf{K}_A\mathsf{C}_A p$, so $\mathsf{K}_A\mathsf{C}_A p$ has to be added to $cl(\mathsf{C}_A p)$.

set of L-formulae. The generic semantic tableaux method considered here is assumed to have the following features:

- it is applied to a finite **input set** of (possibly signed) formulae Γ. I will call the tableaux method **essentially analytic** if it only operates on formulae in $cl(\Gamma)$. Unless indicated otherwise, I will only consider here essentially analytic tableaux.
- The procedure builds step-by-step a finite directed **tableau graph** with labelled nodes (**tableau states**). The **label of a tableau state** X is a set $\ell(X)$ of (signed) formulae in $cl(\Gamma)$ to be satisfied by the model under construction.
- The step-by-step expansion of the tableau graph is rule-based and usually nondeterministic. Generally, one can distinguish between three kinds of tableau expansion rules: **logical rules**, associated with the logical connectives; **structural rules**, ensuring the structural properties of the constructed models; and **control rules**, governing the process of extension and termination of the construction of the tableau graph.
- For modal and other logics with possible worlds semantics, the tableaux nodes are also prefixed with indices indicating possible worlds in the model under construction where the formulae in the state label are to be satisfied.
- Often (but not always) the tableau graph is presented as a *search tree* consisting of **branches**. There are two types of branching in the tableau graph: **logical/disjunctive branching**, in the search tree, and **structural branching**, in the model under construction. Unless otherwise specified, by a branch in a tableau I will mean a *logical branch*, whereas the structural branching will be implicit and encoded in the prefixes and labels of the nodes.

  The **label of a branch** is the union of the labels of its nodes.
- A branch in the tableau **closes** if, for some ϕ, both $\mathrm{T}\phi$ and $\mathrm{F}\phi$ (or, ϕ and $\neg\phi$ in the unsigned version) occur on it. Also, if $\top$ and $\bot$ are primitives in the language, the branch closes if $\mathrm{T}\bot$ or $\mathrm{F}\top$ occur on it.
- A branch in the tableau is **saturated** (aka, **fully expanded**), when no new formulae can be produced on the branch by applying any of the rules.
- The tableau expansion may terminate, or may run forever. The tableau graph keeps extending until every branch is either closed, or saturated, if ever; otherwise it extends forever.
- If a branch in the tableau does not close upon saturation, or extends forever, it is declared **open**. If at least one branch ends up open, the entire tableau is

declared **open**; else it **closes**.

- If the construction terminates and the tableau closes, that is a proof of unsatisfiability of the input set.
- When the tableau ends up open because it cannot close, it provides a partly defined “pre-model" (aka *Hintikka structure*) from which a **satisfying context of evaluation** for the input formula set can be constructed.

Some view semantic tableaux primarily as a method for testing validity. Others – as a method for testing satisfiability. These tasks are dually reducible to each other, but the choice reflects on the basic terminology. Here I will assume the former and will call a tableaux method:

- **sound**, if whenever the tableau closes, the input (set) is not satisfiable.
- **complete**, if, when the input (set) is not satisfiable, the tableau *can* close.
- **terminating**, if the tableau construction terminates for *any* input, with either the tableau closing, or with saturation leaving at least one open branch.

A sound, complete, and terminating tableaux method provides a decision procedure for satisfiability/validity of the logic.

3 Combining tableaux for fibred logics

3.1 Fibring of logics explained intuitively

Key references on the construction and underlying theory of fibring of logics are [14, 15, 16, 5]. Here I only give an intuitive outline of the syntax and semantics of a generic version of the fibring construction that will be used further. Consider two logical systems, L_1 and L_2, with respective formal semantics and classes of contexts of evaluation $\mathcal{C}_1$ and $\mathcal{C}_2$. The **fibred logic** $L_1 * L_2$ combines the languages of L_1 and L_2 like $L_1 \uplus L_2$ and *interleaves* their semantics.

A model of $L_1 * L_2$ can be specified by means of two partial mappings:

$$f_{12} : \mathcal{C}_1 \dashrightarrow \mathcal{C}_2 \text{ and } f_{21} : \mathcal{C}_2 \dashrightarrow \mathcal{C}_1,$$

possibly satisfying specific conditions for the desired models. Now, the semantics of $L_1 * L_2$ in such a model $\mathcal{M} = (f_{12}, f_{21})$ is defined compositionally as follows:

When evaluating a formula ϕ from $L_1 * L_2$ in a context $c_1 \in \mathcal{C}_1$ and a subformula ψ with a main connective from $\mathcal{L}_{2\setminus 1}$ is reached, the semantics continues the evaluation of ψ in the context $f_{12}(c_1) \in \mathcal{C}_2$.

The procedure when evaluating a formula in a context $c_2 \in \mathcal{C}_2$ goes likewise.

Remark 3.1. Here I consider the most general version of fibring of logics, with unconstrained syntax and semantics and the preservation results obtained here apply to that version. Both the language of $L_1 {*} L_2$, as well as the mappings f_{12}, f_{21} defining models and semantics of $L_1 {*} L_2$, can be constrained in various ways, so any (naturally defined) sublanguage of $L_1 \uplus L_2$ and a pair of classes of such mappings, $(\mathcal{C}_1,\mathcal{C}_2)$, define a version of the fibring $L_1 * L_2$. So, there is a family of such fibrings, rather than a single one. In particular, other well known types of combinations of logics, such as fusions and products, can be represented as suitably constrained fibrings. All such constructions of fibred logics and of the respective fibred tableaux, as well as most – but not all! – results are relativised accordingly from the general case. For example, completeness or decidability need not be preserved from the component logics to such restricted fibred systems.

3.2 Fibring of semantic tableaux

General constructions and methodology for combining (“fibring") tableaux for fibring of logics were first described on high-level and illustrated by some examples in [16], chapter 18, mainly for FOL (originally published as paper [2]) and chapter 19 specifically for modal logics. For each case, proofs are outlined stating that under a number of generic conditions the construction preserves soundness and completeness of the tableaux of the component logics. Here I present a simpler, generic yet concrete, self-contained and arguably somewhat more intuitive alternative to that construction, building first on the simpler special cases of *propositional merger*, then *simple nesting* of logics and their tableaux, and eventually covering the general case.

3.3 Warming up: tableaux for propositional merger of logics

Consider logics L_1 and L_2 with *disjoint languages*, but sharing common boolean connectives and possibly some of the atomic formulae. The simplest meaningful way of combining L_1 and L_2 is what I call the **propositional merger of $\mathcal{L}_1$ and $\mathcal{L}_2$** produced by uniting their sets of formulae and closing under booleans. Formally, this is a logic, hereafter denoted $L_1 \circ L_2$, with formulae defined as:

$$\phi = \phi_1 \mid \phi_2 \mid \neg\phi \mid \phi \wedge \phi \mid$$

where ϕ_1 is a formula of L_1, ϕ_2 is a formula of L_2, and all other boolean connectives are definable as usual. Thus, each component logic in the merger treats the formulae beginning with a private operator of the other logic as atoms.

The semantics is defined compositionally and is unrestricted, on all pairs of contexts of evaluation $(\mathbf{c}_1, \mathbf{c}_2)$. The evaluation of a formula starts at either of these and switches to the other context when the main connective of the currently evaluated formula is private for the other language. The routine details are omitted.

3.3.1 Description of the propositional merger tableau

Given tableau procedures $\mathscr{T}_1$ and $\mathscr{T}_2$ for the logics L_1 and L_2, their **merger tableau** is the pair $\mathscr{T} = \mathfrak{m}(\mathscr{T}_1, \mathscr{T}_2)$, acting as follows.

1. A propositional tableau $\mathscr{T}_0$ is run on the input set Γ in $L_1 \circ L_2$, treating all monolithic subformulae as atoms. Formally, this can be done by applying $\mathscr{T}_0$ to either approximation $\alpha_1(\alpha_2(\Gamma))$ or $\alpha_2(\alpha_1(\Gamma))$.
2. If that tableau closes, then the merger tableau $\mathscr{T}$ closes, too.
3. Otherwise, after saturation we select, non-deterministically or following some strategy, an open branch B in $\mathscr{T}_0$ and take its label $\ell(B)$.

   Let $\ell_i(B)$ be the set of monolithic L_i-formulae in $\ell(B)$, for $i \in \{1, 2\}$.
4. Now, each tableau $\mathscr{T}_i$ on $\ell_i(B)$ is run separately. If any of these closes, the branch B in $\mathscr{T}$ is declared closed and another one (the next in a list) is explored likewise.
5. Otherwise, an open branch from each tableau is chosen, non-deterministically or following some strategy, and the union X of the sets of atomic formulae in the labels of these two branches is checked for atomic consistency (i.e., lack of contradictory pair of atoms).
6. If the set X is inconsistent, the procedure backtracks and looks for another pair of open branches. If all such pairs have already been explored and found locally inconsistent then the branch B in $\mathscr{T}$ is declared closed and another one (the next in a list) is explored likewise.
7. If the set X is consistent, the pair of selected branches is declared **locally consistent** and the branch B in $\mathscr{T}$ is declared open. Then the open branch in each $\mathscr{T}_i$ generates a context of evaluation $\mathbf{c}_i$ satisfying $\ell_i(B)$ and the pair $(\mathbf{c}_1, \mathbf{c}_2)$ is returned as a context of evaluation satisfying Γ.

3.3.2 Two remarks on the propositional merger tableaux construction

1. The merger tableau procedure can be simplified to work only on logics with no shared atomic formulae, by replacing every shared atomic formula A by two distinct copies, A_1 private for L_1 and A_2 private for L_2, and then adding $A_1 \leftrightarrow A_2$ to the input set of formulae for $\mathscr{T}_0$.

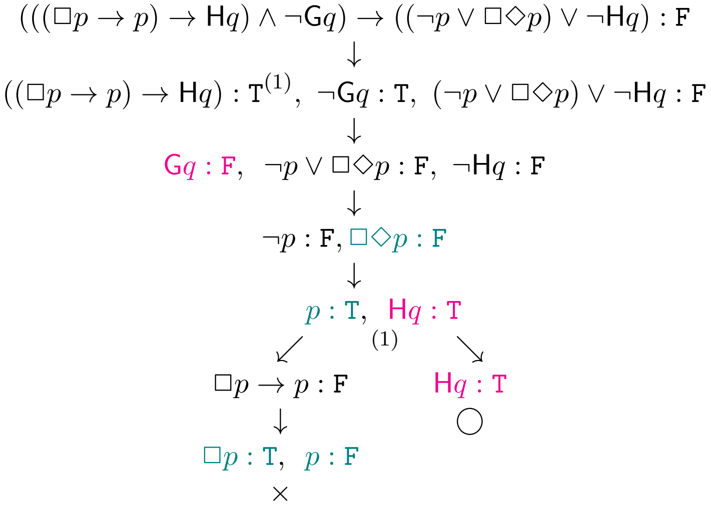
2. The construction of $\mathfrak{m}(\mathscr{T}_1, \mathscr{T}_2)$ can be modified so that $\mathscr{T}_0$ is subsumed by one of $\mathscr{T}_1$ or $\mathscr{T}_2$, and the other one is run on the labels of the open branches.

3.3.3 Example 1: propositional merger of tableaux

Let L_1 be a normal modal logic with primitive modal operators $\Box$ and $\Diamond$ and contexts of evaluation being pointed Kripke models $(\mathcal{M}, w)$. Let L_2 be a Priorian temporal logic with temporal operators H, P, G, F and contexts of evaluation being pointed temporal models $(\mathcal{F}, t)$. Consider the following formula of $L_1 \circ L_2$:

$$\phi = (((\Box p \to p) \to \mathsf{H}q) \wedge \neg \mathsf{G}q) \to ((\neg p \vee \Box\Diamond p) \vee \neg \mathsf{H}q)$$

Let us run the tableau $\mathscr{T} = \mathfrak{m}(\mathscr{T}_1, \mathscr{T}_2)$ for checking validity of ϕ:


$(((\Box p \to p) \to \mathsf{H}q) \wedge \neg \mathsf{G}q) \to ((\neg p \vee \Box\Diamond p) \vee \neg \mathsf{H}q) : \mathtt{F}$
$((\Box p \to p) \to \mathsf{H}q) : \mathtt{T}^{(1)}, \ \neg \mathsf{G}q : \mathtt{T}, \ (\neg p \vee \Box\Diamond p) \vee \neg \mathsf{H}q : \mathtt{F}$
$\mathsf{G}q : \mathtt{F}, \ \neg p \vee \Box\Diamond p : \mathtt{F}, \ \neg \mathsf{H}q : \mathtt{F}$
$\neg p : \mathtt{F}, \Box\Diamond p : \mathtt{F}$
$p : \mathtt{T}, \ \mathsf{H}q : \mathtt{T}$
(1)
$\Box p \to p : \mathtt{F}$
$\mathsf{H}q : \mathtt{T}$
○
$\Box p : \mathtt{T}, \ p : \mathtt{F}$
×

The index $^{(1)}$ indicates where the decomposition rule for the indexed formula is applied. The label of (the leaf of) the open branch B contains all formulae on the branch. The respective sub-labels, taken as input sets for $\mathscr{T}_1$ and $\mathscr{T}_2$ are:

$\ell_1(B) = \{p : \mathtt{T}, \Box\Diamond p : \mathtt{F}\}$ for $\mathscr{T}_1$ and $\ell_2(B) = \{\mathsf{H}q : \mathtt{T}, \mathsf{G}q : \mathtt{F}\}$ for $\mathscr{T}_2$.

Running $\mathscr{T}_1$ on $\{p : \mathtt{T}, \Box\Diamond p : \mathtt{F}\}$ ends open if L_1 is **K**, but closes if L_1 is **B**.

In the latter case, the entire merger tableau $\mathscr{T} = \mathfrak{m}(\mathscr{T}_1, \mathscr{T}_2)$ closes.

Running $\mathscr{T}_2$ on $\{\mathsf{H}q : \mathtt{T}, \mathsf{G}q : \mathtt{F}\}$ ends up open if L_2 is the temporal logic of irreflexive time flows, but closes for the logic of reflexive time flows.

In the latter case, the entire merger tableau $\mathscr{T}$ closes.

In the former case we select any pair of open branches in $\mathscr{T}_1$ and $\mathscr{T}_2$. They will be trivially locally consistent, as the two labels do not share common atoms. Then, assuming e.g., $L_1 = \mathbf{K}$, the tableau $\mathscr{T}$ does not close and a satisfying for ϕ context of evaluation $((\mathcal{M}, w), (\mathcal{F}, t))$ can be constructed.

Now, consider the following modification of the formula ϕ:

$$\phi' = (((\Box p \to p) \to \mathsf{FH}q) \wedge \neg \mathsf{G}q) \to ((\neg p \vee \Box\Diamond p) \vee \neg \mathsf{FH}(\neg p \vee \neg q))$$

The run of the tableau $\mathscr{T}_0$ is essentially the same as before, as the changes are only in monolithic subformulae, but the resulting sub-labels now are:

$\ell'_1(B) = \{p : \mathtt{T}, \Box\Diamond p : \mathtt{F}\}$ for $\mathscr{T}_1$, and
$\ell'_2(B) = \{\mathsf{FH}q : \mathtt{T}, \mathsf{G}q : \mathtt{F}, \mathsf{FH}(\neg p \vee \neg q) : \mathtt{T}\}$ for $\mathscr{T}_2$.

Now, the label $\ell'_1(B)$ is the same as $\ell_1(B)$. Assuming that L_2 is the temporal logic of irreflexive time flows where a time point may have an immediate successor, the $\mathscr{T}_2$ when run on $\ell'_2(B)$ will not close, but any open branch of it will have a label containing $p : \mathtt{F}$, which is locally inconsistent with $\ell'_1(B)$, so the entire merger tableau procedure will close.

Similarly, consider another modification of the formula ϕ:

$$\phi'' = (((\Box p \to p) \to \mathsf{H}q) \wedge \neg \mathsf{G}q) \to ((\neg p \vee \Box\Diamond p) \vee \neg \mathsf{FH}(\neg p \vee q))$$

Again, the run of the tableau $\mathscr{T}_0$ is the same, but the resulting sub-labels now are:

$\ell''_1(B) = \{p : \mathtt{T}, \Box\Diamond p : \mathtt{F}\}$ for $\mathscr{T}_1$, and
$\ell''_2(B) = \{\mathsf{H}q : \mathtt{T}, \mathsf{G}q : \mathtt{F}, \mathsf{FH}(\neg p \vee q) : \mathtt{T}\}$ for $\mathscr{T}_2$.

Thus, the label $\ell''_1(B)$ is the same as before. Now, running $\mathscr{T}_2$ on $\ell''_2(B)$ again ends up open for L_2 being the logic of irreflexive time flows, with two open branches, with labels containing respectively $p : \mathtt{F}$ and $q : \mathtt{T}$. The first one is locally inconsistent with $\ell''_1(B)$, so if it is selected, the entire merger tableau procedure will close. The second one, however, is locally consistent with $\ell''_1(B)$, so if it is selected, the merger tableau procedure will end up open and will produce a satisfying model.

3.3.4 Preservation results for tableaux for propositional mergers

In summary, $\mathfrak{m}(\mathscr{T}_1, \mathscr{T}_2)$ preserves soundness, completeness, and termination of $\mathscr{T}_1$ and $\mathscr{T}_2$.

Theorem 3.2. *Let $\mathscr{T}_1$ and $\mathscr{T}_2$ be tableaux procedures respectively for the logics L_1 and L_2. The following holds for the propositional merger tableau $\mathfrak{m}(\mathscr{T}_1, \mathscr{T}_2)$:*

1. *If $\mathscr{T}_1$ and $\mathscr{T}_2$ are sound respectively for L_1 and L_2, then $\mathfrak{m}(\mathscr{T}_1, \mathscr{T}_2)$ is sound for $L_1 \circ L_2$.*

2. *If $\mathscr{T}_1$ and $\mathscr{T}_2$ are complete respectively for L_1 and L_2, then $\mathfrak{m}(\mathscr{T}_1, \mathscr{T}_2)$ is complete for $L_1 \circ L_2$.*

3. *If each of $\mathscr{T}_1$ and $\mathscr{T}_2$ is terminating then $\mathfrak{m}(\mathscr{T}_1, \mathscr{T}_2)$ is terminating, too.*

Proof.

1. We reason by contraposition. Suppose the input set Γ is satisfiable in $L_1 \circ L_2$ in some context of evaluation $(\mathbf{c}_1, \mathbf{c}_2)$. Then $\alpha_1(\Gamma)$ is satisfiable in L_1 and $\alpha_2(\Gamma)$ is satisfiable in L_2, by suitable modifications respectively of $\mathbf{c}_1$ and $\mathbf{c}_2$. In particular, $\alpha_1(\alpha_2(\Gamma))$ is propositionally satisfiable, so there is an open branch B in $\mathscr{T}_0(\alpha_1(\alpha_2(\Gamma)))$ that generates some propositional assignment $\mathbf{c}_B$ for L_1 satisfying $\ell(B)$. Note that $\alpha_1(\Gamma) \subseteq \alpha_2^{-1}(\ell(B))$ and $\alpha_2(\Gamma) \subseteq \alpha_1^{-1}(\ell(B))$, so we can choose B and $\mathbf{c}_B$ so that both $\ell_1(B)$ and $\ell_2(B)$ are satisfiable respectively in L_1 and L_2, in a way which is consistent for the common atomic formulae. Then each tableau $\mathscr{T}_i$, when run on $\ell_i(B)$ separately (and independently from each other) ends with an open branch generating a context of evaluation $\mathbf{c}_i$ satisfying $\ell_i(B)$ and, moreover, these open branches will be locally consistent. Therefore, $\mathfrak{m}(\mathscr{T}_1, \mathscr{T}_2)$ does not close and the pair $(\mathbf{c}_1, \mathbf{c}_2)$ is a context of evaluation satisfying Γ.

2. We reason again by contraposition. Suppose $\mathscr{T}_0(\alpha_1(\alpha_2(\Gamma)))$ does not close. Then, by construction, it generates an open branch B and runs $\mathscr{T}_1$ and $\mathscr{T}_2$ to generate contexts of evaluation $\mathbf{c}_1$ and $\mathbf{c}_2$ satisfying respectively $\ell_1(B)$ and $\ell_2(B)$. Hence, by completeness of $\mathscr{T}_1$ and $\mathscr{T}_2$, both $\ell_1(B)$ and $\ell_2(B)$ are satisfiable, respectively in L_1 and L_2, and hence Γ is satisfiable in $L_1 \circ L_2$.

3. The preservation of termination is immediate by construction.

□

The time and space complexites for $\mathfrak{m}(\mathscr{T}_1, \mathscr{T}_2)$ are each linearly bounded by the maxima of the respective complexities of the components $\mathscr{T}_1$ and $\mathscr{T}_2$.

3.4 Nesting (layering) of logics

Nesting (aka, *layering* [16]) is a simple way of fibring of two logics L_1 and L_2, by applying one within the other. More precisely, the **nesting of L_2 in L_1** is a logic $L_1(L_2)$ in the combined language of L_1 and L_2 under the restriction that *the formulae* *of L_2 may only appear as atoms in the formulae of L_1.* Note that disjointness of the languages of L_1 and L_2 is no longer required, as all atomic propositions of L_2 can also be regarded as atomic propositions of L_1. Special cases of nesting are *temporalising of logical systems* [8, 21], and *modalising of FOL* [11]. For example:

- for L_1 being a temporal logic TL and L_2 being a single-agent epistemic logic EL, the formula $\mathsf{GPK}_a p \to \mathsf{K}_a p$ is in $\mathsf{TL}(\mathsf{EL})$, whereas $\mathsf{K}_a\mathsf{F}p \to p$ is not.
- $\Box\forall x \neg Q(x)$ is a formula of K(FOL), whereas $\forall x \Box \neg Q(x)$ is not.
- a natural principle, expressible in a nesting of logics, is the *future determinism* $\mathsf{F}\Box p \vee \mathsf{F}\Box\neg p$ in a temporal-alethic logic.

A model for $L_1(L_2)$ can be specified by a single mapping, $f_{12} : \mathcal{C}_1 \to \mathcal{C}_2$. For example, a model for $\mathsf{TL}(\mathsf{EL})$ is defined by first fixing a temporal model $\mathcal{T}$ of TL and then associating with every $t \in \mathcal{T}$ a rooted epistemic model $(\mathcal{M}_t, w)$. Natural constraints can be imposed both on the *global* temporal model (e.g., linear, discrete, etc., cf. [18]), as well as on the *local* epistemic models, e.g., synchrony, no forgetting, no learning, (cf. [7]) etc. Likewise, a model for K(FOL) is constructed by fixing a Kripke model $\mathcal{M}$ and associating with every world w in $\mathcal{M}$ a first-order model $\mathcal{F}_w$. These are naturally packaged together in models of first-order modal logic (FOML). To provide a context of evaluation here, a variable assignment in each $\mathcal{F}_w$ is added. Natural constraints can be imposed on the local FO models, e.g. increasing domains, constant domains, etc.

3.5 Nesting of tableaux: outline

Given tableaux procedures, $\mathscr{T}_1$ and $\mathscr{T}_2$, for the logics L_1 and L_2, the **nested tableau** $\mathscr{T} = \mathscr{T}_1(\mathscr{T}_2)$ is defined to work on formulae of $L_1(L_2)$ as follows:

1. The input set of formulae Γ in the language of $L_1(L_2)$, is replaced by its 1-approximation $\alpha_1(\Gamma)$.
2. The tableau $\mathscr{T}_1$ is applied to $\alpha_1(\Gamma)$. If it closes, then $\mathscr{T}$ closes on Γ, too.
3. Otherwise, each approximating variable a_ψ in $\alpha_1(\Gamma)$ is replaced back in the resulting tableau graph $\mathscr{T}_1(\alpha_1(\Gamma))$ by the respective formula ψ in L_2.
4. Every open branch S in $\mathscr{T}_1(\alpha_1(\Gamma))$ generates at least one context of evaluation $\mathbf{c}_S$ for L_1 satisfying $\alpha_1(\Gamma)$. We non-deterministically select (or guess) and fix such branch S and a context of evaluation $\mathbf{c}_S$.

5. The context of evaluation $\mathbf{c}_S$ is based on a model $\mathcal{M}$ for L_1. It determines a set of associated contexts of evaluation $\mathscr{C}_S$ in $\mathcal{M}$ which are used for evaluating the truth of $\alpha_1(\Gamma)$ in $\mathbf{c}_S$ (e.g., pointed models based on $\mathcal{M}$).
6. For each $\mathbf{c} \in \mathscr{C}_S$, consider the set $\ell_2(\mathbf{c})$ consisting of all monolithic formulae of L_2 in $cl(\Gamma)$ for which the 1-approximations in $cl(\alpha_1(\Gamma))$ are true in $\mathbf{c}$.

   They must all be satisfied in some model of L_2.
7. Now, for each $\mathbf{c} \in \mathscr{C}_S$, the tableau $\mathscr{T}_2$ is applied to the input set $\ell_2(\mathbf{c})$.
8. If any of the resulting tableau graphs $\mathscr{T}_2(\ell_2(\mathbf{c}))$ closes, then the branch S in $\mathscr{T}_1(\alpha_1(\Gamma))$ closes in $\mathscr{T} = \mathscr{T}_1(\mathscr{T}_2)$. If so, another open branch S in $\mathscr{T}_1(\alpha_1(\Gamma))$ (if any are left) is selected and the same procedure is applied.

   If there are no such open branches still left, then $\mathscr{T}$ is declared closed.
9. Else, every open branch B in each $\mathscr{T}_2(\ell_2(\mathbf{c}))$ generates a context of evaluation $\mathbf{c}_B$ for L_2 satisfying $\ell_2(\mathbf{c})$. We fix one such open branch B and a context $\mathbf{c}_B$ for each $\mathbf{c} \in \mathscr{C}_S$ and declare the respective branch B in $\mathscr{T}(\Gamma)$ open.
10. Now, a model for $L_1(L_2)$ satisfying Γ at $\mathbf{c}_S$ can be constructed by any partial mapping f_{12} such that $f_{12}(\mathbf{c}) = \mathbf{c}_B$ for each $\mathbf{c} \in \mathscr{C}_S$.

This completes the description of the nesting of tableaux procedure $\mathscr{T} = \mathscr{T}_1(\mathscr{T}_2)$.

Remark 3.3. The nesting of tableaux described here resembles the way the usual Satisfiability Modulo Theories (SMT) methods work, viz. complex formulae are replaced with new literals and then the different SAT procedures for the component theories are respectively called and eventually combined. This parallel is worth exploring further.

3.6 Nesting of tableaux: examples in temporal epistemic logic

Let L_1 be the temporal logic over irreflexive linear time flows LTL^{ir} with linear models based on $\mathbb{N}$ as contexts of evaluation and a tableau system $\mathscr{T}_1$.

Let L_2 be the single-agent epistemic logic EL with modal operator K, pointed epistemic (S5) models as contexts of evaluation, and a tableau system $\mathscr{T}_2$.

I will run the tableau $\mathscr{T} = \mathscr{T}_1 \circ \mathscr{T}_2$ for checking satisfiability on two examples of formulae of $L_1(L_2)$.

3.6.1 Example 1 of nesting of tableaux

Consider the $L_1(L_2)$-formula $\phi_1 = \mathsf{FG}\,\mathsf{K}(p \vee q) \wedge \mathsf{G}\,\mathsf{K}\neg p \wedge \mathsf{F}\neg\mathsf{K}q$.

The tableau $\mathscr{T} = \mathscr{T}_1(\mathscr{T}_2)$ for checking satisfiability of ϕ_1 runs as follows.

1. First, it takes the 1-approximation of the input formula:

   $\alpha_1(\phi_1) = \mathsf{FGa}_{\psi_1} \wedge \mathsf{Ga}_{\psi_2} \wedge \mathsf{F}\neg\mathsf{a}_{\psi_3}$, where $\psi_1 = \mathsf{K}(p \vee q)$, $\psi_2 = \mathsf{K}\neg p$, $\psi_3 = \mathsf{K}q$.

2. The tableau $\mathscr{T}_1$ applied to $\alpha_1(\phi_1)$ ends up open. (Easy details are omitted.)

   An open branch of $\mathscr{T}_1(\alpha_1(\phi_1))$ produces a linear temporal model $\mathcal{M}$ over $\mathbb{N}$ with respective labels $\{\}, \{\mathsf{a}_{\psi_2}, \neg\mathsf{a}_{\psi_3}\}, \{\mathsf{a}_{\psi_1}, \mathsf{a}_{\psi_2}\}, \{\mathsf{a}_{\psi_1}, \mathsf{a}_{\psi_2}\}, \ldots$.

3. Each of these corresponds to a set of L_2-formulae which is satisfiable in EL, so $\mathscr{T}_2$ produces for each of these a satisfying pointed EL-model.

   Let Γ_n be the set of L_2-formulae in the label of state n in $\mathcal{M}$, and let $(\mathcal{M}_n, w_n)$ be a pointed EL-model satisfying Γ_n produced from $\mathscr{T}_2(\Gamma_n)$.

4. Then the mapping f_{12} assigning $(\mathcal{M}_n, w_n)$ to $(\mathcal{M}, n)$, for each $n \in \mathbb{N}$, provides a fibred model for $\mathsf{LTL}^{ir}(\mathsf{EL})$ satisfying ϕ_1.

3.6.2 Example 2 of nesting of tableaux

Consider now the $L_1(L_2)$-formula $\phi_2 = \mathsf{FG}\,\mathsf{K}(p \vee q) \wedge \mathsf{G}\,\mathsf{K}\neg p \wedge \mathsf{GF}\neg\mathsf{K}q$. (It is very similar, but *not* the same as ϕ_1 in Example 2, because of the last conjunct.)

The tableau $\mathscr{T} = \mathscr{T}_1(\mathscr{T}_2)$ for checking its satisfiability runs as follows.

1. First, it takes the 1-approximation of the input formula:

   $\alpha_1(\phi_2) = \mathsf{FGa}_{\psi_1} \wedge \mathsf{Ga}_{\psi_2} \wedge \mathsf{GF}\neg\mathsf{a}_{\psi_3}$, where $\psi_1 = \mathsf{K}(p \vee q)$, $\psi_2 = \mathsf{K}\neg p$, $\psi_3 = \mathsf{K}q$. The tableau $\mathscr{T}_1$ applied to $\alpha_1(\phi_2)$ ends up open. (Easy details are omitted.)

2. Then, any open branch of $\mathscr{T}_1(\alpha_1(\phi_2))$ produces a linear temporal model $\mathcal{M}$ that has a state with a label $\{\mathsf{a}_{\psi_1}, \mathsf{a}_{\psi_2}, \neg\mathsf{a}_{\psi_3}\}$.

3. However, the corresponding set of (EL)-formulae $\{\psi_1, \psi_2, \neg\psi_3\}$ is not satisfiable in any (EL)-model, so $\mathscr{T}_2$ closes on it.

4. Therefore, $\mathscr{T}$ closes on ϕ_2.

3.7 Nesting of tableaux: preservation results

Theorem 3.4. *Let $\mathscr{T}_1$ and $\mathscr{T}_2$ be tableaux procedures respectively for the logics L_1 and L_2. The following holds for the nested tableau $\mathscr{T} = \mathscr{T}_1(\mathscr{T}_2)$:*

1. *If $\mathscr{T}_1$ and $\mathscr{T}_2$ are sound respectively for L_1 and L_2, then $\mathscr{T}_1(\mathscr{T}_2)$ is sound for $L_1(L_2)$.*

2. *If $\mathscr{T}_1$ and $\mathscr{T}_2$ are complete respectively for L_1 and L_2, then $\mathscr{T}_1(\mathscr{T}_2)$ is complete for $L_1(L_2)$.*

3. *If each of $\mathscr{T}_1$ and $\mathscr{T}_2$ is terminating and, whenever $\mathscr{T}_1$ does not close it produces a finitary model for L_1, then $\mathscr{T}_1(\mathscr{T}_2)$ is terminating, too.*

Proof. I only provide proof sketches here and leave the mostly routine, but lengthy, details to the reader.

1. We reason by contraposition. Suppose the input set Γ is satisfiable in $L_1(L_2)$. Then $\alpha_1(\Gamma)$ is satisfiable in L_1, so $\mathscr{T}_1(\alpha_1(\Gamma))$ does not close. Then, there is some open branch B in $\mathscr{T}_1(\alpha_1(\Gamma))$ that generates some context of evaluation $\mathbf{c}_B$ for L_1 satisfying $\alpha_1(\Gamma)$ and it can be expanded to a context of evaluation $\mathbf{c}^*$ for $L_1(L_2)$ satisfying Γ. Hence, every set $\ell_2(\mathbf{c})$ for a context of evaluation $\mathbf{c}_B \in \mathscr{C}_B$ is satisfiable in L_2, hence each $\mathscr{T}_2(\ell_2(\mathbf{c}))$ does not close. Therefore, $\mathscr{T}(\Gamma)$ does not close.

2. We reason again by contraposition. If $\mathscr{T}(\Gamma)$ does not close, then, by construction, it generates a satisfying fibred model from the satisfying contexts of evaluation $\mathbf{c}_S$ for $\alpha_1(\Gamma)$ and $\mathbf{c}_B$ for each open branch B in $\mathscr{T}_2(\ell_2(\mathbf{c}))$, for each $\mathbf{c} \in \mathscr{C}_S$.

3. The preservation of termination is by construction, assuming that the correctly selected context of evaluation $\mathbf{c}_S$ satisfying $\alpha_1(\Gamma)$ (if there is one) is finite, up to the set of labels of nodes.

□

The time and space complexites for $\mathscr{T}_1(\mathscr{T}_2)$ are each effectively bounded in terms of the respective complexities of the component tableaux and the maximum of the size bounds of the models they produce. The predictable details are omitted.

3.8 Extension to interleaving of logics and their tableaux

By a **(simple) interleaving** of logics L_1 and L_2 I mean the logic

$$L_1 \between L_2 := L_1(L_2) \circ L_2(L_1),$$

i.e., the propositional merger of the nestings $L_1(L_2)$ and $L_2(L_1)$ over the same set of atoms and classical connectives.

Note that this simple interleaving is often sufficiently expressive to capture important principles of interaction between the component logics, for example:

- In first-order modal logic (FOML): the *Barcan formula* $\forall x \Box A(x) \to \forall x \Box A(x)$ and its converse $\forall x \Box A(x) \to \forall x \Box A(x)$;
- in temporal-epistemic logics: the principles of *no learning*: $\mathsf{GK}p \to \mathsf{KG}p$ (where G is the operator “always in the future") and *perfect recall* (aka, *no forgetting*): $\mathsf{KG}p \to \mathsf{GK}p$;
- in products of modal logics: the *commutation* $\Diamond_i \Diamond_j p \leftrightarrow \Diamond_j \Diamond_i p$ and the *confluence* $\Diamond_i \Box_j j \to \Box_j \Diamond_i p$ of the component modalities, etc.

If $\mathscr{T}_1$ and $\mathscr{T}_2$ are respective tableaux for L_1 and L_2, then their **interleaved tableau** $\mathscr{T} = \mathscr{T}_1 \between \mathscr{T}_2$ is defined similarly to $\mathscr{T}_1(\mathscr{T}_2)$, but with an additional step of approximations. Alternatively, it can be constructed as a propositional merger of the two nested tableaux, $\mathscr{T}_1(\mathscr{T}_2)$ and $\mathscr{T}_2(\mathscr{T}_1)$. The preservation results in Theorem 3.4 for $\mathscr{T}_1(\mathscr{T}_2)$ apply accordingly here.

3.9 The extension to full fibring of logics

The full fibring $L_1 * L_2$ of the logical systems L_1 and L_2 can be constructed by iterating the process of interleaving described earlier. Indeed, one can define a hierarchy of deeper interleavings as follows

$$L_1 \between^1 L_2 := L_1 \between L_2$$

$$L_1 \between^{n+1} L_2 := L_1(L_1 \between^n L_2) \circ L_2(L_1 \between^n L_2)$$

Intuitively, $L_1 \between^n L_2$ is the partial fibring allowing for nested alternation of operators from the two logics up to depth n. Note that these partial fibrings form a chain by inclusion: $L_1 \between^1 L_2 \subseteq L_1 \between^2 L_2 \subseteq \ldots$. These syntactic inclusions also correspond to a chain of semantic expansions: for each n, the semantics of $L_1 \between^{n+1} L_2$ subsumes that of $L_1 \between^n L_2$ in a natural way. I leave out the predictable, but somewhat messy details. Now, I define the full fibring as follows:

$$L_1 * L_2 := \bigcup_{n=0}^{\infty} L_1 \between^n L_2$$

Thus, every formula of $L_1 * L_2$ belongs to some $L_1 \between^n L_2$.

Accordingly, the constructions of nesting and interleaving of tableaux can be iterated to produce **full fibring** $\mathcal{F}_1 * \mathcal{F}_2$ of the component tableaux. Eventually, upon saturation, a satisfying context of evaluation can be constructed from any open branch of the fully fibred tableau iteratively, by grafting intermediate contexts of evaluation produced at every step to the already constructed “approximating” context of evaluation. However, defining the full fibring of tableaux is unnecessary if we only consider finite input sets of formulae, for in such case it suffices to consider the partial fibring $\mathscr{T}_1 \between^n \mathscr{T}_2$, for a large enough n, defined for $L_1 \between^n L_2$ by the same recurrent constructions.

The proofs of the respective preservation results for partial and full fibrings are conceptually routine, but technically cumbersome, so I leave out the details. Here I only remark that the fibred semantics adopted here is usually more general than the intended standard semantics for the logic constructed by fibring, so there is an additional challenge here to strengthen the preservation results with respect to that standard semantics, which is to be addressed in every particular case.

4 Some applications and concluding remarks

4.1 On some applications of combining tableaux for fibred logics

Here I outline two applications to natural and important cases of fibred logics.

4.1.1 Application to first-order modal logics

First-order modal logic (FOML) can be defined as a fibring of an underlying propositional modal logic **ML** and a first-order theory **FOL**. The respective (partial) mappings act as follows:

- To any pair $(\mathcal{M}, w)$ of a model $\mathcal{M}$ for **ML** and $w \in \mathcal{M}$, f_{12} assigns a pair $(\mathcal{F}, \mathbf{v})$, where $\mathcal{F}$ is a first-order model for **FOL** and $\mathbf{v}$ is a variable assignment in it.

- Conversely, to any pair $(\mathcal{F}, \mathbf{v})$ of a first-order model $\mathcal{F}$ for **FOL** plus a variable assignment $\mathbf{v}$ in it, f_{21} assigns a pair $(\mathcal{M}, w)$, where $\mathcal{M}$ is a model for **ML** and $w \in \mathcal{M}$.

These mappings together generate a class of fibred models for FOML, which are more general than standard FOML models, because their fibring is not subjected to any constraints, but they can be converted to such models. Thus, fibring of tableaux can be generically applied to designing new types of tableaux procedures for FOML and fragments of it. Note that in [11] some tableaux systems for FOML over constant domains and over varying domains are presented, where the modal component is a normal modal logic from the so called *Lesser Modal Cube*. However, to our knowledge, no general tableaux method for FOML going beyond such concrete results seems to be available so far. One challenging case, that can be handled with the tableaux fibring technology outlined here, is to obtain a sound and complete tableau system for FO(S4.2), where S4.2 = S4 + $\Diamond\Box p \rightarrow \Box\Diamond p$ (the Church-Rosser confluence axiom).

4.1.2 Application to temporalising tableaux

Temporalising a logical system means combining it with – usually, by nesting into – a temporal logic. Key references are [8], [9], [10]. Respectively, the problem of *temporalising tableaux* arises. That problem was explored in [21] for the case of constructing tableau methods for the so called *monodic fragments* of first-order temporal logics, based on decidable fragments of first-order logic like the two-variable fragment or the guarded fragment. In particular, a general framework is presented there that shows how existing decision procedures for first-order fragments can be used for constructing a tableau algorithm for the corresponding monodic fragment of [first-order temporal logic] FOTL. This framework is applied there only to the case of LTL over $(\mathbb{N}, <)$. The general challenge arising here is to use the fibring tableaux technology outlined here to extend these results to tableaux for FOTL e.g., over branching time temporal logics for which sound, complete, and terminating tableaux systems have been developed.

4.2 Concluding remarks

This paper initiates a research project devoted to combining tableaux for various combinations of logics. Here I have considered the case of a generic version of fibring of logics with most general fibrings of their languages and semantics, and have outlined a method for full “fibring" of their tableaux, based on iteration of the procedure of “interleaving’" of tableaux for simple nesting (interleaving) of the logics. The main points of this work are as follows.

1. Tableaux-based deductive systems for combinations of logics can be developed in a systematic and modular way, rather than ad hoc.

2. The very concept of tableaux for combined logics can be extended from single tableau-building procedures to more complex systems of relatively autonomous and interactive simpler tableau components.

3. The advantages of that approach include: modularity, flexibility, relative simplicity, and uniform preservation of important properties (soundness, completeness, possibly termination).

Subsequent intended papers will explore the cases of *fusions* and *products* and will develop methods for combining tableaux for these cases. Furthermore, all these methods will hopefully be eventually implemented.

References

[1] Franz Baader and Klaus U. Schulz, editors. *Frontiers of Combining Systems, First International Workshop FroCoS 1996, Munich, Germany, March 26-29, 1996, Proceedings*, volume 3 of *Applied Logic Series*. Kluwer Academic Publishers, 1996.

[2] Bernhard Beckert and Dov Gabbay. Fibring semantic tableaux. In Harrie de Swart, editor, *Automated Reasoning with Analytic Tableaux and Related Methods*, pages 77–92, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

[3] P. Blackburn, M. de Rijke, and V. Venema. *Modal Logic.* CUP, 2001.

[4] P. Blackburn, J. van Benthem, and F. Wolter, editors. *Handbook of Modal Logic.* Elsevier, 2006.

[5] Carlos Caleiro, Amílcar Sernadas, and Cristina Sernadas. Fibring logics: Past, present and future. In Sergei N. Artëmov, Howard Barringer, Artur S. d'Avila Garcez, Luís C. Lamb, and John Woods, editors, *We Will Show Them! Essays in Honour of Dov Gabbay, Volume One*, pages 363–388. College Publications, 2005.

[6] M. D'Agostino, D. Gabbay, R. Hähnle, and J. Posegga, editors. *Handbook of Tableau Methods.* Kluwer, Dordrecht, 1999.

[7] R. Fagin, J.Y. Halpern, Y. Moses, and M.Y. Vardi. *Reasoning about Knowledge.* MIT Press: Cambridge, MA, 1995.

[8] M. Finger and D.M. Gabbay. Adding a temporal dimension to a logic system. *Journal of Logic, Language and Information*, 1:203–233, 1992.

[9] Marcelo Finger and Dov Gabbay. Combining temporal logic systems. *Notre Dame Journal of Formal Logic*, 37(2):204–232, 1996.

[10] Marcelo Finger and M. Weiss. The unrestricted combination of temporal logic systems. *Logic Journal of the IGPL*, 10(2):165–189, 2002.

[11] Melvin Fitting and Richard L. Mendelsohn. *First-Order Modal Logic.* Springer, 2nd edition, 2023.

[12] Melvin C. Fitting. Modal proof theory. In Patrick Blackburn, Johan van Bentem, and Frank Wolter, editors, *Handbook of Modal Logic*, pages 85–138. Elsevier, 2007.

[13] D.M. Gabbay, A. Kurucz, F. Wolter, and M. Zakharyaschev. *Many-dimensional modal logics: theory and practice.* Cambridge University Press, 2003.

[14] Dov M. Gabbay. Fibred semantics and the weaving of logics. Part 1. Modal and intuitionistic logics. *The Journal of Symbolic Logic*, 61(4):1057–1120, December 1996.

[15] Dov M. Gabbay. An overview of fibred semantics and the combination of logics. In F. Baader and K.U. Schulz, editors, *Frontiers of Combining Systems: Proceedings of the 1st International Workshop, Munich (Germany)*, Applied Logic, pages 1–56. Kluwer Academic Publishers, March 1996.

[16] Dov M. Gabbay. *Fibring Logics.* Clarendon Press, New York, 1999.

[17] Dov M. Gabbay and Maarten de Rijke, editors. *Frontiers of Combining Systems 2.* Research Studies Press, Philadelphia, PA, 2000.

[18] Valentin Goranko. *Temporal Logics.* Elements in Philosophy and Logic. Cambridge University Press, 2023.

[19] Valentin Goranko. *Logic as a Tool - A Guide to Formal Logical Reasoning.* College Publications, 2nd edition, 2024.

[20] R. Goré. Tableau methods for modal and temporal logics. In M. D'Agostino et al., editor, *Handbook of Tableau Methods.* Springer, 1999.

[21] R. Kontchakov, C. Lutz, F. Wolter, and M. Zakharyashev. Temporalising tableaux. *Studia Logica*, 76:91–134, 2004.

[22] Agi Kurucz. Combining modal logics. In Patrick Blackburn, Johan Van Benthem, and Frank Wolter, editors, *Handbook of Modal Logic*, volume 3 of *Studies in Logic and Practical Reasoning*, pages 869–924. Elsevier, 2007.

Received August 2025

The Small Index Property — A Thank-you Note to Dov Gabbay

Ian Hodkinson
Department of Computing, Imperial College London, London SW7 2AZ, UK
i.hodkinson@imperial.ac.uk

Abstract

This note is to thank Dov Gabbay on his 80th birthday for some help he gave me, over 35 years ago. It contributed to a solution of a prominent problem in model theory and permutation groups, on whether the random graph has the small index property. Surprisingly, this work led before long to a proof that some modal fragments of first-order logic have the finite model property, topics that were of interest to Dov at the time.

1 Introduction

When I was small, I had to write 'thank-you letters' to people who had kindly given me birthday presents. Sad to say, it was not something I enjoyed. But on this occasion, I am writing a thank-you note going in the other direction, as it were, and it is a happier task for me.

I joined Dov Gabbay's research group in the Department of Computing at Imperial College London in November 1988, and stayed for several years. Much of our work was in temporal logic, a prominent part of logic in computer science, and one that Dov was already well known for: e.g., [7, 8, 9, 10, 11, 12, 13, 14]. The work led to a number of publications and took up most of my time, but I am not going to reminisce about it here. It would be of gravely limited interest, risk falling into sentimentality, and my memories after over 35 years are partial and unreliable and I would probably fail to give credit to many who deserve it.

Although all the same risks apply, I am going to reminisce instead about one curious outlying piece of work that I became involved with at that time. It seemed to have no relation to temporal logic or computing at all, though that impression

Many thanks to Jane Spurr for organising the production of this note.

was wrong. I, and as we will see, others too, owe considerable thanks to Dov in regard to it.

I arrived in Dov's group having recently done a PhD and postdoc with Wilfrid Hodges in mathematical logic (more precisely, model theory), and one particular open problem from that field came with me and lay on my desk. It asked *whether the random graph has the small index property.*

This question concerns the topological structure of the automorphism group of a certain infinite graph. It seemed about as far from any possible application as one could easily get. Nonetheless, I was preoccupied by it. I do not quite remember why, but it may have been because at least two of the very few papers that I had already written or co-written involved the small index property, and I was still attending some model theory seminars in Wilfrid's group at Queen Mary and Westfield College (now 'Queen Mary University of London'). The small index property and the random-graph instance of it were of wide interest in these circles and had attracted people who I had actually met, such as Peter Cameron, David Evans, Dugald Macpherson, Πeter Neumann, Simon Thomas, and John Truss, as well as Wilfrid and his associates at QMW and visitors including Ehud Hrushovski.

So I said to Dov that I would like to spend some 'work time' looking at this problem — not to the exclusion of my day job, but still seriously. (I might have been rather forceful in my request: I was a difficult person to work with.)

Not only did Dov permit this, *when he absolutely didn't have to,* but he actively encouraged me and indeed got involved in the technicalities of the work itself.

As a result of this generosity and enthusiasm, I had a small role in a collaboration of people working on this problem that started when Saharon Shelah visited Wilfrid in London in summer 1989. Daniel Lascar also became involved, in early 1991. Eventually we showed that the random graph does in fact have the small index property [21].[1]

This paper laid out a new method of proving the small index property, and also stimulated interest in another area that was involved in the proof: 'EPPA', the 'extension property for partial automorphisms'.

In 1991, EPPA was a nascent field, with (as I recall) only one result to its name — by Truss [30] — if indeed it had a name at all. A strengthening of Truss's result, wanted for [21], was proved by Hrushovski at a conference in Banff, Alberta, in spring 1991 [23]. This really got the EPPA ball rolling, and the field advanced quite rapidly, initially through work of Bernhard Herwig [16, 17] and Herwig–Lascar [18], with contributions by many others later. The study of EPPA is now very extensive, perhaps more so than the study of the small index property itself. See, e.g., [24] for

[1] We also showed the same for totally categorical structures — not discussed here.

recent references.

Within ten years, EPPA became the crucial ingredient in proving the 'finite algebra on finite base property' in algebraic logic [19, 2], and in Grädel's proof [15] of the finite model property for the *guarded fragment.* This is a 'modal fragment' of first-order logic. Such fragments were of interest to modal logicians at the time, including Dov himself; and the finite model property was also an interest of Dov's [7, 8]. EPPA was used later to prove the finite model property for other algebras and modal fragments, and these attempts in turn probably stimulated more work in EPPA itself.

So the story came full circle. A seemingly utterly abstract theoretical problem led unpredictably but rather directly and within a decade to contributions to modal logic, which is one of Dov's core interests and is used in computer science, philosophy, linguistics, economics, and even mathematical logic. People who use the phrase 'blue sky research' often claim as a justification for it that this kind of thing can happen, though they are probably hoping for lucrative applications. Although the finite model property of modal fragments is not yet among these, I still find this example rather striking — even salutory.

Of course the random graph would have been shown in time to have the small index property without my involvement, and perhaps better and more quickly. But the fact is that I *was* involved, and I will always be grateful to Dov for allowing me time to work on this beautiful problem. I, and I believe other workers in the affected areas too, owe him great gratitude for this.

So: *thank you Dov, and I wish you a very happy birthday!*

In the rest of this note, I will flesh out the picture with a little more technical detail for those interested. Any experts who have read this far need not read further. Mostly I will not give full proofs, as they were published over 30 years ago and would take too much space, but I will at least try to indicate how each topic involves the next. Sections 2 and 3 discuss the the random graph and the small index property. Section 4 concerns EPPA, and section 5 its applications to the finite model property.

2 The random graph

By a *graph,* we will mean an undirected loopfree graph, which is to say a set of 'nodes' endowed with an irreflexive symmetric binary 'edge' relation.

The so-called *random graph,* also known as the *Rado graph,* the *Erdős–Renyi graph,* and the *countable universal homogeneous graph,* is a particular countably infinite graph. It is generally defined only up to isomorphism. There are several equivalent characterisations of it:

1. It is the unique (up to isomorphism) graph on a countably infinite set N such that for each pair X, Y of disjoint finite subsets of N, there is a node in N that is connected by an edge to every node in X but to no node in Y.

2. (Erdős–Renyi) It is (up to isomorphism) the graph with the same set of nodes N that is obtained with probability 1 when a coin is tossed for each unordered pair of distinct nodes, and an edge placed between them just when the coin comes up 'heads'. Probably the name 'random graph' comes from this.

3. It is the unique (up to isomorphism) countable graph that is:

   - *universal:* it embeds all finite graphs as induced subgraphs,
   - *homogeneous:* every finite partial isomorphism (isomorphism between finite induced subgraphs) of it extends to an automorphism (an isomorphism from the whole graph onto itself).

   It plays the role for graphs that the rational order $(\mathbb{Q}, <)$ plays for linear orders: they are both in a certain sense the 'limit' of a class of finite structures (the class of all finite graphs and the class of all finite linear orders, respectively).

   This notion of limit can be made precise using a general theorem of Fraïssé [20, theorem 7.1.2]. The *Fraïssé limits* so obtained are countable universal homogeneous structures, unique up to isomorphism, and include the random graph, $(\mathbb{Q}, <)$, partial orders, tournaments, boolean algebras, and so on.

4. There are characterisations by binary expansions and quadratic residues [4, exercise 2.10(1)], and in other ways. As Cameron once said, since the random graph has probability 1, any construction is likely to give it! Ackermann used such devices as early as 1937 to construct directed versions of the random graph.

5. There is a connection to 0–1 laws in finite model theory, of relevance to computing. A graph can be seen as a structure for a relational signature L comprising a single binary relation symbol, interpreted as the edge relation. A first-order L-sentence is true in the random graph just when it holds in almost all finite graphs, which means that the proportion of graphs with nodes $\{0, 1, \ldots, n\}$ in which the sentence is true tends to 1 as $n \to \infty$. Up to isomorphism, the random graph is the only countable graph for which this holds.

The random graph is an attractive structure of great interest in several areas. But does it have the small index property?

3 The small index property

Let M be a (model-theoretic) structure. As we said, an *automorphism* of M is an isomorphism from M onto itself. For example, the automorphisms of a graph are the permutations of its set of nodes that take edges to edges and non-edges to non-edges.

The identity map on the domain of M is certainly an automorphism, and the inverse of an automorphism is an automorphism. Also, the composition $f \circ g$ of automorphisms f, g is an automorphism too. So the set of automorphisms of M forms a *group,* written $\mathrm{Aut}\, M$. Fixing M, I will write it as just G.

The group G carries a topology called the *topology of pointwise convergence.* For a finite sequence or 'tuple' $\bar{a}$ of elements of M, write $G_{\bar{a}}$ for the set of automorphisms of M that fix each element of $\bar{a}$. This is a subgroup of G. The basic open subsets of G are now defined to be the left cosets of the subgroups $G_{\bar{a}}$, taken over all tuples $\bar{a}$ of elements of M. Arbitrary open sets are the unions of such cosets — these are closed under finite intersections. Group inverse and composition are continuous with respect to this topology, so G becomes a *topological group.* Its open subgroups are precisely those that contain some $G_{\bar{a}}$.

The basic open subsets of G, together perhaps with $\emptyset$, are precisely the sets $\{g \in G : g \supseteq p\}$, taken over all finite partial isomorphisms p of M (that is, isomorphisms between finite substructures of M), and so these p determine the topology. The left cosets of any $G_{\bar{a}}$ are in one-one correspondence with the images $g(\bar{a})$ of $\bar{a}$ under automorphisms $g \in G$. For, if $g, h \in G$, then $gG_{\bar{a}} = hG_{\bar{a}}$ iff $h^{-1}g \in G_{\bar{a}}$, iff $h^{-1}g(\bar{a}) = \bar{a}$, iff $g(\bar{a}) = h(\bar{a})$.

Now if M is countable, there are only countably many such images $g(\bar{a})$, and it follows that the index $|G : G_{\bar{a}}|$ of any $G_{\bar{a}}$ in G — the number of cosets it has — is countable. Every open subgroup of G contains some $G_{\bar{a}}$, so its index is no bigger. Hence, when M is countable, *every open subgroup has countable index in G.* (Hence also, G has at most 2^{ω} open subgroups.[2])

DEFINITION 3.1. We say that (countable) *M has the small index property* if a strong converse holds: every subgroup of $\mathrm{Aut}\, M$ of index $< 2^{\omega}$ (we say 'of small index') is open in $\mathrm{Aut}\, M$.

This 'small index property' is good to have because if M has it, then the abstract group $\mathrm{Aut}\, M$ determines its topology: the open subsets are the unions of left cosets of subgroups of countable index. That is useful because in many cases, we can recover M up to bi-interpretability from $\mathrm{Aut}\, M$ as a topological group [1].

When I was working on this topic, I was mainly interested in whether (certain) Fraïssé limits have the small index property. A few of them were known to have

[2] This can often be improved to 'countably many' by a result of Evans [20, exercise 7.3.15].

it, including the infinite set without structure — unsurprisingly the first case to be established, by Semmes and Dixon–Neumann–Thomas [28, 5] — and the rational order $(\mathbb{Q}, <)$, by Truss [29]. (For a longer list of examples known at the time, see [21, example 1.1].) But for the random graph, in spite of some effort, it was open, and no general method of establishing the small index property was known.

It should be noted that not every Fraïssé limit has the small index property. A counterexample was given by Hrushovski.[3] Its automorphism group has 2^{2^ω} subgroups of index 2, which is too many for them all to be open.

However, a weaker small index property does hold generally: for every countable structure M, every *closed* subgroup of $\operatorname{Aut} M$ of small index is open. This was proved by Evans [6], extending a result of Kueker [25].

And Dov proved it too, from scratch, in discussions with me on the problem.

A generalisation of this was involved in the proof in [21] that the random graph has the small index property. I will write Γ for the random graph, and G for $\operatorname{Aut}\Gamma$. The useful topological notions of meagre and comeagre sets are applicable in this context. A subset of G is *comeagre* (think 'large') if it contains a countable intersection of dense open sets, and *meagre* (or *of first category*) if its complement is comeagre. It turns out that:

- every subgroup of G whose intersection with some $G_{\bar{a}}$ is meagre in $G_{\bar{a}}$ (in the subspace topology) has large index (2^ω) in G,
- every subgroup of G whose intersection with some $G_{\bar{a}}$ is comeagre in $G_{\bar{a}}$ must contain that $G_{\bar{a}}$, and is therefore open.

So we can restrict attention to subgroups H of G that are neither meagre nor comeagre within every subgroup $G_{\bar{a}}$. *Every comeagre subset of every $G_{\bar{a}}$ overlaps both H and its complement.* We complete the proof by showing that *H has large index in G.*

To indicate why this is, I will take a shortcut suggested by Hrushovski [23, p.412]. Even so, I can give only a scandalously bare outline of the argument. Given EPPA for graphs, defined in definition 4.1 below and established in [23], we can choose a sequence $f_0, f_1, \ldots \in G$ of 'mutually generic automorphisms'[4] of Γ such that:

G1. Each permutation of the f_n is induced by conjugation by some element of G. Formally, for each permutation σ of $\{0, 1, \ldots\}$ we have $(\Gamma, f_{\sigma(0)}, f_{\sigma(1)}, \ldots) \cong (\Gamma, f_0, f_1, \ldots)$, so there is $g \in G$ with $g^{-1} f_n g = f_{\sigma(n)}$ for each $n = 0, 1, \ldots$.

[3] See [26, §2.1], [4, p.108], and [20, exercise 7.4.15]. According to [26], the example was found by Cherlin, to refute a different but related conjecture; Hrushovski rediscovered it independently.

[4] Individual 'generic' automorphisms are typically those whose conjugacy class is comeagre. They were considered earlier by (e.g.) Lascar, Rubin, and Truss.

G2. $f_n \in H$ iff n is even, for each $n = 0, 1, \ldots$.

Since the set of even numbers has 2^ω images under permutations of $\{0, 1, \ldots\}$, it follows that H has 2^ω conjugates, so must have large index in G, as claimed.

The f_n are chosen by induction. Given $f_0, \ldots, f_{n-1}$, we use EPPA and another Fraïssé limit to find (within some $G_{\bar{a}}$) a comeagre set of suitable f_n. Comeagre sets arise from the topology on G, which in turn, as we saw, arises from *finite partial isomorphisms* of Γ. Fraïssé's machine ingests a special case of this: *automorphisms of finite substructures* of Γ. *The role of EPPA is to pass between these two.*

Having found a comeagre set of candidates, overlapping both H and its complement, we can then choose f_n in H or not, according to G2. By increasing $\bar{a}$ at each step, we can ensure that the resulting infinite sequence $f_0, f_1, \ldots$ satisfies G1. □

This idea works for the random graph and other Fraïssé limits where EPPA applies, but apparently not $(\mathbb{Q}, <)$ — Truss's method in [29] is still needed there.

4 EPPA

To construct the mutually generic automorphisms just mentioned, we used EPPA: the *extension property for partial automorphisms,* also called the *Hrushovski property.* As the term 'partial automorphism' can seem circular in this context, I will say 'partial isomorphism' instead. A *partial isomorphism* of a structure is an isomorphism between substructures of it.

DEFINITION 4.1. A class $\mathcal{C}$ of finite structures is said to have EPPA if each $A \in \mathcal{C}$ has an extension $B \in \mathcal{C}$ such that every partial isomorphism of A extends to an automorphism of B.

'Extension' just means that A is a substructure of B. For graphs, A would be an induced subgraph of B.

I believe the first EPPA-style result was due to Truss [30], who showed that any (single) partial isomorphism of a given finite graph extends to an automorphism of some larger finite graph. Hrushovski [23] then showed in a different way that the class of finite graphs has EPPA. This allowed our construction above of the mutually generic automorphisms $f_0, f_1, \ldots$ of the random graph, and it followed that the random graph has the small index property.

Since then, a number of other proofs of Hrushovski's theorem have appeared, both algebraic and combinatorial. A very short one can be found in [18, §4.1]. EPPA has been established for more classes of finite structures, and sometimes with extra benefits (as did [23] implicitly). For example, an early result of Herwig extended it beyond graphs, and gave more information:

THEOREM 4.2 (Herwig, [17]). *Let A be a finite structure in a finite relational signature L. Then there is a finite L-structure $B \supseteq A$, which we will call a* Herwig extension, *such that*

1. *every partial isomorphism of A extends to an automorphism of B,*
2. *every* 'live' *tuple of elements of B (that is, a single element or a tuple $\bar{b}$ such that $B \models R(\bar{b})$ for some relation symbol $R \in L$) is mapped into A by some automorphism of B.*

In addition, [16, 17] proved EPPA for the class of finite triangle-free and K_m-free graphs (and more), and it follows that the corresponding Fraïssé limits have the small index property. Herwig and Lascar [18] linked the subject with free groups. The study of EPPA is now extensive, but as far as I know, it remains open whether the classes of finite partial orders and finite tournaments have EPPA, and whether the corresponding Fraïssé limits have the small index property.

5 Finite model property

Some modal logicians in the 1990s were interested in finding *modal fragments* of first-order logic with the 'nice' properties of modal logic — decidability with good complexity, Craig interpolation, Beth property, finite model property, and so on.

Dov proposed the *finite-variable fragments* of first-order logic, perhaps because modal formulas can be translated to first-order formulas with only two variables — for example, $\Box(p \to \Diamond q)$ translates to

$$\forall y(R(x,y) \to (P(y) \to \exists x(R(y,x) \wedge Q(x)))). \tag{1}$$

For temporal formulas written with Until and Since, some of Dov's favourite connectives, three variables suffice.

For many-dimensional modal logics, Dov's proposal is correct, and the one- and two-variable fragments are quite well behaved, though unfortunately the rest are undecidable.

There were other ideas around. In [3], Andréka, van Benthem and Németi proposed the *bounded* or *guarded fragment,* in which all quantification is relativised to atomic formulas. More formally, assuming a finite relational signature, any atomic formula is guarded; boolean combinations of guarded formulas are guarded; and if $\psi(\bar{x}, \bar{y})$ is guarded, and α is an atomic formula — the *guard* — in which all of $\bar{x}, \bar{y}$ occur, then $\exists \bar{y}(\alpha \wedge \psi)$ is guarded. Hence, $\forall \bar{y}(\alpha \to \psi)$ is guarded. Evidently, (1) is guarded.

The guarded fragment and some of its extensions do have good properties, including the *finite model property:* every satisfiable guarded sentence is true in some finite model [15].

Very strikingly, *EPPA was used to prove this.* To indicate how on earth it could be involved, here is a sketch of the idea, based on [2, 15]. Similar arguments work in algebraic logic [19, 2]. Let σ be a guarded sentence written with n variables, say, and true in some structure M. We find a finite model of σ.

By expanding M by a new relation symbol $P_\varphi(\bar{x})$ for each subformula $\varphi(\bar{x})$ of σ, with $M \models \forall\bar{x}(P_\varphi(\bar{x}) \leftrightarrow \varphi(\bar{x}))$, we can suppose that (†) the quantifier-free type of any tuple $\bar{a}$ in M determines whether or not $M \models \varphi(\bar{a})$, for each subformula φ of σ.

Choose a finite substructure $A \subseteq M$ containing representatives of all quantifier-free types of n-tuples occurring in M. Using theorem 4.2, choose a finite Herwig extension $B \supseteq A$. We show that for each subformula $\varphi(\bar{x})$ of σ,

(∗) $M \models \varphi(\bar{a})$ iff $B \models \varphi(\bar{a})$, for every tuple $\bar{a}$ of elements of A.

Applying this to σ will then show that B is our desired model of σ.

We prove (∗) by induction on φ. It is clear for atomic formulas, and the boolean cases are easy. So consider the case $\varphi(\bar{x}) = \exists\bar{y}(\alpha(\bar{x},\bar{y}) \wedge \psi(\bar{x},\bar{y}))$, assume (∗) for ψ inductively (it also holds trivially for α, which is atomic), and let $\bar{a}$ be a tuple in A.

Suppose that $M \models \varphi(\bar{a})$. Then there is $\bar{b}$ in M with $M \models \alpha(\bar{a},\bar{b}) \wedge \psi(\bar{a},\bar{b})$. Choose a representative $\bar{a}'\bar{b}'$ in A of the quantifier-free type in M of $\bar{a}\bar{b}$. By (†), $M \models \alpha(\bar{a}',\bar{b}') \wedge \psi(\bar{a}',\bar{b}')$ as well. Inductively, $B \models \alpha(\bar{a}',\bar{b}') \wedge \psi(\bar{a}',\bar{b}')$. Now the map $(\bar{a}' \mapsto \bar{a})$ is a partial isomorphism of M, hence also a partial isomorphism of its substructure A. As B is a Herwig extension of A, there is $g \in \operatorname{Aut} B$ with $g(\bar{a}') = \bar{a}$. Let $\bar{c} = g(\bar{b}')$, a tuple in B. As automorphisms preserve all formulas, $B \models \alpha(\bar{a},\bar{c}) \wedge \psi(\bar{a},\bar{c})$. Hence, $B \models \varphi(\bar{a})$ by definition of φ.

Conversely suppose that $B \models \varphi(\bar{a})$. Then there is $\bar{b}$ in B with $B \models \alpha(\bar{a},\bar{b}) \wedge \psi(\bar{a},\bar{b})$. The guard α is an atomic formula in which all of $\bar{x},\bar{y}$ occur. So $\bar{a}\bar{b}$ is a live tuple in B. As B is a Herwig extension of A, there is $g \in \operatorname{Aut} B$ with $g(\bar{a}\bar{b}) = \bar{a}'\bar{b}'$, a tuple in A. Then $B \models \alpha(\bar{a}',\bar{b}') \wedge \psi(\bar{a}',\bar{b}')$ since automorphisms preserve all formulas. Inductively, $M \models \alpha(\bar{a}',\bar{b}') \wedge \psi(\bar{a}',\bar{b}')$, so $M \models \varphi(\bar{a}')$. But the quantifier-free types of $\bar{a},\bar{a}'$ are the same in B (since $g(\bar{a}) = \bar{a}'$) and hence in M since A is a substructure of both. So by (†), $M \models \varphi(\bar{a})$ as required. □

This result extends to stronger guarded fragments such as the loosely guarded, packed, and clique-guarded fragments, into which temporal formulas involving Until and Since can be translated. The proofs needed stronger Herwig-style theorems, such as [22]. So perhaps the finite model property for modal fragments of first-order logic stimulated development of EPPA.

6 Conclusion

Once again, I thank Dov for allowing me to work on the small index property 'in work time' all those years ago. Probably neither of us expected that it would be relevant to Dov's own core interests, but I have tried to show that it was, and in no small way, so his generosity (and forbearance) paid off. Happy 80th birthday, Dov!

References

[1] G. Ahlbrandt and M. Ziegler. Quasi finitely axiomatizable totally categorical theories. *Ann. Pure. Appl. Logic*, 30:63–82, 1986.

[2] H. Andréka, I. Hodkinson, and I. Németi. Finite algebras of relations are representable on finite sets. *J. Symbolic Logic*, 64:243–267, 1999.

[3] H. Andréka, J. van Benthem, and I. Németi. Modal logics and bounded fragments of predicate logic. *J. Philosophical Logic*, 27:217–274, 1998.

[4] P. J. Cameron. *Oligomorphic permutation groups*, volume 152 of *London Math. Soc. Lecture Note Series*. Cambridge University Press, 1990.

[5] J. D. Dixon, P. M. Neumann, and S. Thomas. Subgroups of small index in infinite symmetric groups. *Bull. London Math. Soc.*, 18:580–586, 1986.

[6] D. M. Evans. A note on automorphism groups of countably infinite structures. *Archiv der Mathematik*, 49:479–483, 1987.

[7] D. M. Gabbay. On decidable finitely axiomatisable modal and tense logics without the finite model property, part I. *Israel Journal of Mathematics*, 10:478–495, 1971.

[8] D. M. Gabbay. On decidable finitely axiomatisable modal and tense logics without the finite model property, part II. *Israel Journal of Mathematics*, 10:496–503, 1971.

[9] D. M. Gabbay. Decidability results in non-classical logics, part I. *Ann. Math. Logic*, 8: 237–295, 1975.

[10] D. M. Gabbay. Model theory for tense logics. *Ann Math Logic*, 8:185–235, 1975.

[11] D. M. Gabbay. *Investigations in Modal and Tense Logics with Applications to Problems in Philosophy and Linguistics.* Synthese volume 92. D. Reidel, Dordrecht, 1976.

[12] D. M. Gabbay. Expressive functional completeness in tense logic (preliminary report). In U. Monnich, editor, *Aspects of Philosophical Logic*, pages 91–117. D. Reidel, Dordrecht, 1981.

[13] D. M. Gabbay. An irreflexivity lemma with applications to axiomatizations of conditions on tense frames. In U. Monnich, editor, *Aspects of Philosophical Logic*, pages 67–89. D. Reidel, Dordrecht, 1981.

[14] D. M. Gabbay. Declarative past and imperative future: Executable temporal logic for interactive systems. In B. Banieqbal, H. Barringer, and A. Pnueli, editors, *Proceedings of Colloquium on Temporal Logic in Specification, Altrincham, 1987*, volume 398 of *Lecture Notes in Computer Science*, pages 67–89. Springer-Verlag, 1989.

[15] E. Grädel. On the restraining power of guards. *J. Symbolic Logic*, 64:1719–1742, 1999.

[16] B. Herwig. Extending partial isomorphisms on finite structures. *Combinatorica*, 15: 365–371, 1995.

[17] B. Herwig. Extending partial isomorphisms for the small index property of many ω–categorical structures. *Israel J. Math.*, 107:93–123, 1998.

[18] B. Herwig and D. Lascar. Extending partial isomorphisms and the profinite topology on the free groups. *Trans. Amer. Math. Soc.*, 352:1985–2021, 2000.

[19] R. Hirsch, I. Hodkinson, M. Marx, S. Mikulás, and M. Reynolds. Mosaics and step-by-step. In Orłowska [27], pages 158–167. Appendix to [31].

[20] W. Hodges. *Model theory*, volume 42 of *Encyclopedia of mathematics and its applications.* Cambridge University Press, 1993.

[21] W. Hodges, I. Hodkinson, D. Lascar, and S. Shelah. The small index property for ω-stable ω-categorical structures and for the random graph. *J. London Math. Soc.*, 48: 204–218, 1993.

[22] I. Hodkinson and M. Otto. Finite conformal hypergraph covers and Gaifman cliques in finite structures. *Bull. Symbolic Logic*, 9:387–405, 2003.

[23] E. Hrushovski. Extending partial isomorphisms of graphs. *Combinatorica*, 12:411–416, 1992.

[24] J. Hubička, M. Konečný, and J. Nešetřil. All those EPPA classes (strengthenings of the Herwig–Lascar theorem). *Trans. Amer. Math. Soc.*, 375:7601–7667, 2022.

[25] David W. Kueker. Definability, automorphisms, and infinitary languages. In Jon Barwise, editor, *The Syntax and Semantics of Infinitary Languages*, pages 152–165. Springer-Verlag Berlin, Heidelberg, 1968. ISBN 978-3-540-35900-5.

[26] D. Lascar. Autour de la propriété du petit indice. *Proc. London Math. Soc.*, 62:25–53, 1991.

[27] E. Orłowska, editor. *Logic at Work*, volume 24 of *Studies in Fuzziness and Soft Computing.* Physica-Verlag, Heidelberg/New York, 1998. Essays dedicated to the memory of Elena Rasiowa. ISBN 3-7908-1164-5.

[28] S. W. Semmes. Endomorphisms of infinite symmetric groups. *Abstracts Amer. Math. Soc.*, 2:426, 1981.

[29] J. K. Truss. Infinite permutation groups; subgroups of small index. *J. Algebra*, 120: 494–515, 1989.

[30] J. K. Truss. Generic automorphisms of homogeneous structures. *Proc. London Math. Soc.*, 65:121–141, 1992.

[31] Y. Venema and M. Marx. A modal logic of relations. In Orłowska [27]. Report no. IR-396, Faculteit der Wiskunde en Informatica, Vrije Universiteit Amsterdam, 1995.

Received August 2025

What Follows from all that Data? Logic in the Methodology of Data-Intensive and AI-Driven Science

Hykel Hosni[*]
Logic Uncertainty Computation and Information (LUCI) Lab, Department of Philosophy, University of Milan
hykel.hosni@unimi.it

Jürgen Landes[†]
Munich Center for Mathematical Philosophy, LMU Munich
juergen_landes@yahoo.de

In honour of Dov Gabbay's 80th birthday

Abstract

There is no foreseeable future in which science is not about data and the inferences data license. For centuries, logic has been the tool to analyse inference. And yet, logic is vastly underappreciated in the current methodology of data-driven science, as we argue in this paper. We first outline two historical reasons behind this mismatch, then highlight the need to bridge it by examining a widely used form of scientific inference: Null Hypothesis Significance Testing. Finally, we argue that the question: what follows from data? is ripe to be tackled by logicians. We submit that this will help lay a sound methodological foundation for the practice of data-intensive and AI-driven science.

[*] Hykel Hosni acknowledges generous funding from the Italian Minister for University and Research (MUR) in the form of the FIS 2021 *Advanced Grant* "Reasoning with Data" (ReDa) G53C23000510001.

[†] Jürgen Landes reports financial support was provided by German Research Foundation project number 528031869.

1 Introduction

There is no foreseeable future in which science is not Data-Intensive and AI-Driven (DIAD) [34, 40, 55]. While the "transformative" promise of AI techniques is ubiquitous in science, it is particularly detectable in the biomedical sciences, spanning drug discovery [71] and diagnostics [11]. This is, of course, great news for healthcare [6]. Provided, that is, the key questions of transparency and trustworthiness in AI-driven science are addressed quickly and effectively enough [61, 68].

The logical community has responded impressively to the problem of transparency of AI systems, with the field of Formal/Abstract Argumentation Theory emerging as cornerstone of explainable AI [3, 5, 19, 47]. The logicians' involvement in formulating and addressing the challenge of trustworthy AI is relatively more recent [81], but it is delivering very promising results [16]. Those approaches typically combine logical and statistical methods, used to account for the stochastic aspects of data. There is, in short, no doubt that logicians are making an impact in this new AI spring [28].

Those recent developments add to the contributions made over the past four decades to the field of Knowledge Representation and Reasoning subfield of AI. Well-known success stories here include non-monotonic logics, paraconsistent logics, probabilistic and fuzzy logics, logics for multi-agent systems, logics for collaboration – the list is *very* long, see for example [18, 32, 57, 58, 75]. Much of the cutting-edge contribution of logic to AI today is inspired, if not grounded, on those logical systems. Those, in turn, largely developed starting from the 1970s in response to the pressing needs arising in computer science and symbolic AI. The kind of humus was pinned down in the inaugural editorial of the *Journal of Logic and Computation*

> We do not mean 'Logic' as it is now. We mean 'Logic' as it will be, as a result of the interaction with computing.[29]

The ensuing three decades witnessed the co-evolution of logic and computation, paving the way for a *turn towards the practical* [30] which challenged the academic boundaries of mathematics, computer science, philosophy, and law.

And yet, logic is vastly underappreciated in the methodology of data-driven *science*. As we noted in [44], and will argue in greater detail herein, this neglect is detrimental to both logicians and scientists. To be able to serve the methodology of data-intensive and AI-driven science, logic and logicians would benefit from undergoing yet another turn. This time towards the stochastic, the practical problem solving.

Timothy Gowers put forward a distinction between the mathematicians who work on theory and those that solve problems [63]. There is a very long and successful tradition of logicians solving problems in probabilistic logic, which starts with Boole and De Morgan, and includes [37, 38, 62, 64, 65, 80] as very influential frameworks. As we point out in [43] those are proposals which seek to extend classical logic with the ability to express and do inference with probabilities. This has led and keeps leading to deep mathematical results. It is however unlikely that those results will be directly applicable to methodological purposes of data-intensive and AI-driven science for the same sort of reasons classical logic won't do (see below). To do so, borrowing again from Gowers, we need more theory. As we argue below, we need to think rather carefully at the building blocks of data-driven inference, all the way down to what we think valid data-driven inference should (not) be. Another Fields medalist, David Mumford, has rather radical ideas on this score. He urges us to put random variables into the foundations of mathematics, thereby marking a *dawning age of stochasticity* [15]. Among the consequences of doing so, one needs to reject the continuum hypothesis. Theory (re)building need not go that far, of course, but Mumford's point brings to the foreground that our understanding of probability theory has profound consequences for foundational questions of logic. And probability, as Laplace pointed out very clearly [53], arises precisely where data and the lack thereof play an equally important, and intuition-defying, role in scientific inference.

Using Gowers' terms, we thus urge logicians to solve real-world (inspired) problems *and* for modern logical theory to be applicable in science, again.

The rest of this paper is organised as follows. We next identify two potential reasons for the disconnect between scientists and logic (Section 2) and then zoom in briefly on the logical foundations of statistical inference (Section 3). We conclude with some reflections on the challenges that lie ahead (Section 4).

2 How working scientists lost contact with logic

Logic in vastly under-appreciated by working scientists and virtually absent from the methodology of data-driven science [44]. We briefly discuss two potential reasons for this state of affairs. The first is rooted in the enduring influence of Alfred Tarski's stance on the role of logic in the methodology of empirical science. The second originates in the very enthusiastic reception of hypothetico-deductive and falsificationist approaches within the sciences, resulting in a very narrow perspective on logic.

2.1 Tarski's restriction and its influence on logicians

Alfred Tarski, as reported by his student John Corcoran, purportedly referred to himself as "the greatest living sane logician." The qualification was, of course, instrumental to rank himself above Kurt Gödel. Be this as it may, Tarski is certainly one of the great modern logicians. He is also the author of the first popular-science book in logic, which shaped the subject and its image within the wider scientific landscape. Written in Polish in 1936, it appeared in German with the title *Einführung in die mathematische Logik und in die Methodologie der Mathematik* and it was turned into a textbook for the 1941 English translation, titled *Introduction to Logic and to the Methodology of Deductive Sciences.* As explained in the Preface, Tarski contrasts the latter with "the empirical sciences" which do not lend themselves purposefully to logical methods:

> I see little rational justification for combining the discussion on logic and the methodology of empirical sciences in the same college course [77, p. xiii].

This should not be read as Tarski dismissing outright the importance of empirical science. In a comment to a seminar quoted in Chapter 10 of [24] Tarski states:

> It would be more than desirable to have concrete examples of scientific theories (from the realm of the natural sciences) organized into deductive systems. Without such examples there is always the danger that the methodological investigation of these theories will, so to speak, hang in the air. Unfortunately, very few examples are known which would meet the standards of the present-day conception of deductive method and would be ripe for methodological investigations [. . .] The development of metamathematics, that is, the methodology of mathematics, would hardly have been possible if various branches of mathematics had not previously been organized into deductive systems.

Requiring axiomatisation as a necessary condition for the applicability of logic to scientific methodology, effectively means restricting the applicability of logic to metamathematics. Owing to Tarski's scientific and academic prominence, this view has been very influential in shaping the twentieth-century perception of the subject, in and outside logic. The recent [25] testifies the enduring importance of this perception.

However, as noted by Wilfried Hodges in his 2011 *Division of Logic, Methodology and Philosophy of Science* Presidential Address [41], not very many would go on doing the methodology of the empirical sciences as Tarski suggested. The birth and development of DLMPS owes much to Tarski, as recounted by [24], so his views shaped the way Logic, Methodology and Philosophy of science have been construed there, especially with respect to their lack of interaction. Indeed, inspection of the DLMPS tables of contents over the decades shows very little trace of logic being part of the methodology of "empirical" sciences. The emerging picture is rather one in which DLMPS provides a rich umbrella encompassing both "metamathematics" and the methodology of "empirical" sciences, but quite separately.

2.2 The Popper/Hempel influence on scientists

A second reason for the current underappreciation of logic in science may be rooted in the enthusiastic acceptance, by working scientists, of hypotetico-deductivist disconfirmation as the signature of the scientific method. In this picture, which has been incredibly influential in the construction of the self-image of generations of scientists [74], logic contributes only with *modus tollens* (Popper, 2005,p. 89). Interestingly, Ken Aizawa argues that Hempel may have exerted an even stronger influence than Popper in elevating *modus tollens* to the only logical snippet deemed relevant to experimental science [2]. This picture has percolated magnificently in the mindset of the methodologically conscious scientist: scientific hypotheses can only be *disconfirmed.* For confirmation relies, as Hempel notes, on the deductive fallacy of *affirming the consequent* [39]. Yet, as argued by Pólya, "affirming the consequent" is what scientists do when taking stock of (a positive) correlation [70]. To mark its importance in data-driven reasoning, Pólya rebrands the long-vituperated fallacy as the *fundamental pattern of induction*, on which we will come back soon. Some scientists seem to have taken notice, notably [8, 48], but they appear to be rare exceptions.

We have a logical blind spot: many working scientists are unaware of the rich variety of logical nuances that would provide a much more adequate methodological pillar in data-driven inquiry compared to classical validity. Aizawa suggests [1] that fixing this blind spot requires replacing the HD view of disconfirmation with a framework centred on *abductive reasoning.* This is good news, because practical logicians have contributed to this area consistently for decades, see [31, 56] for overviews. However, there are some usually neglected challenges, identified in Section 4 below, that must be addressed.

Summing up, the restriction of logic to the methodology of deductive sciences on the one hand, and the extraordinary success of the Popper/Hempel view impacted negatively on the scope of logic and logical methods outside mathematics. As a consequence, the logical questions on data-driven inference were left to statisticians.

3 *Inference* in statistical inference

Around the same time Hempel was advocating for the primacy of classical logic, Richard Feynman was crystallizing the role of modus tollens in the methodology of science: "It doesn't make any difference how beautiful your guess is [..]. If it disagrees with experiment, it's wrong" [26]. We take no issue, of course, with the message delivered by the Feynman dictum. However, it may not be clear what it means for data to (dis)agree with a hypothesis, as pointed out long ago by Duhem and Quine. In fact, with the exception of very simple cases, it is *almost never* clear, as the one century-old debates on statistical significance clearly testify [7, 54, 59, 67, 82].

3.1 Case study: Null Hypothesis Significance Testing

There is a commonplace view to the effect that statistics *is* "the logic of science" [48, 49]. In an overview of R.A. Fisher's contributions to statistics, Efron says:

> Fisher believed that there must exist a logic of inductive inference that would yield a correct answer to any statistical problem, in the same way that ordinary logic solves deductive problems [21].

While this is a very logical way of putting it, likely borrowed from Boole's formulation of his *general method* [9], identifying criteria of validity for data-driven inference is not what Fisher (or any other statistician, for that matter) went on to do. Rather than being loosely inspired by logic, the logic of data-driven science should be logic; even better, a logical framework for the sciences.

To see why, take the procedure known as *Null Hypothesis Significance Testing* (NHST). Statisticians are often appalled by this acronym, because it juxtaposes the conflicting ideas of Fisher on one hand, and the Neyman–Pearson duo on the other. But given that NHST is what most working scientists do, it suits our present purposes very well.

The key idea behind NHST is quite entrenched in scientific thinking: *to assess a hypothesis, look at its consequences.* Specifically, we want to draw a conclusion about a statistical hypothesis of interest on the basis of the data it would generate, if true. So let H_0 be the statistical hypothesis of interest, referred to (after Fisher) as the *null hypothesis*, and denote the observed (low probability) data by D.

Specific instances of NHST are variations on the following pattern:

(NHST1) Data D are observed.

(NHST2) If H_0 were true, then data at least as improbable (according to some test statistic) as D would be very improbable.

(NHST3) Either we accept that very improbable events have occurred or we reject H_0.

(NHST4) Very improbable events do not occur.

(NHST5) Therefore, we reject H_0.

Since Fisher's [27] it has become customary to translate "very improbable", with $p < 0.05$. This threshold stuck, so 0.05 is indeed the most used *p-value*. Informally, this is the calculated (hypothetical) conditional probability of observing data as improbable as, or more improbable than D, given that H_0 is true. Hence, the purpose of NHST so construed is to let the data speak about H_0: a small-enough p-value being licence to *reject* H_0.

Given the observations in Section 2, it is unsurprising that concluding (NHST5) from the conjunction of the premisses (NHST1-NHST4) is often justified by invoking *modus tollens*. Here are a few notable examples: [17] speaks of "inductive modus tollens", [59] refers to "statistical modus tollens", whereas [72] tellingly notes that the analogy with *modus tollens* lends tests of significance their prominence in the "scientific method":

> [modus tollens] is at the heart of the philosophy of science, according to Popper. Its statistical manifestation is in [the] formulation of hypothesis testing that we will call 'rejection trials'. ([72], p.72)

Quite interestingly, this view is shared by prominent critics of NHST as well, see e.g. [76], more on this below.

The NHST pattern of inference can be mapped to something that looks like *modus tollens* by making the following assumptions:

(A1) Boolean implication is the adequate formalisation of the conditional statement featuring in the p-value.

(A2) The clause that the observed data given H_0 should be at least as improbable as D (as measured by some test statistics) can be omitted with no loss of generality;

(A3) Small (but non-zero) probability events are false.

With (A1-A3) in place we get the following instance of *modus tollens*:

(NHST'1) D

(NHST'2) $H_0 \to \neg D$

(NHST'3) Therefore, $\neg H_0$

Before going into some detail as to why (A1-A3) are rather unpalatable, note that (A1) and (A3) are compatible with the so-called *Fisher Disjunction*:

> The force with which such a conclusion is supported is logically that of the simple disjunction: *Either* an exceptionally rare event has occurred, *or* the theory of random distribution is not true [27].

The untenability of (A2) is easily ascertained by probabilistic means, and hence well-appreciated in the statistical literature. Its implausibility is in fact the reason why the awkward "as improbable or more improbable" bit cannot be omitted from a correct conceptualisation of the p-value [46]. So let's focus on the other two assumptions.

The problem with (A1) is more subtle, and quite hard to spot unless one looks at it from a probability logic angle. Recall that the probability of a Boolean implication does not equal, in general, the "associated" conditional probability, i.e:

$$P(D \mid H) \neq P(H \to D),$$

where P is a probability function on the propositional language which includes H and D, and which is closed under the usual Boolean connectives. This obvious fact may easily go unnoticed to the working scientist as a consequence of the ambiguity of natural language. More precisely, it may be hard to see the difference between

- The probability of the data given the null hypothesis.
- The probability that the null hypothesis implies the data.

It is no mystery that implication is vastly misunderstood outside its mathematical usage.

(A3) amounts to a metaphysical assumption on the nature of probability. Sometimes it is referred to as the *Cournot Principle* after [14], and it lies at the heart of the frequentist view of probability underlying NHST [73]. Loosely put, the Cournot principle says that the (physical) meaning of probability is that very small-probability events do not happen. This relates to a more general question: *how small should* *a small probability be before it is rationally neglected*? Debated at least since Jacob Bernoulli's *Ars Conjectandi*, it isn't likely to be decided any time soon. So rather than untenable, the adequacy of (A3) seems not to be tackled meaningfully from the logical point of view. There is some irony in this, of course, in light of the rhetoric that views NHST as purely data-driven, and hence objective. An irony that of course has not escaped the attention of critics. See [36, Chapter 14] for a concise, witty overview by I.J. Good.

3.2 On the analogy between statistical inference and classical deduction

In light of the above discussion, it is curious that the statistical camp seems to be generally happy with justifying NHST through *modus tollens*. Indeed, some have strongly dissented, pointing to the failure of *modus tollens* as a strong line of criticism of p-value based significance [23, 50, 76]. Consider the following (now) standard example made popular by [69]:

(P1) Harold is a member of Congress;

(P2) If Harold is a US citizen, than he is most probably
not a member of Congress;

(P3) Therefore, Harold is likely not a US citizen.

While premisses (P1-P2) may be easy to accept, the conclusion (P3) is not, since being a US citizen is a necessary condition to sit in the US Congress. Hence, the example shows a context in which (NHST'1-NHST'2) above may be satisfied, but not (NHST'3).

The 'Harold example' certainly suggests that there is something to be careful about in NHST inference, but it still concedes a lot to assumptions (A1) and (A3) above. For it shows the *classical* invalidity of NHST inference. However, similarly to what was noted above regarding (A1-A3), classical validity is hardly adequate to capture the "most probably" and the "likely" that appear in (P2) and (P3). As argued in detail in [4] turning to *non-monotonic consequence relations* pays nice dividends. In addition to expressing adequately the qualitative uncertainty conveyed by (P2) and (P3), suitably defined non-monotonic consequence relations are capable of vindicating some of the intuition linking conceptually NHST to *modus tollens*. Indeed, from the non-monotonic-logic point of view, the natural conclusion of (P1-P2) above is that *Harold is not a typical US citizen*.

The discussion above leads to the more abstract and general question as to whether data-driven patterns of reasoning can at all "borrow their validity", so to speak, from analogous classical patterns. Let’s call this the Analogy Principle. We have seen that *modus tollens* is classically valid, but fails in contexts of interest to significance-based inference. But the Analogy Principle is unreliable also in the other direction. To see this, consider Pólya’s *fundamental inductive pattern* mentioned briefly above:

$$\frac{\begin{array}{r} A \to B \\ B \end{array}}{A \text{ [becomes] more plausible.}} \tag{FIP}$$

As a scheme of inference, (FIP) has no direct counterpart in classical logic, but if one reasons by analogy and takes “A [becomes] more plausible" as “A [is] true", then (FIP) is precisely the deductive fallacy of “affirming the consequent” despised by Hempel.

So the Analogy Principle can mislead in both directions: data-driven inference may not get methodological support from analogous classical patterns of inference, and conversely, classically invalid patterns of inference may be crucial to their analogues for data-driven reasoning. No wonder then that one sees rather spectacular failures of the Analogy Principle in connection to the (mis)applications of NHST [42, 50, 66].

Taking stock from the failure of the Analogy Principle we may conclude that validity does not form a continuum spanning mathematical proof and data-driven inference. Indeed, it is well known that *modus tollens* need not be probabilistically valid [10], and in general fails to deliver a point-valued probability [79]. Hence, it is not advisable to trust a statistical pattern of inference because it can be articulated as the stochastic analogue of a classically valid one. Interestingly, this view resonates with Tarski’s insistence that logic’s proper role resides exclusively in the methodology of deductive sciences. However, one century on, both the scientific and the logical landscapes have changed dramatically. The coming of age of data-intensive and AI-driven science, with its ability to immediately impact society and ultimately our daily lives, compels us to devise a fail-safe “general method” for valid data-driven inference. The time is ripe for logicians to venture where Tarski saw no role for logic.

4 Challenges

The coming of age of data-intensive and AI-driven science raises questions revolving around a key one: *what follows from stochastic data*? Addressing it involves taking a step back, and asking *logical* questions about statistical inference. The previous Section demonstrates that we must be cautious before we leap. We conclude this paper by offering some reflections on two rather distinct challenges inherent in making this leap.

4.1 Choosing a good trade-off between context-dependent and formal nature of validity

Suppose we agreed to play dice, no money involved. The first who rolls three "ones" wins. If you win on the first try, we'd be witnessing a 1/216 probability event. Would this be evidence that the dice are loaded? NHST would say so, with rather strong "significance". But an event of this improbable kind is certainly not impossible. Second round, and you score again. The probability of you winning twice in a row now gets quite small at 1/46656. This is roughly four times smaller than the chance of a (natural) triplet birth. Again not impossible, but uncommon enough to justify jumping to the conclusion that maybe the dice *are* loaded after all. Doing so is reasoning along the lines of NHST. And so do particle physicists when claiming discoveries [20]. Famously, the Higgs Boson was announced with a "5 sigma" significance, which roughly corresponds to a p-value of around 1 in 3.5 million. Those are just two of very many circumstances in which significance inference delivers unquestionable results. But apparently minor variations on the NHST pattern of inference soon lead to very questionable outcomes [13, 45]. So, what appears to be the same pattern of inference seems to be sometimes valid and sometimes not. This is a clear call for logical scrutiny.

Defining the meaning of *formality* may not be straightforward. However, one rather uncontroversial view holds formality to be pinned down by closure under uniform substitution, sometimes also referred to as *structurality*. While it is a characteristic property of classical logic, closure under substitution fails for many logics for practical reasoning, notably non-monotonic logics [57, P. 14] and paraconsistent logics [32, P. 76]. However, since classical logic is manifestly unfit for purpose, a trade-off between formality and concreteness must be identified.

It is well-known from the psychology of reasoning that, while we may struggle with basic inferential patterns in decontextualized tests, we do very well in dialogical, argumentative contexts. The key difference being that in the latter we are genuinely trying to evaluate or convey practically directed information [60]. So, if we are interested in providing logical criteria for the validity of data-driven inference, we must strike a balance between:

- offering criteria that are applicable beyond the single, specific, case;

- ensuring that some context-specific aspects of the data-driven inference we are interested in must be represented.

It turns out that the terms of this trade-off have a lot to do with opposing statistical methodologies. Discounting some inevitable and very human attachment to one's own ideology, it is quite easy to see scientific communities naturally clustering around certain statistical methodologies. The kind of data that is typically available in a given field plays a big role in matching probabilistic ideology and research contexts. This may account for the fact that, say geneticists rarely identify as "Bayesians", whereas the strongest opposition to the NHST hard-core frequentism comes from the social sciences [82]. So it is the nature of the data-generating mechanisms that provides the context for data-driven reasoning. And this is what probably marks the difference between apparently similar, but logically distinct, patterns of data-driven inference.

Creating families of logical systems capable of adapting to the context-depending demands of practical reasoning is the core methodology of Labelled Deductive Systems (LDS). While we are not aware of any LDS framework motivated by the problem of accounting for data-driven inference, a recent retrospective [33] demonstrates the applicability of LDS to the analysis of legal evidence, which is a particularly regimented case of data-driven inference. Similarly, the logical framework for data-driven reasoning recently put forward in [4] seems to be promising by striking a reasonable balance between formality and context-dependence. The formality of the concept of validity is provided by a *blueprint consequence relation*, whose intended semantics is built on the idea that data can reject, to some degree, and possibly by mistake, any well-formed (statistical) hypothesis. Then, different data-generating contexts give rise to distinct consequence relations, fine-tuning the "blueprint".

4.2 Communicating across academic disciplines and societal sectors

Let us close with a challenge which is hardly ever on the logician's agenda, but which is crucial nonetheless: finding an effective way to communicate the criteria for valid data-driven reasoning across disciplines and, ideally, across sectors.

First, as noted in Section 2, working scientists have very limited familiarity with the subject. This owes to the fact that most scientists acquire only indirect training in logic, chiefly through introductory chapters in mathematics textbooks. Those typically cover rudimentary classical logic, often expressed in the form of naive set theory. This results in scientists typically identifying logic with an informal version of classical logic, very much in line with the Popper/Hempel view recalled above. Of course, we can't expect, say, a translational biologist to get acquainted with the formal detail of logics for data-driven reasoning. So, the challenge arises to make the results which will (hopefully) spring by addressing the logical issues raised above *usable* for the translational biologist. It is no small challenge, of course, but we can look at statistical software for inspiration. The computational turn in statistics made data-intensive, and then AI-driven inference, unprecedentedly accessible [22]. Modelling the construction of data-driven scientific knowledge with abstract argumentation theory stands out as a promising route to achieving that [52].

Second, the curious scientist, policy-maker or ideal (in our view!) citizen who occasionally enjoys the popular science book, will only find logic-as-metamathematics titles on the shelves. Communicating the golden age of mathematical logic with the extraordinary contributions of Cantor, Russell, Hilbert, Gödel, Tarski, and Turing, certainly makes for good storytelling which is moreover naturally in tune with what most readers expect. Again, we take no issue with the narrative recounting how logicians ascended the highest peaks of human intellect. Yet, we badly need logical methods to enter inter-sectoral dissemination. Scientists, policy-makers, and curious citizens must have a chance to appreciate the role of logic in shaping the methodology of data-intensive and AI-driven science. Hence, the popular science bookshelves must hold logic books that extend well beyond the foundations of mathematics. Eugenia Cheng's [12] and Adam Kucharski's [51] are recent and successful books that go in this direction. Many more are needed.

The history of the subject is filled with logical types thriving on practical problems. I.J. Good's [35] reports on previously classified material of the work done by Alan Turing at Bletchley Park. Among other things, Good describes how his line manager worked out from scratch the (odds form of the) Bayes rule while trying to crack Enigma. A few years later, on 20 February 1947, Turing delivered a lecture at the London Mathematical Society, reporting on the construction of the Automated Computing Engine (ACE), where he argues extensively for giving priority to large enough and accessible enough data storage:

> I have spent a considerable time in this lecture on this question of memory, because I believe that the provision of proper storage is the key to the problem of the digital computer, and certainly if they are to be persuaded to show any sort of genuine intelligence much larger capacities than are yet available must be provided. [78]

Three quarters of a century later, the historical events have been turned into an award-winning movie (The Imitation Game) and the digital computer with proper storage (measured in terabytes) has become ubiquitous. We contend that after solving the challenges of building a digital computer and storing large amounts of data the next challenge is to *determine what follows from all that data.*

5 Dedication

It is an honour and a pleasure to offer this note in celebration of Dov Gabbay's 80th birthday. Dov's remarkable *approach* to logic has been a great inspiration to us since our days as graduate students. It is an inspiration that continues to this day.

References

[1] K. Aizawa. *Compositional Abduction and Scientific Interpretation : A Granular Approach.* Cambridge University Press, Cambridge, 2025.

[2] K. Aizawa. Disconfirmation is not Modus Tollens. *The Reasoner*, 19(2):26–32, 2025.

[3] K. Atkinson, P. Baroni, M. Giacomin, A. Hunter, H. Prakken, C. Reed, G. Simari, M. Thimm, and S. Villata. Toward Artificial Argumentation. *AI Magazine*, 38(3):25–36, 2017.

[4] P. Baldi, E. A. Corsi, and H. Hosni. A logical framework for data-driven reasoning. *Logic Journal of the IGLP*, 33, 2025.

[5] P. Baroni, D. Gabbay, M. Giacomin, and L. van der Torre, editors. *Handbook of Formal Argumentation.* College Publications, Milton Keynes, 2018.

[6] K. Batko and A. Ślęzak. The use of Big Data Analytics in healthcare. *Journal of Big Data*, 9(1), 2022.

[7] Y. Benjamini, R. D. De Veaux, B. Efron, S. Evans, M. Glickman, B. I. Graubard, X. L. Meng, N. Reid, S. M. Stigler, S. B. Vardeman, C. K. Wikle, T. Wright, L. J. Young, and K. Kafadar. The ASA president's task force statement on statistical significance and replicability. *Annals of Applied Statistics*, 15(3):1084–1085, 2021.

[8] F. L. Bookstein. *Measuring and Reasoning.* Cambridge University Press, Cambridge, 2014.

[9] G. Boole. *An investigation of the laws of thought, on which are founded the mathematical theories of logic and probabilities.* Walton & Maberly, London, 1854.

[10] R. Bosley. Modus Tollens. *Notre Dame Journal of Formal Logic*, (1):103–111, 1979.

[11] T. Chanda et al. Dermatologist-like explainable AI enhances melanoma diagnosis accuracy: eye-tracking study. *Nature Communications*, 16(1), 2025.

[12] E. Cheng. The Art of Logic How to Make Sense in a World that Doesn't, 2018.

[13] A. Clayton. *Bernoulli's Fallacy. Statistical Illogic and the Crisis of Modern Science.* Columbia University Press, New York, 2021.

[14] A.-A. Cournot. *Exposition de la théorie des chances et des probabilités.* Hachette, Paris, 1843.

[15] D. Mumford. The Dawning of the Age of Stochasticity. In V. Arnold, M. Atiyha, P. Lax, and P. Mazur, editors, *Mathematics: Frontiers and Perspectives*, pages 197–218. International Mathematical Union, USA, 2000.

[16] F. A. D'Asaro, F. A. Genco, and G. Primiero. Checking trustworthiness of probabilistic computations in a typed natural deduction system. *Journal of Logic and Computation*, 2025.

[17] M. Dickson and D. Baird. Significance Testing. In *Handbook of the Philosophy of Science, vol 7: Philosophy of Statistics*, volume 7, pages 199–229. Elsevier, 2011.

[18] D. Dubois and H. Prade. The three semantics of fuzzy sets. *Fuzzy Sets and Systems*, 90(2):141–150, 1997.

[19] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77(2):321–357, 1995.

[20] D. A. Van Dyk. The role of statistics in the discovery of a higgs boson. *Annual Review of Statistics and Its Application*, 1(October 2013):41–59, 2014.

[21] B. Efron. R. A. Fisher in the 21st century: Invited paper presented at the 1996 R. A. Fisher lecture. *Statistical Science*, 13(2):95–122, 1998.

[22] B. Efron and T. J. Hastie. *Computer Age Statistical Inference.* Cambridge University Press, Cambridge, 2016.

[23] R. Falk and C. W. Greenbaum. Significance Tests Die Hard: The Amazing Persistence of a Probabilistic Misconception. *Theory & Psychology*, 5(1):75–98, 1995.

[24] A. Burdman Feferman and S. Feferman. *Alfred Tarski: Life and Logic.* Cambridge University Press, Cambridge, 2004.

[25] F. Ferrari and M. Carrara. *Logic and Science.* Cambridge University Press, Cambridge, 2025.

[26] R. Feynman. *The Character of Physical Law.* MIT Press, Cambridge, 1965.

[27] R. A. Fisher. *Statistical Methods and Statistical Inference.* Oliver & Boyd, Edinburgh, 1956.

[28] T. Flaminio and H. Hosni. Logics for the new AI spring. *International Journal of Approximate Reasoning*, 170, 2024.

[29] D. M. Gabbay. Editorial. *Journal of Logic and Computation*, 1(1):1–4, 1990.

[30] D. M. Gabbay and J. Woods. The practical turn in logic. In D. M. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic, 2nd Edition, Vol 13*, volume 13, pages 15–122. Springer, 2005.

[31] D. M. Gabbay and J. Woods. *The reach of Abduction: Insight and Trial.* Elsevier, Amsterdam, 2005.

[32] D. M. Gabbay and J. Woods, editors. *Handbook of the History of Logic Vol 8. The Many Valued and Non Monotonic Turn in Logic.* North-Holland, Amsterdam, 2007.

[33] D. M. Gabbay and J. Woods. The Law of Evidence and Labelled Deduction: Ten Years Later. In G. Casini, L. Robaldo, L. van der Torre, and S. Villata, editors, *Handbook of Legal AI*, pages 364–434. College Publications, London, 2022.

[34] A. Gandomi and M. Haider. Beyond the hype: Big data concepts, methods, and analytics. *International Journal of Information Management*, 35(2):137–144, 2015.

[35] I. J. Good. A. M. Turing's Statistical Work in World War II. *Biometrika*, 66(2):393–396, 1979.

[36] I. J. Good. *Good thinking: The foundations of probability and its applications.* The University of Minnesota Press, 1983.

[37] R. Haenni, J.-W. Romeijn, J. Williamson, and G. Wheeler. *Probabilistic Logics and Probabilistic Networks.* Springer, Dordrecht, 2011.

[38] T. Hailperin. *Sentential Probability Logic. Origins, Development, Current Status, and Technical Applications.* Lehigh University Press, Bethlehem, 1996.

[39] C. Hempel. *Philosophy of Natural Science.* Prentice Hall, Upper Saddle River, 1966.

[40] T. Hey, K. Butler, S. Jackson, and J. Thiyagalingam. Machine learning and big scientific data. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 378(2166), 2020.

[41] W. Hodges. DLMPS – Tarski's vision and ours. In B. Loewe and D. Sarikaya, editors, *60 Jahre DVMLG*, pages 103–119. 2022.

[42] S. G. Hofmann. Fisher's fallacy and NHST's flawed logic. *The American psychologist*, 57(1):69–70, 2002.

[43] H. Hosni and J. Landes. Logical perspectives on the foundations of probability. *Open Mathematics*, 21(1), 2023.

[44] H. Hosni and J. Landes, editors. *Perspectives on Logics for Data-Driven Reasoning.* Springer, Cham, 2025.

[45] C. Howson. *Hume's Problem: Induction and the Justification of Belief.* Oxford University Press, Oxford, 2003.

[46] R. Hubbard and M. J. Bayarri. Confusion Over Measures of Evidence (p's) Versus Errors (α's) in Classical Statistical Testing. *American Statistician*, 57(3):171–178, 2003.

[47] A. Hunter, S. Polberg, N. Potyka, T. Rienstra, and M. Thimm. Probabilistic Argumentation: A Survey. In D. Gabbay, M. Giacomin, G. R. Simari, and M. Thimm, editors, *Handbook of Formal Argumentation*, volume 2, chapter 7, pages 397–444. College Publications, London, 2021.

[48] E. T. Jaynes. *Probability Theory: The Logic of Science.* Cambridge University Press, Cambridge, 2003.

[49] H. Jeffreys. *Scientific inference.* Cambridge University Press, Cambridge, 1931.

[50] J. Krueger. Null hypothesis significance testing: On the survival of a flawed method. *American Psychologist*, 56(1):16–26, 2001.

[51] A. Kucharski. *Proof: The Uncertain Science of Certainty.* Profile Books, London, 2025.

[52] J. Landes, E. A. Corsi, and P. Baldi. Knowledge Representation, Scientific Argumentation and Non-monotonic Logic. In H. Hosni and J. Landes, editors, *Perspectives on Logics for Data-Driven Reasoning*, pages 155–179. Springer, 2025.

[53] P.-S. Laplace. *A Philosophical Essay on Probabilities.* Dover Publications, 1825.

[54] B. Lecoutre and J. Poitevineau. *The Significance Test Controversy.* Springer, 2nd edition, 2022.

[55] S. Leonelli and N. Tempini, editors. *Data Journeys in the Sciences.* Springer, Cham, 2020.

[56] L. Magnani, editor. *Handbook of Abductive Cognition.* Springer, Cham, 2023.

[57] D. Makinson. *Bridges from classical to nonmonotonic logic.* College Publications, London, 2005.

[58] P. Marquis, O. Papini, and H. Prade. *A Guided Tour of Artificial Intelligence Research 1: Knowledge Representation, Reasoning and Learning.* Springer, Cham, 2020.

[59] D. G. Mayo. *Statistical Inference as Severe Testing: How to Get Beyond the Statistics Wars.* Cambridge University Press, 2018.

[60] H. Mercier and D. Sperber. Why do humans reason? Arguments for an argumentative theory. *Behavioral and Brain Sciences*, 34(02):57–74, 2011.

[61] L. Messeri and M. J. Crockett. Artificial intelligence and illusions of understanding in scientific research. *Nature*, 627(8002):49–58, 2024.

[62] Z. Ognjanovic, M. Raskovic, and Z. Markovic. *Probability Logics - Probability based formalization of uncertainty reasoning.* Springer, 2016.

[63] P. T. Gowers. The Two Cultures of Mathematics. In V. Arnold, M. Atiyha, P. Lax, and P. Mazur, editors, *Mathematics: Frontiers and Perspectives*, pages 65–78. International Mathematical Union, USA, 2000.

[64] J. B. Paris. *The uncertain reasoner's companion: A mathematical perspective.* Cambridge University Press, Cambrdige, 1994.

[65] J. B. Paris and A. Vencovská. *Pure Inductive Logic.* Cambridge University Press, Cambridge, 2015.

[66] A. G. Patriota. Is NHST logically flawed? Commentary on: "NHST is still logically flawed". *Scientometrics*, 116(3):2189–2191, 2018.

[67] Y. Pawitan. Defending the P-value. *arXiv, 2009.02099*, 2022.

[68] J. Pearl. Radical empiricism and machine learning research. *Journal of Causal Inference*, 9(1):78–81, 2021.

[69] P. Pollard and J. T. Richardson. On the Probability of Making Type I Errors. *Psychological Bulletin*, 102(1):159–163, 1987.

[70] G. Pólya. *Mathematics and Plausible Reasoning: Patterns of plausible inference.* Princeton University Press, Princeton, 1954.

[71] S. Rodriguez, C. Hug, P. Todorov, N. Moret, S. A. Boswell, K. Evans, G. Zhou, N. T. Johnson, B. T. Hyman, P. K. Sorger, M. W. Albers, and A. Sokolov. Machine learning identifies candidates for drug repurposing in Alzheimer's disease. *Nature Communications*, 12(1), 2021.

[72] R. Royall. *Statistical Evidence: A likelihood paradigm.* Chapman and Hall, London, 1997.

[73] G. Shafer. Testing by betting: A strategy for statistical and scientific communication. *Journal of the Royal Statistical Society. Series A: Statistics in Society*, 184(2):407–431, 2021.

[74] M. Singham. *The Great Paradox of Science.* Oxford University Press, Oxford, 2020.

[75] L. Sonenberg. Logics and collaboration. *Logic Journal of the IGPL*, 31:1024–1046, 2023.

[76] D. Szucs and J. P. A. Ioannidis. When null hypothesis significance testing is unsuitable for research: A reassessment. *Frontiers in Human Neuroscience*, 11, 2017.

[77] A. Tarski. *Introduction to Logic and the Methodology of Deductive Sciences (1941).* Oxford University Press, Oxford, fourth edition, 1994.

[78] A. M. Turing. Lecture to the London Mathematical Society on 20 February 1947. In *Mechanical Intelligence, vol 1*, volume 1. North Holland, 1992.

[79] C. G. Wagner. Modus Tollens Probabilized. *British Journal for Philosophy of Science*, 55:747–753, 2004.

[80] J. Williamson. *In Defence of Objective Bayesianism.* Oxford University Press, Oxford, 2010.

[81] J. M. Wing. Trustworthy AI. *Communications of the ACM*, 64(10):64–71, 2021.

[82] S. Ziliak and D. McCloskey. *The Cult of Statistical Significance.* University of Michigan Press, 2008.

Received August 2025

We Love Inconsistency

Anthony Hunter
University College London,Gower Street, London, UK
anthony.hunter@ucl.ac.uk

Abstract

Inconsistency is an important phenomenon for agents operating in the real-world. And it is now recognized as a key issue in many areas of artificial intelligence, and more broadly in computer science. Here I revisit a proposal by Dov Gabbay for switching from thinking about inconsistency as being necessarily bad, to something that may be desirable. This led to the idea that inconsistency is not something that we have to eliminate, but rather it is something that we need to act upon. Actions may range from isolating an inconsistency, seeking more clarification on the information involved in an inconsistency, through to exploiting an inconsistency. As an example of the latter, consider a researcher finding inconsistencies in the literature, which can then be exploited by constructing valuable new research questions.

1 Introduction

Inconsistency happens all around us on a constant basis. Think of any real-world scenario, and of the inconsistencies that can arise. For example, consider you're leaving the house, and you can't find your keys where you normally leave them. So, there's an inconsistency between what you remember and what is true. After finding the keys, suppose you look at a travel app and see that the trains are running on time, only to find that when you arrive at the station, there is a delay. Hence, there is an inconsistency between the travel app and what is happening at the station. Anyway, you go to the ticket office to get a ticket. You think the ticket is one price but the ticket office gives you a price that is much higher. Again, there is an inconsistency. This time it is because you hadn't noticed the restricted travel times on the cheaper tickets. The first inconsistency you need to resolve since you need your keys but there is little you can for the other two inconsistencies, since the delay is occurring and ticket office will charge you the higher price, but perhaps you can learn from these inconsistencies.

Often when inconsistencies arise, you have a choice on whether to resolve them or not, and you have a choice on how and when to resolve them. However, some situations need a resolution. For example, if you are building a new house, and you want a house with two stories, and your partner wants a bungalow, then this is an inconsistency that needs to be resolved before the architect and builder can do their work.

Whilst many inconsistencies can be problematical, there are many situations where they are indicators of something useful. As in the adage, "*Man Bites Dog*" is interesting news; "*Dog Bites Man*" is not. However, most information published is of the "*Dog Bites Man*" variety, particularly in the case of business, technical, and scientific news. This information continues existing trends or confirms expected information. Nonetheless, unexpected information is often more useful (or at least, it is more interesting) to the reader. We look for interesting news based on the expectations we have about the world. If a news report violates an expectation, this indicates the news is interesting.

As further examples of useful inconsistencies, a lawyer in court is looking for inconsistencies in the opposition case, where exposing such inconsistencies can considerably weaken the opposition case, and in brainstorming sessions, researchers can find exciting new research questions by looking at inconsistencies in the literature.

2 Making inconsistency respectable

In 1991, Dov Gabbay suggested that we wrote a position paper on how we need to revisit how we regard inconsistency [2]. He had identified that there is a fundamental difference between the way humans handle inconsistency, and the way it was currently handled in formal logical systems. To a human, resolving inconsistency is *not necessarily done* by "restoring" consistency but by supplying rules telling one how to act when the inconsistency arises. Then, the position that inconsistency is a "bad" thing can shift to it often being a "good" thing. Inconsistencies can be read as signals to take external action, like find more information, or internal action, like treat the inconsistency with caution.

The first paper laid out the motivation for Dov's idea using examples based on a *Professor Nobody* and *Professor Somebody* using Dov's wonderful combination of humour and very effective explanation of ideas. We followed this up with an article on a meta-reasoning system for acting on inconsistency [3].

To explore this proposal, we collaborated with Anthony Finkelstein, Jeff Kramer, and Bashar Nuseibeh, on applying the approach to software requirements specifications [1, 4]. The development of large and complex systems can involve many people, each with their own perspectives on the system defined by their knowledge, responsibilities, and commitments. In this context, maintaining absolute consistency is not always possible, nor is it even desirable since this can unnecessarily constrain the development process, and can lead to the loss of important information. To deal with this, we combined the ViewPoints framework for perspective development, interaction and organisation, and a logic-based approach to inconsistency handling based on rules.

Some empirical evaluation was reported by Bashar Nuseibeh, Alessandra Russo, Jeff Kramer, and Steve Easterbrook, in [6, 5] on studies regarding requirements for satellite software. Here, various kinds of inconsistency were identified and investigated. Various lessons were drawn including domain-independent and domain-dependent rules can be identified that provide explicit information which is often missing in specifications and which is needed for identifying inconsistencies. How this extra information is needed for consistent change management was also explored. For instance, they identified inconsistencies that the satellite engineers are aware of, but for practical reasons are ignored. This was because it can be more cost-effective to not fix some inconsistencies, but just ensure that the risks involved are managed.

3 Conclusions

When writing the paper on "*Making inconsistency respectable*", Dov wanted to call the paper "*We love inconsistency*". I thought this sounded a little too much, and so we agreed on the published title. Since then, most of my research has considered aspects of inconsistency, in particular, measures of inconsistency, and argumentation. Furthermore, I see that inconsistency is increasingly important in many areas of AI (e.g. merging information from heterogeneous sources, negotiation in multi-agent systems, understanding natural language dialogues, and commonsense reasoning in robotics). Moreover, through many interactions with colleagues around the world, I have found studying, and discussing aspects of inconsistency, has been fascinating. Hence, at least as researchers, we may want to say that we do indeed love inconsistency.

References

[1] Anthony Finkelstein, Dov M. Gabbay, Anthony Hunter, Jeff Kramer, and Bashar Nuseibeh. Inconsistency handling in multperspective specifications. *IEEE Transactions on Software Engineering*, 20(8):569–578, 1994.

[2] Dov M. Gabbay and Anthony Hunter. Making inconsistency respectable: a logical framework for inconsistency in reasoning. In Philippe Jorrand and Jozef Kelemen, editors, *Proceedings of the International Workshop on Fundamentals of Artificial Intelligence Research (FAIR'91)*, volume 535 of *Lecture Notes in Computer Science*, pages 19–32. Springer, 1991.

[3] Dov M. Gabbay and Anthony Hunter. Making inconsistency respectable: Part 2—meta-level handling of inconsistency. In *Proceedings of the European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty (ECSQARU'93)*, volume 747 of *Lecture Notes in Computer Science*, pages 129–136. Springer, 1993.

[4] Anthony Hunter and Bashar Nuseibeh. Managing inconsistent specifications: Reasoning, analysis, and action. *ACM Transactions on Software Engineering Methodology*, 7(4):335–367, 1998.

[5] Bashar Nuseibeh, Steve M. Easterbrook, and Alessandra Russo. Making inconsistency respectable in software development. *Journal of Systems and Software*, 58(2):171–180, 2001.

[6] Alessandra Russo, Bashar Nuseibeh, and Jeff Kramer. Restructuring requirements specifications for managing inconsistency and change: A case study. In *Proceedings of the International Conference on Requirements Engineering (ICRE'98)*, page 51. IEEE Computer Society, 1998.

Received July 2025

Cautious Nonmonotonicity

Timotheus Kampik
Department of Computing Science, Umeå University,
Umeå, Sweden
tkampik@cs.umu.se

Abstract

A key intuition in symbolic artificial intelligence is that an intelligent system should be *non-monotonic*, but *cautiously* so: previous conclusions should only be revised if a compelling reason for doing so exists. In this paper, I trace the evolution of this intuition, which emerged from Dov Gabbay's seminal 1985 paper and gained additional prominence as *cautious monotonicity* in the 1990 KLM paper, as well as in an earlier paper by Makinson. I introduce the term *cautious* ***non****monotonicity* for the general idea of assuring that monotonicity is satisfied *given some condition*, thus highlighting that it is the *violation*, and not the satisfaction, of monotonicity that we need to be careful about. Also, I discuss why cautious nonmonotonicity still is an open problem in theory and practice, and present some results that highlight the intricacy of cautious nonmonotonicity in the simple case of abstract argumentation, where inferences are drawn from directed graphs without further structure.

1 Introduction

Intelligent agents must be able to reason *nonmonotonically* in that they may first draw a conclusion to later revise it, given new evidence. Even humans may systematically fail to do so, for example when experiencing *sunk cost fallacies*, i.e., by staying committed to plans that can no longer be executed successfully, given newly obtained knowledge, typically about changes in the environment. More formally, we may model nonmonotonic inference assuming a Knowledge Base (KB) K—sets of (here not further specified) formulas—and a conclusion A (again, some set of formulas) inferred by an agent from K, with inference denoted by $K \models A$. Our agent

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. I thank Vicenç Torra for commenting on a draft of this paper.

must, at least under some circumstances, be allowed to violate $K \cup K' \models A$, given some KB K'. This stands in contrast to classical logic, where a statement when inferred as either true or false must remain so, even in face of new observations.

It is easy to be nonmonotonic: intuitively, one merely needs to be entirely unhinged—uttering random incoherent garbage will pretty much guarantee nonmonotonicity. Instead, what one wants to achieve is the design of a somewhat "intelligent" agent that can proactively justify why its inference violates monotonicity, explaining that the violation is based on sound reasoning principles. This requires principles that stipulate under which conditions monotonicity may or should[1] be violated. One such principle is *cautious monotonicity*, first introduced by Gabbay as *restricted monotonicity* [5] (whereas the latter term was coined by Makinson [9] and is also used in the seminal KLM paper [8]): if we can infer conclusion A from KB K and B from K, then we can infer B from $K \cup A$.

Generalising the idea underlying cautious monotonicity, in this paper I introduce *cautious* ***non****monotonicity*, a meta-principle stipulating that monotonicity may be violated under some condition, which then needs to be specified by instantiations of the principle (Section 2). Based on this I characterise existing reasoning principles as cautious nonmonotonicity[2] in theory (Section 3 for argumentation, with some basic observations about logics already in Section 2) and practice (Section 4). Finally, I conclude the paper by discussing potential future research that may shed more light on how to achieve principled cautious nonmonotonicity (Section 5).

As I have written this paper for the occasion of Dov Gabbay's 80th birthday, I have added some personal anecdotes about Dov and how he has helped me during my PhD and, in recent years, prior to obtaining a tenure-track position. These small stories are intended to highlight Dov's kindness, wisdom, and humour. Obviously, my recollection may be subjective and inaccurate. For the sake of readability, I have used textboxes to visually separate the anecdotes from the rest of the paper[3].

[1] Here, we focus on the *may* case.

[2] For this, I partially rely on some results obtained as a side-line of earlier studies [7].

[3] As this paper is heavy on anecdotes, I sometimes use first-person singular "I", although I still use the more usual first-person plural "we" when it feels more natural.

2 Introducing Cautious Nonmonotonicity

Meeting Dov. I first met Dov when visiting Leon van der Torre's group in Luxembourg. As a PhD student from the northern European periphery who was—academically speaking—semi-literate at best, I was glad that Leon was so kind to receive me. Of course, I had heard of Dov, and knew he regularly spent time with Leon's group. After some days, I got the chance to meet him. Although I had essentially nothing to show for besides some ideas and crude sketches, Dov strongly encouraged me to continue studying the apparent conflict between the semantics of abstract argumentation and economic rationality, which eventually became the cornerstone of my PhD. Beyond that, Dov took it on himself to mentor me and we eventually started collaborating, which initially must have been rather frustrating, given my lack of knowledge. However, I do think that by now, we have obtained some reasonably interesting results, for example on modelling burdens of persuasion in abstract argumentation (jointly with Giovanni Sartor).

Let me now introduce my central notion of the meta-principle of cautious nonmonotonicity. Again, assume sets of not further specified formulas K, K', A, and B, which we may call *KBs* when drawing inferences from them and *conclusions* when inferring them from KBs. Consider monotonicity and cautious monotonicity:

Monotonicity. If $K \models A$ then $K \cup K' \models A$.

Cautious Monotonicity. If $K \models A$ and $K \models B$ then $K \cup A \models B$.

Clearly, monotonicity is stricter than cautious monotonicity: assume $K \models B$; for monotonicity, $K \cup K' \models B$ must hold for *any* K', whereas for cautious monotonicity, $K \cup K' \models B$ must hold for the special case that $K \models K'$: cautious monotonicity can, intuitively, be re-phrased as "monotonicity must be satisfied if K' (from the definition of monotonicity) can be inferred from K". Generalising this, we define our *cautious nonmonotonicity* meta-principle: we *must* satisfy monotonicity *under some condition* (and thus *may* violate monotonicity only if this condition is not met).

Cautious Nonmonotonicity. If $K \models A$ then $K \cup K' \models A$ if $f_{\models}(K, K', A)$ holds, where $f_{\models} : \mathcal{K} \times \mathcal{K} \times \mathcal{K} \rightarrow \{true, false\}$ and $\mathcal{K}$ serves as the background set of sets of formulas. We may call $f_{\models}$ the (function that defines the) *monotonicity condition.*

Using cautious *non*monotonicity, we can characterise cautious monotonicity for the case that $f_{\models}$ is defined as follows:

$$f_{\models}(K, K', A) := \begin{cases} true, & \text{if } K \models K'; \\ false, & \text{otherwise.} \end{cases}$$

Clearly, we can also characterise *monotonicity* as cautious nonmonotonicity by stipulating that $f_{\models}(K, K', A) := true$; intuitively, the fewer cases we have for which $f_{\models}$ yields $true$, the more relaxed is our notion of cautious nonmonotonicity, until we reach what we may call *unhinged monotonicity* with $f_{\models}(K, K', A) := false$. Note that we provide A as a function argument to $f_{\models}$, although it is not needed above, because one may reasonably want to make the violation of monotonicity depend on it. For example, one may define $f_{\models}$ as follows:

$$f_{\models}(K, K', A) := \begin{cases} true, & \text{if } K' \models A; \\ false, & \text{otherwise.} \end{cases}$$

Reasonable inference systems that violate this principle can be assumed to exist; for example, $K \cup K'$ may be inconsistent (but consistent when separated) and $\models$ may not allow us to infer anything given an inconsistent KB.

Figuring out when to violate monotonicity is tricky. As Dov observed already in the 90s, cautious monotonicity may sometimes be useful, but at other times too strong, or not strong enough [6]. For our cautious nonmonotonicity meta-principle, this means it is unclear how to define $f_{\models}$. It is reasonable to assume that a good definition of $f_{\models}$ depends on context, and a key question is how semantics-, application-, and domain-specific this context is. I explore the conceptual intricacy of the monotonicity condition in the following two sections—first using a fundamentally different formalisation for some theoretical investigations, and then discussing applied perspectives. Obviously, I will not be able to provide a definitive answer.

3 Cautious Nonmonotonicity in Argumentation

The Garbage Man-Logician Hypothesis. After completing my PhD, I spent the time waiting for a suitable tenure-track position working in industry, preferring the stability the job offered to my family, as well as the ability to conduct well-funded applied research, over a precarious post-doc position. Dov encouraged me to move back to the academy rather sooner than later. Still, he exhibited a keen understanding of the realities of academic life and their sometimes detrimental effects on the ability to carry out research. Thought-provokingly, he told me that a decent professional occupation for a logician was *garbage man*: as such, one can spend most of one's time outdoors engaging in moderate physical activity, while thinking about problems in logic. By the time one is home from work, one merely needs to write down the solutions to the problems one has solved during the day. To the best of my knowledge, there are, as of now, no hybrid curricula and structured career paths for garbage man logicians. Given the increasing burden of academic bureaucracy, exchanging endless paperwork and pseudo-performative meetings with some days of honest garbage collection work would certainly make our jobs more meaningful.

We now apply our conceptual idea about cautious nonmonotonicity to formal argumentation, and specifically to Dung's *abstract argumentation* [4]. In abstract argumentation, inferences are drawn from directed graphs, called *Argumentation Frameworks* (AFs). The nodes of the graphs are called *arguments* and the graph's binary relation models conflicts between arguments, called *attacks*. *Argumentation semantics* infer sets of arguments, called *extensions*, from AFs. Formally, let $\mathcal{F}$ denote the class of AFs and $\mathcal{A}$ the class of arguments; an (argumentation) semantics is a function $\sigma : \mathcal{F} \to 2^{2^{\mathcal{A}}}$ that given an argumentation framework $F = (A, R) \in \mathcal{F}$ infers a set of extensions $ES \subseteq 2^{A}$.

For better or for worse (this is much debated), arguments in abstract argumentation have no further structure. One argument for the "better" case is that studying abstract argumentation may allow us to obtain fundamental insights into reasoning in face of conflicts, forcing the focus on aspects rich enough to be interesting even given directed graphs and nothing more. Conceptually, in nonmonotonic reasoning one would expect implicit or explicit conflicts between what we have initially inferred and new information we have obtained: given $K \models A$, if $K \cup K' \models A$ is violated, we would expect that something in K', or perhaps only in $K' \setminus K$, causes a conflict. Thus, using abstract argumentation as the formal machinery allows for a focused study of key aspects of nonmonotonic inference. Note that the bridge between abstract argumentation and nonmonotonic reasoning is, of course, not new; indeed, it stems back to Dung's seminal paper that, for example, makes the connection to Reiter's default logic[4].

In abstract argumentation, the notions of *attack*, *conflict*, and *defence* play an important role. Here and henceforth, we assume an AF $F = (A, R)$ and sets $S, S' \subseteq A$, as well as arguments $\mathsf{x}, \mathsf{y} \in A$ if not specified otherwise. We indicate which AF we are referring to only if the context may not be clear.

Attack. x attacks y iff $(\mathsf{x}, \mathsf{y}) \in R$; S attacks S' iff for some $\mathsf{x} \in S$, $\mathsf{y} \in S'$ it holds that x attacks y. Also, $S^+ := \{\mathsf{x} | S \text{ attacks } \{\mathsf{x}\}\}$ and $S^- := \{\mathsf{x} | \{\mathsf{x}\} \text{ attacks } S\}$. We may refer to S^- as the *attackers* of S and use the same term when it is a single argument that is attacked.

Conflict. S is conflicted iff S attacks S; otherwise, S is conflict-free.

Defence. S defends x iff it attacks all $\{\mathsf{y}\}$ for which it holds that y attacks x.

Figure 1 provides an example of an AF that highlights differences between argumentation semantics. Given the AF, we may: *(i)* observe that no non-empty set of arguments can defend itself and infer $\emptyset$ (or even *nothing*); *(ii)* discard the odd cycle as an indirect self-attack and infer $\{\mathsf{d}\}$; *(iii)* consider a, b, and c as equally plausible and infer either of the maximal conflict-free sets $\{\mathsf{a}, \mathsf{d}\}$, $\{\mathsf{b}\}$, or $\{\mathsf{c}, \mathsf{d}\}$. Semantics following the first intuition are somewhat more *sceptical*, whereas semantics following the last intuition are quite *credulous*; semantics following the second intuition may be somewhere in between, but are more intricate to define. To avoid notational overhead, we give example semantics for the other two intuitions only. The former is based on the notion of an *admissible* set.

Admissible sets. S is admissible iff it is conflict-free and for every $\mathsf{x} \in S$ it holds that S defends x.

We can now introduce the two semantics: the relatively sceptical *preferred* semantics (introduced by Dung) and the more credulous *stage* semantics (introduced later [12]).

Preferred semantics. $\sigma_p(F) := \{S | S \text{ is admissible}, \nexists S' \subseteq A \text{ s.t. } S \text{ is admissible and } S \subset S'\}$. $E \in \sigma_p(F)$ is a preferred or σ_p-extension of F.

Stage semantics. $\sigma_s(F) := \{S | S \text{ is conflict-free and } \nexists S' \subseteq A \text{ s.t. } S' \text{ is conflict-free and } S \cup S^+ \subset S' \cup S'^+\}$. $E \in \sigma_s(F)$ is a stage or σ_s-extension of F.

[4] Also, Dov has made important contributions to the field of formal argumentation, both in terms of original research and as the editor of several handbooks on the topic. Thus, argumentation is a good fit for this small contribution to his Festschrift.

In the AF of Figure 1, the only preferred extension is $\emptyset$, and the stage extensions are $\{\mathsf{a},\mathsf{d}\}$, $\{\mathsf{b}\}$, and $\{\mathsf{c},\mathsf{d}\}$—as expected given the intuitions provided above.

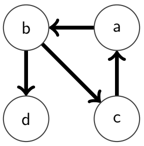
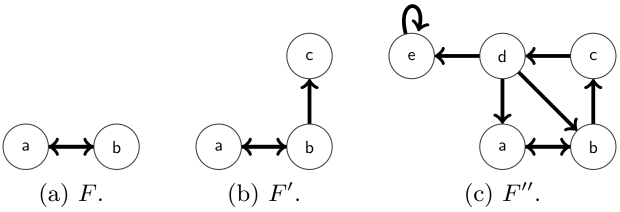


Figure 1: This AF helps highlight different ideas behind argumentation semantics.

Let us now explore what nonmonotonic inference is in abstract argumentation. As a key assumption, we consider a dynamic reasoning process, during which we consecutively *normally expand* AFs by adding new arguments and attacks to them, though without adding attacks between existing arguments. More formally, an AF $F' = (A', R')$ normally expands an AF $F = (A, R)$, denoted by $F \prec_N F'$, iff $A \subset A'$, $R \subseteq R'$ and $\forall(\mathsf{x},\mathsf{y}) \in R' \setminus R$ it holds that $\mathsf{x} \in A' \setminus A$ or $\mathsf{y} \in A' \setminus A$ [1].

As we normally expand an AF, we draw inferences from it and its expansions by computing and selecting extensions. Consider the AFs F, F', and F'' in Figure 2. Observe that $F \prec_N F'$ and $F' \prec_N F''$. Table 1 shows the extensions of the AFs. In each row, we can see extensions of a given semantics such that each extension of F is contained by an extension of F', analogously for F' and F''. ✗ indicates that there is no extension that can contain the previous one. In such case, we say that monotonicity is violated, as we can no longer infer what we have inferred before. Intuitively, we assume that an agent draws inferences by determining the extensions of the first AF and selecting one of them (e.g., by random choice). When the AF is normally expanded, the agent again determines the extensions and, if possible, keeps its inference monotonic by selecting an extension of the expansion that is a superset of or equal to the previously inferred extension.



(a) F. (b) F'. (c) F''.

Figure 2: Nonmonotonic inference in abstract argumentation.

σ	F	F'	F''
σ_p	{a} {b}	{a, c} {b}	{a, c} ✗
σ_s	{a} {b}	{a, c} {b}	{a, c} ✗ {d}

Table 1: σ_p- and σ_s-extensions of the argumentation frameworks in Figure 1.

Given this intuition, we can define monotonicity as an *argumentation principle*[5].

Here and henceforth, we assume two AFs $F = (A, R)$ and $F' = (A', R')$ s.t. $F \prec_N F'$; a principle is satisfied iff the corresponding statement holds given a semantics σ and all such AFs.

Monotonicity (argumentation).
$\forall E \in \sigma(F)$ it holds that $\exists E' \in \sigma(F')$ s.t. $E \subseteq E'$.

Next, we define the meta-principle of cautious nonmonotonicity for (abstract) argumentation. Here, the condition for violating monotonicity is specific to an extension.

Cautious Nonmonotonicity (argumentation). $\forall E \in \sigma(F)$ it holds that $\exists E' \in \sigma(F')$ s.t. $E \subseteq E'$ if $f_\sigma(E, F, F')$ holds, where $f_\sigma : \mathcal{A} \times \mathcal{F} \times \mathcal{F} \rightarrow \{true, false\}$.

We call f_σ the *monotonicity condition.* By stipulating that $f_\sigma(E, F, F') := true$, we can instantiate monotonicity from the cautious nonmonotonicity meta-principle.

Cautious monotonicity is an argumentation principle that somewhat reflects the idea of admissibility: if we can no longer infer an extension, this must be due to a newly occurring attack to just that extension[6].

Cautious Monotonicity (argumentation). $\forall E \in \sigma(F)$ it holds that
$\exists E' \in \sigma(F')$ s.t. $E \subseteq E'$ if $A' \setminus A$ does not attack E (in F').

Indeed, we can observe that preferred semantics satisfies cautious monotonicity: as a preferred extension is a maximal admissible set, if such extension is no longer included in any "new" extension after normal expansion, a new attack to the former extension must have been added—otherwise, we would still have the extension as an admissible set, and hence as a subset of a maximal admissible set.

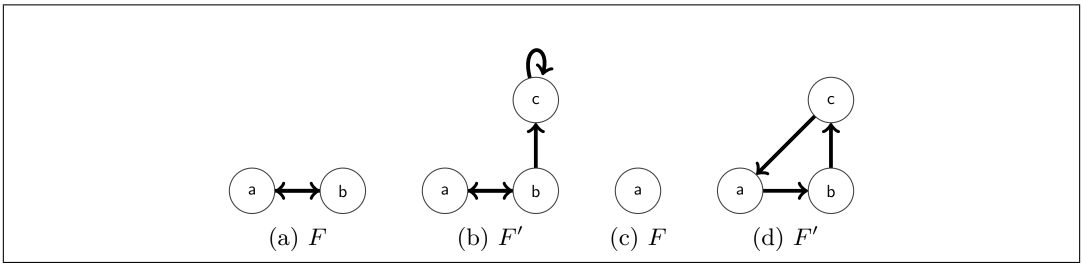
[5] For an overview of substantial studies of argumentation principles, see [11].

[6] As arguments have no structure (and we infer sets of sets of nodes from graphs), there is no clear formal connection between cautious monotonicity in abstract argumentation and Dov's cautious monotonicity. However, there may (speculative) be a clearer connection to a notion of cautious monotonicity as defined by Çyras and Toni for a structured argumentation variant [2].

We can characterise cautious monotonicity as an instantiation of cautious nonmonotonicity by writing the former principle's monotonicity condition as follows:

$$f_\sigma(E, F, F') := \begin{cases} true, & \text{if } A' \setminus A \text{ does not attack } E \text{ (in } F'\text{);} \\ false, & \text{otherwise.} \end{cases}$$

In contrast, stage semantics violates cautious monotonicity. A counter-example is provided in Figure 3 (F and F' in Sub-figures 3a and 3b, respectively), where the expansion of the initial AF with a self-attacking argument c, attacked by b, means that we can no longer infer {a} (or a superset thereof).



(a) F (b) F' (c) F (d) F'

Figure 3: Counter-examples: stage semantics violates cautious monotonicity (3a, 3b) and preferred semantics violates extension attack nonmonotonicity (3c, 3d).

Can we define a cautious nonmonotonicity instantiation that somewhat reflects the idea of maximal conflict-free set-based semantics?

Intuitively, maximal conflict-freeness itself can satisfy monotonicity: if we have a maximal conflict-free set S in an AF F and normally expand to F', S remains conflict-free in F', i.e., it must be contained by some maximal conflict-free set in F'.

Still, in *reasonable* maximal conflict-free semantics, we may speculate that the culprit of nonmonotonicity is an attack by an extension of the expanded AF, which we formalise as *extension attack nonmonotonicity.*

Extension Attack Nonmonotonicity. $\forall E \in \sigma(F)$ it holds that $\exists E' \in \sigma(F')$ s.t. $E \subseteq E'$ if $\nexists E'' \in \sigma(F')$ s.t. E'' attacks E (in F').

Extension attack nonmonotonicity can be instantiated from cautious nonmonotonicity by setting the monotonicity condition as follows:

$$f_\sigma(E, F, F') := \begin{cases} true, & \text{if } \nexists E'' \in \sigma(F') \text{ s.t. } E'' \text{ attacks } E \text{ (in } F'\text{);} \\ false, & \text{otherwise.} \end{cases}$$

Stage semantics satisfies extension attack nonmonotonicity. To demonstrate this, consider any $E \in \sigma_s(F)$. Assume that $\nexists E' \in \sigma_s(F')$ s.t. $E \subseteq E'$. This means that there exists $E'' \subseteq A'$ s.t. E'' is conflict-free in F' and $E \cup E^+ \subset E'' \cup E''^+$ but $E \not\subseteq E''$ (otherwise, E would still be contained by a stage extension). From $E \cup E^+ \subset E'' \cup E''^+$ and $E \not\subseteq E''$ it follows that $E \cap E''^+ \neq \emptyset$, which means that E'' attacks E.

Preferred semantics violates extension attack nonmonotonicity. Consider the (F, F')-pair provided by Sub-figures (c, d) in Figure 3. The only preferred extension of F is $E = \{\mathsf{a}\}$, whereas the only preferred extension of F' is $E' = \emptyset$. Clearly, $E \not\subseteq E'$ and E' does not attack E.

The reader may have noticed that so far, we have only been concerned with what we infer, and not with what we do *not*. Intuitively, we may consider not inferring previously rejected arguments without good reasons just as important as not rejecting arguments that have already been inferred. To study the former through a principle-based lens, we introduce the notion of *complement monotonicity*.

For this, we define the complement of a set of arguments S given an AF (A, R) as follows: $S^{C,(A,R)} := A \setminus S$. We drop the superscript (A, R) where the context is clear: for $E \in \sigma(F)$, E^C implicitly means $E^{C,F}$; for $E' \in \sigma(F')$, E'^C means $E'^{C,F'}$.

Complement Monotonicity
$\forall S \in \{E^C | E \in \sigma(F)\}$ it holds that $\exists S' \in \{E'^C | E' \in \sigma(F')\}$ s.t. $S \subseteq S'$.

We can now define *cautious complement nonmonotonicity* as a meta-principle, analogous to cautious nonmonotonicity, stipulating that complement monotonicity may only be violated if a specific condition is met.

Cautious Complement Nonmonotonicity $\forall S \in \{E^C | E \in \sigma(F)\}$ it holds that $\exists S' \in \{E'^C | E' \in \sigma(F')\}$ s.t. $S \subseteq S'$ if $f_\sigma(E, F, F')$ holds,
where $f_\sigma : \mathcal{A} \times \mathcal{F} \times \mathcal{F} \rightarrow \{true, false\}$.

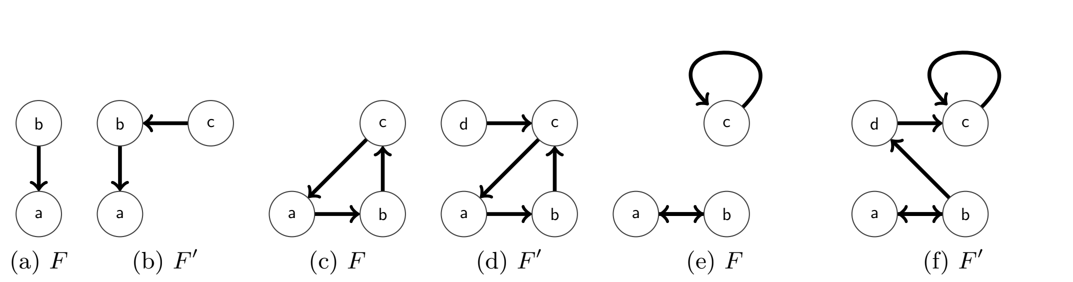
Again, we may call f_σ the *monotonicity condition*. Analogously to monotonicity, if we stipulate that $f_\sigma(E, F, F') := true$, we obtain complement monotonicity. Let us claim that we can trivially characterise the principles that follow as cautious complement nonmonotonicity without explicitly specifying them in terms of the monotonicity condition f_σ, for the sake of conciseness.

Intuitively, we may speculate that for at least one of our two example semantics, we can find a cautious complement nonmonotonicity principle whose monotonicity condition is that either a given extension or its complement must not be attacked by arguments added by a normal expansion. After all, for the *extension* (and not *complement*) case we have found an analogous principle with cautious monotonicity, requiring the absence of new attacks to an extension. However, this is not possible for the *complement* case. As counter-examples, consider as F and F', respectively, each of the following sub-figure pairs of Figures 4: (a, b), (c, d), and (e, f). Let us refer to them as Pair 1, Pair 2, and Pair 3, respectively.

Pair 1 (Sub-figures 4a and 4b) demonstrates, for both preferred and stage semantics σ, that attacks to a given extension can lead to the violation of complement nonmonotonicity: for F, the only σ-extension is $E = \{\mathsf{b}\}$, with $E^C = \{\mathsf{a}\}$. For F', the only σ-extension is $E' = \{\mathsf{a}, \mathsf{c}\}$ and thus $E'^C = \{\mathsf{b}\} \not\supseteq E^C$.

Pair 2 (Sub-figures 4c and 4d) demonstrates, for preferred semantics, that attacks to an extension's complement can lead to the violation of complement nonmonotonicity as well: for F, the only σ_p-extension is $E = \emptyset$, with $E^C = \{\mathsf{a}, \mathsf{b}, \mathsf{c}\}$. For F', the only σ_p-extension is $E' = \{\mathsf{a}, \mathsf{d}\}$, with $E'^C = \{\mathsf{b}, \mathsf{c}\} \not\supseteq E^C$.

Pair 3 (Sub-figures 4e and 4f) demonstrates, for stage semantics, that attacks to an extension's complement can lead to the violation of complement nonmonotonicity: $\sigma_s(F) = \{\{\mathsf{a}\}, \{\mathsf{b}\}\}$. For $E = \{\mathsf{b}\}$ we have the complement $E^C = \{\mathsf{a}, \mathsf{c}\}$. For F' the only σ_s-extension is $E' = \{\mathsf{a}, \mathsf{d}\}$, with $E'^C = \{\mathsf{b}, \mathsf{c}\} \not\supseteq E^C$.



(a) F (b) F' (c) F (d) F' (e) F (f) F'

Figure 4: Pairs of AF F and F's normal expansion F'. Complement monotonicity can be violated as a result of attacks to an extension (4a, 4b) or to the extension's complement (4c, 4d for σ_p; 4e, 4f for σ_s).

This means we need to follow more nuanced intuitions to develop interesting principles. A reasonable expectation is that inferring a previously not inferred argument requires an attack from a "new" argument that we are able to infer as part of an extension to any of the "old" arguments (in the original AF) that attack arguments not in the extension. Such attack(s) may be necessary to "defend" (in the technical or intuitive sense) arguments that move from "not inferred" to "inferred". Let us define *complement defence nonmonotonicity*, a principle that follows this intuition.

Complement Defence Nonmonotonicity

$\forall S \in \{E^C | E \in \sigma(F)\}$ it holds that $\exists S' \in \{E'^C | E' \in \sigma(F')\}$ s.t. $S \subseteq S'$ if $\nexists E'' \in \sigma(F')$ s.t. $E'' \setminus AR$ attacks $(E^C)^-$ (in F').

Preferred semantics satisfies complement defence nonmonotonicity. Assume complement monotonicity is violated; thus for some $E \in \sigma_p(F)$ it holds that $\nexists E' \in \sigma_p(F')$ s.t. $E^C \subseteq E'^C$. Also, then every maximal admissible set S in F', i.e., every $S \in \sigma_p(F')$ must defend at least one $\mathsf{x} \in E^C$ (in F') and not be conflicted with $\{\mathsf{x}\}$; as this has not been the case for $E \in \sigma_p(F)$ (in F, otherwise $\mathsf{x} \in E$ would have been the case) and as conflicts between arguments in A remain the same, there must exist some $E'' \in \sigma_p(F')$ s.t. $E'' \setminus AR$ attacks the attackers of some $\mathsf{x} \in E^C$ in F' (intuitively, to defend against some argument in either E or E^C).

Stage semantics does not satisfy complement defence nonmonotonicity. For a counter-example see F and F' in Sub-figures 3a and 3b, respectively; observe that $F \prec_N F'$. Given the σ_s-extension $E = \{\mathsf{a}\}$ of F and its complement $E^C = \{\mathsf{b}\}$, there exists no σ_s-extension E' of F' s.t. $E^C \subseteq E'^C$, as $E' = \{\mathsf{b}\}$ is the only stage extension of F' (due to b's attack of c, so to speak) and $E'^C = \{\mathsf{a}, \mathsf{c}\} \not\supseteq \{\mathsf{b}\}$.

We can search for a different principle that stage semantics may be able to satisfy. What about conditioning complement nonmonotonicity on the existence of a "new" extension that attacks the corresponding "old" extension? We can call such principle *c-extension attack nonmonotonicity* (*c* stands for *complement*).

C-Extension Attack Nonmonotonicity

$\forall S \in \{E^C | E \in \sigma(F)\}$ it holds that $\exists S' \in \{E'^C | E' \in \sigma(F')\}$ s.t. $S \subseteq S'$ if $\nexists E'' \in \sigma(F')$ s.t. E'' attacks E.

Indeed, stage semantics satisfies this principle. Observe that as stage extensions are maximal conflict-free sets, $\forall E \in \sigma_s(F), \forall \mathsf{x} \in E^C$ it must hold that $E \cup \{\mathsf{x}\}$ is conflicted. If complement monotonicity is violated, for some $E \in \sigma_s(F)$ it then must hold that $\nexists E' \in \sigma_s(F')$ s.t. $E \subseteq E'$—otherwise, $E' \cup \{\mathsf{x}\}$ must be conflicted (in F') for every $\mathsf{x} \in E^C$, which would mean that $E^C \subseteq E'^C$. Now, if $\nexists E' \in \sigma_s(F')$ s.t. $E \subseteq E'$ then $\exists E'' \in \sigma_s(F')$ s.t. E'' attacks E (see observation about σ_s and extension attack nonmonotonicity).

Observe that preferred semantics violates complement attack nonmonotonicity. For a counter-example, consider the (F, F')-pair of Sub-figures 4c and 4d, respectively. For $E = \emptyset$, the only preferred extension of F, the complement is $E^C = \{\mathsf{a}, \mathsf{b}, \mathsf{c}\}$. Clearly, no set of arguments attacks $\emptyset$ in F'. Still, given $E' = \{\mathsf{a}, \mathsf{d}\}$, i.e., the only preferred extension of F', we have $E'^C = \{\mathsf{b}, \mathsf{c}\} \not\supseteq E^C$.

To conclude, we can see that systematically relaxing monotonicity is challenging also for the case of (abstract) argumentation, somewhat analogously to challenges

Dov has observed with nonmonotonic logics. There appears to be no obvious "one principle fits all semantics"-relaxation of monotonicity; further studies may shed light on additional nuances, drawing from a substantial body of related work [3].

4 Can We Build Cautiously Nonmonotonic Systems?

Professors of Plumbing. Being aware of my applied research activities, Dov admonished me to not become a *professor of plumbing.* According to Dov, in Victorian England, indoor plumbing was the subject of serious academic study, with professors specialising in the field and writing text books about the topic. By focusing one's attention on mundane problems that practitioners can easily solve (not to speak of problems so trivial that practitioners do not care about them at all), one becomes a professor of plumbing in the metaphorical sense. Unfortunately, studying plumbing problems is what—in my experience—often yields the most immediate extrinsic reward, given the current academic system. Accordingly, I heed Dov's advice and try to spend as much time as possible studying the intrinsically and philosophically interesting.

Let us now move on to provide a practically-minded perspective on the idea of cautious nonmonotonicity. As mentioned in the introduction, unconstrained nonmonotonicity can be considered—somewhat colloquially speaking—*unhinged.* Certainly, one would question the cognitive stability or social adequateness of an agent that revises their inferences for no apparent reason, e.g., by flip-flopping with respect to basic facts about themselves and their environment; indeed, such behaviours, when observed in prominent people, are frequently criticised and ridiculed.

While unhinged-ness has traditionally been a characteristic primarily ascribed to humans, advances in artificial intelligence research and applications have achieved impressingly unhinged machines, most recently utilising Large Language Models (LLMs), possibly prompted with a high *temperature* parameter value. Even more "symbolic" systems can easily draw nonmonotonic inferences without exhibiting any signs of intelligence. As an example, consider JavaScript, the arguably most unhinged of mainstream programming languages. Take the following JavaScript function:

```
r = k => !!k.isSnowing,
```

where `k` is the input argument, `!!` denotes double negation, and `k.isSnowing` queries the value assigned to the `isSnowing` key in object/KB `k` (all then assigned as a function to variable `r`). Now, let us define KBs `k0 = {}` and `k1 = {isSnowing: true}` (using `k0`, `k1` instead of K, K' due to language-specific conventions); `r(k0)` yields `false`, while `r(k1)` yields `true`. As `k1` can be considered a superset of `k0`, monotonicity is violated. In contrast to the unhingedness of many modern subsymbolic systems, observe that one can consider the violation of monotonicity as the application of negation as failure and thus principle-based: as the key `isSnowing` does not occur in `kb1`, failure to infer anything about `isSnowing` leads to its negation, as a consequence of the double negation operator's intended behaviour. This means that in this example, JavaScript is actually not particularly unhinged, anecdotally contradicting some common narratives about the nastiness of the language.

Obviously, JavaScript can hardly be considered a language for intelligent inference. Indeed, it would be odd to argue for the use of JavaScript—e.g., instead of the latest and greatest of subsymbolic research advancements—for instilling "intelligence" into software-based computing systems. Still, it is striking that production-grade systems utilising LLMs tend to be heavily reliant on knowledge-based systems (and often on substantial human work for data labelling or similar tasks); i.e., human-crafted domain knowledge still is of central importance. This stands in contrast to the prevalent idea that algorithms and compute is all that matters—which is known as Sutton's *Bitter Lesson* [10]. Accordingly, it is not clear whether conditions for when to violate monotonicity can, in a practically reasonable way, be made on the level of a generic inference function (be it symbolic, subsymbolic, or hybrid), or whether the modelling of domain knowledge is required, specifically pertaining to the conditions under which monotonicity can be violated. An example of a rule for cautiously nonmonotonic inference that requires both domain knowledge and the ability to reason on both object- and meta-levels is: *a person that is dead cannot be revived, unless a meta-level error has caused the incorrect logging of the person as "dead" in the first place*[7]. Even 40 years after Dov's seminal paper, computing systems struggle with such rules, and evaluating the abilities of real-world systems to behave nonmonotonically in an—intuitively or formally—"principled" way seems to be both difficult and under-explored.

[7] See the following link for a related anecdotal observation of the shortcomings of LLM-based systems: `https://davebarry.substack.com/p/death-by-ai`, *accessed at 2025-08-06.*

5 Conclusions

Argument over Dogma. Dov's interest in studying and discussing religion with an entirely open mind has always impressed me. I recall that the first time I met Dov, he asked me whether I believe in God. Like many academically inclined people of my generation, my relationship with religion is complicated at best; I dodged the question by remarking that surely, an answer requires providing a definition of God first. Still, I have since taken an interest in understanding Dov's studies of religion and logic. Indeed, early on in my academic journey I had the pleasure to join Dov in giving an (online) seminar about Talmudic logic as part of the *Logica Universalis* series. We presented some results (as of now still not properly published) about the iterative inference from argumentation graphs that are constructed step-by-step, one argument at a time, assuming these ideas would be interesting in this context. Certainly, giving this seminar was the furthest I have moved out of my comfort-zone, with respect to topic and competence.

In this paper, I have given an overview of the notion of *cautious nonmonotonicity*, which, intuitively, describes the meta-principle of remaining monotonic (in the inferences that one draws), as long as some condition remains satisfied. I have connected this idea to *restricted monotonicity* from Dov's seminal 1985 paper on nonmonotonic reasoning, as well as to notions relating to nonmonotonicity in formal argumentation, highlighting difficulties with instantiating generally applicable principles from cautious nonmonotonicity. Finally, I have discussed analogous challenges in a practical context, where it is unclear whether or to what extent "reasonable" instantiations of cautious nonmonotonicity require the hard-wiring of specific domain knowledge. Below, I sketch some ideas for future research.

Theory. The cautious nonmonotonicity meta-principle supports the definition of *necessary* conditions for nonmonotonicity; future work may focus on *sufficiency*, e.g., by surveying such sufficient conditions for argumentation semantics, as well as for inference functions of other approaches to nonmonotonic reasoning.

Practice. From an applied perspective, one may want to study the trade-off between domain knowledge and general-purpose capabilities that is required for cautiously nonmonotonic inference. For example, subsymbolic systems such as LLMs can potentially be integrated with symbolic reasoners that may, at least to some extent, rely on formally modelled domain knowledge—perhaps somewhat similar to the currently popular architectures for *retrieval augmented generation*—in order to decide whether or not a previously provided inference should be revised.

References

[1] Ringo Baumann and Gerhard Brewka. Expanding argumentation frameworks: Enforcing and monotonicity results. *COMMA*, 10:75–86, 2010.

[2] Kristijonas Čyras and Francesca Toni. Non-monotonic inference properties for assumption-based argumentation. In Elizabeth Black, Sanjay Modgil, and Nir Oren, editors, *Theory and Applications of Formal Argumentation*, pages 92–111, Cham, 2015. Springer International Publishing.

[3] Sylvie Doutre and Jean-Guy Mailly. Constraints and changes: A survey of abstract argumentation dynamics. *Argument & Computation*, 9:223–248, 2018.

[4] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial intelligence*, 77(2):321–357, 1995.

[5] D. M. Gabbay. Theoretical foundations for non-monotonic reasoning in expert systems. In Krzysztof R. Apt, editor, *Logics and Models of Concurrent Systems*, pages 439–457, Berlin, Heidelberg, 1985. Springer Berlin Heidelberg.

[6] Dov M. Gabbay. Classical vs non-classical logics (the universality of classical logic). In Dov M. Gabbay, Christopher J. Hogger, J. A. Robinson, and Jörg H. Siekmann, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming, Volume 2, Deduction Methodologies*, pages 359–495. Oxford University Press, 1994.

[7] Timotheus Kampik and Juan Carlos Nieves. Disagree and commit: degrees of argumentation-based agreements. *Auton. Agents Multi Agent Syst.*, 39(1):8, 2025.

[8] Sarit Kraus, Daniel Lehmann, and Menachem Magidor. Nonmonotonic reasoning, preferential models and cumulative logics. *Artificial Intelligence*, 44(1):167–207, 1990.

[9] David Makinson. General theory of cumulative inference. In Michael Reinfrank, Johan de Kleer, Matthew L. Ginsberg, and Erik Sandewall, editors, *Non-Monotonic Reasoning, 2nd International Workshop, Grassau, FRG, June 13-15, 1988, Proceedings*, volume 346 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 1988.

[10] Richard Sutton. The bitter lesson, 2019. Available at `http://www.incompleteideas.net/IncIdeas/BitterLesson.html`, accessed at 2025-08-06.

[11] Leon van der Torre and Srdjan Vesic. The principle-based approach to abstract argumentation semantics. *IfCoLog Journal of Logics and Their Applications*, 4(8), October 2017.

[12] Bart Verheij. Two approaches to dialectical argumentation: admissible sets and argumentation stages. *Proc. NAIC*, 96:357–368, 1996.

Received August 2025

Neuro-symbolic Models in AI

Shalom Lappin*
Queen Mary University of London, University of Gothenburg, and King's College London
s.lappin@qmul.ac.uk

Abstract

Deep Neural Networks (DNNs) in general, and transformers in particular, have revolutionised AI by achieving high levels of performance across a wide range of tasks. However, they remain limited in their capacity to handle domain general real world reasoning. They also require large amounts of training data to obtain reasonable learning outcomes. A number of researchers have attempted to combine DNNs with symbolic representations and rule systems to overcome these limitations. Hybrid models of this kind fall into two broad

Earlier versions of this paper were presented to the Chalmers Big Data and AI Seminar, and to the Centre for Linguistic Theory and Studies in Probability Seminar at the University of Gothenburg, in February 2025. I thank the audiences of these venues for helpful comments and discussion. I am also grateful to Devdatt Dubhashi, Moa Johansson, and Alexander Koller for valuable suggestions and references. I bear sole responsibility for the views expressed in this paper, and for any errors that it may contain. My research for this work was supported, in part, by grant 2014-39 from the Swedish Research Council, which funds the Centre for Linguistic Theory and Studies in Probability (CLASP) in the Department of Philosophy, Linguistics, and Theory of Science at the University of Gothenburg.

[*] It is a privilege and a pleasure to contribute to this volume, compiled in honour of Dov Gabbay's eightieth birthday. I first met Dov in Israel in the early 1970s, when he was a lecturer at Bar-Ilan University, and I was teaching at Ben Gurion University of the Negev and Tel Aviv University. In the years that followed he went on to become a distinguished figure, and a major influence, in computational logic. We were colleagues in the Department of Computer Science at King's College London from 2000-2005. Our friendship now spans many decades, and it encompasses a range of common concerns. These include a scientific focus on the role of logic in artificial intelligence and natural language processing, strong Jewish commitments, and a shared Israeli identity, in which Hebrew is a central component. In Pirkei Avot (Ethics of the Fathers) 5:24 Yehuda ben Tema describes the defining points in a peson's life. He tells us that when someone turns 80, he/she achieves *gevura*, which Rabbinic commentaries take to indicate the physical and spiritual strength required to reach this age. The literal meaning of *gevura* is courage, which adds a moral dimension to this description. I wish Dov many more years of *gevura*, so that he can continue to enrich our understanding of computational logic, and of Rabbinic reasoning, with the gentleness and compassion that he has always shown towards the people around him.

classes. The first includes DNNs in which symbolic representations and constraints are injected directly into the internal processing operations of the system, or they are incorporated into the data on which it is trained. In the second class, DNNs and symbolic reasoning systems operate autonomously, with the independence of each sustained. DNNs extract features from the input, which are made accessible to a symbolic rule system through an interface. I consider several instances of each type of hybrid neuro-symbolic model, with application to a number of AI tasks. The available evidence suggests that while the first class of models has, in general, not yielded substantial improvements over their non-hybrid counterpart systems, the second variety has produced more hopeful results. I will briefly consider the implications of this contrast in the architecture of hybrid models for future research in deep learning.

1 Some AI History: The Deep Learning Revolution

In the early years of AI both neural networks and symbolic systems were unable to go beyond small scale models, which had to be adjusted, often by handcrafted extensions, to new cases. This was, in large measure, the result of the hardware limitations of the time, and the absence of digitalised data for training and testing.

Feed forward neural networks lacked memory for tracking long distance dependency relations in input data. Symbolic systems did not incorporate learning procedures, and so their rules had to be devised by hand. Minsky ([16]) suggested that hybrid systems, combining neural networks for lower level perceptual classification and symbolic components for reasoning, were needed for progress in AI.

In the past three decades the emergence of powerful hardware (GPUs), the abundance of online data, and radical innovations in the architecture of neural networks, have produced the deep learning revolution. Transformers, which drive Large Language Models (LLMs), consist entirely of blocks of attention heads. These are trained independently of each other, and they can identify fine grained patterns in data across distinct modalities (text, visual images, sound, etc.). They have equalled or surpassed human performance over a wide variety of cognitively challenging tasks that had resisted earlier AI systems. They define the state of the art for most AI applications, and they have all but displaced symbolic systems.[1]

LLMs do not perform reliably on natural language inference (NLI) tasks, when subject to adversarial testing ([21], [20]). They also do not do well on many real world reasoning tasks ([12]). While transformers learn superficial patterns of inference and they are sensitive to some lexical semantic content in arguments, they do not

[1] [10] provides a brief history of AI, and the factors that have generated the deep learning revolution.

acquire stable deep reasoning abilities. LLMs are notorious for hallucinating fluent but fictional content, which undermines their reliability for question-answering, and a variety of other applications. Transformers are computationally opaque, in large measure because their activation and probability generating functions (such as ReLU and softmax) are non-linear.[2]

2 Injective Hybrid Models

Some theorists have revived Minksy's call for the development of hybrid neuro-symbolic models ([14]). Proponents of neuro-symbolic models assert that they significantly reduce training time by encoding information in symbolic features and rule systems, which would require additional data to extract. They argue that these models are more transparent than non-enriched DNNs, by virtue of the explainable nature of their symbolic content. They maintain that the symbolic component of these models substantially improves their performance, relative to non-symbolic DNNs, over a wide variety of tasks. In fact, the evidence for these claims is far from clear, in at least one major class of neuro-symbolic models.

One way of constructing a hybrid framework is to inject symbolic representations into the processing operations of a Deep Neural Network (DNN). This can be done directly, by revising the architecture of the DNN to incorporate the biases of a symbolic system into its computation, at different levels of the network. Injection can also be achieved indirectly, through training the DNN on a biased distribution that a symbolic system generates (knowledge distillation). Symbolic markers, or structures, can also be inserted into the data on which a DNN is trained.

Tree DNNs incorporate syntactic structure into a Deep Neural Network (DNN), either directly through its architecture, or indirectly through knowledge distillation and training data. [19], [3], [23], [4], [22], [13], [7] consider LSTM-based Tree DNNs. These have been applied to NLP tasks like sentiment analysis, NLI, and the prediction of human sentence acceptability judgments. They have yielded small improvements in performance, which do not provide strong motivation for inserting trees, or syntactic and semantic markers into LSTMs.[3]

More recent work has incorporated syntactic tree structure into transformers like BERT, and applied them to a broader range of tasks. [2] integrate tree structure recognition into the attention head blocks of BERT and RoBERTa. They test different versions of these transformers on the GLUE benchmark tasks, which include sentence acceptability assessment, paraphrase recognition, and NLI. The structure

[2] [9] discusses the strengths and the limitations of LLMs.

[3] See [8] for detailed discussion of these LSTM-based tree DNNs.

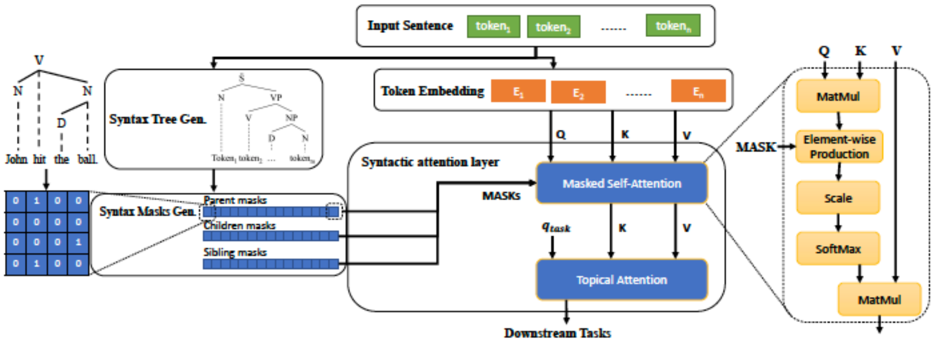


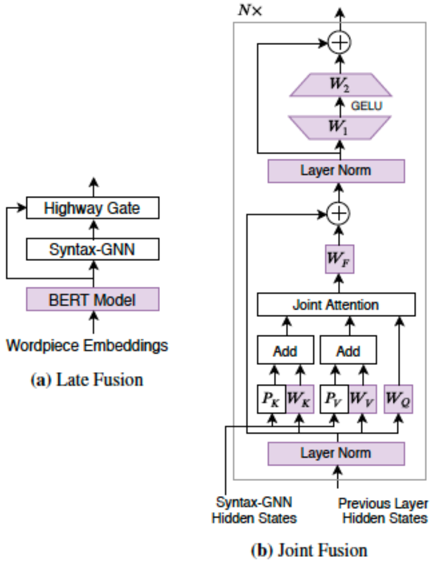
Figure 1: Syntax-BERT, Bai et al. (2021)

of Bai et al.'s syntax enriched BERT is shown in Figure 1. For the overwhelming majority of cases they report an accuracy gain of the tree enriched model, relative to its non-tree counterpart, of between 1% and 2%. These results suggest that the contribution of the implemented tree structure enrichment to BERT and RoBERTa's performance on the GLUE tasks is marginal.

[17] enrich BERT and RoBERTa with dependency tree graphs. They test them on semantic role labelling, named entity recognition, and relation extraction. Figure 2 displays two versions of their tree graph enriched model. In the (a) variant the graphs are fused with BERT at a late layer of the transformer. In the (b) version they are injected into earlier layers. For in domain test sets the graph versions of the models achieve F1 scores that are 1%-2% higher than their non-enriched counterparts. In an out of domain test on semantic role labelling, the gain in F1 score was 2%-5%. These results are similar to those that [2] report for their syntactic tree versions of BERT and RoBERTa.

[1] enrich a CNN by infusing handcrafted knowledge features for segmenting brain aneurism images. They experiment with feature infusion at different levels of the network. They use Intersection over Union (IoU) as the metric to compare several versions of the feature infused CNN with its non-enriched baseline. Let M_{image} be the image that the model identifies, and GT_{image} be the ground truth image. Then IoU is defined as follows.

1. $\text{IoU} = \dfrac{area(M_{image}) \cap area(GT_{image})}{area(M_{image}) \cup area(GT_{image})}$



(a) Late Fusion

(b) Joint Fusion

Figure 2: Dependency Tree Graph BERT, Sachan et al. (2021)

Figure 3 displays [1]'s knowledge feature diffusion CNN. Their best model scored an IoU of 0.9676, while the non-enriched CNN achieved 0.9158.

[11] modify the hidden units of a CNN to function as probabilistic logical operators. They train the network to extract rules for diagnosing diabetes on the basis of data encoded as feature vectors. They compare alternative implementations of their rule learning CNN with traditional machine learning methods used for medical diagnosis. Their highest scoring model obtains an F1 score of 0.6875 on their test set, while they report Random Forest as achieving the best traditional ML result at 0.6380. Their best enriched CNN for Area Under the Curve (AUC) binary classification scores 0.8457, while Random Forest achieves 0.8342. Interestingly, they do not provide a comparison between their logically enriched CNN and a baseline version of the same model. [11]'s results are given in Table 1

Injective models provide small gains in performance relative to their unenriched counterparts. These gains tend to diminish with additional training data for non-enriched DNNs. The claim that injective models offer greater transparency than non-injective DNNs is open to question. In most cases injective DNNs remain non-

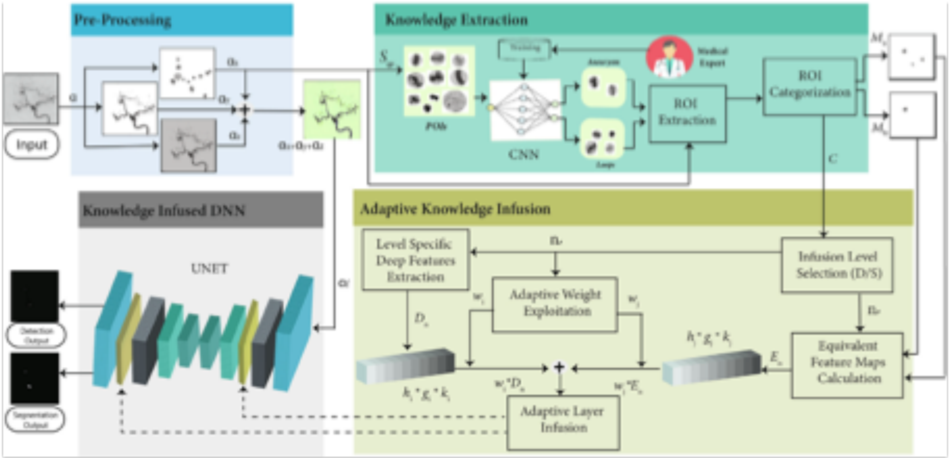
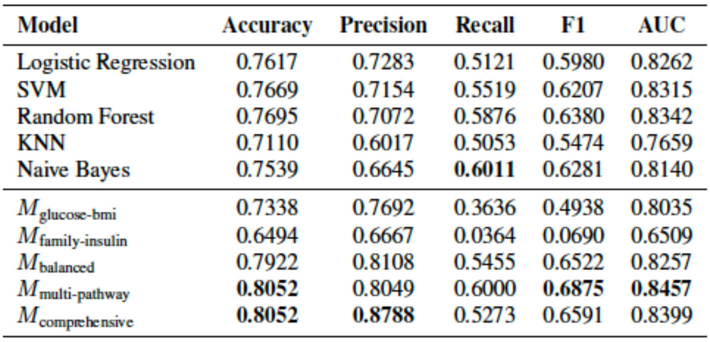


Figure 3: Knowledge Feature Infusion CNN, Abdullah et al. 2023



Model	Accuracy	Precision	Recall	F1	AUC
Logistic Regression	0.7617	0.7283	0.5121	0.5980	0.8262
SVM	0.7669	0.7154	0.5519	0.6207	0.8315
Random Forest	0.7695	0.7072	0.5876	0.6380	0.8342
KNN	0.7110	0.6017	0.5053	0.5474	0.7659
Naive Bayes	0.7539	0.6645	**0.6011**	0.6281	0.8140
$M_{\text{glucose-bmi}}$	0.7338	0.7692	0.3636	0.4938	0.8035
$M_{\text{family-insulin}}$	0.6494	0.6667	0.0364	0.0690	0.6509
M_{balanced}	0.7922	0.8108	0.5455	0.6522	0.8257
$M_{\text{multi-pathway}}$	**0.8052**	0.8049	0.6000	**0.6875**	**0.8457**
$M_{\text{comprehensive}}$	**0.8052**	**0.8788**	0.5273	0.6591	0.8399

Table 1: Lu et al. (2025) Results

compositional in their output at each level, as they continue to use non-linear functions like ReLU and softmax to generate output vectors.

Advocates of injective models tend to assume that humans acquire and represent most knowledge as rule sets that are best modelled as algebraic systems (grammars, logics, sets of constraints, etc.). It is far from obvious that this is the case for all types of knowledge. It is entirely possible that humans encode many aspects of their discriminatory classification knowledge in non-symbolic, distributed representations of regularities, as [18] and [15], among others, suggest. It is also possible that, by virtue of their design, DNNs are unable to easily integrate symbolic components into their distributed representations of information, in a way that significantly improves learning or inference.

3 Federative Hybrid Models

A federative hybrid model does not inject symbolic content into a DNN. It combines a DNN with a symbolic reasoning module within a framework in which each of these systems functions autonomously. The framework sustains the distinct computational procedures that its two central components apply for representing information. In one version of this architecture the DNN extracts features for an interface that labels them, and feeds them to a logic based inference program. This approach seems closer than an injection model to Minsky's original proposal.

[5] present a Feed Forward Neural-Symbolic Learner (FFNSL) for image classification. It consists of a DNN for extracting features from images, an interface component that assigns labels to these features, and a logic based system, an Inductive Logic Program (ILP), that learns rules from these labelled features. They test variants of this model on a suite of image classification tasks in which knowledge of a game, or a problem, are necessary for the correct solution. They use distributional shifts of training and test data (through image rotation) to ascertain the robustness of the system under variation. The architecture of FFNSL is given in Figure 4.

FFNSL models exhibit significant gains over non-symbolic ML and DNN baselines. They require significantly less training data to achieve high accuracy in complex image classification tasks. They remain stable over higher levels of distributional shift in the images of both training and test data. They generate transparent rule-based hypotheses. [5] test FFNSL on a series of image recognition tasks which require different sorts of knowledge. One of these tasks is the identification of valid sudoko grids in 4 X 4 and 9 X 9 squares. The graphs in Figure 5 show the accuracy of FFNSL for this task, relative to baselines systems without ILP enrichment, over training size (320 vs 32000 samples) and percentage of distributional shift in the

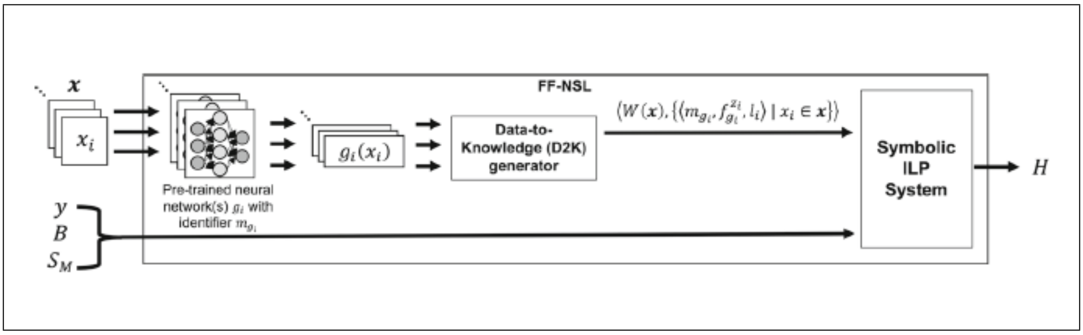
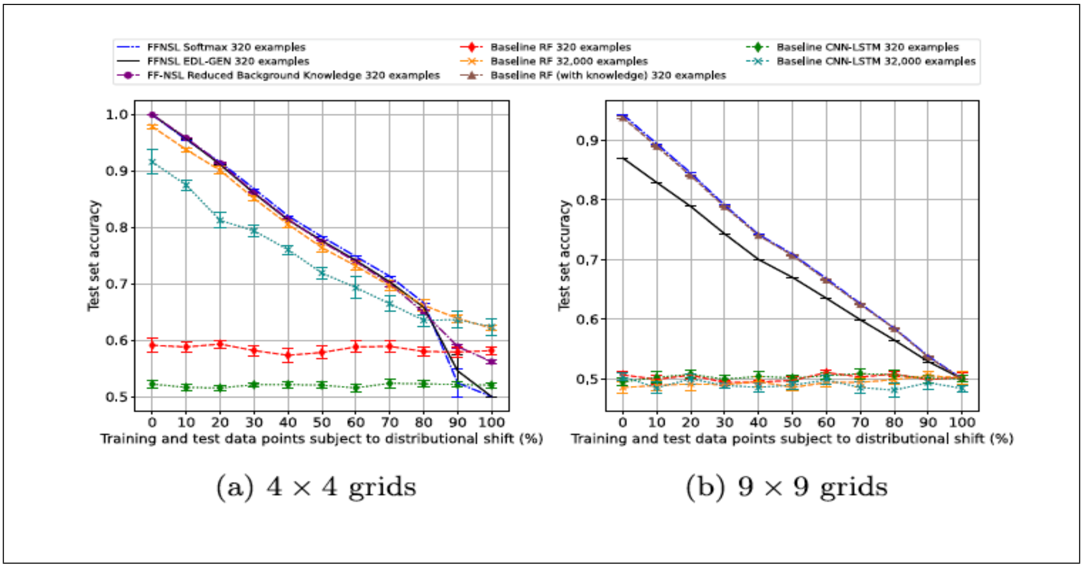

x
x_i
FF-NSL
Pre-trained neural network(s) g_i with identifier m_{g_i}
$g_i(x_i)$
Data-to-Knowledge (D2K) generator
$\langle W(x), \{\langle m_{g_i}, f_{g_i}^{z_i}, l_i \rangle \mid x_i \in x\}\rangle$
Symbolic ILP System
H
y
B
S_M

Figure 4: FFNSL, Cunnington et al. 2023



(a) 4 × 4 grids

(b) 9 × 9 grids

Figure 5: FFNSL Accuracy for Sudoku Grid Identification, Cunnington et al. (2023)

training and test sets.

[6] propose another instance of a federative hybrid model. They consider three well known NP-hard optimisation problems: Graph Colouring, Knapsack, and Travelling Salesman. The first and third problems involve finding an optimal solution for adjacent colour distribution, and non-redundant routes, respectively, through the nodes of a graph. The second requires satisfying a weight constraint for the largest number of items placed in a knapsack. As the number of nodes in the graph, or items to be placed in the knapsack, grows, the complexity of the problem increases exponentially, in a way that renders the task NP-hard.

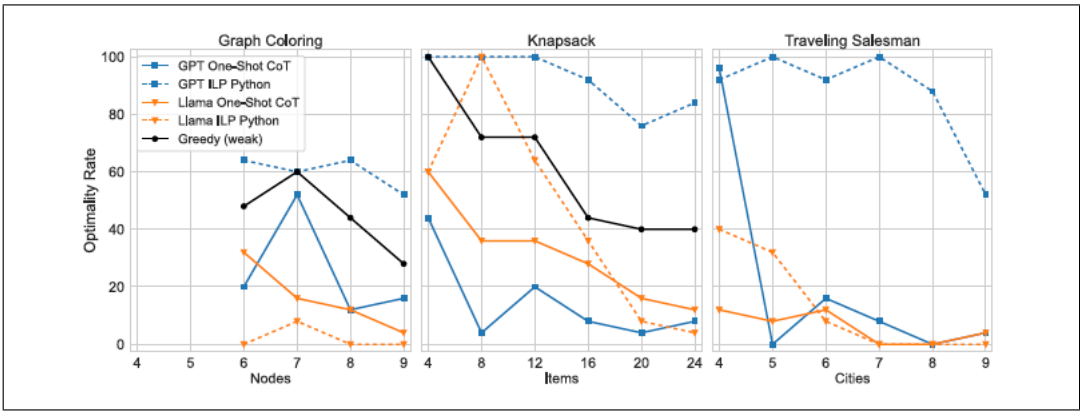


Figure 6: Results for NP-Hard Optimisation Problems, Duchnowski et al. (2025)

[6] experiment with two LLMs, GPT 4o and Llama 3.1 70B Instruct, as well as several greedy algorithms, on different versions of these problems. The main distinction in these versions is between textbook specifications of the problems, and informal natural language formulations. They find that the LLMs perform more successfully on the textbook specifications than on the informal versions.

Their federative model consists of the LLM feeding an optimisation task to a Python Integer Linear Program encoder, which applies a Gurobi optimisation solver to the problem. They find that the GPT 4o hybrid model significantly outperforms its non-enriched counterpart on both textbook and natural language versions of the problems. Figure 6 shows the results for the models that [6] test on the textbook versions of the three problems. GPT 4o ILP Python and Llama ILP Python are the hybrid systems calling the Gurobi solver.

There are two significant limitations worth noting in [6]'s reported experiments. First, they do not test their models against a human baseline. Therefore, it is not clear how any of their models compare with human performance on these problems. It is important to know how both the Gurobi solver enriched models, and their non-enriched counterparts, reflect or diverge from human abilities for NP-hard optimisation tasks.

Second, [6] test only two LLMs, GPT 4o and Llama 3.1 70B. While GPT 4o ILP Python does well on the three tasks Llama ILP Python does not. Is this because of its size, or its architecture? More experimental data for additional LLMs, and a larger variety of optimisation problems, are required before we can draw firm conclusions on the capacity of LLMs, symbolically augmented or not, to handle this type of task.

4 Conclusions and Future Research

This overview of two approaches to constructing hybrid neuro-symbolic models suggests several preliminary conclusions. First, the injection of symbolic features or rule-based biases directly into a DNN does not seem to significantly improve its performance, relative to a non-enriched version of the same model. Second, this limitation of injective DNNs may be due to the difference in the way DNNs and symbolic systems represent patterns of regularity. Third, federative neuro-symbolic models sustain the internal integrity and autonomy of both types of processing system. Finally, they appear to offer a more effective way of combining the strengths of each framework.

Further research on both injective and federative models is required to ascertain the extent to which the final conclusion, which is still a conjecture, actually holds. More extensive comparisons of injective and non-injective state of the art transformers, over a wider variety of tasks, is needed to obtain a better sense of the limits of this approach. Similarly, federative models in which current transformers are used as the DNN, with testing against the unenriched transformers, will help to clarify the prospects of this version of neuro-symbolic machine learning. At this point, federative models may present the most efficient way of augmenting the reasoning and inference capacities of DNNs. They also suggest a route to greater transparency in DNN driven machine learning.

References

[1] Iram Abdullah, Ali Javed, Khalid Mahmood Malik, and Ghaus Malik. DeepInfusion: A dynamic infusion based-neuro-symbolic AI model for segmentation of intracranial aneurysms. *Neurocomputing*, 2023.

[2] Jiangang Bai, Yujing Wang, Yiren Chen, Yaming Yang, Jing Bai, Jing Yu, and Yunhai Tong. Syntax-bert: Improving pre-trained transformers with syntax trees. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3011–3020, Stroudsburg, PA, 2021. Association for Computational Linguistics.

[3] Samuel R. Bowman, Jon Gauthier, Abhinav Rastogi, Raghav Gupta, Christopher D. Manning, and Christopher Potts. A fast unified model for parsing and sentence understanding. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1466–1477, Berlin, Germany, August 2016. Association for Computational Linguistics.

[4] Jihun Choi, Kang Min Yoo, and Sang goo Lee. Learning to compose task-specific tree structures. In *AAAI Conference on Artificial Intelligence*, 2018.

[5] Daniel Cunnington, Mark Law, Jorge Lobo, and Alessandra Russo. FFNSL: Feed-forward neural-symbolic learner. *Machine Learning*, 112:515–569, 2023.

[6] Alex Duchnowski, Ellie Pavlick, and Alexander Koller. EHOP: A dataset of everyday np-hard optimization problems. *arXiv 2502.13776*, 2025.

[7] Adam Ek, Jean-Philippe Bernardy, and Shalom Lappin. Language modeling with syntactic and semantic representation for sentence acceptability predictions. In *Proceedings of the 22nd Nordic Conference on Computational Linguistics*, pages 76–85, Turku, Finland, 2019.

[8] Shalom Lappin. *Deep Learning and Linguistic Representation.* CRC Press, Taylor & Francis, Boca Raton, London, New York, 2021.

[9] Shalom Lappin. Assessing the strengths and weaknesses of large language models. *Journal of Logic, Language and Information*, 33(1):9–20, 2024.

[10] Shalom Lappin. *Understanding the Artificial Intelligence Revolution: Between Catastrophe and Utopia.* CRC Press, Taylor & Francis, Boca Raton, London, New York, 2025.

[11] Qiuhao Lu, Rui Li, Elham Sagheb, Andrew Wen, Jinlian Wang, Liwei Wang, Jungwei W. Fan, and Hongfang Liu. Explainable diagnosis prediction through neuro-symbolic integration. *arXiv 2410.01855*, 2025.

[12] Kyle Mahowald, Anna A. Ivanova, Idan A. Blank, Nancy Kanwisher, Joshua B. Tenenbaum, and Evelina Fedorenko. Dissociating language and thought in large language models. *Trends in Cognitive Sciences*, 28(6):517–540, 2024.

[13] Jean Maillard, Stephen Clark, and Dani Yogatama. Jointly learning sentence embeddings and syntax with unsupervised tree-lstms. *Natural Language Engineering*, 25(4):433–449, 2019.

[14] Gary Marcus. Deep learning alone isn't getting us to human-like AI. *Noema*, August 11,2022, 2022.

[15] James L. McClelland. Capturing gradience, continuous change, and quasi-regularity in sound, word, phrase, and meaning. In Brian MacWhinney and William O'Grady, editors, *The Handbook of Language Emergence*, pages 54–80. John Wiley and Sons, Hoboken, NJ, 2016.

[16] Marvin Minsky. Logical versus analogical or symbolic versus connectionist or neat versus scruffy. *AI Magazine*, 12(2):34–51, 1991.

[17] Devendra Singh Sachan, Yuhao Zhang, Peng Qi, and William Hamilton. Do syntax trees help pre-trained transformers extract information? In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2647–2661, Stroudsburg, PA, 2021. Association for Computational Linguistics.

[18] Paul Smolensky. Connectionist AI, symbolic AI, and the brain. *Artificial Intelligence Review*, 1(2):95–109, 1987.

[19] Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Lan-*

guage Processing, pages 151–161, Edinburgh, Scotland, UK., July 2011. Association for Computational Linguistics.

[20] Aarne Talman, Marianna Apidianaki, Stergios Chatzikyriakidis, and Jörg Tiedemann. NLI data sanity check: Assessing the effect of data corruption on model performance. In *Proceedings of the 23rd Nordic Conference on Computational Linguistics (NoDaLiDa)*, pages 276–287, Reykjavik, Iceland (Online), 2021. Linköping University Electronic Press, Sweden.

[21] Aarne Talman and Stergios Chatzikyriakidis. Testing the generalization power of neural network models across NLI benchmarks. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 85–94, Florence, Italy, August 2019. Association for Computational Linguistics.

[22] Adina Williams, Nikita Nangia, and Samuel Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana, 2018. Association for Computational Linguistics.

[23] Dani Yogatama, Phil Blunsom, Chris Dyer, Edward Grefenstette, and Wang Ling. Learning to compose words into sentences with reinforcement learning. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

Received May 2025

Boole Between the Equations

David Makinson
School of Historical and Philosophical Inquiry, University of Queensland
d.makinson@uq.edu.au

Dedicated to Dov Gabbay on his 80th birthday: a life in logic and a friend for life

Abstract

Two questions are investigated. Why is it that, despite his general explanations, Boole's logical operations can be difficult to pin down when he solves specific problems? Why did his late manuscript attempt to get rid of division by zero fall short of its goal? It is suggested that the former difficulty arises from his segmentation of solution-routines into stages, with the identity of the operations differing from stage to stage; while the latter failure stems from his continuing confinement to strictly equational reasoning.

1 Two questions

Why is it that despite Boole's general explanations, it is so difficult to pin down even his basic operations of multiplication, addition and subtraction when reading his solutions to logical problems?

Why is it that when, in a late manuscript only now being published, he finally got down to the task of trying to eliminate the controverted operation of division from his procedures, he did not fully succeed?

These are the central questions of the present paper. In outline, the conclusions are as follows:

- For Boole, solving a logical problem consisted essentially in running a routine. The routines have stages, and the identities of even the basic operations of multiplication, addition and subtraction are not fixed, but vary according to the stage of the routine in which they appear. These stages and their operations are analysed in section 3, revealing also several different ways of reading Boole's celebrated $\{0, 1\}$ principle.

The author thanks David Waszek for generously making available the edited text of Boole's manuscript 'The nature of thought' before its publication and general discussions of Boole's work. Thanks also to Alex Citkin, Antonielly Garcia Rodrigues and again David Waszek for helpful information, comments and corrections on drafts.

- While the late manuscript succeeded in removing division from most of Boole's routines, it remained in his solution of the central 'find x' problem. One may suspect that, at bottom, he was not very highly motivated about the task, or that he intended to iron out the wrinkle from his draft at a later date. But on a substantive level, his self-imposed restriction to equational reasoning held him back from applying several techniques any one of which would have rendered the task trivial, as detailed in section 6.

2 Background: Equations and the $\{0, 1\}$ principle

We briefly recall two features of Boole's work that are particularly relevant to the issues raised above: his focus on equational reasoning, and his celebrated $\{0, 1\}$ principle.

Equational reasoning

The Introduction to MAL[1] contains the following laconic declaration:

> A logical proposition is, according to the method of this Essay, expressible by an equation.

Although not repeated in MAL, unmentioned in CL, and barely hinted at in LT pp. 34-5, this principle is applied with utter strictness in all three publications: logical relationships are always expressed symbolically as equalities. The constancy of this practice manifests itself in the symbolism, where the equality sign reigns supreme. In the whole of MAL, CL and (as far as the present author has noticed) the lengthy text of LT, there is not a single use of the sign $\neq$, nor any sign for class

[1] Boole published three texts on logic. The first was a booklet of 1847 of under ninety pages, entitled *The Mathematical Analysis of Logic*. The second was an 1848 article called 'The calculus of logic' summarizing the booklet's essentials in just thirteen pages, for an audience of professional mathematicians. The third was *The Laws of Thought*, a full-length treatment in 1854, whose first fifteen chapters are on logic (the remaining seven are mainly on probability, reworked in the light of the logic). Three main posthumous publications are currently available: some letters to Jevons in Grattan-Guinness [20], some manuscript selections in Grattan-Guinness & Bornet [21] and, most recently, the important manuscript 'The nature of thought' in Waszek (to appear) [36], to which frequent reference is made in sections 5 and 6 of the present paper. For brevity, the acronyms MAL, CL, LT are used for the texts published in Boole's lifetime, and NT for the last-mentioned manuscript. Page numbers for MAL, CL and LT are those of the originals (preserved in the annotated texts of Burris [10, 11, 12] respectively as well as in the Dover reprint of LT). For NT they are as recorded by Waszek, e.g. B.46 means page 46 of the manuscript. In the present paper, double quotation marks are used when citing specific passages from a text; otherwise single ones.

inclusion. Even the signs $<, \leq$ are used only in the chapters dealing with probability, where we find expressions $r < s$ between probability values. To be sure, from time to time and especially in later manuscripts, Boole does occasionally use the word 'inclusion'; for example, in NT B.19 he tells us that the equation $y = vx$ "expresses that inclusion of the class y in the class x". But every formal statement in the logic of classes is written as an equation. In the opinion of Jevons [24], "The great reform effected by Boole was that of making the equation the corner-stone of logic, as it had always been that of mathematical science."[2]

Restriction to the language of equations gave Boole sharp focus, but it also brought costs. Notoriously, the only way he could express traditional 'particular' propositions *Some Xs are Ys* by equations was with the help of what he called 'indefinite symbols' $v, w, \ldots$ as one of $vx = wy, vx = vy, xy = v, vx(1-y) = 0$ (see e.g. MAL pp. 22–5). The obscurity surrounding the meaning of such indefinite symbols, as well as their clumsiness, led critics beginning with Cayley [14] to point out that they are unnecessary; if we allow inequalities, we can express *Some Xs are Ys* simply as $xy \neq 0$. To be sure, following the seminal work of Peirce and Frege in the 1870s on the logic of quantifiers, Boole's indefinite symbols could be clarified. As observed by Couturat [15, §34], Broad [6, p. 92], and more recently Hailperin [22, §§1.10, 2.5], the above equalities may be read with v and w implicitly bound by existential quantifiers; with hindsight, we can see Boole struggling to express existential quantification without ever giving it symbolic dress. However, the clarification has not led to revival, for there is no point in machinery for work that is unnecessary.[3]

The cost that interests us in this paper is less visible. We will contend in section 6 that strict adherence to an equational straitjacket created technical hurdles for Boole's attempt to eliminate division from his procedures.

[2] The idea of expressing logical relationships as equations was already to some extent 'in the air' when Boole wrote MAL. In the seventeenth and eighteenth centuries, Leibniz and Ploucquet had tried representing *some* logical relations in that way, and the position that they should *always* be so expressed seems to have appeared early in the nineteenth century when mooted by George Bentham [1, pp. 127–135]. It was also pronounced by De Morgan [16, p. 8] who, however, did not apply it in practice and later explicitly rejected it (see De Morgan [17, p. 294]). It was declared as doctrine by Sir William Hamilton who, however, proudly disdained all mathematical symbolism in logic (see Makinson [28]).

[3] Boole also used indefinite symbols to replace coefficients $0/0$ in his solutions to logical problems. In section 6 we will see that there too they are quite unnecessary, even within the context of his general methods.

The $\{0,1\}$ principle

The algebra of the integers (positive, negative and zero) uses three basic operations: addition, subtraction, multiplication. Their properties inspired Boole in his search for an algebra of classes, but he realized that there are important differences. For example (LT pp. 36–7), the former but not the latter satisfies the law that whenever $zx = zy$ and $z \neq 0$ then $x = y$; while the latter but not the former satisfies the equation $xx = x$. Here, following Boole, we juxtapose to express both multiplication in algebras of numbers and meet in the algebra of classes, while $0, 1$ are zero and one in the former context and the empty class and universe of discourse in the latter context.

However, in LT pp. 37–8, Boole also tells us that the algebra of classes corresponds exactly with an algebra based on the numbers 0 and 1 alone. Although well known, the passage is worth citing again in full; the ellipses are introduced by the present author to replace omitted text, but the italics are Boole's.

> … instead of determining the measure of formal agreement of the symbols of Logic with those of Number generally, it is more immediately suggested to us to compare them with symbols of quantity *admitting only of the values* 0 *and* 1. Let us conceive, then, of an Algebra in which the symbols x, y, z&c admit indifferently of the values 0 and 1, and of these values alone. The laws, the axioms, and the processes of such an Algebra will be identical in their whole extent with the laws, the axioms, and the processes of an Algebra of Logic … Upon this principle the method of the following work is established.

This is the celebrated $\{0,1\}$ principle, also called the rule of 0 and 1. It is repeated in a more procedural language in LT p. 70, where it is written entirely in italics:

> *We may in fact lay aside the logical interpretation of the symbols in the given equation; convert them into quantitative symbols, susceptible only of the values 0 and 1; perform upon them as such all the requisite processes of solution; and finally restore to them their logical interpretation.*

Many questions of interpretation arise. To begin with, one may ask what kinds of statement are covered by the "laws, the axioms, and the processes" that Boole claimed to be identical. Clearly, he would not have wished them to cover the law that if $zx = zy$ and $z \neq 0$ then $x = y$ since he recognized that this holds in the algebra of $\{0,1\}$ but fails for classes in general (e.g. when z, x, y are the classes of plane figures with three, four and five sides respectively). But the principle does hold for what we now call Horn clause statements made up of equational components, and it is generally accepted that Boole vaguely had them in mind.

More subtly, one may ask what kinds of expression are "susceptible only of the values 0 and 1". In both of the above passages, Boole speaks of "symbols", by which he means what we would call variables; but what about compound expressions formed from those variables? A casual reader may assume that in the context of the principle, they too may take only those two values, but it has also been read as allowing them to take other integer values (see e.g. Burris & Sankappanavar [13], Burris [8, 9]).

It turns out that one's reading of the $\{0, 1\}$ principle is linked with one's understanding of the operations used when forming compound expressions, and in the following section we shall see that those operations are not fixed, but vary with the *stage* in a problem-solving routine in they appear. The $\{0, 1\}$ principle takes different forms according to which stages it is linking.

3 Problem-solving routines and their stages

When one asks how exactly Boole understood his operations of multiplication, addition and subtraction, it is all too easy to presume that each of them was assigned a fixed meaning. But that, we shall argue, was not the case. For him, doing logic was essentially a matter of solving logical problems using quasi-algorithmic routines.[4] He had different routines for different kinds of problem but, we contend, they are all made up of the same stages, and the nature of each operation varies according to the stage. To know what is meant by an operation, and in particular its domain of application and possible values, one thus needs to know at what stage of a given routine it is being applied.[5]

[4] This point has been made by several authors, e.g. Peckhaus [31, p. 18] Wilson [38, pp. 522-530], Waszek & Schlimm [37, *passim*], although it is left unmentioned in many presentations of Boole's work, and in the general histories of Bocheński [2, §40] and Kneale [25, pp. 404–20].

[5] This view is not standard among commentators. Expositions of Boole's routines do often break them down into various steps or stages (see e.g. Wilson [38, pp. 527–8], Brown [7, pp. 321ff], Burris [9, §6.1], Waszek [36, §5.1]) but, as far as the present author is aware, none have systematically correlated the steps with shifts in the identity of the operations themselves, and they tend to assume that the operations have a principal identity on which one should focus. In particular, modern algebraists such as Hailperin, Brown and Burris find the centre of gravity of Boole's system in the operations of what we call Stage 3; that is, the familiar operations of multiplication, addition, and subtraction with variables taking values only in $\{0, 1\}$ but with compound expressions ending up with any positive or negative integers as values.

Stage 1: Symbolizing the data in terms of classes

Boole usually presents his logical problems in words, and the first step in all his routines is to symbolize their data. This is done using one or more equations with variables $x, y, z, \ldots$ standing for classes, the constants 0, 1, and with compound expressions featuring three operations of what he calls joint application, gathering together, and separation of classes (LT pp. 27-34). For the first of these operations there is no special sign: the output is conveyed by juxtaposition, representing what we would now call the meet (or intersection) of classes x and y, and no restrictions are placed on its application. But for gathering two classes together, and for separating one from another, certain constraints are imposed. These constraints are syntactic in so far as they concern the admissibility of expressions, but they use criteria that are semantic, requiring that certain classes are disjoint or that one is included in another. They are thus not constraints on bare expressions, but constraints on expressions under given interpretations of their variables.

Specifically, compound signs $x+y$ are allowed as expressions in Stage 1 only when class x is disjoint from class y. This constraint is both semantic and contingent, since the admissibility of an expression $x+y$ thereby depends on what classes one happens to take the variables to stand for. For example, $x + y$ is a permissible expression when x stands for the class of penguins and y for the class of parrots, but not when x represents the class of poets and y the class of parsons. The restriction is meant recursively: when A, B are themselves complex expressions, then again $A + B$ is allowed only if A is disjoint from B; thus, $(x + y) + z$ is admitted only if x is disjoint from y and also $x + y$ is disjoint from z. When the restriction on admissibility is satisfied, the expression $x + y$ represents the class whose members are just those items that are in exactly one (equivalently in that context, at least one) of x, y (LT pp. 66-7).

Analogously, compound signs $x - y$ are allowed in Stage 1 only when class y is included in class x. Again the restriction is meant recursively: for example, $(x-y)-z$ is admitted only if y is included in x and also z is included in $x - y$. Semantically, $x - y$ represents the class whose members are just those items that are in x but not in y, so that it coincides (when the restriction is satisfied) with set difference as we know it today.

Boole says little about why he makes these restrictions but, at least in LT, it was not because of any belief that they correspond to ordinary English usage. On the contrary, in the case of addition he acknowledges that "the *jus et norma loquendi* seems rather to favour" a reading of the class of y's or x's "to include things that are y's and x's at the same time, together with things that come under the one, but not the other" (LT p. 56). To be sure, in a late manuscript fragment he takes an opposite view, saying that "Such conditions of interpretability are really implied in ordinary language" (Grattan-Guinness & Bornet [21, p . 183n]); but, at least at the time of writing his seminal works, the motivation lay elsewhere. We suggest that it was formal, issuing from what he does in later stages of his routines.

Stage 2: Re-reading the operations in the domain $\{0, 1\}$

As soon as the data for a logical problem have been expressed in the form of one or more equations, Boole is ready to re-read them. In Stage 2, the variables $x, y, z \ldots$ are allowed to take as possible values just 0 and 1, which are understood as the familiar numbers zero and one, rather than truth-values as a modern reader might expect (Gastaldi [19, pp. 238–141]). Each of $xy, x + y, x - y$ is also re-read as an the output of an arithmetic operation, full or partial, defined for those values of x, y alone.

Since the set $\{0, 1\}$ is closed under ordinary multiplication, xy is understood in Stage 2 as arithmetic product. Nevertheless, the passage from classes in Stage 1 to 0, 1 in Stage 2 affects the non-Horn laws that hold for multiplication: for example, while $zx = zy$ and $z \neq 0$ together imply $x = y$ in the latter context, they do not do so in the former one. Indeed, the same step alters the status of certain non-Horn operation-free laws that Boole does not mention; for example, $x \neq y \neq z$ implies $x = z$ in Stage 2, although it does not do so in Stage 1.

On the other hand, since the set $\{0, 1\}$ is not closed under ordinary sum and difference, in Stage 2 Boole treats $x + y$ and $x - y$ as only partial operations: the former is undefined when $x = 1, y = 0$ and the latter is undefined when $x = 0, y = 1$. Again, little is said about the grounds for choosing this option as contrasted with taking them as full operations $\{0, 1\}^2 \to \{0, 1\}$ as is done in arithmetic modulo 2 and in Boolean algebras today, but they will become apparent as we continue into Stage 3.

One way of reading Boole's $\{0, 1\}$ principle is as connecting Stage 2 to Stage 1, that is, as relating equations about 0,1 to equations about classes. To state it with reasonable precision, we refer to expressions formed from variables $x, y, z \ldots$ and the constants 0, 1 by means of the two-place connectives $\cdot, +, -$ (without semantic interpretation or syntactic constraints) as *basic expressions*, and equations between them as *basic equations*. We say that a basic equation is *partially valid for* $\{0, 1\}$ iff there are no values of the variables in $\{0, 1\}$ for which all expressions on the LHS and RHS are well-defined under their Stage 2 reading, but LHS $\neq$ RHS. Similarly, call an equation of the kind allowed in Stage 1 *partially valid for classes* iff for every non-empty domain U of discourse, there are no values for the variables taken as subclasses of U for which all expressions on the LHS and RHS are well-defined under their Stage 1 reading, but LHS $\neq$ RHS. Then the first form of the $\{0,1\}$ principle tells us that a basic equation is partially valid for $\{0,1\}$ iff it is partially valid for classes.

Stage 3: Allowing output values beyond 0, 1

So far, Boole's routines begin by taking a logical problem that is given in ordinary language, symbolizing the data as equations interpreted in terms of classes (Stage 1), then re-interpreting the equations in the domain of the numbers 0, 1 (Stage 2). But in Stage 3, the value-range of the basic operations is extended: ordinary arithmetical addition and subtraction are permitted even when they take us from numbers in $\{0,1\}$ to integers outside $\{0,1\}$. For example, we are allowed to read $1+1$ as 2 and $0-1$ as -1. However, variables must still be given values in $\{0,1\}$.[6]

This gives rise to a second reading of the $\{0,1\}$ principle, which Boole does not clearly distinguish from the first one, and which is emphasized by Burris op. cit. It relates the numerical operations of Stage 3 to those of Stage 2, as follows: If a basic equation is valid under the ordinary arithmetic understanding of its operations but with all values of its variables taken in $\{0,1\}$, then it is partially valid in $\{0,1\}$ under the partial operations of Stage 2. Unlike the first version, its converse fails.[7]

Putting the two versions of the $\{0,1\}$ principle together evidently links Stages 3 and 1: if a basic equation is valid for all values of its variables taken in $\{0,1\}$, under the ordinary arithmetic understanding of its operations in the integers, then it is partially valid for classes. This version would perhaps be better called the integer-class principle, since it does not directly concern the set $\{0,1\}$. But articulating it helps explain why, back in Stage 1, Boole doggedly insisted on treating the addition

[6] From the point of view of logic today, this restriction is awkward because it creates a conspicuous failure of what we now call the rule of substitution: while $xx = x$ is counted as valid in Stage 3, its substitution instance $(x+y)(x+y) = (x+y)$ fails, since when $x = 1 = y$ then $x+y = 2$, so that $(x+y)(x+y) = 4 \neq (x+y)$. If we were to drop that restriction in its formulation, the second version of the $\{0,1\}$ principle would evidently remain true but it would be rather weaker since, as Boole was acutely aware (e.g. LT pp. 92–3, 157), the equalities $xx = x$ and $x(1-x) = 0$ are valid for the integers when the variable x is confined to $\{0,1\}$ but fail when x is allowed to take other integer values, e.g. 2. Boole saw the equalities $xx = x$ and $x(1-x) = 0$ and equivalent to each other (distribute the LHS of the latter and add xx to each side) and as fundamental to logic (see e.g. LT p. 49); for him, they present the most conspicuous difference between the laws of thought for classes and the common laws of number.

[7] Two simple counterexamples are $x+x = 0$ and $x+x = x$: both are partially valid in $\{0,1\}$, but none is valid in ordinary arithmetic even for the value $x = 1$. Together, they illustrate the awkward fact that partial validity of equations for $\{0,1\}$, as also for classes, is not transitive We note in passing that the two examples are quite different in character: if we re-read $+$ as a full operation on $\{0,1\}$, then the first equality becomes valid in $\{0,1\}$ under an exclusive reading (where $1+1 = 0$) and invalid under an inclusive one (where $1+1 = 1$), the second has the reverse behaviour.

and subtraction of classes as only partial operations, in the face of ordinary usage. He dearly wanted his logic of classes to preserve some kind of validity for all familiar, valid, equations of arithmetic; the only way to do that was to treat the operations on classes as partial, understand validity for class equations as likewise partial, and make use of this third version of the $\{0, 1\}$ principle.

Stage 4: Introducing division

In Stage 4, division is permitted as an operation on pairs of integers, not only as a partial operation defined when the divisor is distinct from zero, but as a full operation including that case. In particular, one may work with the quotient expressions $1/0$ and $0/0$ as well as with others such as $-1/0$ and $2/0$.[8] However, Boole gives no hint of a version of the $\{0, 1\}$ principle for this broadened context, nor does one seem possible.

For Boole's contemporaries, the freedoms created by Stages 3 and 4 met with increasing levels of criticism. For Stage 3, integers other than 0 and 1 have no visible interpretations as classes, but that could perhaps be tolerated. For Stage 4, division does not correspond to any recognizable operation on classes and takes us beyond the limits of the $\{0, 1\}$ principle, raising more doubts. Moreover, the legitimacy of division by zero was contested even for familiar number systems. As we shall see in section 5, the purpose of Boole's manuscript 'On the nature of thought' was to meet these criticisms by dispensing entirely with division.

Stage 5: Back to $\{0, 1\}$

Stage 4 typically outputs equations of the form $X = c_1Y_1 + \ldots + c_nY_n$ where X and the Y_i are expressions that are readable in the domain $\{0, 1\}$ under the conventions of Stage 2, while the c_i are certain purely numerical 'coefficients'. Stage 5 works on those coefficients so as to obtain a RHS that is again readable in the domain $\{0, 1\}$ and thus also (Stage 6) in the domain of classes and in ordinary language. The changes of operation after Stage 4 in effect reverse those effected before it. In practice, in the sample routines worked through in LT, they are traversed rapidly and tend to bleed into each other.

Specifically, the coefficients c_i of the equation $X = c_1Y_1 + \ldots + c_nY_n$ that need to be dealt with in Stage 5 are $1/1, 0/1, 0/0$ together with a notorious fourth group,

[8] There is an element of vagueness here, since Boole does not make it clear how far quotients, and in particular those with denominator zero, can in turn serve as inputs for operations. For example, while in Stage 4 he works with expressions $1/0, 0/0$, it is not clear whether, say, the product $(1/0)(0/0)$ and quotient $(0/0)/(1/0)$ are also legitimate expressions. Since Boole does not need them, he does not talk about them; nor will we.

which contains 1/0 but also, on occasion, other numerical fractions such as 1/2 (e.g. LT p. 95), $-1/0$ (e.g. LT p. 94), and plain numbers such as 2 (e.g. LT pp. 125, 127).

Eliminating the first two fractions 1/1 and 0/1 is hardly a problem: as one leaves Stage 4 one need only note that $(1/1)Y = 1Y = Y$ and $(0/1)Y = 0Y = 0$, and effect the replacements. But the others are less straightforward. Boole treats 0/0 as a variable v that "admits of the numerical values 0 and 1 indifferently" (LT p. 91), with implicit existential quantification. In contrast, all terms c_iY_i with coefficient c_i in the fourth group are simply deleted, on the obscure and unconvincing ground (which refers only to the most frequently occurring item in the group) that 1/0 "is the algebraic symbol of infinity" and so fails the law $a(1-a) = 0$ that holds in $\{0, 1\}$ (LT p. 91).

Clearly, Boole's intuitions for dealing with coefficients other than 1/1 and 0/1 far outrun his justifications. But, by manhandling them in this manner, he gets back to an equation that is readable in the domain $\{0, 1\}$ under the conventions of Stage 2.

Stage 6: Back to classes and words

Finally, Stage 6 re-reads the output equation of Stage 5 in terms of classes, with the indefinite symbols v that were introduced in Stage 5 now ranging over all subclasses of the universe. This class-equation, implicitly existentially generalized with respect to v, is then translated into the same everyday language as the initial data in Stage 1, completing the cycle.

Figure 1. Six stages in Boole's problem-solving routines

Stage 1: classes (expressed in words, then symbols)

Stage 2: 0, 1

Stage 3: integers (positive, negative, and zero)

Stage 4: integers and quotients of integers

Stage 5: 0, 1, indefinite symbols

Stage 6: classes (expressed in symbols, then words)

The entire process is pictured in Figure 1. The indentation levels indicate 'how far' each stage has gone from the verbal starting/end points; the annotations recall the ranges beyond which the operation-outputs may not pass. Note the asymmetry between the outward and return halves, due to the elimination in one fell swoop of both quotients and all integers other than 0, 1 when passing from Stage 4 to Stage 5.

Boole did not articulate this six-stage segmentation explicitly when describing his general methods. But it corresponds to his practice when solving specific problems. To illustrate, we transcribe a very simple example from LT pp. 89–90. The punctuation is as in the original but we transcribe quotients with an oblique bar rather than a horizontal line and place our comments between braces; the suspension points are for omitted text. Due to its simplicity, the example passes directly from Stage 2 to Stage 4, and from Stage 4 to Stage 6, skipping Stages 3, 5.

> [**Data and goal**] Let us take the Proposition "Men not mortal do not exist;" represent this Proposition by symbols; and seek, in obedience to the laws to which those symbols have been proved to be subject, a reverse definition of "mortal beings," in terms of "men."
>
> [**Beginning of Stage 1**] Now if we represent "men" by y, and "mortal beings" by x, the Proposition "Men not mortal do not exist," will be expressed by the equation
>
> $$y(1-x)=0,$$
>
> from which we are to seek the value of x. [**Transition from Stage 1 to Stage 2**] Now the above equation gives
>
> $$y-yx=0, \text{ or } yx=y.$$
>
> Were this an ordinary algebraic equation, we should, in the next place, divide both sides of it by y. But it has been remarked in Chap. II. that the operation of division cannot be *performed* with the symbols with which we are now engaged. [**Transition from Stage 2 to Stage 4**] Our resource, then, is to *express* the operation, and develop the result by the method of the preceding chapter. We have, then, first,
>
> $$x=y/y,$$
>
> and, expanding the second member as directed,
>
> $$x=y+0/0(1-y).$$
>
> This implies [**Transition from Stage 4 to Stage 6**] that mortals (x) consist of all men (y) together with such a remainder of beings which are not men ($1-y$), as will be indicated by the coefficient 0/0. Now let us inquire what remainder of "not men" is implied by the premiss ... the expression 0/0 here indicates that *all*, *some*, or *none* of the class to whose expression it is affixed must be taken.

In LT, Boole works through many other examples that can be decomposed into the same six stages. Examples 2–4 on pp. 94–98 are fairly simple. More elaborate ones, where Stages 3 and 5 become apparent, include Example 1 on pp. 118–120, Example 2 on pp. 125–9 (with a confused treatment of relations such as geometric similarity, as classes just like the class of triangles), and Examples 1–5 on pp. 134–149. Of these, the very last became a 'test case' for later nineteenth-century authors wishing to demonstrate the superiority of their methods over those of Boole (see Brown [7, p. 305] for references). In each of the examples, the pattern of stages, with shift in operations in the progression from stage to stage, is essentially the same.

Our conclusion is that Boole did not have a unique way of reading his basic operations. Their domains of application and their possible values, as well as the laws that are valid (or partially valid) for them, vary according to the stage of the routine where the operations are used. According to the stage, we can be looking at partial class operations (Stages 1, 6), partial operations on $\{0,1\}$ (Stage 2, 5), operations which, while originating in $\{0,1\}$ may take other integers as values (Stage 3), the same but with applications of division as a full operation (Stage 4). Versions of the $\{0,1\}$ principle serve as bridges between Stages 1, 2, 3, as well as between Stages 5 and 6 (but not between Stage 4 and its neighbours) while the *ad hoc* instructions for eliminating and interpreting coefficients force passage from Stage 4 to 5.

4 The dark tunnel

Memorably, Jevons [23, p. 75] referred to Boole's calculations as leading us through "dark and symbolic processes". Echoing his words, we refer more specifically to Stages 3 and 4 of the routines described above as Boole's 'dark tunnel', in which the operations of addition, subtraction and division become obscure.

In the cases of addition and subtraction, the obscurity is easily lifted. As seen in the preceding section, the only ground for treating the operations of Stages 1 and 2 as partial is to retain as partially valid all those equations that are valid over the integers in the sense of Stage 3. As already realized by Jevons [23], Peirce [29] and others in the 1870s (see the preface of Peirce [30]), once that consideration is given less importance and one is willing to develop a logic of classes with its own laws, there is no obstacle to taking addition and subtraction as defined for all values of their arguments. When x, y are classes, we can legitimately take $x + y$ as what we now call inclusive union; and when x, y are elements of $\{0,1\}$ we can take it to be the full operation $max(x, y)$ on that set. Similarly for $x - y$. If one does that from the beginning, the operations of Stages 1 and 2 become full and Stage 3 can be re-read as using exactly the same operations as Stage 2 on $\{0, 1\}$.

But for division, introduced in Stage 4, the problem is much more refractory. Recall that even in the context of the rational or real numbers, if we define x/y by inversion as 'the unique' z such that $yz = x$, we have an 'almost full' operation with x/y defined everywhere except for the unique divisor $y = 0$. In that context, the undefined case can be treated as a limiting one; but in the domain $\{0, 1\}$, it makes up half of the four possible values for the argument-pair. Since Boole makes essential use of the fractions 1/0 and 0/0 in Stage 4, he has a serious problem on his hands; for division, the darkness is not just an artefact of an unnecessary restriction.

Notoriously, in his published work Boole dodged the problem. At one point, he claims that if division by zero is not allowed then "it is manifest that no such thing as a general method in Logic is possible" (LT p. 67), and boldly asserts that "it is an unquestionable fact that the validity of a conclusion arrived at by any symbolical process of reasoning, does not depend upon our ability to interpret the formal results which have presented themselves in the different stages of the investigation" (LT pp. 67–8). He found comfort in the fact that the British school of mathematicians in which he had been brought up routinely did similar things in algebra, analysis and geometry (see e.g. Rice [33], Wilson [39]), and he did so himself for certain 'inverse' functions in his work on differential equations (see e.g. Waszek [36, §5.3]).

But it is not surprising that this created an air of mystery and sleight-of-hand that he was never able to dispel. The mystery became even greater as corresponding practices of British algebraists and analysts accumulated criticism. In a late manuscript, perhaps prepared at the prompting of Jevons, Boole finally attempted to do better. We now examine that attempt.

5 Boole's manuscript attempt to avoid the dark tunnel

In a letter to Jevons dated 17 August 1863 (Grattan-Guinness [20, p. 26]), Boole said that he was quite sure that one can always reformulate his procedures to ensure preservation of interpretability.

> It is certainly possible to work out logical problems according to the general laws developed in my work, *with such added restrictions* as shall make all intermediate results interpretable. I have somewhere laid by, a paper on this subject which I wrote about two years ago. I cannot at this moment put my hand upon it, but I remember that it involved the application of such instructions as should make the elementary operation

> always interpretable in ordinary language. Thus $x - y = 0$ would become $x - xy = 0$, and so on.

A surviving manuscript finally edited and published with an introductory guide by Waszek [36], attempts to do just that. Entitled ‘On the nature of thought’, it appears to have been written shortly after the above letter to Jevons, and is perhaps based on the text there mentioned (op. cit. §3.2). In it, Boole seeks to reformulate his principal routines using only what in section 3 above we have called basic expressions.

He managed to do so successfully for the important tasks of developing or expanding an expression (NT B.29–32), a precursor of what we know today as perfect disjunctive normal form; putting equations into normal form LHS $= 0$ (NT B.33–8); and elimination, which extracts from an expression information that omits one or more of its variables (NT pp. B.38–41). These reformulations are also discussed in detail in section 3.3 of Waszek's introductory guide to the manuscript. So far so good. But trouble arises when Boole turns to another kind of problem (NT B.42):

> From any system of equations, to deduce the complete expression of any one of the symbols x in terms either of all the others or of any portion of them and to interpret the result.

We will refer to this as the *find x problem.* Taking a little more care with the syntax than Boole does, we can put it as follows. Recall from section 3 above the notion of a *basic expression*: one formed from variables $x, y, z, \ldots$ and the constants $0, 1$ by means of the two-place connectives $\cdot, +, -$ without semantic interpretation or syntactic constraints. Take *indefinite expressions* to be like basic expressions except that they also contain one or more ‘indefinite symbols’ $v, w, \ldots$. And recall from sections 2 and 3 above that indefinite symbols served as a clumsy, pre-Fregean way of expressing existential quantification. Then the ‘find x’ problem may be formulated thus:

> Given an equation $X = 0$, where X is a basic expression containing x, find a basic or indefinite expression Z not containing x such that $x = Z$.

Boole's solution to this problem is: $x = A'B + vA'B'$, where A, B are basic expressions without x, A' is our abbreviation of his $1 - A$, and v is an indefinite symbol. Adding the implicit existential quantification, the solution becomes, in more modern language: $x = A'B + vA'B'$ for some v.

To reach his solution, he begins by using his routine for development, which he has successfully justified without division, to write X as $Ax + Bx'$ where A, B are basic expressions without x, so that we have the equality $Ax + Bx' = 0$, which we will call the *datum.* From this, Boole infers by transformations still justifiable without division that $(AB' + A'B)x = B$. Then he isolates x on the left by **dividing** both sides by $AB' + A'B$:

$$x - B/(AB' + A'B).$$

Applying development (again effected justified without division) to the RHS of this equality, he gets:

$$x = sAB + tAB' + uA'B + vA'B'$$

for some coefficients s, t, u, v of the kinds described in our account of Stage 5 (section 3 above). Breaking the argument into cases by considering the four ways of assigning $0, 1$ to A, B and implicitly appealing to the $\{0, 1\}$ principle, he infers from those two equalities that $s = 1/0, t = 0/1, u = 1/1, v = 0/0$. For example, when $A = 1 = B$ the second of the above equations gives us $s = x$ while the first gives us $x = 1/(0+0) = 1/0$; similarly for the others. Thus he has the 'pre-solution'

$$x = (1/0)AB + (0/1)AB' + (1/1)A'B + (0/0)A'B'.$$

Using his conventions for interpreting the terms with coefficients $1/0$ and $0/0$ in Stage 5 (section 3), the coefficient $1/0$ on the first term tells us that AB "does not exist" (i.e. is empty), and the coefficient $0/0$ on the last term is tantamount to an 'indefinite symbol'. That gives him the equality:

$$x = A'B + vA'B'$$

where, implicitly, v is existentially quantified. Complementing this result, Boole notes (without appeal to division) that the datum $Ax + Bx' = 0$ immediately implies that $Ax = 0$ and $Bx' = 0$, which together imply that $AB = 0$ (which in LT e.g. p. 108 he had called an 'independent relation' and in later literature also came to be known as the 'condition' or 'resultant' of the solution). He seems to recognize that while $x = A'B + vA'B'$ taken alone does not suffice to recuperate the datum, it does so when taken together with $AB = 0$.

Thus, just as in LT, the argument elaborated in 'The nature of thought' continues to apply division to isolate the 'unknown' variable x on the left side of an equation, creating a quotient expression on the right. Once that is done, no further applications of division are needed, but in the end one needs to 'interpret' the numerical fractions $1/0$, $0/0$ thus created. The assurance that one can do without division, given to Jevons in the letter of 17 August 1863, is left unsubstantiated.[9]

[9] Despite its failure to eliminate division, Boole's discussion of the 'find x' problem in NT

Subsequently, proofs of the equivalence of $Ax + Bx' = 0$ with the solution $x = A'B + vA'B'$ conjoined with its 'equation of condition' $AB = 0$ were offered by Schröder [34, 35] without using division, and it interesting to compare them briefly with that of Boole.

Schröder [34, pp. 20–22] presents Boole's solution of the 'find x' problem (his Theorem 20,) as the *Haupttheorem* of the entire calculus of logic. He proves it without appealing to division, by a rather tortuous argument that makes use of two indefinite symbols, their complements, and the expression of one of them in terms of the other (Pollard [32, p. 315 n.23] carefully disentangles the existential quantifications that are implicit). There is no appeal to the $\{0, 1\}$ principle, nor even to what he calls "cumbersome development schemes" (Schröder [34, p. 20], Pollard [32, p. 313]).

However, influenced by the central role of the inclusion relation in work of Peirce and MacColl, Schröder later gave a quite different proof. The posthumously published volume Schröder [35, §52 p. 448–9] expresses the solution (taken along with its adjunct) $AB = 0$ in the much more perspicuous form $B \subseteq x \subseteq A'$. This formulation permitted him to bypass not only division, the $\{0, 1\}$ principle, and development (as before), but also indefinite symbols, with a very succinct, elegant, and reversible argument: given $Ax + Bx' = 0$ we have $Ax = 0$ and $Bx' = 0$, so $x \subseteq A'$ and $B \subseteq x$, so $B \subseteq x \subseteq A'$; end of story. This presentation was echoed in the outline of Couturat [15, §30] (who called it "Schröder's Theorem") and the survey of Lewis [26, p. 146 observation 7.2].

introduces some interesting conceptual novelties. At one point it breaks the argument down into four cases, according to four possible combinations of the values 0 and 1 for expressions A, B, justifying the procedure by the $\{0, 1\}$ principle — a technique of proof rarely if ever used in LT. The account of fractional coefficients in NT is also less brutal. Whereas in LT the fractions $0/1, 1/1, 0/0$ were read numerically in Stage 5 and eliminated there before passing to classes in Stage 6, in NT B.45–46 Boole proposes reading them directly in terms of classes. Specifically, $0/1$ is read as the empty class 0, because that is the unique class which when intersected with the denominator gives the numerator, and $1/1$ is read as 1 for the same reason. In a similar spirit, $0/0$ is read as "any class" because any class intersected with the denominator gives the numerator (NT B.45–46). The ground for deleting terms with coefficient $1/0$ is also explained (rather obscurely) in the language of classes: Boole tells us that the case $A = 1, B = 1$ generating that coefficient "... implies not simply the nonexistence of any individuals of the class AB in the class x but the impossibility of conceiving the class AB as existing at all compatibly with the relations expressed in the actual data" (NT B.46).

6 Why didn't Boole's attempt succeed?

Why was Boole unable to avoid division in 'The nature of thought', despite his announced intention? Of course, it is impossible to read his mind, but there seem to have been several factors at work.

On a general level, one may suspect that, at bottom, he was not highly motivated to do so. The manuscript was written at the urging of Jevons and others to meet widespread misgivings about division but, even then, he continued to see the procedure as perfectly legitimate; eliminating it was merely a concession to the mathematically unsophisticated (see e.g. NT B.43, other parts of the letter to Jevons cited in section 5 above and the pronouncements in LT pp. 67–8 cited in section 4 above).

Again, NT was a draft text, which Boole seems to have put aside to complete work for a second edition of his *Treatise on Differential Equations* (letter to Jevons of 30 January 1864, in Grattan-Guinness [20, p. 32]). Perhaps he intended to iron out this remaining wrinkle at a later date but was prevented by his death.

On a technical level, however, the main obstacle seems to have been his strict confinement to equational reasoning which, while giving him sharp focus, cramped his style in several ways.

A cumbersome notation to work in. As is well known, the restriction forced him to express class inclusion $x \subseteq y$ indirectly as an equality. The simplest way of doing so is as $x = xy$, which Boole used in MAL p. 21 to express the traditional 'All Xs are Ys' – although, curiously, he seems to have settled on the more cumbersome version $x = vy$ for that purpose, where v is an indefinite symbol (see LT p. 61), and when finally he speaks of 'inclusion' between classes in the manuscript NT B.19, he continues to use the $x = vy$ formulation.[10] This policy cut Boole off completely from the kind of reasoning used by Schröder 1905, expressed directly in terms of inclusion.

A narrow view of what counts as a 'solution' of a problem. When solving equations, there is a strong presumption that any solution in which x occurs on both left and right is "mathematically objectionable", as Lewis [26, p. 149] describes it. This perspective disbars any account of the 'find x' problem that allows x in the solution itself. But once one appreciates the implicit existential quantification of

[10] The restriction to equational reasoning not only led Boole to introduce 'indefinite symbols' into his system, it also discouraged any attempt to make explicit the existential quantifiers implicitly associated with them, since introducing special operations for that purpose would destroy the purity of the equational format. One may suspect that adhesion to that format was, at bottom, one of the reasons for the long gap between the publication of MAL in 1847 and the articulation of an adequate account of the quantifiers in Frege [18] and Peirce [30].

the indefinite symbol v in Boole's solution $x = A'B + vA'B'$, one realizes that this solution follows immediately, by what we now call the EG or $\exists^+$ rule, from the equality $x = A'B + xA'B'$, in which x occurs on both sides. That simple fact suggests another proof, rivalling Schröder's in its succinctness but in equational language.

> Given $Ax + Bx' = 0$ we infer that $Ax = 0$ and $Bx' = 0$. Clearly, $x = x(AB + AB' + A'B + A'B') = xAB + xAB' + xA'B + xA'B'$. Now since $Ax = 0$ we have $xAB = 0 = xAB'$, and since $Bx' = 0$ we have $Bx = B$, so $xA'B = A'B$, so that $x = A'B + xA'B'$.

Like Schröder's [35] argument, this appeals neither to the $\{0, 1\}$ principle, nor to "cumbersome development schemes". Moreover, it turns out that the equality $x = A'B + xA'B'$ not only implies, but is also implied by the existence of a v such that $x = A'B + vA'B'$, so that the two versions of the solution are equivalent. For suppose $x = A'B + vA'B'$; it suffices to show that $vA'B' = xA'B'$. By the supposition we have $xA'B' = A'B'(A'B + vA'B') = A'B'A'B + A'B'vA'B' = vA'B'$.

Note that the proof of equivalence of these two versions of the solution-equality does not depend on the problem hypothesis $Ax + Bx' = 0$. It tells us that for arbitrary expressions $A, B, x = A'B + xA'B'$ iff there is a v with $x = A'B + vA'B'$. Roughly speaking, *the indefinite symbol on the right of the solution tells us no more or less than the unknown variable itself.* [11]

On a procedural level, Boole tended to follow the familiar strategy of isolating the unknown variable on the left as soon as possible, and it was precisely that step that forced him to apply division, transforming $xY = Z$ into $x = Z/Y$. For example, in the NT solution to the general 'find x' problem (see section 5 above), it is just at that point that division is applied. The same pattern occurs in the problem-example in LT pp. 89–90, as well as in the other examples from LT mentioned in section 3. In each of them, an act of division appears as the natural (and perhaps the only) way to get rid of everything on the left of the equation other than the 'unknown' variable.

Under-exploitation of the $\{0, 1\}$ *principle.* Boole set great store by the $\{0, 1\}$ principle, describing it as "a fact of great moment and significance" and "the basis of the Calculus of Logic" (Grattan-Guinness & Bornet [21, pp. 94, 182]), but its position in his work changed over time.

[11] Essentially this fact was noticed by Lewis [26, pp. 146–151]. More specifically, he observed that a variant of Boole's solution due to Poretsky, $x = A'v + Bv'$ where v is an indefinite symbol, is equivalent to $x = A'x + Bx'$. While he appreciated that the latter "is in reality a useful and logically simple form of the solution", he did not see it as a way of dispensing with arbitrary symbols, and regarded the former kind of solution as "the most 'mathematical' form".

In MAL it is articulated (in a rather vague form that seems to correspond to the second of the two versions distinguished in section 3 above) and is justified for the special case of expressions containing a single variable, with a remark that the argument "may be extended to functions of more than one symbol" (MAL pp. 60–2). The justification appeals to development, which in turn was obtained by an absurdly baroque argument appealing to Maclaurin's/Taylor's Theorem on power series in algebra.

In response to criticisms of that strategy, in LT he proceeded in the reverse direction, validating development using the $\{0,1\}$ principle and treating the latter as unproven and fundamental (LT p. 70ff).[12] Indeed, a later manuscript remark seems to suggest that one could do no more: "it is a conclusion founded on actual comparison and which apparently could not have been predicted" (Grattan-Guinness & Bornet 1997 p. 182).[13] But once development is proven in LT, it again shoulders all the heavy work that it had been doing in MAL; the $\{0,1\}$ principle is retired from action, reappearing mainly in chapter X to simplify some of the calculations of preceding chapters. Boole did not exploit the possibility of using the $\{0,1\}$ principle as a master tool for quick proofs by cases of facts about classes, replacing long-winded applications of development.

Boole's solution to the 'find x' problem in NT does apply the $\{0,1\}$ principle – but only as a last resort. Its application is delayed to Stage 4, *after* the act of division

[12] There is an intimate connection between the $\{0,1\}$ principle and development; one might even say that at some deep level, they are the same. But there are also differences in the ways they are deployed. On the one hand, development is applied to a given expression and provides another expression such that the statement of their equality is valid; typically, this is then chained with other equalities. On the other hand, the $\{0,1\}$ principle is applied to a given equation (or to the implication of an equation by one or more others) to determine whether it is valid; typically, it breaks the argument into 2^n cases according to the possible choices of values in $\{0,1\}$ of n component variables (or suitably chosen compound expressions).

[13] We recall in passing a simple, direct proof of the $\{0,1\}$ principle (in the first of the versions distinguished in the text, linking Stages 1 and 2), sketching the non-trivial half, with minimal notation, as follows. For basic expressions Y, Z (as defined in the account of Stage 2, section 3 above), suppose that for some valuation of the variables as classes, the values of both Y and Z are well-defined and $Y \neq Z$. Then there is an i in the value of Y but not in the value of Z or conversely; say the former. Define a valuation into $\{0,1\}$ by giving each variable the value 1 if i is in that variable's class-value, and 0 if i is not in its class-value, and induce on the complexity of the expressions to conclude that Y receives value 1 while Z receives value 0. Evidently, this argument uses a principle of extensionality for classes and 'digs inside' them in a manner quite foreign to Boole's equational perspectives in LT. It is interesting to note, however, that in NT he proposed an extensional definition of identity between classes: "The sign will be used … to denote that any two expressions between which it is placed represent classes of things which are identical as respects the individuals of which they consist…" (NT B.17). But he still did not use this definition as a tool of proof.

that got x alone on the left. But by advancing its application, one immediately obtains the following argument that dispenses with both division and development.

> Suppose as datum that $Ax+Bx' = 0$. Consider any valuation of variables in $\{0,1\}$ that satisfies this datum. By the $\{0,1\}$ principle, it suffices to show that the valuation also satisfies Boole's solution, written as $\exists v : x = A'B + vA'B'$. Consider four cases according to the values of A,B.
>
> *Case 1.* $A = 1, B = 1$. This case is inconsistent with the datum, for if $x = 1$ then $Ax = 1$ so $Ax + Bx' = 1 \neq 0$; while if $x = 0$ then $Bx' = 1$ so $Ax + Bx' = 1 \neq 0$.
>
> *Case 2.* $A = 1, B = 0$ By the datum $Ax = 0$, so $x = 0$. Also $A' = 0$, so for any choice of v we have $A'B = 0 = vA'B'$ and thus $A'B + vA'B' = 0$. Thus $\exists v : x = A'B + vA'B'$ as needed.
>
> *Case 3.* $A = 0, B = 1$. By the datum $Bx' = 0$, so $x' = 0$ so $x = 1$. Also $A'B = 1$ while $vA'B' = 0$, so $A'B + vA'B' = 1$ for any choice of v. Thus $\exists v : x = A'B + vA'B'$ as needed.
>
> *Case 4.* $A = 0, B = 0$. Then $A'B = 0$ and $A'B' = 1$. So we have $x = A'B + xA'B'$, so $\exists v : x = A'B + vA'B'$ as needed.

These four cases correspond to the four terms (in the same order) in Boole's 'pre-solution' $x = (1/0)AB + (0/1)AB' + (1/1)A'B + (0/0)A'B'$ for the 'find x' problem, and also to the four cases that he considers (in a different order) *after* having applied division and development (NT B.45-6). Roughly speaking, by advancing application of the $\{0,1\}$ principle and breaking the argument into cases one eliminates the need for both division and development.[14] When so deployed, the $\{0,1\}$ principle (in the first of the three forms distinguished in section 3) functions like a truth-table; bearing in mind its verification in footnote 13, one may think of it as an 'element-table'.

Summarizing this section, we can say that Boole's strict confinement to equational reasoning created obstacles to the elimination of division from his problem-solving procedures. Relaxation of that constraint in any one of three ways would

[14] The same procedure may be carried out in the problem examples mentioned in section 3. For instance, in the simple one reproduced from LT pp. 89-90, we are given the datum $y(1-x) = 0$ and asked to solve for x. Boole gets the unknown x on the left by expressing the datum as $yx = y$, and then dividing both sides by y to obtain $x = y/y$; to which he applies development to end up with the solution $x = y + vy'$. But if we apply the $\{0,1\}$ principle (in its first version) from the outset, we consider four cases according to values of x, y in $\{0,1\}$ that satisfy the datum. *Case 1.* $y = 1$. Then by the datum $x = 1$ and also $y + vy' = 1$ for arbitrary choice of v, so that $\exists v : x = y + vy'$. *Case 2.* $y = 0$. Then $xy = 0 = y$. But $x = xy + xy' = y + xy'$, so $\exists v : x = y + vy'$. Division is obviated and development is replaced by a corresponding consideration of cases.

have permitted him to complete the task without difficulty: making use of the inclusion relation, allowing the unknown to occur on the right hand side of a solution, or applying the $\{0, 1\}$ principle in a more determined way, early in the problem-solving process.

References

[1] Bentham, G. 1827. *Outline of a New System of Logic, with a Critical Examination of Dr. Whately's Elements of Logic*. London: Hunt & Clarke.

[2] Bocheński, I.M. 1962. *A History of Formal Logic*. Notre Dame, Indiana: University of Notre Dame Press.

[3] Boole, G. 1847. *The Mathematical Analysis of Logic*. Cambridge: MacMillan, Barclay & MacMillan.

[4] Boole, G. 1848. The calculus of logic. *Cambridge and Dublin Mathematical Journal* 3: 183-198.

[5] Boole, G. 1854. *An Investigation of the Laws of Thought*. London: Walton.

[6] Broad, C.D. 1917. Review of George Boole 'Collected Logical Works. Vol. II. Laws of Thought'. *Mind* 26: 81-99.

[7] Brown, F.M. 2009. George Boole's deductive system. *Notre Dame Journal of Formal Logic* 50: 303–30.

[8] Burris, S.N. 2015. George Boole and Boolean algebra. *European Mathematical Society Newsletter* 98: 27-31.

[9] Burris, S.N. 2019. George Boole. *Stanford Encyclopedia of Philosophy*, `https://plato.stanford.edu/entries/boole/`.

[10] Burris, S.N. 2022. *Annotated Version of Boole's Algebra of Logic.* `https://math.uwaterloo.ca/~snburris/htdocs/MAL_Nov_20_2022.pdf`

[11] Burris, S.N. 2022a. *Annotated Logic Chapters from Boole's Laws of Thought.* `https://math.uwaterloo.ca/~snburris/htdocs/LT_15_CHAPS.pdf`

[12] Burris, S.N. 2024. *Annotated Version of Boole's 1848 paper The Calculus of Logic.* https://math.uwaterloo.ca/~snburris/htdocs/CL.pdf

[13] Burris, S.N. & H.P. Sankappanavar 2013. The Horn theory of Boole's partial algebras. *Bulletin of Symbolic Logic* 19: 97-105.

[14] Cayley, A. 1871. A note on the calculus of logic. *Quarterly Journal of Pure and Applied Mathematics* 7: 282-3.

[15] Couturat, L. 1905. *L'algèbre de la logique.* Paris : Gauthier-Villars. English translation: *The Algebra of Logic.* Chicago: Open Court, 1914.

[16] De Morgan, A. 1847. *Formal Logic.* London: Taylor & Watson.

[17] De Morgan, A. 1966. *On the Syllogism and Other Logical Writings,* ed. P. Heath. London: Routledge & Kegan Paul.

[18] Frege, G. 1879. *Begriffschrift, eine der arithmetischen nachgebildete Formelsprache des reinen Denkens.* Halle: L. Nebert.

[19] Gastaldi, J.L. 2022. Boole's untruth tables: the formal conditions of meaning before the emergence of propositional logic. In *Logic in Question* ed. J.-Y. Béziau et al. pp. 119-149. Cham: Birkhauser.

[20] Grattan-Guinness, I. 1991. The Correspondence between George Boole and Stanley Jevons, 1863–1864. *History and Philosophy of Logic* 12: 15–35.

[21] Grattan-Guinness, I. & G. Bornet eds 1997. *George Boole: Selected Manuscripts on Logic and Philosophy.* Basel: Birkhauser.

[22] Hailperin, T. 1986. *Boole's Logic and Probability*, 2nd edition. Amsterdam: North-Holland.

[23] Jevons, W. S. 1864. *Pure Logic.* London: Edward Stanford

[24] Jevons, W. S. 1881. Recent mathematico-logical memoirs. *Nature*, 23: 485-487.

[25] Kneale, W.&M. 1962. *The Development of Logic.* Oxford: Clarendon Press.

[26] Lewis, C.I. 1918. *A Survey of Symbolic Logic.* Berkeley.

[27] Makinson, D. 2022. Boole's indefinite symbols re-examined. *Australasian Journal of Logic* 2022, 19: 165-181. Post-publication text with additions at `https://sites.google.com/site/davidcmakinson/`

[28] Makinson, D. 2023. Hamilton's cumulative conception of quantifying particles: an exercise in third-order logic. *J. Logic & Computation*, online 01/12/2023, DOI 10.1093/logcom/exad072. Post-publication text with additions at `https://sites.google.com/site/davidcmakinson/`

[29] Peirce, C.S. 1867. On an improvement in Boole's calculus of logic. *Proceedings of the American Academy of Arts and Sciences* 7: 250-261.

[30] Peirce, C.S. 1883. Pages iii-vi (Preface) and 187-203 (Note B) of C.S. Peirce et al. *Studies in Logic by Members of the Johns Hopkins University.* Boston: Little, Brown, and Company.

[31] Peckhaus, V. 1998. Hugh MacColl and the German algebra of logic. *Nordic Journal of Philosophical Logic* 3: 17-34.

[32] Pollard, S. 2022. *Ernst Schröder on Algebra and Logic.* Synthese Library 465. Cham: Springer.

[33] Rice, A. 2024. De Morgan and mathematics. Pages 3-30 of Attar et al. *Augustus de Morgan, Polymath: New Perspectives on his Life and Legacy.* Cambridge UK: Open Book Publishers.

[34] Schröder, E. 1877. *Operationskreis des Logikkalkuls.* Leipzig: Teubner.

[35] Schröder, E. 1905. *Vorlesungen über die Algebra der Logik (Exacte Logik) Bande 2, Zweiter Abteilung.* Leipzig: Teubner. Also: https://archive.org/details/bub_gb_zQ9VAAAAMAAJ/page/n82/mode/1up

[36] Waszek, D. (to appear). Boole's late manuscript "On the nature of thought": a rewriting of "The Laws of Thought" without uninterpretables. *History and Philosophy of Logic.*

[37] Waszek, D. & Schlimm, D. 2021. Calculus as method or calculus as rules? Boole and Frege on the aims of a logical calculus. *Synthese* 199: 11913–11943.

[38] Wilson, M. 2006. *Wandering Significance: an Essay on Conceptual Behaviour*. Oxford University Press.

[39] Wilson, M. 2021. *Innovation and Certainty*. Cambridge University Press.

Received August 2025

From Hilbert Axioms to Frame Properties in Modal Logics

Hans Jürgen Ohlbach

Abstract

In the 1990s Dov Gabbay and myself developed an algorithmic method for eliminating existentially quantified predicates from second-order predicate logic formulae. We used it to automatically map Hilbert axioms of normal propositional modal logics to the corresponding properties of the accessibility relation in its possible worlds semantics. "And what about completeness?" was a question Dov kept asking. In this note I shortly present the method and put the finger on the points where the still unsatisfactorily solved completeness issues arise.

1 Two Ways for Characterizing Normal Propositional Modal Logics

The formulae of propositional modal logic are like the formulae of standard propositional logic extended with the modal operators $\Box$ (necessarily) and $\Diamond$ (possibly). Their semantics can be characterized in different ways.

Hilbert Calculus

Kurt Gödel introduced the now traditional way for characterizing a modal logic by means of axioms and inference rules. The calculus for the basic normal propositional modal logic K consists of

- the axioms for standard propositional logic,
- the axiom K: $\Box(A \to B) \to (\Box A \to \Box B)$,
- the Modus Ponens inference rule: From P and $P \to Q$ infer Q,
- and the Necessitation Rule: If P is a theorem then $\Box P$ can be inferred.

In principle one can add arbitrarily additional axioms, as long as the calculus remains consistent. Well known axioms are:

axiom D:	$\Box P \to \Diamond P$	axiom 4:	$\Box P \to \Box\Box P$
axiom T:	$\Box P \to P$	axiom 5:	$\Diamond P \to \Box\Diamond P$
axiom B:	$P \to \Box\Diamond P$	axiom .2	$\Diamond\Box P \to \Box\Diamond P$
axiom M	$\Box\Diamond P \to \Diamond\Box P$	McKinsey Axiom	
axiom L:	$\Box(\Box P \to P) \to \Box P$	Löb-Axiom	

Most modal systems discussed in the literature are combinations of these axioms. Examples are:

S4	axioms K, T and 4	
S4.2	axioms K, T, 4 and .2	
GL	axioms K and L	(Gödel-Löb logic, axiom 4 is actually a consequence in GL)

Possible Worlds Semantics

The possible worlds semantics is a very intuitive semantics for normal modal logics. It goes back to Jónsson and Tarski's representation theorem for Boolean Algebras with operators [4, 5] and later on Kripke's completeness results [6, 7].

A *Kripke frame* consists of a set $\mathcal{W}$ of so called possible worlds as points of evaluation, and a binary relation $\mathcal{R}$ between these worlds. A particular model assigns truth values to each predicate symbol in each world. A formula $\Box P$ is true in a world w if P itself is true in all worlds which are $\mathcal{R}$-accessible from w. A formula $\Diamond P$ is true in a world w if P itself is true in at least one world which is $\mathcal{R}$-accessible from w.

It turned out that some Hilbert-axioms characterize properties of the accessibility relation $\mathcal{R}$.

axiom D:	$\Box P \to \Diamond P$	Seriality of $\mathcal{R}$
axiom T:	$\Box P \to P$	Reflexivity of $\mathcal{R}$
axiom B:	$P \to \Box\Diamond P$	Symmetry of $\mathcal{R}$
axiom 4:	$\Box P \to \Box\Box P$	Transitivity of $\mathcal{R}$
axiom 5:	$\Diamond P \to \Box\Diamond P$	Euclidicity of $\mathcal{R}$
axiom .2:	$\Diamond\Box P \to \Box\Diamond P$	Convergence of $\mathcal{R}$
axiom M:	$\Box\Diamond P \to \Diamond\Box P$	Some singleton successor
axiom L:	$\Box(\Box P \to P) \to \Box P$	No infinite ascending $\mathcal{R}$-chains.

Over time *Completeness Results* have been proved for the correspondences between Modal Logics characterized by Hilbert calculi and the corresponding semantically characterized modal logics. This means that a formula is a theorem in the Hilbert calculus if and only if it is true in all its Kripke models.

2 Quantifier Elimination

In [1] Dov Gabbay and I presented a method for algorithmically mapping Hilbert axioms to properties of the accessibility relation. It is based on a *quantifier elimination* algorithm for existentially quantified predicates from formulae like $\exists P\varphi$ in second-order predicate logic. The algorithm (we called it the SCAN-algorithm) performs the following steps:

- Transform φ into clause form (existentially quantified variables must be *skolemized*).
- Exhaustively generate all resolvents and factors with literals containing the predicate P (this step may not terminate). The resolution operation needed here is slightly more general than the usual resolution rule: If two terms s and t in the resolution literals do not unify, generate the resolvent anyway, but add a literal $s \neq t$ to the resolvent.
- If possible unskolemize those resulting clauses which do not contain the predicate P.

Universally quantified predicate variables can be eliminated by first negating the formula, eliminating the existentially quantified predicates, and finally negating the result.

What about completeness?

The completeness result for the elimination of quantified predicate variables states that if the procedure succeeds then the output formula is equivalent to the input formula. The procedure may fail in three different ways:
1. The generation of the resolvents may not terminate. In some cases it may still be possible to comprise the infinitely many resolvents into a finite formula schema which describes the property of the accessibility relation. The Löb-Axiom is an example for this phenomenon (see below).
2. The conversion of the formula into clause form may require the replacement of existentially quantified variables by Skolem functions. From a pure logic point of view, these are existentially quantified function variables. If they are still contained in the resolvents they can or cannot be unskolemized, i.e. transformed back into existentially quantified variables. If this is not possible they either need to remain as existentially quantified functions, or, in some cases, become part of parallel Henkin quantifiers [3]. As we shall see in the McKinsey example, the resolvents can be simplified and transformed such that first-order unskolemization is possible. So far,

however, there is no algorithmic procedure to do this.
3. Most modal logics are specified with more than one Hilbert axiom. In this case the correspondence properties of the different Hilbert axioms can interact such that overall they become simpler. The Löb-Axiom is again an example where this can be exploited.

3 Hilbert Axioms ⇒ Properties of the Accessibility Relation

The standard relational translation ST of Hilbert axioms into predicate logic uses the possible-worlds semantics to replace the modal operators by predicate logic formulae:

$$\begin{aligned} &ST(\Box\varphi)_w = \forall v\ R(w,v) \rightarrow ST(\varphi)_v \text{ and} \\ &ST(\Diamond\varphi)_w = \exists v\ R(w,v) \wedge ST(\varphi)_v \end{aligned}$$

where w and v denote the worlds at which the modal expression is to be evaluated. R is the predicate symbol that denotes the accessibility relation $\mathcal{R}$.

The translation of Hilbert axioms replaces in addition the formula variables by universally quantified predicate variables and, because the axiom is supposed to hold in all worlds, it adds a universal quantifier over all worlds.

For example: $ST(\Box P \rightarrow \Diamond P) = \forall P\ \forall w\ (\forall x\ R(w,x) \rightarrow P(x)) \rightarrow (\exists y\ R(w,y) \wedge P(y))$

What about completeness?

The replacement of a formula variable P by a universally quantified one-place predicate $\forall P\ \ldots P(\ldots)$ in the standard translation seems to be crucial here. Since the formula variables in the Hilbert axiom are to be replaced by all modal formulae (rule of uniform substitution), a proper standard translation should translate all the instances of the axioms (usually infinitely many). If we instead introduce the universally quantified predicate, this means a quantification over *all* subsets of the possible worlds, not only those subsets where one of the modal formulae is true. Since the set of modal formulae is enumerable, the set of their truth-sets is also enumerable. If there are infinitely many possible worlds, the set of all subsets is not enumerable. Therefore the quantifier over a predicate variable may introduce a stronger condition than the formula variables in the Hilbert axiom. The completeness results for many of the modal logics and the equivalence result of the quantifier elimination algorithm (where it succeeds) show that this is not a problem for these logics. In general, however, this point should be investigated in more detail.

Application to Hilbert Axioms

The standard translation of modal axioms introduces universally quantified predicate variables. In order to apply the quantifier elimination algorithm one has to negate the translated formula, thus turning the universal quantifiers into existential quantifiers, apply quantifier elimination, and negate the result again.

We illustrate the method with a few examples. The first two examples work as expected. The last two examples show results which still make sense, but which are not covered by the equivalence result of the quantifier elimination algorithm. Since modal axioms are more compact than their standard translation, the examples become more readable if we negate the modal axiom first, and then translate it.

Axiom D: $\Box P \rightarrow \Diamond P$ (Seriality)
Negated and Translated:
$ST(\Box P \wedge \Box\neg P) = \exists P\ \exists w\ (\forall x\ R(w,x) \rightarrow P(x)) \wedge (\forall y\ R(w,y) \rightarrow \neg P(y))$

Clauses:
$\neg R(w_s,x), P(x)$ (w_s is the Skolem constant for w)
$\neg R(w_s,y), \neg P(y)$

Resolvent with P: $\neg R(w_s,x)$

Unskolemized: $\exists w \forall x\ \neg R(w,x)$

Negated: $\forall w \exists x\ R(w,x)$, which specifies the seriality of $\mathcal{R}$.

Convergence Axiom: $\Diamond\Box P \rightarrow \Box\Diamond P$
Negated and Translated:
$$ST(\Diamond\Box P \wedge \Diamond\Box\neg P)$$
$$= \exists P\ \exists w\ (\exists x\ R(w,x) \wedge (\forall y R(x,y) \rightarrow P(y))) \wedge$$
$$(\exists u\ R(w,u) \wedge (\forall v R(u,v) \rightarrow \neg P(v)))$$

Clauses:
$R(w_s,x_s)$ x_s is the Skolem constant for x
$\neg R(x_s,y), P(y)$
$\neg R(w_s,u_s)$ u_s is the Skolem constant for u
$\neg R(u_s,v), \neg P(v)$

Resolvent with P: $\neg P(x_s,y), \neg P(u_s,y)$

Unskolemized: $\exists w,x,u\ R(w,x) \wedge R(w,u) \wedge \forall y\ (\neg R(x,y) \vee \neg R(u,y))$

Negated: $\forall w,x,u\ (R(w,x) \wedge R(w,u)) \rightarrow \exists y\ (R(x,y) \wedge R(u,y))$

and this means convergence: from two accessible worlds there is a common one which is accessible from both worlds.

The McKinsey Axiom: $\Box\Diamond P \to \Diamond\Box P$
This is an example where the unskolemization step becomes critical. With a relatively simple transformation, however, we can find a meaningful result.

Negated and Translated:
$ST(\Box\Diamond P \wedge \Box\Diamond\neg P)$
$= \exists P\ \exists w\ (\forall x\ R(w,x) \to (\exists y\ R(x,y) \wedge P(y)) \wedge$
$\quad\quad (\forall u\ R(w,u) \to \exists v\ R(u,v) \wedge \neg P(v))$

The negation means informally: for every accessible world there are at least two different accessible worlds.

Clauses: (the variables in different clauses must be renamed)

$\neg R(w_s, x_1), R(x_1, y_s(x_1))$ (y_s is the Skolem function for y)
$\neg R(w_s, x_2), P(y_s(x_2))$
$\neg R(w_s, u_1), R(u_1, v_s(u_1))$ (v_s is the Skolem function for v)
$\neg R(w_s, u_2), \neg P(v_s(u_2))$

P resolved away:, together with the remaining clauses:

$\neg R(w_s, x_1), R(x_1, y_s(x_1))$
$\neg R(w_s, u_1), R(u_1, v_s(u_1))$
$\neg R(w_s, x_2), \neg R(w_s, u_2), y_s(x_2) \neq v_s(u_2)$

Unskolemized with second-order Henkin quantifiers [Hen61]:

$$\exists w \begin{pmatrix} \forall x\ \exists y \\ \forall u\ \exists v \end{pmatrix} \begin{array}{l} (R(w,x) \to R(x,y)) \wedge \\ (R(w,u) \to R(u,v)) \wedge \\ (\neg R(w,x) \vee \neg R(w,u) \vee y \neq v) \end{array}$$

which is not a first-order formula.

In the resolution calculus, variables in different clauses are usually renamed such that two different clauses don't share variables. It is, however, logically possible to unify these variables again. By unifying x_1, x_2, u_1 and u_2 we obtain

$\neg R(w_s, x), R(x, y_s(x))$
$\neg R(w_s, x), R(x, v_s(x))$
$\neg R(w_s, x), y_s(x) \neq v_s(x)$

Unskolemized:
$\exists w\ \forall x\ R(w,x) \to \exists y_1, y_2\ (R(x,y_1) \wedge R(x,y_2) \wedge y_1 \neq y_2.$

Negated:
$\forall w\ \exists x\ R(w,x) \wedge \forall y_1, y_2\ (R(x,y_1) \wedge R(x,y_2)) \to y_1 = y_2.$

Axiom L (Gödel-Löb): $\Box(\Box P \to P) \to \Box P$
In this example the resolution operations do not terminate. With meta-reasoning and induction it is possible to generate an formula schema which describes the property of the accessibility relation.

Negated and Translated:
$ST(\Box(\Diamond\neg P \lor P) \land \Diamond\neg P)$
$= \exists P\ \exists w\ (\forall x\ R(w,x) \to ((\exists y\ R(x,y) \land \neg P(y)) \lor P(x)) \land (\exists z\ R(w,z) \land \neg P(z))$

Clauses

C1	$\neg R(w_s,x), R(x,y_s(x)), P(x)$	y_s is the Skolem function for y.
C2	$\neg R(w_s,x), \neg P(y_s(x)), P(x)$	
C3	$R(w_s,z_s)$	z_s is the Skolem constant for z.
C4	$\neg P(z_s)$	

We can exploit that the axiom 4 ($\Box P \to \Box\Box P$ holds in the Gödel-Löb logic). That means the accessibility relation is transitive.

Resolutions

C1&C4	C5:	$\neg R(w_s,z_s), R(z_s,y_s(z_s))$ using C3 simplified to $R(z_s,y_s(z_s))$
C2&C4	C6:	$\neg R(w_s,z_s), \neg P(y_s(z_s))$ using C3 simplified to $\neg P(y_s(z_s))$
C1&C6	C7:	$\neg R(w_s,y_s(z_s)), R(y_s(z_s),y_s(y_s(z_s)))$ using C5 and transitivity simplified to $R(y_s(z_s),y_s(y_s(z_s)))$
C2&C6	C8:	$\neg R(w_s,y_s(z_s)), \neg P(y_s(y_s(z_s)))$ using C5 and transitivity simplified to $\neg P(y_s(y_s(z_s)))$

and so on. One can comprise the results to $\forall i\ R(y_s^i(z_s), y_s^{i+1}(z_s))$
where y_s^i means i-times nesting of y_s.

After negation we get $\exists i\ \neg R(y_s^i(z_s), y_s^{i+1}(z_s))$,
which means the sequence of accessible worlds eventually ends, i.e. the accessibility relation is converse well-founded.

Summary

As we have seen, the transformation of Hilbert axioms of normal model logics to properties of the accessibility relation by means of the quantifier elimination procedure can be fully automated only in relatively simple cases. Logical transformations may help in more complex cases. Some of them might also be automatable, but this seems to be a never ending story. Soundness and completeness of the transformations at the level of predicate logic can be proven with the usual means. What is still unclear with respect to completeness is the replacement of formula variables in the Hilbert axioms by universally quantified second-order predicate variables.

Acknowledgement

Dov Gabbay visited me a few times while I was still in the Max-Planck-Institute in Saarbrücken. I remember him enjoying good German Weissbier. From 1996 - 2000 I joined his group, first at Imperial College and then at King's College in London. I am very grateful, both for his and his wife Lydia's hospitality, and for his and Jane Spurr's (his secretary) professional support.

References

[1] Dov M. Gabbay and Hans Jürgen Ohlbach. Quantifier elimination in second–order predicate logic. In Bernhard Nebel, Charles Rich, and William Swartout, editors, *Principles of Knowledge Representation and Reasoning (KR92)*, pages 425–435. Morgan Kaufmann, 1992. also published in [2].

[2] Dov M. Gabbay and Hans Jürgen Ohlbach. Quantifier elimination in second–order predicate logic. *South African Computer Journal*, 7:35–43, July 1992.

[3] L. Henkin. Some remarks on infinitely long formulas. In *Infinitistic Methods*, pages 167–183. Pergamon Press, Oxford, 1961.

[4] B. Jónsson and A. Tarski. Boolean algebras with operators, Part I. *American Journal of Mathematics*, 73:891–939, 1951.

[5] B. Jónsson and A. Tarski. Boolean algebras with operators, Part II. *American Journal of Mathematics*, 74:127–162, 1952.

[6] S. A. Kripke. A completeness theorem in modal logic. *Journal of Symbolic Logic*, 24:1–14, 1959.

[7] S. A. Kripke. Semantical analysis of modal logic I, normal propositional calculi. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, 9:67–96, 1963.

Received July 2025

Words Without World: The Non-Epistemic Predictions of LLMs. A Note for Dov Gabbay's 80th Birthday

Gabriella Pigozzi
LAMSADE, Université Paris-Dauphine, PSL, Paris, France
gabriella.pigozzi@lamsade.dauphine.fr

1 Tides

I'm writing this note while on holiday on the Île de Ré, an island off the Atlantic coast of France. This week, the tidal coefficient is relatively high, indicating that the tide's amplitude is also high. As an Italian who had only known the Mediterranean Sea, the first time I visited Britanny and went to the beach at low tide, I alarmingly and (ashamedly) loudly asked, "*Where* is the sea?". So, I smiled today when I heard a young girl, who was also swimming and playing with the waves, tell her father that last night she saw fewer rocks and asked him why these rocks had moved here, so close to the shore today. I wonder if she would get a correct explanation if she asked the same question to an AI chatbot (yes, she would).

I pondered this because, some months ago, I tested how an AI would solve an exercise I had given my students. It was an axiomatic proof in propositional logic. It didn't go well. I gave ChatGPT the set of axioms and the theorem to prove. It seemed very confident, but the proof was wrong. It assumed non-existent theorems and created some convenient hypotheses to make the proof almost trivial. I engaged in a discussion with the chatbot, correcting its mistakes for some time. However, after a while, I became angry as it continued making the same mistakes over and over again, much like a stubborn student. After one hour, my husband told me, "You know you're training it, right?". I stopped.

One part of me was relieved because I could be sure that (for the moment, at least), my students could not hand me over a correct AI-generated axiomatic proof. But it also raised many questions about the value of current large language models (LLMs). This is also one of Dov's current concerns. In recent weeks, on several

Thanks to Flavia Padovani for her constructive feedback on the manuscript.

occasions during our online meetings, he raised the issue of LLMs being unable to distinguish truth from falsehood. Would that be the end of logic?

What these tools can do is impressive. They are extremely useful in various fields, e.g., medical research. Their use is becoming more and more pervasive in our society. These are technologies that are here to stay. Given their extremely rapid development, it is hard to predict how they will evolve and be used in the future. In the face of this uncertainty and with the possibility of arriving at an Artificial General Intelligence (AGI), it is natural to have polarised views concerning AI. On one side, there are the enthusiasts, those who think that AI is a revolution that will significantly impact every aspect of human existence, that AGI is attainable, and its effects will be everlasting. People like Sam Altman obviously belong to this group. One of the reasons for this conviction is the scaling law for LLMs, supported by a 2020 paper [10], which is the idea that by increasing the size of the training set, the performance of LLMs would also increase.[1] On the other side, some are sceptical about the virtually limitless potential of AI. Gary Marcus is one of them and has been very critical and vocal about the current state of LLMs. He argues that current LLMs fail to capture some crucial aspects of human intelligence, like the ability to generalise from limited information and apply knowledge in new contexts. The reason is that neural networks seem to work well within a distribution of data they have been trained on, but fail outside that distribution. So, training the models on more and more data won't solve the problem. Marcus calls for a *neurosymbolic AI*, where predictive models are coupled with symbolic reasoning based on human logic.

The discussion between these two positions is heated. The group of "negationists" is often ridiculed, and their worries have been compared to Socrates' ones about the dangers of writing, as exposed in Plato's Phaedrus:

> Your invention will enable them to hear many things without being properly taught, and they will imagine that they have come to know much, while for the most part, they will know nothing. And they will be difficult to get along with, since they will merely appear to be wise instead of really being so.

Those accused of being blind to a technological revolution hurry to defend themselves, explaining that they are not Luddites (nobody likes to be accused of being an old grump trying to resist the advances of a new culture). However, in the face of a technology that promises to revolutionise the world as we know it and threatens

[1] Notice that [10] admit that "At present we do not have a solid theoretical understanding for any of our proposed scaling laws." (p.22)

to take the jobs of millions of people (Sam Altman's words), while consuming a country-sized amount of energy, water, etc., perhaps some caution should be warranted. The "negationists" point to news that hints at an AI bubble [3, 11, 12, 14], like the disastrous release of OpenAI's GPT-5 (which cast more doubts on the existence of an AI scaling law and led to the embrace of post-training techniques[2]). Other news that recently made headlines is that a staggering 95% of companies are witnessing the failure of generative AI pilots, the decline of AI adoption rates in large companies and the historic $1.5 billion AI copyright agreement between Anthropic and authors who accused the company of copyright infringement as they mass digitised books to train LLMs.

So, like our societies, the debate around AI is heavily polarised.

2 Statistical 'truths'

LLMs are fed with and trained on a vast corpus of text (books, articles, web pages, etc.). The training process allows the model to make predictions based on statistical patterns in the data. Because these models do not understand the meaning of the text they process, but they repeat what they have learned from the training data, they have been referred to as *stochastic parrots* [1]. These systems cannot distinguish truth from falsehood because statistics, not logic, govern them.

In the rest of this paper, I will list some of the issues that emerge from such an architecture.

- *Training data*: Training data is a collection of annotated information that is used to teach a machine learning model how to make predictions, recognise patterns or generate content. Kate Crawford warns us about the social and political implications of what appears as a purely technical approach:

  > [T]raining data is a brittle form of ground truth – and even the largest troves of data cannot escape the fundamental slippages that occur when an infinitely complex world is simplified and sliced into categories. [...] [H]ow and what they are fed has an enormous impact on how they will interpret the world, and the priorities of their masters will always shape how that vision is monetised. By looking at the layers of training data that shape and inform AI

[2] Yet, some are sceptical about the actual reach of these post-training approaches [14]. Post-training happens on some benchmark problems, whose inputs and outputs are much simpler and more clearly defined than those we encounter in general problems. The fact that we get better scores is no surprise.

> models and algorithms, we can see that gathering and labelling data about the world is a social and political intervention, even as it masquerades as a purely technical one. ([5], p. 98 and 121)

The traceability of data used for training is a desirable requirement. Yet, training data are not open access. After the pre-training phase, the Reinforcement Learning from Human Feedback (RLHF) technique aims at defining a reward function that aligns with human preferences and values by using feedback from external, paid annotators. Whoever ranks outputs determines what ‘truth’ is. Moreover, if the human evaluators come from a narrow demographic, the model may exhibit biases. Model providers are increasingly interested in obtaining additional feedback directly from their users. This incurs no costs and ensures user satisfaction. However, as [20] show, this can lead an AI to deploy manipulative tactics to receive positive feedback from vulnerable users. This can go as far as supporting harmful behaviours in users rather than steering them away, when this would cause the user to evaluate the model negatively.

- *Fake news:* The historian Carlo Ginzburg observes that “fake news” is a new name for a phenomenon that already existed in the far past [7]. For this reason, he believes that talking of “post-truth” is inappropriate, a position shared by [15] since “there has never been a ‘truth era’”. What changed is the medium: the web allows unprecedented production and distribution speed. Unfortunately, LLMs can also contribute to the automation of many aspects of fake news generation, making it easier to create believable and compelling counterfeit stories. At the same time, these systems can be used to build polished blogs or websites, which can be employed to diffuse fake news and AI-generated deepfakes. A recent audit of the leading AI tools conducted by the online news fact-checking service NewsGuard[3] revealed that these tools repeat false news claims more than one-third of the times (which represents an increase of 18% in the last year). The reason is the progressively polluted online information ecosystem from which LLMs get their information. Malign actors ensure that fake news is well spread. This, coupled with the AI chatbots’ current inability to distinguish volume from accuracy, leads to treating unreliable sources as credible ones. According to the audit’s results, parallel to such a decline in accuracy, AI chatbots have increased their responsiveness. If they previously avoided answering some prompts, they now respond by drawing information from non-credible sources.

- *Autofagy*: With the diffusion of LLMs, gradually more and more content

[3] https://www.newsguardtech.com/ai-monitor/august-2025-ai-false-claim-monitor/

will be generated by LLMs. [18] investigated what may happen when LLMs train increasingly on data produced by their predecessors rather than human-generated text. What if one day, much of the online content is generated by LLMs? They discovered that "indiscriminately learning from data produced by other models causes 'model collapse' – a degenerative process whereby, over time, models forget the true underlying data distribution. [. . .] Being trained on polluted data, they then misperceive reality." If what is generated is injected back into the next generation of models, they could eventually collapse into nonsense and noise.

- *Cognitive flattening*: An MIT study compared essays produced by a group of university students allowed to use ChatGPT with those written by another group of students who could only rely on their brains. The results indicate that texts produced with the help of AI showed less diversity, a lack of originality, and a convergence on similar ideas [4]. In the long run, the cognitive cost of relying on AI could be huge, especially if such a homogenising effect is combined with the feeding loop mentioned above.
- *Reasoning errors*: We still lack a comprehensive understanding of how AI models reason. Some defend that this happened before with previous technological inventions:

> Reducing a complex phenomenon to its building blocks is always a mistake. [. . .] AI is much more than the algorithms that govern the probabilities of its elements. [. . .] We build the bricks that make up LLMs, but do we understand the infinitely complicated series of causal relationships that run from our prompts to the AI's output? No. Then again, this has always been the case: Volta built the battery a century before Maxwell came up with a theoretical formulation of electromagnetism. Watt built the famous valve for the steam engine a century before Wiener's control theory. ([13], Author's translation)

However, a battery does not decide whether I will get a loan, nor does it help a judge rule in a court case. To trust AI decisions, we need to understand what happens in that black box. Thus, understanding AI-generated mathematical proofs can be a starting point. [16] evaluated the reasoning of LLMs in proofs of problems from the 2025 USA Math Olympiad. Their findings revealed several reasoning errors, like flawed logic and the use of unjustified assumptions. In June 2025, some researchers at Apple tested LLMs and LRMs (large reasoning models) across a variety of puzzles [17]. They showed that these models collapse as complexity increases. Particularly telling were the results on the Tower of Hanoi puzzle, a standard game in computer science. Even when the authors provided the solution algorithm to the models, their performance did not improve. A promising research avenue comes from neurosymbolic models. AlphaGeometry, for example, combines a neural language model with a symbolic engine that guides the deductive steps in proving geometry problems, as seen in the International Mathematical Olympiad competition [19]. As in [8]'s two systems governing how humans think, neurosymbolic models combine 'fast' LLMs, whose strength is to identify data patterns, with 'slow' symbolic deduction engines based on formal logic.

- *Hallucinations*: Wikipedia defines an AI hallucination as "a response generated by AI that contains false or misleading information presented as fact." The use of an anthropocentric term like 'hallucination' has been criticised. Adopting a term that evokes a psychological human condition could seem a successful marketing strategy ('hallucination' sounds better than 'failure'). A recent OpenAI paper affirms that AI hallucinations happen because "the training and evaluation procedures reward guessing over acknowledging uncertainty" [9]. Harry Frankfurt studied different dishonesty classes [6]: lies, deceptions and bullshits. Bullshit happens when we make statements about things we are not knowledgeable about, because we have to say something (and want to appear knowledgeable) about a particular complex topic. In bullshit, unlike in a lie, there is no consideration for the truth or falsity of the statement. Finally, deception occurs when a piece of correct information is given. However, the person receiving the information will likely make an incorrect inference. If we take Frankfurt's classification, it seems that AI hallucinations should be classified as bullshit, arguably an even less appealing term than 'failure'.

3 Conclusion

LLMs need a vast amounts of text. The more data is produced, the more data is injected. We are trying to discover textual statistical patterns to make sense of the world. But are we sure we are heading in the right direction? Jorge Luis Borges was aware of the dangers of the illusion of total knowledge. In *The Library of Babel*, we are taken to an infinite library, containing every possible book. Each book includes every possible ordering of just 25 basic symbols. Although most of the volumes in the library are nonsensical, the laws of probability ensure that books containing perfect truths can also be found:

> When it was announced that the Library contained all books, the first reaction was unbounded joy. All men felt themselves the possessors of an intact and secret treasure. There was no personal problem, no world problem, whose eloquent solution did not exist – somewhere in some hexagon. ([2], p. 115)

However, finding coherent books could only happen thanks to "an improbable gift of chance". This threw the inhabitants of the library into madness and depression. As for the inhabitants of Borges' library, in the end, it will all depend on the use we make of such tools. We cannot expect all solutions to personal or world problems to be found in an LLM library.

What worries me most is the installation of a cognitive dependence that may, in the end, become cognitive laziness and finally atrophy. When a colleague told me that some researchers are now subcontracting to an AI to do their reviews for conferences or journals, I was surprised, partly due to my naïveté. What interrogates me most is *why* we may want to delegate a machine to write a paper or to review one. It makes even less sense to me why one major AI conference recently decided to employ an 'AI-Powered Peer Review Assessment System'[4]. Writing and reviewing papers are among the primary activities of our jobs, and one that allows us to generate new ideas. We have struggled to get these positions, and we have the chance to have jobs that (put aside the administrative burdens) we chose out of passion. So, why are we so ready to give them up to robots?

I realise that this is not an uplifting note, especially considering the occasion of Dov's birthday. Of course, not all is grim. Research on explainable AI is fundamental, striving to meet the demands of transparency on the AI processes and outputs. This would also enable us to understand better when and why AI make mistakes. Another pillar is education, both in critical thinking skills to make citizens less vulnerable to false information, and in what Ginzburg calls 'education in digital philology', where AI chatbots are not only used to get answers, but also to produce new questions that, in turn, lead to further questions.

To conclude on a more optimistic note, let me assure you that, unlike my students with their axiomatic proof exercise, the kid on the shore got the correct answer from her father.

References

[1] E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell. On the dangers of stochastic parrots: can language models be too big? In *Proceedings of the 2021 ACM*

[4] https://aaai.org/aaai-launches-ai-powered-peer-review-assessment-system/

Conference on Fairness, Accountability, and Transparency, 2021.

[2] Jorge Luis Borges. The library of Babel. In *Collected Fictions*, pages 112–118. Viking Press, 1998.

[3] J. Cassidy. Is the A.I. boom turning into an A.I. bubble? *The New Yorker*, August 12, 2025.

[4] K. Chayka. A.I. is homogenizing our thoughts. *The New Yorker*, June 25, 2025.

[5] Kate Crawford. *Atlas of AI. Power, Politics, and the Planetary Costs of Artificial Intelligence.* Yale University Press, 2021.

[6] H.G. Frankfurt. *On Bullshit.* Princeton University Press, 2005.

[7] C. Ginzburg. Fake news, tra passato e futuro. *Gli Asini*, 107:3–6, 2025.

[8] Daniel Kahneman. *Thinking, Fast and Slow.* Farrar, Straus and Giroux, 2011.

[9] Adam Tauman Kalai, Ofir Nachum, Santosh S. Vempala, and Edwin Zhang. Why language models hallucinate, 2025.

[10] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020.

[11] H. Langley. AI hype is crashing into reality. Stay calm. *Business Insider*, September 4, 2025.

[12] N. Lichtenberg. ‘It’s almost tragic’: Bubble or not, the AI backlash is validating what one researcher and critic has been saying for years. *Fortune*, August 24, 2025.

[13] R. Manzotti. I Don Ferrante della IA. *L’Indiscreto*, July 2025, 2025.

[14] C. Newport. What if A.I. doesn’t get much better than this? *The New Yorker*, August 12, 2025.

[15] C. D. Novaes and J. de Ridder. Is fake news old news? In *The Epistemology of Fake News*, pages 156–179. Oxford University Press, 2021.

[16] Ivo Petrov, Jasper Dekoninck, Lyuben Baltadzhiev, Maria Drencheva, Kristian Minchev, Mislav Balunovic, Nikola Jovanovic, and Martin Vechev. Proof or bluff? Evaluating LLMs on 2025 USA Math Olympiad, 2025.

[17] Parshin Shojaee, Iman Mirzadeh, Keivan Alizadeh, Maxwell Horton, Samy Bengio, and Mehrdad Farajtabar. The illusion of thinking: Understanding the strengths and limitations of reasoning models via the lens of problem complexity, 2025.

[18] I. Shumailov, Y. Gal, Z. Shumaylov, Y. Zhao, N. Papernot, and R. Anderson. AI models collapse when trained on recursively generated data. *Nature*, 631:755–760, 2024.

[19] Trieu H. Trinh, Yuhuai Wu, Quoc V. Le, He He, and Thang Luong. Solving olympiad geometry without human demonstrations. *Nature*, 625(7995):476–482, 2024.

[20] Marcus Williams, Micah Carroll, Adhyyan Narang, Constantin Weisser, Brendan Murphy, and Anca Dragan. On targeted manipulation and deception when optimizing LLMs for user feedback. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu, editors, *International Conference on Representation Learning*, volume 2025, pages 89542–89593, 2025.

Received September 2025

Reasoning Alignment for Agentic AI: Argumentation, Belief Revision, and Dialogue

Tjitze Rienstra
Maastricht University, Maastricht, The Netherlands
t.rienstra@maastrichtuniversity.nl

Leendert van der Torre
University of Luxembourg, Esch-sur-Alzette, Luxembourg and Zhejiang University, China
leon.vandertorre@uni.lu

Liuwen Yu
University of Luxembourg, Esch-sur-Alzette, Luxembourg
liuwen.yu@uni.lu

Abstract

Agentic AI—deployed as technical systems that perceive, decide, and act via tools—faces requirements of safety, accountability, controlled adaptivity, and compositionality. We develop *Reasoning Alignment Diagrams* (RADs), commutative reasoning representations that align a source specification with an argumentation-based explanation route. As illustrative examples, we first show that *full-meet belief base revision* admits an exact representation within *base argumentation* via a restricted-attack construction: the revised base equals the intersection of the premises appearing in all stable extensions of the modified framework. This yields a RAD from input to sanctioned output that doubles as an explanation engine. We then compose the "listen" (revision) and "assert" (inference) RADs to model dialogue among agents, enabling explainable and auditable autonomy. Although our results are entirely symbolic, the RAD template can serve as a specification layer even when other components are opaque or learned. The approach realizes core themes of Gabbay's programme—logic as a toolbox, combining logics, and argumentation as a host formalism—and supports a principle-based analysis of correctness, transparency, and modularity.

1 Dov's vision and approach

Dov Gabbay's research programme treats logic not as a single monolithic calculus but as an *engineerable repertoire of mechanisms* that we assemble to model reasoning in context. Three leitmotifs run through his work and give us a methodological compass:

Logic as a toolbox. Beginning with *Labelled Deductive Systems* (LDS[46]), Gabbay advocates building logics from reusable components—labels for time, agents, resources, priorities; modalities for knowledge and obligation; non-monotonic rules and preference orderings: chosen to fit an application. The point is methodological pluralism with engineering discipline: pick the right tools, and make the choices explicit.

Combining logics. The toolbox needs assembly rules. In *Fibring Logics* and related work [45, 47, 48], Gabbay and collaborators developed operations (fibring, fusion, possible-translations) and meta-results that tell us when properties (soundness, completeness, decidability, interpolation) are preserved or lost when we cut and paste reasoning systems. This provides the algebra of composition required to scale from local components to full systems.

Argumentation as the logic for our century. Gabbay has consistently argued that the centre of gravity in logic must shift from static consequence to *interactive, defeasible, and explainable* reasoning. His contributions to abstract and structured argumentation—e.g., equational/numerical semantics for argumentation networks and higher-order attacks—frame argumentation as the *host formalism* where heterogeneous mechanisms live together and where explanations are native [10, 51].

Two further threads reinforce this picture. First, *temporal and executable views of logic* (e.g., MetateM) recast formulas as rules that drive processes over time; second, *reactive Kripke semantics* let models change as evaluation proceeds [50]. Together, they emphasize dynamics and interaction—exactly what argumentation systems aim to capture.

From this programme we distill five working principles that guide our paper.

P1 (Mechanisms, not monoliths). Choose and expose the representational and inferential devices appropriate to the task (labels, priorities, numerical updates, etc.).

P2 (Modularity and composition). Combine mechanisms using disciplined operations with meta-theoretic guarantees (fibring/fusion/translation), and state what is preserved.

P3 (Dynamics). Treat reasoning as a process that updates states (bases, networks, labels) rather than a once-for-all closure.

P4 (Commutation as explanation). When two perspectives on the same reasoning task coexist (e.g., direct belief change vs. argumentation), require a *commutative diagram*: different routes yield the same outcome. The commuting rectangle provides both a specification and an explanation.

P5 (Transparency). Prefer formalisms that surface *why* an outcome holds. Argumentation earns its keep by turning outcomes into attack/defence structures and (in equational treatments) explicit numerical/functional dependencies.

Our paper instantiates this programme at three levels.

Toolbox to construction. We take two standard "tools" from the KR toolbox—*belief base revision* (AGM-style) and *base argumentation*—and make the design choice that matters for our purpose: in the argumentation framework over $K \cup \{\varphi\}$, we restrict *attacks originating from* $K \setminus \{\varphi\}$. This single, explicit mechanism plays the role of contraction by $\neg\varphi$. The rest is cleanly delegated to Dung-style semantics.

Combining logics to commutation. Rather than embedding one formalism into the other ad hoc, we enforce *commutation* between two routes from input (K, φ) to output $K * \varphi$: revise directly by full-meet, or construct the modified base-AF and compute stable extensions, then extract conclusions. The equality of outcomes is our main theorem. In Gabbay's terms, this is a preservation result: our combination of mechanisms (base arguments + restricted attacks + stable semantics) *preserves* the specification given by full-meet revision.

Argumentation as host for interaction. Argumentation is not merely a target representation; it is the *operating system* in which different reasoning components—revision, inference, explanation—cohabit. We therefore compose two commuting diagrams to model dialogue: *assert* (argumentation-as-inference) and *listen* (revision). This realizes Gabbay's "logic for the 21st century" stance: logic that is interactive (speech acts as moves), defeasible (bases may conflict), dynamic (states evolve), and explainable (attacks/defences or equations justify outcomes) [51, 10].

In contemporary deployments of agentic AI—systems that perceive, decide, and act in tool-rich environments—the need for disciplined composition, dynamics, and explanation resurfaces as an engineering requirement. We instantiate Gabbay's toolbox and combination principles in a reusable pattern we call *Reasoning Alignment Diagrams* (RADs): commutative reasoning representations that align a source specification with an argumentation-based explanation path.

In what follows we first introduce RADs as a general template and discuss the role of this concept in existing applications of Dung's model of abstract argumentation. We then introduce *revision-as-argumentation* RADs to demonstrate that this role extends to belief revision as well. We prove an exact alignment result for revision-as-argumentation RADs (full-meet base revision via restricted attacks in base argumentation), and finally compose a "listen" (revision) RAD with an "assert" (inference) RAD to model dialogue—thus operationalising Gabbay's view of argumentation as the host formalism for interactive, dynamic, and explainable reasoning.

2 Reasoning Alignment Diagrams

In this section we introduce the notion of *reasoning alignment diagram* (RAD) as an architectural pattern for aligning different forms of reasoning. In simple terms, a RAD is a commutative diagram that shows how different forms of reasoning align in the sense that different 'routes' from input to output lead to the same result. As a general concept, a RAD may apply to different types of input and output, and each route between input and output serves a distinct purpose. For instance, the input may be a knowledge base, a state description, a query, a learned model or a combination thereof, while the output may be a set of sanctioned consequences, a prediction, an updated state description, and so on.

A classical example of a RAD arises in logic from the distinction between *semantic entailment* $\models$, typically defined in terms of truth preservation across models, and *proof-theoretic inference* $\vdash$, typically defined in terms of syntactic derivations (see Figure 1). Here, the purpose of the proof-theoretic route is to provide a syntactic, human-interpretable and typically mechanisable procedure that aligns with the semantic definition of entailment. In general, the strongest form of alignment in a RAD is where the outputs of the different routes are equivalent, meaning that the different routes differ only in *how* they arrive at this output. We call this *exact alignment*. In the semantic/proof-theoretic RAD, exact alignment amounts to soundness and completeness of the proof-theoretic inference route. Note that there are other ways to think about alignment, such as *partial alignment*, for instance if an inference procedure is sound but not complete, and *approximate alignment*, which may be defined in terms of a distance measure on outcomes or probabilistically.

$$\begin{tikzcd}
\text{Premises} \arrow[r, bend left, "\text{(1) \textit{Entailment} } (\models)"] \arrow[r, bend right, "\text{(2) \textit{Inference} } (\vdash)"'] & \text{Conclusions}
\end{tikzcd}$$

Figure 1: The entailment-as-inference RAD, showing how proof-theoretic inference (route 2) provides an alternative to semantic entailment (route 1).

Another application of the RAD concept is that one route provides a normative justification while another route provides explanations, i.e., one route 'explains' the other. Such RADs may involve more than two arrows to encode translation steps that are needed to obtain explanations. This aligns with how Dung's abstract argumentation is used, which we use in the remainder to develop the concept of RAD. We will discuss three examples of argumentation-based RADs. The first two are *inference-as-argumentation* and *argumentation-as-discussion*, and we discuss prior work that fit these two RAD concepts. The third, which we call *revision-as-argumentation*, is a completely new kind of RAD, where argumentation provides a means to explain the process of *base revision*, a specific form of belief change. In Section 3 we formalise this concept and provide a proof of exact alignment.

The RAD examples that we consider in this section deal with the alignment of different forms of symbolic reasoning. However, as a general concept, it can also be used to align subsymbolic approaches or a combination of symbolic and subsymbolic approaches. Some examples of such RADs will be discussed in Sections 4 and 5.

2.1 Inference-as-Argumentation

An *abstract argumentation framework* consists of a set of arguments together with a binary attack relation [35]. Arguments are treated abstractly, without internal structure, and the attack relation between arguments, where one argument attacks another if the former counts as a counterargument to the latter, determines whether a set of arguments is collectively acceptable. These sets, called *extensions*, represent coherent positions that can be adopted in light of conflicting information. Different criteria can be used to determine the extensions of an argumentation framework, and these criteria define a so-called semantics. Under this view, inference is defined relative to the extensions of the argumentation framework, for example by requiring that an argument appears in all (*skeptical acceptance*) or some (*credulous acceptance*) extensions under a given semantics.

Dung showed that various forms of defeasible inference can be represented within his theory of abstract argumentation. This elevates the abstract theory into a theory of *structured argumentation*, where an argumentation framework consists of arguments with structured content (e.g. premises, conclusion, and rules that derive the latter from the former) and attacks are induced by conflicts between arguments. This approach is depicted by the RAD shown in Figure 2 [71]. This diagram relates two approaches to deriving conclusions from a knowledge base. The first (arrow 1) is the 'canonical' route defined by the defeasible inference formalism. The other (arrows 2, 3 and 4) is the structured argumentation route. It starts with the translation (arrow 2) of the knowledge base into a structured argumentation framework. Then a semantics (arrow 3) determines the extensions, followed by the extraction of the conclusions of the arguments in the extensions (arrow 4).

$$\begin{tikzcd}
\text{knowledge base} \arrow[r, "(1)\ \textit{logic}"] \arrow[d, "\textit{(2) argument construction, attack assignment}"'] & \text{conclusion extension} \\
\text{argumentation framework} \arrow[r, "\textit{(3) arg. semantics}"] & \text{accepted arguments} \arrow[u, "\textit{(4) conclusion extraction}"']
\end{tikzcd}$$

Figure 2: The Inference-as-Argumentation RAD, where the purpose of the argumentation route (arrows 2-3-4) is to explain inferences performed by the source formalism (arrow 1).

Dung's approach can be understood as providing a general framework for *Inference-as-Argumentation* in the sense that the RAD shown in Figure 2 applies to different source formalisms (arrow 1) and corresponding translations (arrows 2 and 4). Dung showed that this scheme applies to Reiter's default logic, Pollock's defeasible reasoning, and logic programming with negation as failure. Others have shown that the approach applies to other forms of inference, such as reasoning with inconsistent knowledge bases using maximally consistent subsets [17, 28]. All of these can be represented as structured argumentation such that the conclusions arrived at using both routes (1 and 2-3-4) coincide. In addition, Dung presented a game-theoretic application of his theory, where arrow 1 corresponds to solving an instance of the stable marriage problem, while arrows 2-3-4 provide an equivalent solution in terms of extensions of an argumentation framework.

Dung's approach can be understood as an *abstraction* in the sense that computing extensions under a given semantics (arrow 3 in Figure 2) does not depend on the source formalism, or on the content of the constructed arguments. This abstraction is part of its power, as seemingly different forms of inference share the same underlying abstract model of argumentation, a fact that has generated various new insights.

Beyond abstraction, Dung's theory of argumentation provides *explanations*, that is, the derivation of conclusions via an argumentation framework and semantics (arrows 2-3-4) provides a way of *explaining* inferences in the source formalism (arrow 1) as a form of argumentation. An explanation for a conclusion is given by the arguments that justify the conclusion, while the extensions in which these arguments appear demonstrate that these arguments are part of a coherent position. The link between argumentation and explanation has also been investigated in the social sciences, where argumentation and explanation are seen as two closely related human activities [5]. This perspective underpins the role of Dung's model in the wider field of Explainable AI, where argumentation frameworks are used to make reasoning in complex systems more transparent [30, 66].

2.2 Argumentation-as-Discussion

The RAD perspective can also be used to model inference as argumentation at further levels of abstraction. Consider the problem of determining the extensions of an argumentation framework under a given semantics (arrow 3 in Figure 2). This problem has been reformulated in terms of *two-player discussion games*. Such discussion games have been defined to capture acceptance under a number of commonly used argumentation semantics (see [22] for an overview). These are two-player discussion games where the players are typically referred to as *proponent* and *opponent*. Starting from an initial argument put forward by the proponent, the proponent and opponent take turns attacking each other's arguments, with the proponent seeking to defend the claim. The exchange of arguments follows a fixed set of rules, defined in such a way that the existence of a winning strategy for the proponent proves that the initial argument is credulously or skeptically accepted. We can represent this concept with the *Argumentation-as-Discussion* RAD shown in Figure 3. In this RAD, arrow 1 represents the canonical way of determining the extensions of an argumentation framework under a given semantics. The other route (arrows 2-3-4) represents the computation of the extensions using a discussion game (we assume here that winning strategies correspond to extensions). Note that arrow 1 in Figure 3 corresponds to arrow 3 in Figure 2. Combining the RADs in Figure 3 and 2 provides a multi-layer explanatory perspective on inference, with the first layer providing explanations in terms of the extensions of argumentation frameworks, and the second layer in terms of a two-player discussion that establishes the acceptance of arguments.

$$\begin{tikzcd}
\text{argumentation framework} \arrow[r, "(1)\ \textit{arg. semantics}"] \arrow[d, "(2)\ \textit{define valid discussions}"'] & \text{accepted arguments} \\
\text{discussion game} \arrow[r, "(3)\ \textit{solve game}"] & \text{winning strategies} \arrow[u, "(4)\ \textit{argument extraction}"']
\end{tikzcd}$$

Figure 3: The Argumentation-as-Discussion RAD, showing how discussion games (arrows 2-3-4) explain argument acceptability (arrow 1).

2.3 Revision-As-Argumentation

Belief revision studies how a rational agent should update its beliefs in light of new, possibly conflicting, information [2, 31, 41]. The connection between argumentation and belief revision has been widely studied in the literature, grounded in the view that argumentation is an inherently dynamic process. We discuss work that connect argumentation and belief revision in the related work section. Surprisingly, however, the direct representation of belief revision itself in structured argumentation is largely unexplored. This representation is depicted in the RAD shown in Figure 4. The figure depicts the representation of a specific form of revision (full-meet base revision) as a specific form of structured argumentation (base argumentation), the details of which will be presented in the next section. One route (arrow 1) represents the canonical way of performing full-meet revision: given a belief base K (i.e., a finite, consistent set of formulas) and input ϕ, the revised belief base $K * \phi$ is obtained, in the full-meet case, by contracting $\neg\phi$ and then adding ϕ, where contraction by $\neg\phi$ amounts to taking the intersection of the maximal subsets of K that do not entail $\neg\phi$. The other route (arrows 2-3-4) provides an alternative argumentation route: constructing a base argumentation framework over $K \cup \{\phi\}$, (modifying attacks to encode the contraction step) (arrow 2), computing the extensions (arrow 3), and extracting the revised base (arrow 4). Like in the case of inference-as-argumentation, the argumentation route provides a way to explain the revision process. In the next section we formalise this approach, including the exact alignment of the revision and argumentation routes, thereby demonstrating the possibility of revision-as-argumentation.

$$\begin{tikzcd}
(K, \phi) \arrow[r, "(1)\ \textit{belief base revision}"] \arrow[d, "\begin{array}{c}(2)\ \textit{argument}\\ \textit{construction,}\\ \textit{attack assignment}\end{array}"'] & (K * \phi) \\
\begin{array}{c}\text{argumentation}\\ \text{framework}\end{array} \arrow[r, "(3)\ \textit{arg. semantics}"] & \begin{array}{c}\text{accepted}\\ \text{arguments}\end{array} \arrow[u, "\begin{array}{c}(4)\ \textit{conclusion}\\ \textit{extraction}\end{array}"']
\end{tikzcd}$$

Figure 4: The Revision-as-Argumentation RAD

3 Revision-as-Argumentation Formalised

We now formally develop the *revision-as-argumentation* RAD introduced in the previous section. Like formal argumentation, belief revision [2, 31, 41] is one of the major branches of knowledge representation and reasoning. Connections between formal argumentation and belief revision have been studied from various perspectives in the field of *dynamics of argumentation* [20]. We discuss some of this work in the related work section. However, and perhaps surprisingly, there have been no results to date showing how belief revision itself can be represented as a form of structured argumentation. We believe that the absence of representation results for belief revision as a form of formal argumentation indicates that a general result may be difficult to obtain. Instead, a more promising approach is to seek results connecting specific types of belief revision with particular forms of structured argumentation, which is precisely what we do in this section.

Building on the observation that formal argumentation often operates with sets of formulas that are not deductively closed, we focus on revising such sets: a process referred to as *base revision.* In base revision, a belief base K, a consistent and finite set of formulas, is revised with a new piece of information ϕ, which may be inconsistent with K, to obtain a revised belief base $K * \phi$. Various constructive [56] and postulate-based characterizations [2] of the revision operator $*$ have been proposed. We focus on the simplest form of base revision: *full-meet* base revision [52]. As the formalism for structured argumentation, we adopt *base argumentation*, a recently introduced formalism that is structurally simpler than traditional approaches. Base argumentation represents arguments as minimal consistent subsets of a belief base, with attacks defined by logical contradiction between premises. Despite lacking explicit conclusions, it is equivalent to premise-conclusion argumentation under standard Dung-style semantics. This equivalence is formally established via bisimulation, as shown by Chen et al. [28]. We will show in this section that full-meet belief base revision can be represented as a kind of base argumentation, thus instantiating the RAD shown in Figure 4. Our aim is to use structured argumentation not merely to reproduce the revised outcome, but to *explain* the revision process by generating an argumentation framework and computing its extensions that yield the same result. Our approach instantiates steps (1, . . . , 4) in Figure 4 as follows.

Belief Base Revision: Full-meet base revision of a consistent base K by ϕ is defined via remainder sets: we define $K * \phi = (K - \neg\phi) + \phi$, where contraction $K - \neg\phi$ is the intersection of all remainders of K that do not imply $\neg\phi$.

Argument and attack assignment: Given the belief base $K \cup \{\phi\}$, we construct a base argumentation framework where arguments are minimal consistent subsets of $K \cup \{\phi\}$, and attacks are defined by logical inconsistency. Crucially, we use a *restricted attack relation* $\Rightarrow^{-X}$, which disables attacks originating from arguments entirely within a chosen remainder set X, effectively modeling the contraction step.

Argumentation semantics: We compute the *stable extensions* of this modified base argumentation framework $F(K * \phi) = (\rho(K \cup \{\phi\}), \Rightarrow^{-(K \setminus \{\phi\})})$. Each stable extension corresponds to one maximal consistent subset S of $K \cup \{\phi\}$ that includes ϕ.

Conclusion extraction: The revised base $K * \phi$ is then obtained by intersecting all stable extensions. Thus, the conclusion drawn from the argumentation framework matches the one derived via the belief revision operation.

We first present the necessary definitions of abstract argumentation [35], base argumentation [28] and base revision [52] that we use. We work with the logic $(\mathcal{L}, \vdash)$ where $\mathcal{L}$ is a propositional language defined as usual, and $\vdash$ is the associated consequence relation. Given a set $K \subseteq \mathcal{L}$ we define $Cn(K)$ by $Cn(K) = \{\phi \in \mathcal{L} \mid K \vdash \phi\}$.

3.1 Abstract Argumentation

An abstract argumentation framework is defined as a pair of a set of arguments, and a binary relation representing the attack relationship between arguments [35].

DEFINITION 3.1. *An argumentation framework is a pair $F = (\mathbf{A}, \Rightarrow)$ where $\mathbf{A}$ is a set of arguments and $\Rightarrow \subseteq \mathbf{A} \times \mathbf{A}$ the attack relation.*

Various semantics have been defined as criteria for deciding which sets of arguments are collectively acceptable. For our purpose we will use the *stable semantics.*

DEFINITION 3.2. *Let $F = (\mathbf{A}, \Rightarrow)$ be an argumentation framework, and let $E \subseteq \mathbf{A}$. We say that E is* conflict-free *if there are no $x, y \in E$ such that $(x, y) \in \Rightarrow$, and that E is* stable *if it is conflict-free and attacks every $x \in \mathbf{A} \setminus E$. We denote by $st(F)$ the set of stable extensions of F.*

3.2 Base Argumentation

In most of the instances of inference-as-argumentation (see Figure 2) based on Dung's model of abstract argumentation, an argument is either a recursive structure containing premises, rules and a conclusion, or a pair (Γ, ϕ) where Γ is a set of premises and ϕ is the conclusion. The *base argumentation* formalism was introduced as a simpler and more compact alternative, where a *base argument* is simply a finite and consistent set Γ of formulas. We can think of a base argument Γ as representing the set of all premise-conclusion pairs (Γ, ϕ) such that ϕ follows from Γ. It was shown in [28] that this simpler approach is *extensionally equivalent* to the *deductive argumentation* formalism that uses premise-conclusion pairs [17]: via a simple mapping between base arguments and premise-conclusion pairs, the two induce the same extensions.

Formally, a *base* is a (possibly inconsistent) set $K \subseteq \mathcal{L}$ of formulas that acts as a knowledge base. A common approach to reason with an inconsistent base K is to look at the maximally consistent subsets $MC(K)$ of K defined by

$$MC(K) = \{K' \subseteq K \mid K' \nvdash \bot \text{ and } \forall K'' \text{ s.t. } K' \subset K'' \subseteq K, K'' \vdash \bot\}.$$

Base argumentation provides a way to represent this kind of reasoning as a form of structured argumentation. This corresponds to the RAD depicted in Figure 2, where the source formalism (arrow 1) amounts to computing $MC(K)$ and the argumentation route (arrows 2-3-4) constructing the argumentation framework F_K (arrow 2), computing the stable extensions of F_K (arrow 3), and converting the stable extensions to sets of formulas (arrow 4). The basic definitions for base argumentation are as follows [28].

DEFINITION 3.3. *Let $K \subseteq \mathcal{L}$ be base. A* base argument *of K is a finite set $\Gamma \subseteq K$ that is consistent ($\Gamma \nvdash \bot$) such that there exists no $\Gamma' \subset \Gamma$ with $Cn(\Gamma') = Cn(\Gamma)$. We denote by $\rho(K)$ the set of base arguments of K.*

A base argument Γ attacks another base argument Γ' when Γ entails the negation of some premise in Γ'.

DEFINITION 3.4. *Let $K \subseteq \mathcal{L}$ and let Γ, Γ' be two base arguments of K. Then Γ attacks Γ' iff $\Gamma \vdash \neg\phi$ for some $\phi \in \Gamma'$. We define the attack relation $\rightarrow_K \subseteq \rho(K) \times \rho(K)$ by $\Gamma \rightarrow_K \Gamma'$ iff Γ attacks Γ'.*

We can now define how a base induces a *base argumentation framework.*

DEFINITION 3.5. *The base argumentation framework induced by the set $K \subseteq \mathcal{L}$ is the argumentation framework $F_K = (\rho(K), \rightarrow_K)$.*

EXAMPLE 3.6. *Consider the base $K = \{p, \neg p, q\}$. The base argumentation framework F_K is the pair $(\rho(K), \rightarrow_K)$, where $\rho(K) = \{\emptyset, \{p\}, \{\neg p\}, \{q\}, \{p, q\}, \{\neg p, q\}\}$ and $\rightarrow_K = \{(\{p\}, \{\neg p\}), (\{p\}, \{\neg p, q\}), (\{\neg p\}, \{p\}), (\{\neg p\}, \{p, q\}), (\{p, q\}, \{\neg p\}), (\{p, q\}, \{\neg p, q\}), (\{\neg p, q\}, \{p\}), (\{\neg p, q\}, \{p, q\})\}$.*

The following result (Proposition 3.7) was established by Chen et al. [28]: the stable extensions of the base argumentation framework F_K correspond to the maximal consistent subsets of K. This result establishes the exact alignment of an instance of the RAD depicted in Figure 2 for the case of base argumentation.

PROPOSITION 3.7. *[28] Let $K \subseteq \mathcal{L}$.*

1. *If $S \in MC(K)$ then $\rho(S) \in st((\rho(K), \rightarrow_K))$.*
2. *If $E \in st((\rho(K), \rightarrow_K))$ then $E = \rho(S)$ for some $S \in MC(K)$.*

3.3 Base Revision

Base revision refers to the revision of a base K with a new piece of information ϕ. In the remainder we assume that K is initially consistent, but that K is not necessarily consistent with ϕ. Base revision is modeled by a base revision operator $*$ where $K * \phi$ represents a revision of K with ϕ. Various constructive and postulate-based characterisations for a base revision operator $*$ have been considered (see, e.g., [52]). We present here the constructive definition of the *full-meet revision* operator. First, a *remainder* of K with respect to ϕ is a maximal subset of K that does not imply ϕ [3]:

DEFINITION 3.8 (Remainders)**.** *Let $K \subseteq \mathcal{L}$ and $\phi \in \mathcal{L}$. The set of* remainders *of K with respect to ϕ, denoted $K \perp \phi$, is the set of all subsets $K' \subseteq K$ such that $K' \nvdash \phi$, and there is no K'' with $K' \subset K'' \subseteq K$ and $K'' \nvdash \phi$.*

Revision is defined in terms of *expansion* and *contraction* [2]. Expansion refers to the addition of a belief without checking consistency. Full-meet contraction takes the intersection of all maximal subsets of the original belief set that do not entail the proposition being contracted. Full-meet revision of K with ϕ is defined by the contraction of K by $\neg\phi$ followed by the expansion with ϕ [2].

DEFINITION 3.9. *Let $K \subseteq \mathcal{L}$ and $\phi \in \mathcal{L}$.*

- *The* expansion *of K by ϕ is defined as: $K + \phi = K \cup \{\phi\}$.*
- *The* full-meet contraction *of K by ϕ is defined as: $K - \phi = \bigcap(K \perp \phi)$.*
- *The* full-meet revision *of K by ϕ is defined as: $K * \phi = (K - \neg\phi) + \phi$.*

In what follows we will focus on the operation of full-meet revision as defined above. For further discussion and motivation for these operators, including their characterisation in terms of postulates, we refer the reader to [52].

3.4 Explaining Base Revision Using Base Argumentation

We now show how to model the revision $K * \phi$ as a form of base argumentation. Below we define, given a base K and set $X \subseteq K$, the attack relation $\Rightarrow_X$. In words, given two arguments Γ, Δ the attack $\Gamma \Rightarrow_X \Delta$ holds whenever $\Gamma \rightarrow_K \Delta$ (i.e., Γ attacks Δ according to Definition 3.4) and Γ is not constructed only from elements of X.

DEFINITION 3.10. *Let $K \subseteq \mathcal{L}$ and $X \in MC(K)$. We define the attack relation $\Rightarrow_X \subseteq \rho(K) \times \rho(K)$ by $\Gamma \Rightarrow_X \Delta$ iff $\Gamma \rightarrow_K \Delta$ and $\Gamma \not\subseteq X$.*

Proposition 3.7 states that, if K is inconsistent then The following lemma states that, if K is inconsistent and X is a maximally consistent subset of K, then the base argumentation framework constructed using the attack relation $\Rightarrow_X$ excludes X.

LEMMA 3.11. *If $K \subseteq \mathcal{L}$ is consistent then for every $X \subseteq K$, $st((\rho(K), \Rightarrow_X)) = \{\rho(K)\}$. Now suppose K is inconsistent and let $X \in MC(K)$.*

1. *If $S \in MC(K) \setminus \{X\}$ then $\rho(S) \in st((\rho(K), \Rightarrow_X))$.*
2. *If $E \in st((\rho(K), \Rightarrow_X))$ then $E = \rho(S)$ for some $S \in MC(K) \setminus \{X\}$.*

Proof. If K is consistent then $\Rightarrow_X = \emptyset$ for every $X \subseteq K$. It then follows that $st((\rho(K), \Rightarrow_X)) = \{\rho(K)\}$. Now assume that K is inconsistent. Let $X \in MC(K)$.

(1) Suppose $S \in MC(K) \setminus \{X\}$. We show that $\rho(S) \in st((\rho(K), \Rightarrow_X))$. We first have that $\rho(S)$ is conflict-free. Otherwise, there exist $\Gamma, \Gamma' \in \rho(S)$, $\Gamma \Rightarrow_X \Gamma'$ and hence $\Gamma \rightarrow_K \Gamma'$. But this implies that S is not consistent, which is false, hence $\rho(S)$ is conflict-free. Now let $\Delta \in \rho(K) \setminus \rho(S)$. Since $S \in MC(K)$, we know $S \cup \Delta$ is inconsistent. Hence there is a $\Gamma \in \rho(S)$ with $Cn(\Gamma) = Cn(S)$ and $\Gamma \rightarrow_K \Delta$. Since $S \neq X$, $\Gamma \not\subseteq X$ and hence $\Gamma \Rightarrow_X \Delta$. Therefore $\rho(S)$ attacks every $\Delta \in \rho(K) \setminus \rho(S)$. It follows that $\rho(S) \in st((\rho(K), \Rightarrow_X))$.

(2) Let $E \in st((\rho(K), \Rightarrow_X))$. Let $S = \cup E$. We will show that (i) $E = \rho(S)$, (ii) $S \in MC(K)$, and (iii) $S \neq X$.

(i) We will show that $E = \rho(S)$. Since K is inconsistent we have that $E \subset \rho(K)$. We first prove that $E = \rho(S)$. Suppose $\Gamma \in \rho(S)$. Suppose for contradiction that $\Gamma \notin E$. Then since E is stable, there is a $\Delta \in E$ s.t. $\Delta \Rightarrow_X \Gamma$. Then $\Delta \vdash \neg\phi$ for some $\phi \in \Gamma$. But since $\Gamma \in \rho(S)$ and $E = \rho(S)$ it follows that there is a $\Gamma' \in E$ such that $\phi \in \Gamma'$. It follows that $\Delta \rightarrow_K \Gamma'$ and (since $\Delta \Rightarrow_X \Gamma$) also $\Delta \Rightarrow_X \Gamma'$. Since $\Delta, \Gamma' \in E$ it follows that E is not conflict-free, which is false. Hence $\Gamma \in E$ and it follows that $E = \rho(S)$.

(ii) We now prove that $S \in MC(K)$. We first prove that S is consistent. Let $\{\Gamma_1, \ldots, \Gamma_n\} = MC(S)$. Since $E = \rho(S)$ it follows that $\Gamma_1, \ldots, \Gamma_n \in E$. If $n = 1$ then $\Gamma_1 = S$ is consistent and we are done. Now suppose for contradiction that $n > 1$. Let $\Delta \in \rho(K) \setminus E$ (existence of Δ follows our assumption that $E \subset \rho(K)$). Stability of E implies that there is a $\Delta' \in E$ such that $\Delta' \Rightarrow_X \Delta$. Since $\Delta' \in \Gamma_i$ for some i it follows that $\Delta' \rightarrow_K \Gamma_j$ for some $i \neq j$. Then, since $\Delta' \Rightarrow_X \Delta$, we also have $\Delta' \Rightarrow_X \Gamma_j$. Since $\Delta, \Gamma_j \in E$ it follows that E is not conflict-free, which is false. Hence S is consistent. Now assume for contradiction that there is consistent S' such that $S \subset S' \subseteq K$. Let $\phi \in S' \setminus S$. Then $\{\phi\} \in \rho(K)$. Since $E = \rho(S)$ it follows that $\{\phi\} \notin E$. Stability of E then implies that for some $\Gamma \in E$, $\Gamma \Rightarrow_X \{\phi\}$ and hence $\Gamma \rightarrow_K \{\phi\}$. This implies that $S \vdash \neg\phi$ which is false since S' is consistent. Hence $S \in MC(K)$.

(iii) We now prove that $S \neq X$. Suppose for contradiction that $S = X$. Let $\Delta \in \rho(K) \setminus E$ (existence of Δ follows our assumption that $E \subset \rho(K)$). Stability of E implies that there is a $\Delta' \in E$ such that $\Delta' \Rightarrow_X \Delta$. But since $\Delta' \subseteq S$ and $S = X$ we have $\Delta' \subseteq X$ and hence $\Delta' \not\Rightarrow_X \Delta$, which is a contradiction. Hence $S \neq X$.

□

We can use this lemma to model full-meet revision of a consistent belief base. Recall that full-meet revision is defined by $K * \phi = (K - \neg\phi) + \phi$. We define the argumentation framework $F_{(K*\phi)}$ by

$$F_{(K*\phi)} = (\rho(K \cup \{\phi\}), \Rightarrow_{(K \setminus \{\phi\})}).$$

The following theorem establishes the link between full-meet base revision $K * \phi$ and the stable extensions of the argumentation framework $F_{(K*\phi)}$.

THEOREM 3.12. *For every consistent $K \subseteq \mathcal{L}$ and $\phi \in \mathcal{L}$:*

$$K * \phi = \bigcap \{\cup_{\Gamma \in E} \Gamma | E \in st(F_{(K*\phi)})\}.$$

Proof. Let $K \subseteq \mathcal{L}$ be a consistent belief base and let $\phi \in \mathcal{L}$ be a consistent formula. We then have

$$\begin{aligned}
K * \phi &= (K - \neg\phi) + \phi && (1)\\
&= (\bigcap K \bot \neg\phi) + \phi && (2)\\
&= \{\phi\} \cup \bigcap (K \bot \neg\phi) && (3)\\
&= \bigcap (\{\phi\} \cup K' | K' \in K \bot \neg\phi) && (4)\\
&= \bigcap \{K' \in MC(K \cup \{\phi\}) | \phi \in K'\} && (5)\\
&= \bigcap \{\cup_{\Gamma \in E} \Gamma | E \in st(\rho(K \cup \{\phi\}), \Rightarrow_{K \setminus \{\phi\}})\} && (6)\\
&= \bigcap \{\cup_{\Gamma \in E} \Gamma | E \in st(F_{(K*\phi)})\} && (7)
\end{aligned}$$

Justification: In steps (1), (2) and (3) we apply, respectively, the definitions of revision, contraction and expansion. Step (4) follows directly. For (5) we will prove that $(\{\phi\} \cup K' | K' \in K \bot \neg\phi) = \{K' \in MC(K \cup \{\phi\}) | \phi \in K'\}$:

($\subseteq$) Suppose $L \in (\{\phi\} \cup K' | K' \in K \bot \neg\phi)$. Two cases:

Case 1: $\phi \in K$. Then $\forall K' \in K \bot \neg\phi$, $\phi \in K'$. Hence (by our assumption) $L \in K \bot \neg\phi$. Then there is no K' such that $L \subsetneq K' \subseteq K$ and $K' \vdash \neg\phi$ and hence (since $\phi \in K'$) no such K' such that $K' \vdash \bot$. This implies that $L \in MC(K \cup \{\phi\})$ and $\phi \in L$.

Case 2: $\phi \notin K$ Then $K \setminus \{\phi\} \in K \bot \neg\phi$. Suppose for contradiction that $L \notin MC(K \cup \{\phi\})$. Then let K' be s.t. $L \subset K' \subseteq K$ s,t. $K' \in MC(K \cup \{\phi\})$. Then $K' \nvdash \neg\phi$ (since $K' \vdash \neg\phi$ would imply $K' \vdash \bot$) and hence $K' \in K \bot \neg\phi$, which is a contradiction.

It follows that $L \in MC(K \cup \{\phi\})$ and $\phi \in K'$.

($\supseteq$) Suppose $L \in \{K' \in MC(K \cup \{\phi\}) | \phi \in K'\}$. Then $L \in MC(K \cup \{\phi\})$ and $\phi \in L$. Suppose $L \notin K \bot \neg\phi$. Then there exists K' with $L \subset K' \subseteq K \cup \{\phi\}$ s.t. $K' \nvdash \neg\phi$. But then K' is a consistent subset of $K \cup \{\phi\}$, contradicting $L \in MC(K \cup \{\phi\})$. It follows that $L \in K \bot \neg\phi$ and hence $L \in (\{\phi\} \cup K' | K' \in K \bot \neg\phi)$.

It follows that $\bigcap(\{\phi\} \cup K' | K' \in K \bot \neg\phi) = \bigcap\{K' \in MC(K \cup \{\phi\}) | \phi \in K'\}$, which proves step (5). Step (6) follows from Lemma 3.11 and step (7) from the definition of $F_{(K*\phi)}$. □

This result establishes a direct correspondence between full-meet belief base revision and structured argumentation, thereby proving the exact alignment of the RAD depicted in Figure 4. We thus show that this form of belief revision can be naturally represented within a structured argumentation framework, which underlines Dov Gabbay's vision of argumentation as a transparent *host formalism* where heterogeneous mechanisms live together [51, 10].

4 Outlook: Multi-RAD Systems and Neuro-symbolic RADs for Agentic AI

In this section we present our vision for how *Reasoning Alignment Diagrams* extend to AI reasoning in the sense of *agentic AI.* At the heart of alignment is making a logical (symbolic) reasoner and a subsymbolic reasoner cohere; our outlook is to develop RADs for *subsymbolic and neurosymbolic* components. In Section 4.1 we state our general vision of *combining RADs* (Multi-RADs) and use the composition of inference-as-argumentation with revision-as-argumentation as an example. In Section 4.2, we use the composition for single-agent decision making. In Section 4.3 we envision a *principle-based* approach for specifying and verifying Multi-RAD systems at their input–output interfaces, independent of internal implementation.

4.1 Multi-RAD for Dialogue

In this section, we take a broader perspective and envision how Inference-as-Argumentation RAD and Revision as Argumentation RAD can be combined for dialogue between agents. The idea is that belief revision and argument generation are essential components of *listening* and *asserting.* This duality allows us to interpret dialogue protocols as the composition of two commutative diagrams. A simple form of dialogue, such as that used in chatbot interaction, can be modeled by speech acts that assert a formula ϕ, thereby informing another agent. This interaction involves belief revision on the receiving side and argument generation on the asserting side. We analyze this kind of dialogue from the perspective of formal argumentation by combining argumentation-as-inference (Figure 2) and argumentation-as-revision (Figure 4), thereby integrating the explanatory structure of base revision with the generative structure of argumentation.

Figure 5 illustrates how revision and inference interact in dialogue. There are three agents, α, β, and γ, each with its own knowledge base: KB_α, KB_β, and KB_γ, respectively. The figure is structured as a temporal and concurrent model: vertical swimlanes represent the internal evolution of each agent's knowledge over time, while horizontal arrows indicate inter-agent communication via speech acts. The process begins with Agent α, who performs argumentation as inference over its knowledge base KB_α. Based on the resulting extension E, α asserts a formula ψ to Agent β. Upon receiving ψ, Agent β listens and revises its knowledge base from KB_β to KB'_β. From this updated base, β constructs a new argumentation framework, infers an extension E, and generates a new assertion δ. In a multi-agent context, an agent may interact with several other agents. Here, Agent γ also asserts δ to Agent β, who integrates it via another revision step from $KB'\beta$ to $KB''\beta$. This update leads to the generation of a new assertion λ, which is sent to Agent α. Agent α then revises its own knowledge base accordingly, from KB_α to KB'_α.

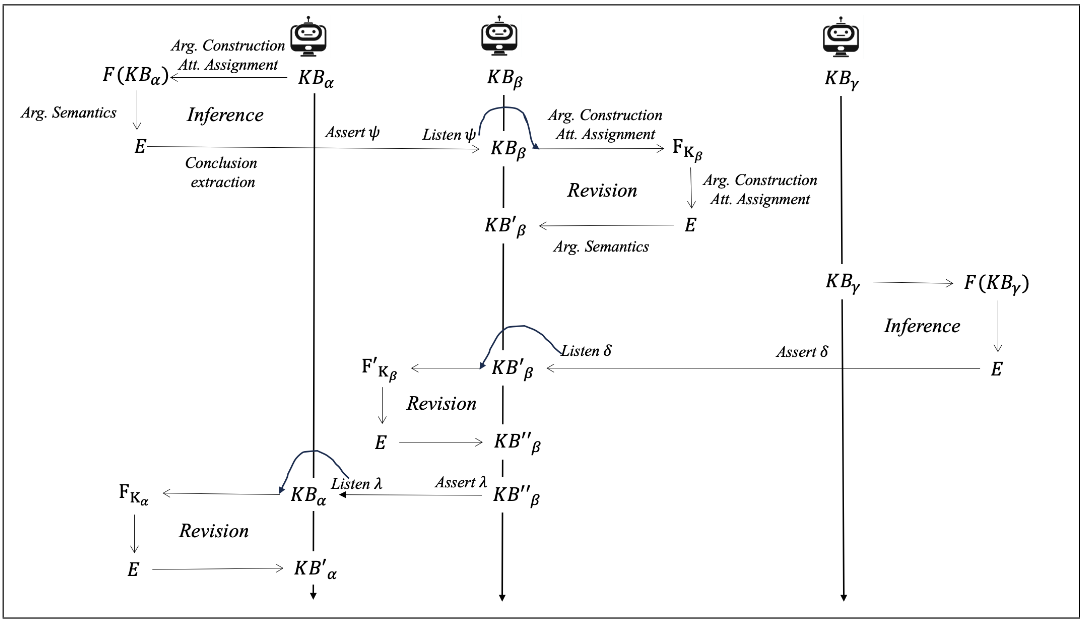


Figure 5: Multi-RAD for dialogue.

4.2 Multi-RAD for Individual Agent Decision Model

In Figure 5, we illustrated combining two RADs in a dialogue context. In this section, we turn to the agent decision model depicted in Figure 6. Whereas Figure 5 considers multiple agents and the evolution of time, and is thus mainly concerned with the synchronization of the agents, Figure 6 examines the interaction from the perspective of a single agent at a fixed point in time.

From the perspective of a single agent α, that agent can either assert something or listen to other agents. If α decides to make an assertion, it must also choose strategically *what* to say. Alternatively, if α is listening and another agent informs it of some proposition ψ, then α has several options: it can revise its knowledge base with the new information, ignore ψ, question the other agents about it by asking for a justification, challenge ψ, and so on. Such agent communication languages based on speech acts have been developed, for example, by FIPA [42].

The strategic decision-making aspect of the dialogue can be implemented using a traditional or a qualitative decision theory [32], as well as techniques from generative

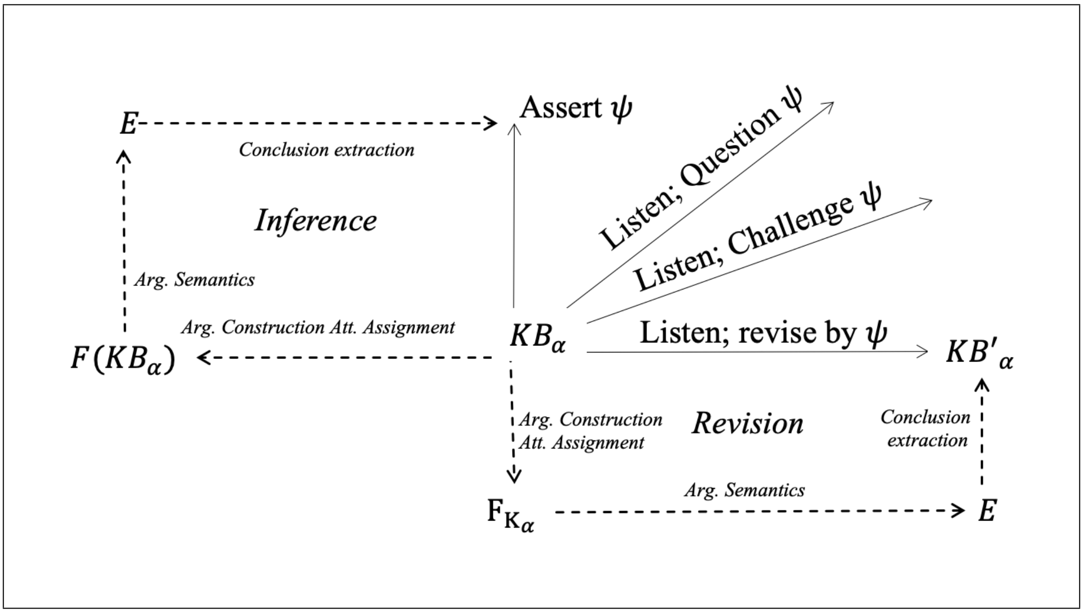


Figure 6: Multi-RAD for individual agent decision

AI. In fact, techniques from neurosymbolic and agentic AI can be adopted. The main question is control: does the symbolic layer or the subsymbolic layer decide?

Moreover, multiple types of dialogues [67] may be distinguished. In persuasion and deliberation dialogues, agents can question or challenge each other's assertions, whereas in information-seeking dialogues, agents primarily ask questions (including requests for clarification). Once an agent is equipped with a range of communicative actions, a meta-level control mechanism is required for deciding when to switch between these actions.

Building on this multi-agent dialogue model, one can investigate a wide range of topics discussed in the multi-agent systems community. For instance, the model could be extended in the direction of hybrid human–machine intelligence, using hybrid human+machine argumentation and notions of cognitive delegation [43] to address questions of autonomy and discretion. Finally, shifting focus from the inter-agent protocol to the viewpoint of a single agent raises broader philosophical questions about agency, which would need to be addressed as this line of research progresses.

4.3 Multi-RAD for Agentic AI

Until now we have shown how individual RADs model inference and revision as argumentation, and how they compose into dialogue and decision-making. Our outlook now moves to apply *Multi-RAD systems* to AI reasoning in the sense of agentic AI.

Agentic AI concerns capabilities such as autonomy, goal pursuit, planning, tool use, and interaction [1]. These capabilities do not presuppose a single implementation style but in hybrid combinations. What matters for us is how the underlying reasoning routes are *aligned* and *composed.* The following example from Gabbay [49] illustrates why multiple kinds of reasoning often co-occur and must be coordinated.

EXAMPLE 4.1 (Untidy room [49])**.** *A mother goes into her teenage daughter's bedroom. Her instant impression is that it is a big mess. There is stuff scattered everywhere. The mother's feeling is that it is not like her daughter to be like this. What happened?*

Conjecture: *The girl may be experiencing boyfriend issues.*

Further Analysis: *The mother notices a collapsed shelf and realizes that the disarray is due to the shelf collapsing under excessive weight which, upon reflection, follows a logical (gravitational) pattern.*

Several types of reasoning are illustrated through this scenario:

Neural network reasoning: *The mess is perceived instantly, similarly to facial recognition by neural networks.*

Nonmonotonic deduction: *The mother deduces from the context and her knowledge that her daughter does not typically live in disarray. Thus, something extraordinary must have happened.*

Abductive reasoning: *She hypothesizes a plausible explanation that her daughter has social-emotional issues, which is common among teenagers.*

Database AI deduction: *A reevaluation leads to the understanding that the mess is due to gravitational effects rather than disorganization on the part of her daughter.*

Pattern recognition: *Someone accustomed to similar patterns may identify the cause as easily as they might recognize a face.*

This example illustrates the need for *hybrid* reasoning, where symbolic and subsymbolic approaches complement one another. By *subsymbolic* reasoning we mean parameterised function-approximation methods (e.g., neural networks and pattern recognition) that support fast perception and intuitive classification; by *symbolic* reasoning we mean rule- or graph-based methods (e.g., deduction, abduction, database-style re-evaluation) that support structured analysis, logical constraints, and explanations [39]. Neither alone suffices: subsymbolic methods lack transparency, while symbolic methods lack perceptual grounding and flexibility.

To address this, we outline *Multi-RAD systems*: architectures that interconnect several RADs so that each route aligns a source specification with an argumentation-based explanation, and the composed diagram preserves alignment from input to outcome. In practice this means (i) supporting heterogeneity (symbolic and/or subsymbolic components), (ii) ensuring compositionality (routes do not break one another), and (iii) maintaining alignment (each route has a specification and an explanation path that commute).

Multi-RAD systems can embody several conceptualisations of formal argumentation —*inference*, *dialogue*, and *balancing*—which should not be treated in isolation (see [71]). A higher-level metamodel such as A-BDI [70] provides the necessary abstraction, showing how these conceptualisations relate. The next step is a methodology for specifying and verifying such systems, which we introduce below.

4.4 The principle-based approach for multi-RAD systems

To manage the diversity of reasoning methods, we use *principle-based analysis* as a methodology for selecting among existing methods or designing new ones. The idea is to describe mechanisms at a higher level of abstraction, focusing not on their implementation but on the *properties* they satisfy. In mathematics, abstraction extracts the underlying structures of a concept; in computer science, it generalises from concrete details to reusable specifications. Principle-based analysis applies this idea to reasoning.

This approach has a long tradition across domains. In voting theory, Arrow's axioms [8] specify criteria for voting systems and yield impossibility results. In belief revision, AGM postulates ensure the rationality of revision operations [2]. In nonmonotonic logic, Gabbay discussed the central requirements reflexivity, cut, and cautious monotony [44]. In formal argumentation, the principle-based view has been applied at the different stages of the Inference-as-Argumentation RAD in Figure 2. For arrow (1), the Kraus–Lehmann–Magidor (KLM) principles [57] axiomatise properties a nonmonotonic consequence relation ought to satisfy. For arrows (2)–(4), axiomatic analyses compare *attack assignments* among arguments [36, 38, 59]. For the whole RAD, *rationality postulates* guarantee that the overall conclusions satisfy desirable properties such as direct/indirect consistency and closure [24, 25, 23].

For abstract argumentation, the semantics has been classified via principles [11], with further extensions in [65]. The same methodology carries over to extended argumentation frameworks, yielding principle-based analyses for ranking-based semantics [4], gradual semantics [12, 18], multi-agent argumentation [69], and bipolar argumentation [68], etc.

Principle-based approach supports two main aims. First, it can be used to *define new input–output behaviours* of reasoning by selecting the principles one wishes to enforce as desiderata. Second, it can be used to *compare existing input–output behaviours* by identifying which principles they satisfy or fail to satisfy. Beyond comparison, principle-based analysis can yield *characterisation theorems*, where a given set of principles uniquely determines a function, and *impossibility results*, which show that no function can satisfy certain sets of principles simultaneously.

In the age of agentic AI, the same style of analysis must also apply to *subsymbolic* components. Even if a component is opaque internally, we can still analyse its input–output behaviour against principles. This connects naturally with ongoing work on verification of machine learning models [55]: for example, proving that a classifier respects monotonicity, fairness, or robustness constraints. In this way, the principle-based approach provides a unifying layer for both symbolic and subsymbolic components. It extends the toolbox idea: principles allow us not only to combine mechanisms but also to constrain and govern them, ensuring that even black-box modules contribute to explainable and accountable agentic systems.

5 Related work

We introduced reasoning alignment diagrams (RADs) as an architectural pattern for aligning different forms of reasoning. Our main technical contribution is the introduction of the revision-as-argumentation RAD and a proof of alignment for the specific case of base revision and base argumentation. The connection between argumentation and belief revision has been widely recognised in the literature, grounded in the view that argumentation is an inherently dynamic process [9, 40, 58]. Both frameworks aim to model rational change in light of new information, and argumentation has been proposed as a natural mechanism for capturing belief dynamics and resolving inconsistencies. Under this perspective, belief addition corresponds to introducing new arguments, contraction may involve removing arguments or introducing counterarguments, and the evolving structure of the argumentation framework reflects the agent's shifting epistemic state.

This conceptual link has been explored from multiple directions. A substantial body of work investigates change in abstract argumentation frameworks, examining how adding or removing arguments and attacks affects extension sets under various semantics [14, 13, 26, 29, 33, 34, 64]. Other lines of research model change in argumentation systems explicitly as a form of belief revision, drawing analogies with AGM-style postulates and operators [21, 15, 54]. Structured argumentation has also been studied in this context, with attention to how changes to the underlying knowledge base or rule set relate to changes in the resulting argument graph [16, 61, 62]. Despite these efforts, representation results for belief revision as a form of formal argumentation are lacking. In Section 3 we provided such a representation result establishing a direct correspondence between full-meet belief base revision and base argumentation. To the best of our knowledge, this is the first work to show that this form of belief revision can be naturally represented within a structured argumentation framework.

$$\begin{tikzcd}
(M,x) \arrow[rr, "\textit{(1) apply } M"] \arrow[dr, "\textit{(2) train surrogate}"'] & & M(x) \approx S_x(x) \\
& S_x = (w_1,\ldots,w_n) \arrow[ur, "\textit{(3) apply } S_x"'] &
\end{tikzcd}$$

Figure 7: The *Interpretable Surrogate* RAD: prediction $M(x)$ of an opaque model M (arrow 1) is explained by training a linear model S_x (arrows 2, 3). The coefficients $w_1, \ldots, w_n$ of S_x represent feature importance values for the prediction $M(x)$.

In this paper we used RADs to represent different forms of symbolic reasoning, where one route provides an argumentation-based explanation for another. Yet the RAD idea is more general and also applies to explanation in subsymbolic approaches. A good example is the *surrogate-model* approach popularised by LIME in the field of Explainable AI [63] (see Figure 7). Given an opaque model M (e.g., a neural network) and an input x, the aim is to provide an explanation for the prediction $M(x)$. One route simply computes $M(x)$ without providing any explanation. The alternative route proceeds in two steps: first, an interpretable surrogate S_x is trained to be *locally faithful* to M (meaning that $M(x) \approx S_x(x)$ for inputs 'close to' x); second, the prediction $S_x(x)$ is obtained, where local faithfulness ensures (approximate) alignment of the RAD. The point of the alternative route is not to provide an alternative way to obtain the prediction, but rather to obtain explanations for the prediction made by the source model. This is typically done by using a linear classifier for the interpretable surrogate, and using the regression coefficients $(w_1, \ldots, w_n)$ as feature importance values. This example shows that the RAD concept can capture explanation even in purely subsymbolic settings.

6 Summary and Conclusion

Our aim was to advance Dov Gabbay's vision of *logic as a toolbox*, of *disciplined combinations*, and of *argumentation as the host formalism* by developing a unifying pattern for aligning different forms of reasoning based on *Reasoning Alignment Diagrams* (RADs). RADs are commutative reasoning representations in which distinct routes from input to output align, enabling the separation of a *normative path* that specifies what is sanctioned, and an *argumentation path* that explains why. The pattern admits exact, partial, and approximate alignment. Routes consisting of multiple steps make explicit the translation from knowledge bases to arguments and attacks, the choice of Dung-style semantics, and the extraction of conclusions from extensions.

We showed that existing work grounded in Dung's model fits the *Inference-as-Argumentation* RAD, which explains defeasible logics via the structured argumentation route (construct an AF, apply semantics, extract conclusions). *Argumentation-as-Discussion* adds a further explanatory layer, by recasting acceptance under a semantics in terms of two-player discussion games with proponent/opponent strategies. Together, these perspectives illustrate how argumentation provides both an abstraction across heterogeneous formalisms and an explanation engine for their outcomes.

Our new contribution is the *Revision-as-Argumentation* RAD that extends this role beyond inference to belief change. We considered full-meet belief *base* revision of a consistent K by φ and showed exact alignment with base argumentation. The construction builds a base-AF over $K \cup \varphi$ and introduces a *restricted-attack* relation that disables attacks originating from arguments contained in a chosen remainder set. We proved that the revised base coincides with the intersection of the premises that appear in all stable extensions of the modified framework. Intuitively, contraction by $\neg\varphi$ is realised by restricting attacks, while expansion by φ is captured by including φ in the base. This approach provides, to our knowledge, the first exact representation of full-meet base revision within base argumentation.

Methodologically, the RAD concept yields three pay-offs. First, it couples *specification and explanation*: the commutation amounts to a correctness guarantee of the explanation route. Second, it supports *modularity*, since Dung-style semantics can be reused across source formalisms. Third, it ensures *transparency*: attack/defence structures make the survival and removal of premises during revision explicit. These benefits instantiate Gabbay's principles: mechanisms rather than monoliths, disciplined combinations, and argumentation as the operating system for heterogeneous reasoning.

We also introduced the idea of composing RADs. At the level of a single agent, combining argumentation-as-inference with revision-as-argumentation clarifies choice points (whether to assert or to listen; what to assert; and whether to revise, ignore, question, or challenge) and allows qualitative decision models to govern these communicative acts. At the dialogue level, composing the *assert* RAD (generation of claims from a base) with the *listen* RAD (revision of the receiver's base) yields explainable and auditable multi-agent interaction over time. We furthermore showed that the RAD concept applies beyond symbolic reasoning, for example by modelling subsymbolic surrogate-based explanation (e.g., LIME) as a surrogate-model RAD. We also envisioned that RADs can be extended to AI reasoning in the context of agentic AI, where symbolic and subsymbolic reasoning can be combined, and a principle-based methodology for specifying and comparing Multi-RAD architectures at their input–output behaviors.

Concerning the notion of revision-as-argumentation, two strands of future work follow. Firstly, Theorem 3.12 can be generalized by considering other kinds of revision operators, including iterated revision, prioritized revision, and external revision (see, e.g., [41] for a discussion of these variants). Furthermore, the problem of revising rules will be interesting to extend to other forms of structured argumentation, such as assumption-based argumentation [37] and ASPIC+ [60]. Moreover, extended forms of argumentation can be explored like ranking semantics [4], bipolar argumentation [27], and multiagent argumentation [7, 19]. Secondly, Theorem 3.12 assumes an initially consistent base. In practice, when an agent with an inconsistent knowledge base receives new information and revises its beliefs, some conflicts may be resolved, but it is unlikely that all inconsistencies will disappear. Consequently, we should consider other revision operators that may have inconsistent knowledge bases as a result [53]. Moreover, the revision of inconsistent knowledge bases can be driven by research on inconsistency measures [6]. The success measure of traditional belief revision, which says that the revised knowledge base must be consistent, can be replaced by a condition that the inconsistency measure of the knowledge bases does not increase. Other success conditions can be given, stating conditions under which the inconsistency measure must decrease.

Acknowledgments

L. van der Torre and L. Yu acknowledge the financial support of the Luxembourg National Research Fund (FNR). L. van der Torre is supported through the projects *The Epistemology of AI Systems (EAI)* (C22/SC/17111440), *DJ4ME – A DJ for Machine Ethics: the Dialogue Jiminy* (O24/18989918/DJ4ME), and *Logical Methods for Deontic Explanations (LoDEx)* (INTER/DFG/23/17415164/LoDEx). L. van der Torre and L. Yu are supported through the project *Symbolic and Explainable Regulatory AI for Finance Innovation (SERAFIN)* (C24/IS/19003061/ SERAFIN). L. Yu is additionally supported by the University of Luxembourg's *Marie Speyer Excellence Grant* (2024) for the project *Formal Analysis of Discretionary Reasoning (MSE-DISCREASON)*.

References

[1] Deepak Bhaskar Acharya, Karthigeyan Kuppan, and B Divya. Agentic AI: Autonomous intelligence for complex goals–a comprehensive survey. *IEEE Access*, 2025.

[2] Carlos E Alchourrón, Peter Gärdenfors, and David Makinson. On the logic of theory change: Partial meet contraction and revision functions. *The journal of symbolic logic*, 50(2):510–530, 1985.

[3] Carlos E Alchourrón and David Makinson. On the logic of theory change: Contraction functions and their associated revision functions. *Theoria*, 48(1):14–37, 1982.

[4] Leila Amgoud and Jonathan Ben-Naim. Ranking-based semantics for argumentation frameworks. In *International Conference on Scalable Uncertainty Management*, pages 134–147. Springer, 2013.

[5] Charles Antaki and Ivan Leudar. Explaining in conversation: Towards an argument model. *European Journal of Social Psychology*, 22(2):181–194, 1992.

[6] Ofer Arieli, Kees van Berkel, Badran Raddaoui, and Christian Straßer. Deontic reasoning based on inconsistency measures. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*, volume 21, pages 71–81, 2024.

[7] Ryuta Arisaka, Ken Satoh, and Leendert van der Torre. Anything you say may be used against you in a court of law: Abstract agent argumentation (Triple-A). In *International Workshop on AI Approaches to the Complexity of Legal Systems*, pages 427–442. Springer, 2015.

[8] Kenneth J. Arrow. *Social Choice and Individual Values*, volume 12 of *Cowles Foundation Monographs*. Yale University Press, New Haven, CT, 1951.

[9] Pietro Baroni, Eduardo Fermé, Massimiliano Giacomin, and Guillermo Ricardo Simari. Belief revision and computational argumentation: A critical comparison. *J. Log. Lang. Inf.*, 31(4):555–589, 2022.

[10] Pietro Baroni, Dov Gabbay, Massimiliano Giacomin, and Leendert van der Torre, editors. *Handbook of Formal Argumentation*, volume 1. College Publications, 2018.

[11] Pietro Baroni and Massimiliano Giacomin. On principle-based evaluation of extension-based argumentation semantics. *Artificial Intelligence*, 171(10-15):675–700, 2007.

[12] Pietro Baroni, Antonio Rago, and Francesca Toni. How many properties do we need for gradual argumentation? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[13] Ringo Baumann. What does it take to enforce an argument? minimal change in abstract argumentation. In Luc De Raedt, Christian Bessiere, Didier Dubois, Patrick Doherty, Paolo Frasconi, Fredrik Heintz, and Peter J. F. Lucas, editors, *ECAI 2012 - 20th European Conference on Artificial Intelligence. Including Prestigious Applications of Artificial Intelligence (PAIS-2012) System Demonstrations Track, Montpellier, France, August 27-31 , 2012*, volume 242 of *Frontiers in Artificial Intelligence and Applications*, pages 127–132. IOS Press, 2012.

[14] Ringo Baumann and Gerhard Brewka. Expanding argumentation frameworks: Enforcing and monotonicity results. In Pietro Baroni, Federico Cerutti, Massimiliano Giacomin, and Guillermo Ricardo Simari, editors, *Computational Models of Argument: Proceedings of COMMA 2010, Desenzano del Garda, Italy, September 8-10, 2010*, volume 216 of *Frontiers in Artificial Intelligence and Applications*, pages 75–86. IOS Press, 2010.

[15] Ringo Baumann and Gerhard Brewka. AGM meets abstract argumentation: Expansion and revision for dung frameworks. In Qiang Yang and Michael J. Wooldridge, editors, *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 2734–2740. AAAI Press, 2015.

[16] Matti Berthold, Anna Rapberger, and Markus Ulbricht. Forgetting aspects in assumption-based argumentation. In Pierre Marquis, Tran Cao Son, and Gabriele Kern-Isberner, editors, *Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning, KR 2023, Rhodes, Greece, September 2-8, 2023*, pages 86–96, 2023.

[17] Philippe Besnard and Anthony Hunter. A review of argumentation based on deductive arguments. *Handbook of Formal Argumentation*, 1(437-484):111, 2018.

[18] Vivien Beuselinck, Jérôme Delobelle, and Srdjan Vesic. A principle-based account of self-attacking arguments in gradual semantics. *Journal of Logic and Computation*, 33(2):230–256, 2023.

[19] Elizabeth Black, Nicolas Maudet, and Simon Parsons. Argumentation-based dialogue. *Handbook of Formal Argumentation, Volume 2*, 2021.

[20] Richard Booth, Souhila Kaci, Tjitze Rienstra, and Leendert van Der Torre. A logical theory about dynamics in abstract argumentation. In *Scalable Uncertainty Management: 7th International Conference, SUM 2013, Washington, DC, USA, September 16-18, 2013. Proceedings 7*, pages 148–161. Springer, 2013.

[21] Richard Booth, Souhila Kaci, Tjitze Rienstra, and Leendert W. N. van der Torre. A

logical theory about dynamics in abstract argumentation. In Weiru Liu, V. S. Subrahmanian, and Jef Wijsen, editors, *Scalable Uncertainty Management - 7th International Conference, SUM 2013, Washington, DC, USA, September 16-18, 2013. Proceedings*, volume 8078 of *Lecture Notes in Computer Science*, pages 148–161. Springer, 2013.

[22] Martin Caminada. Argumentation semantics as formal discussion. *Handbook of Formal Argumentation*, 1, 2017.

[23] Martin Caminada. Rationality postulates: Applying argumentation theory for non-monotonic reasoning. *Handbook of Formal Argumentation, Volume 1*, pages 771–796, 2018.

[24] Martin Caminada and Leila Amgoud. An axiomatic account of formal argumentation. In *AAAI*, volume 6, pages 608–613, 2005.

[25] Martin Caminada and Jonathan Ben-Naim. *Postulates for paraconsistent reasoning and fault tolerant logic programming.* PhD thesis, Department of Information and Computing Sciences, Utrecht University, 2007.

[26] Claudette Cayrol, Florence Dupin de Saint-Cyr, and Marie-Christine Lagasquie-Schiex. Change in abstract argumentation frameworks: Adding an argument. *CoRR*, abs/1401.3838, 2014.

[27] Claudette Cayrol and Marie-Christine Lagasquie-Schiex. Bipolar abstract argumentation systems. In *Argumentation in Artificial Intelligence*, pages 65–84. Springer, 2009.

[28] Jinsheng Chen, Beishui Liao, and Leendert van der Torre. Bisimulation between base argumentation and premise-conclusion argumentation. *Artificial Intelligence*, 336:104203, 2024.

[29] Sylvie Coste-Marquis, Sébastien Konieczny, Jean-Guy Mailly, and Pierre Marquis. On the revision of argumentation systems: Minimal change of arguments statuses. In Chitta Baral, Giuseppe De Giacomo, and Thomas Eiter, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the Fourteenth International Conference, KR 2014, Vienna, Austria, July 20-24, 2014*. AAAI Press, 2014.

[30] Kristijonas Cyras, Antonio Rago, Emanuele Albini, Pietro Baroni, and Francesca Toni. Argumentative XAI: A survey. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 4392–4399. ijcai.org, 2021.

[31] Adnan Darwiche and Judea Pearl. On the logic of iterated belief revision. *Artificial intelligence*, 89(1-2):1–29, 1997.

[32] Mehdi Dastani, Joris Hulstijn, and Leendert Van der Torre. How to decide what to do? *European Journal of Operational Research*, 160(3):762–784, 2005.

[33] Sylvie Doutre, Andreas Herzig, and Laurent Perrussel. Abstract argumentation in dynamic logic: Representation, reasoning and change. In Beishui Liao, Thomas Ågotnes, and Yì N. Wáng, editors, *Dynamics, Uncertainty and Reasoning, The Second Chinese Conference on Logic and Argumentation, CLAR 2018, Hangzhou, China, 16-17 June 2018*, pages 153–185. Springer, 2018.

[34] Sylvie Doutre and Jean-Guy Mailly. Constraints and changes: A survey of abstract

argumentation dynamics. *Argument Comput.*, 9(3):223–248, 2018.

[35] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial intelligence*, 77(2):321–357, 1995.

[36] Phan Minh Dung. An axiomatic analysis of structured argumentation with priorities. *Artif. Intell.*, 231:107–150, 2016.

[37] Phan Minh Dung, Robert A Kowalski, and Francesca Toni. Assumption-based argumentation. In *Argumentation in artificial intelligence*, pages 199–218. Springer, 2009.

[38] Phan Minh Dung and Phan Minh Thang. Fundamental properties of attack relations in structured argumentation with priorities. *Artificial Intelligence*, 255:1–42, 2018.

[39] Artur S d'Avila Garcez, Luis C Lamb, and Dov M Gabbay. *Neural-symbolic cognitive reasoning*. Springer, 2009.

[40] Marcelo Alejandro Falappa, Gabriele Kern-Isberner, and Guillermo Ricardo Simari. Belief revision and argumentation theory. In Guillermo Ricardo Simari and Iyad Rahwan, editors, *Argumentation in Artificial Intelligence*, pages 341–360. Springer, 2009.

[41] Eduardo Fermé and Sven Ove Hansson. *Belief change: introduction and overview*. Springer, 2018.

[42] FIPA. Communicative act library specification. *http://www. fipa. org/specs/fipa00037*, 2002.

[43] Andrew Fuchs, Andrea Passarella, and Marco Conti. A cognitive framework for delegation between error-prone ai and human agents. In *2022 IEEE International Conference on Smart Computing (SMARTCOMP)*, pages 317–322. IEEE, 2022.

[44] Dov M. Gabbay. Theoretical foundations for non-monotonic reasoning in expert systems. In Krzysztof R. Apt, editor, *Logics and Models of Concurrent Systems*, pages 439–457. Springer-Verlag, New York, 1985.

[45] Dov M. Gabbay. Fibred semantics and the weaving of logics. part 1: Modal and intuitionistic logics. *The Journal of Symbolic Logic*, 61(4):1057–1120, 1996.

[46] Dov M Gabbay. *Labelled deductive systems*. Oxford university press, 1996.

[47] Dov M Gabbay. An overview of fibred semantics and the combination of logics. In *Frontiers of Combining Systems: First International Workshop, Munich, March 1996*, pages 1–55. Springer, 1996.

[48] Dov M Gabbay. *Fibring logics*, volume 38. Clarendon Press, 1998.

[49] Dov M Gabbay. Fibring argumentation frames. *Studia Logica*, 93(2):231, 2009.

[50] Dov M Gabbay and Sérgio Marcelino. Modal logics of reactive frames. *Studia Logica*, 93(2):405, 2009.

[51] Dov M Gabbay and Lydia Rivlin. Heal2100: human effective argumentation and logic for the 21st century. the next step in the evolution of logic. *IFCoLog Journal of Logics and Their Applications*, 2017.

[52] Sven Ove Hansson. *A textbook of belief dynamics - theory change and database updating*, volume 11 of *Applied logic series*. Kluwer, 1999.

[53] Sven Ove Hansson and Renata Wassermann. Local change. *Studia Logica*, 70:49–76, 2002.

[54] Adrian Haret, Johannes Peter Wallner, and Stefan Woltran. Two sides of the same coin: Belief revision and enforcing arguments. In Jérôme Lang, editor, *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 1854–1860. ijcai.org, 2018.

[55] Philips George John, Deepak Vijaykeerthy, and Diptikalyan Saha. Verifying individual fairness in machine learning models. In *Conference on Uncertainty in Artificial Intelligence*, pages 749–758. PMLR, 2020.

[56] Hirofumi Katsuno and Alberto O Mendelzon. Propositional knowledge base revision and minimal change. *Artificial Intelligence*, 52(3):263–294, 1991.

[57] Sarit Kraus, Daniel Lehmann, and Menachem Magidor. Nonmonotonic reasoning, preferential models and cumulative logics. *Artificial intelligence*, 44(1-2):167–207, 1990.

[58] Fabio Paglieri and Cristiano Castelfranchi. Revising beliefs through arguments: Bridging the gap between argumentation and belief revision in MAS. In Iyad Rahwan, Pavlos Moraitis, and Chris Reed, editors, *Argumentation in Multi-Agent Systems, First International Workshop, ArgMAS 2004, New York, NY, USA, July 19, 2004, Revised Selected and Invited Papers*, volume 3366 of *Lecture Notes in Computer Science*, pages 78–94. Springer, 2004.

[59] Pere Pardo, Liuwen Yu, Chen Chen, and Leendert van der Torre. Weakest link, prioritized default logic and principles in argumentation. *Journal of Logic and Computation*, 35(4):exaf007, 2025.

[60] Henry Prakken. An abstract framework for argumentation with structured arguments. *Argument & Computation*, 1(2):93–124, 2010.

[61] Henry Prakken. Relating abstract and structured accounts of argumentation dynamics: the case of expansions. In Pierre Marquis, Tran Cao Son, and Gabriele Kern-Isberner, editors, *Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning, KR 2023, Rhodes, Greece, September 2-8, 2023*, pages 562–571, 2023.

[62] Anna Rapberger and Markus Ulbricht. On dynamics in structured argumentation formalisms. *J. Artif. Intell. Res.*, 77:563–643, 2023.

[63] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why should I trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

[64] Tjitze Rienstra, Chiaki Sakama, Leendert van der Torre, and Beishui Liao. A principle-based robustness analysis of admissibility-based argumentation semantics. *Argument Comput.*, 11(3):305–339, 2020.

[65] Leendert van der Torre and Srdjan Vesic. The principle-based approach to abstract argumentation semantics. In Pietro Baroni, Dov Gabbay, Massimiliano Giacomin, and Leendert van der Torre, editors, *Handbook of Formal Argumentation, Volume 1*, pages 797–838. College Publications, 2018.

[66] Alexandros Vassiliades, Nick Bassiliades, and Theodore Patkos. Argumentation and explainable artificial intelligence: a survey. *Knowl. Eng. Rev.*, 36:e5, 2021.

[67] Douglas Walton and Erik CW Krabbe. *Commitment in dialogue: Basic concepts of interpersonal reasoning*. State University of New York Press, 1995.

[68] Liuwen Yu, Caren Al Anaissy, Srdjan Vesic, Xu Li, and Leendert van der Torre. A principle-based analysis of bipolar argumentation semantics. In *European Conference on Logics in Artificial Intelligence*, pages 209–224. Springer, 2023.

[69] Liuwen Yu, Dongheng Chen, Lisha Qiao, Yiqi Shen, and Leendert van der Torre. A Principle-based Analysis of Abstract Agent Argumentation Semantics. In *Proceedings of the 18th International Conference on Principles of Knowledge Representation and Reasoning*, pages 629–639, 11 2021.

[70] Liuwen Yu and Leendert van der Torre. The A-BDI metamodel for human-level AI: Argumentation as balancing, dialogue and inference. In *International Conference on Logic and Argumentation (CLAR 2025)*, Cham, 2025. Springer Nature Switzerland. To appear.

[71] Liuwen Yu, Leendert Van der Torre, and Réka Markovich. Thirteen challenges in formal and computational argumentation. *Handbook of Formal Argumentation*, 3:931–1012, 2024.

Received September 2025

Staged Logic: A Unified Framework Against Paradox

Daniel Rowe

Abstract

The ideas and much of the content here is adapted from a Chapter of my PhD thesis (forthcoming). There (and in an appendix) I present some of the results in more technical detail, including fully worked through soundness and completeness theorems for propositional staged logic. This Chapter is more condensed, but hopefully suffices to express the core idea, as well as to see how it delivers a paradox-resistant logic that remains nevertheless powerful enough to deliver the types of mathematical results we would want from logic.

Dov Gabbay's contributions to logic consistently exemplify the search for systematicity and unification. Among his many unificatory projects, two stand out as especially resonant with the staged framework: Labelled Deductive Systems (1996), where the inferential behaviour of formulas is made sensitive to labels (much as staged logic makes it sensitive to stage indices), and Fibring Logics (1999), which provides a systematic method for combining logics while preserving discipline at their points of interaction. These works provide the closest formal analogues to the staged approach, and show how Gabbay's unificatory vision continues in new guises.

Professor Gabbay has a more direct hand in this too. In 2023 whilst working on a joint paper in Talmudic logic, he encouraged me to explore whether any of the Talmudic logic approaches we were studying might help to dissolve the Liar paradox. That served as the inspiration for this logic. He also offered important feedback and insight at an earlier stage of the development of this logic. Other important feedback came from Alex Paseau, James Studd and Carl Posy as well as audiences at three conferences in which early versions were presented: the Jean Paul van Bendegem Masterclass in the Philosophy of Mathematical Practice, University of Brussels, June 2023; the Israel Philosophical Association annual conference, Ben Gurion University, July 2023; and the 'Perspectives About Truth' online conference, September 2023.

1 Introduction

Logic and the philosophy of mathematics have been shaped by paradoxes: Russell's (1902), Burali-Forti (1897), the Liar sentence,[1] Curry (see e.g. [24]), G´odel's diagonal argument, and many others each challenge our understanding of truth, definability, and totality. While each has attracted technical solutions, these have historically been piecemeal. The iterative conception of set, codified in ZFC, blocks Russell and Burali-Forti but not the liar. Tarski's hierarchy (Tarski,1944/1996) blocks the liar but not Russell. Other solutions exist for Curry's paradox, Berry's, Grelling's and so on. The landscape is fragmented.

[20], p.36, 1908) was the first to observe that a common denominator in many, if not all, paradoxes is indefinite extensibility.[2] Paradoxes occur when the totality of the domain can be extended. Sets, ordinals, truths, and sentences all exhibit this property. Yet existing approaches handle them separately.[3] The common structural mistake in each paradox is the possibility of same-level self-reference: a set applied to itself, a sentence denying its own truth, an ordinal including itself, a Gödel number encoding its own sentence. Staged logic offers a single principle: every logical operation 'negation, conjunction, quantification, predication, abstraction, truth' advances the stage index. Formulas are stratified as $\varphi^{[0]}$, $\varphi^{[1]}$, $\varphi^{[2]}$, … and no formula can contain itself at the same stage. All paradoxes dissolve uniformly.

The outline of the rest of this paper is as follows: Section 2 introduces Staged Propositional Logic including the syntax (2.1), semantics (2.2) a brief outline of its modal versions (2.3) and a metatheoretic section (2.4) discussing soundness, completeness and conservativity over standard propositional logic. Section 3 looks at Staged Predicate Logic, its syntax (3.1) semantics (3.2) and a note on soundness and completeness (3.3). Section 4 looks at Staged Proof Theory regarding the introduction and elimination rules for connectives (4.1) and quantifiers (4.2). We then explore how staged logic differs from Type theory (Section 5) and Predicativism (Section 6). Section 7 demonstrates how Staged logic helps to solve numerous paradoxes from Liar (7.1) and Strengthened Liar (7.2), Yablo's Paradox (7.3) Curry Semantic (7.4) and Syntactic (7.5) paradoxes, Russell (7.6) the 'Set of all sets' (7.7) Burali-Forti (7.8) Frege's Basic Law V (which is consistent in Staged logic) (7.9) Gödel sentences (7.10) Berry's (7.11) and Grelling's paradoxes (7.12). Section 8 sketches the way in

[1] The literature on Liar Paradoxes is vast. For an up-to-date readable overviews see (Beall, Glanzberg & Ripley, 2023), and Visser, A. in Gabbay & Guenther (1989).

[2] Other prominent proponents of this diagnosis include Dummett (1963, p.195) and Priest (1994) though the approaches all three propose differ from the staged logic I propose herein.

[3] This point is raised in (Builes & Wilson 2022).

which Staged logic, despite its paradox-resistance, nevertheless is powerful enough to replicate Primitive Recursive Arithmetic (8.1) and the Peano Axioms (8.2) as well as the sub-systems of Second Order Arithmetic relevant to the Reverse Mathematics program. I also outline strategies for proving Second order completeness of Staged logic (8.3), and for resurrecting Frege's logicist program from a Staged version of his logic and Basic Law V (8.4). These latter two suggest the possibility of some additional benefits of this logic.

2 Staged Propositional Logic

Take a sentence (S) in classical first-order predicate logic:

$$(S) : (\forall x\,(Px \wedge Qx) \wedge \forall y\,(Py \vee Qy)) \tag{1}$$

Given the way that we decompose such complex formulas in order to assess truth valuation, it seems natural to think of such complex formulae as being formed in stages. It thus seems fairly natural to keep track of each stage, and even label each stage appropriately.

In our new logic, in addition to ensuring formulae are well-formed, we also must ensure that they are stage-wise well formed (swff) this means that every formula must be labelled a stage greater than all stages of its component parts. Self-referential paradoxes typically employ a sentence inside itself. In staged logic these will be two distinct sentences, as they will be constructed at two different stages. Thus they could take different truth values without paradox. Since the left hand side labels the right hand side it must be a sentence of the same stage. Alternatively, they both take the same stage, but then the overall sentence will be stage-wise badly formed (we will demonstrate this in 7.1).

3 Syntax

The syntax of staged propositional logic is designed to enforce a strict discipline: every logical operation advances the stage index. This principle is the key move that avoids illicit self-reference. We begin with the familiar ingredients of propositional logic, but we mark each formula with a superscript indicating the stage at which it resides.

- **Propositional variables.** We assume a finite or countable stock of propositional variables $\mathbf{p}, \mathbf{q}, \mathbf{r}, \ldots$ Each variable is taken to be a formula at stage [0].

Thus $p^{[0]}$, $q^{[0]}$, $r^{[0]}, \ldots$are the simplest formulas, belonging to the initial level of the hierarchy.

- **Connectives.** If $\varphi^{[n]}$ and $\psi^{[n]}$ are formulas at stage $[n]$, then by applying a connective we obtain a new formula at the *next* stage $[n+1]$.

3.1 In particular:

- $(\neg\varphi^{[n]})^{[n+1]}$ which can be abbreviated: $(\neg\varphi)^{[n+1]}$
- $(\varphi^{[n]}\wedge\psi^{[n]})^{[n+1]}$which can be abbreviated: $(\varphi\wedge\psi)^{[n+1]}$
- $(\varphi^{[n]}\vee\psi^{[n]})^{[n+1]}$which can be abbreviated: $(\varphi\vee\psi)^{[n+1]}$
- $(\varphi^{[n]}\rightarrow\psi^{[m]})^{[Max(m,n)+1]}$which can be abbreviated: $(\varphi\rightarrow\psi)^{[Max(m,n)+1]}$
- $(\varphi^{[n]}\leftrightarrow\psi^{[m]})^{[Max(m,n)+1]}$which can be abbreviated: $(\varphi\leftrightarrow\psi)^{[Max(m,n)+1]}$

  The rule is uniform: any application of a connective increments the stage index by one. As a result, a formula can never contain its own negation or conditional at the same level; it can only contain the negation or conditional of a formula from the previous level.

- **Predication.** A key distinctive feature of staged logic is that **predication itself is stage-advancing**. If $\varphi^{[n]}$ is a formula, then the claim that that formula is ψ, is one stage advanced from $\varphi^{[n]}$ itself. For example, to say that $\varphi^{[n]}$ is true, in staged logic becomes: $True(\ulcorner\varphi^{[n]}\urcorner)^{[n+1]})$. This ensures that a predicate cannot be applied to an object at the same stage as that object: the act of predication always moves us one step up. Like Tarski, there is a shift that takes place. Unlike Tarski this shift occurs *within* the same language.

- **General form.** Formulas are thus stratified into levels, beginning with propositional variables at [0] and extending indefinitely. Each level depends only on those below it. No formula can ever fold back into itself at the same stage, since every syntactic construction forces a shift in level that is recorded syntactically. Importantly, predicates are not intrinsically limited to fixed stages as they are in type-theory (see section 5). This flexibility allows for the same predicate to appear at distinct stages.

- **Natural Numbers.** Staged logic does not predetermine whether all natural numbers are considered to be members of the initial domain, or whether it is just 0 that is, and others are generated in successive stages by successive application of the Successor function. The latter is very much in line with the type of thinking that motivates staged logic, and also works with the Peano Axioms (see 8.2). It also makes it easier to block diagonalization with Gödel numbers (see 7.10).

This stratified syntax departs minimally from classical logic: we use the same variables and connectives, but add explicit stage indices and require that every new construction shifts up a level. This simple adjustment turns out to suffice to block the formation of paradoxical sentences such as Liar or Curry, which rely precisely on same-stage self-application. We will examine these and other paradoxes in section 7.

4 Semantics

The semantics of staged propositional logic mirrors its syntax: just as the formation rules ensure that each stage of complexity is indexed by stage, so too the truth-conditions are defined in the precise same stages. Truth at stage $[n + 1]$ depends entirely on truth assignments already fixed at stage $[n]$. In this way, staged semantics prevents a formula from ever determining its own value at its own level.

- **Staged valuations.** A model for propositional staged logic is given by a sequence of classical valuations:

$$V = \left\langle V_{[0]}, V_{[1]}, V_{[2]}, \ldots \right\rangle \tag{2}$$

  Here $V_{[0]}$ assigns truth values to propositional variables at stage $[0]$. Each subsequent valuation $V_{[n+1]}$ extends the assignment to formulas at stage$[n+1]$, drawing only on the values already given at (or before) stage $[n]$.

- **Base stage.** At stage [0], the valuation is exactly the same as its non-staged counterpart. Thus each variable $p^{[0]}$ is assigned either 0 (false) or 1 (true). This provides the foundation for the entire hierarchy.

- **Connectives.** The extension from any stage $[n]$ to the next stage $[n + 1]$ is classical in form, but indexed by level:

- $V_{[n+1]}(\neg\varphi^{[n]})^{[n+1]} = 1$ iff $V_{[n]}\varphi^{[n]} = 0$

- $V_{[n+1]}(\varphi^{[n]}\wedge\psi^{[n]})^{[n+1]} = 1$ iff $V_{[n]}\varphi^{[n]} = V_{[n]}\psi^{[n]} = 1$

- And similarly for the valuation of other connectives.

Thus truth-values at stage $[n+1]$ are computed in the usual way, but using the assignments available from stage $[n]$. No formula at $[n+1]$ can ever refer to the truth-value of a formula at $[n+1]$ itself.

- **Predication.** For the truth predicate, we stipulate:
  $V_{[n+1]}(True^{\ulcorner}\varphi^{[n]\urcorner[n+1]}) = 1$ iff $V_{[n]}\varphi^{[n]} = 1$.

In other words, the assertion that a formula from stage $[n]$ is true is itself a formula at stage $[n+1]$, whose truth is determined entirely by the evaluation at the earlier stage. This eliminates the possibility of a Liar sentence, which would require $(False^{\ulcorner}\varphi^{[n]\urcorner[n+1]})$ to be evaluated at the same level (see 7.1).
The semantics therefore preserves the classical truth-tables, but stratifies them. By making truth at each stage depend strictly on the stage below, staged logic closes off the circularities that semantic paradoxes exploit.

2.3 Modal enrichment

Any system in which the domain is never complete faces the problem of generalisation. One natural device is to enrich the language with modal operators $\Box$ ("always at later stages") and $\Diamond$ ("at some later stage").
Unlike the object-level operators ($\neg$, $\wedge$, $\vee$, $\to$), quantifiers, predication, truth-ascription), $\Box$ and $\Diamond$ do not increment the formula stage. Instead, they are governed by the accessibility relation on stages:

- $M, n \vDash \Box\varphi$ iff for each $m \geq nM, m \vDash \varphi$
- $M, n \vDash \Diamond\varphi$ iff for some $m \geq nM, m \vDash \varphi$

On this interpretation, the accessibility relation between stages is the natural order $\geq$, which is reflexive and transitive. That immediately validates the axioms of S4. If one further assumes that the hierarchy of stages is directed – any two stages admit a common extension – then the stronger S4.2 is validated.[4]
Either way, the modal enrichment remains conservative over the core staged system. The operators allow us to express generalisation, induction, and reflection, but cannot collapse a formula back into the stage in which it is defined. They are not needed for the core paradox-blocking results, but they enrich the expressive resources of the system, and will be useful in section 8 when reconstructing induction and second-order principles.

[4] Linnebo (2010, 2013) and Linnebo & Shapiro (2018) argue that S4.2 is needed to capture many important mathematical results.

Note. The quantification "for all $m \geq n$" in these clauses is not a commitment to a completed domain of stages. It is a potentialist schema: at any stage one might construct in the future, the condition will hold.

5 Meta-theory

The propositional system enjoys the expected metatheoretic properties of soundness, completeness, and conservativity. I offer a very brief overview of the strategy for demonstrating each of these (the detailed proofs are available in the appendix of my PhD thesis). These results presuppose the natural-deduction system spelled out in section 4. In particular, they rely on the staged versions of *modus ponens* (4.2) and *ex falso* (4.3) which ensure that derivations respect the stage discipline.

Soundness.
Soundness is proved by induction on derivations. Each inference rule matches the semantic clause at the next stage: if$\Gamma \vdash \varphi^{[n]}$, then φ is satisfied at stage $[n]$ under every staged valuation that makes Γ true. Because every connective ($\neg$, $\wedge$, $\vee$, $\rightarrow$) predication, truth-ascription) raises the stage index, no inference can produce a false conclusion from true premises.

Completeness.
Completeness requires a canonical construction adapted to the staged setting. The key lemma is a **staged Lindenbaum extension**: any consistent set of formulas at stage $[n]$ can be extended to a maximally consistent set that (i) is closed under stage-shifted consequences, and (ii) for every $\varphi^{[n]}$, contains either $\varphi^{[n]}$ or its negation $\neg\varphi^{[n+1]}$. This "offset maximality" preserves the classical Lindenbaum idea while respecting the staged syntax. From such sets we build a staged analogue of the canonical (un-staged) model: a sequence of valuations $\left\langle V_{[0]}, V_{[1]}, V_{[2]}, \ldots \right\rangle$ where $V_{[n]}$ assigns propositional variables at stage $[n]$, and compound clauses at $[n+1]$ are evaluated strictly from $V_{[n]}$. A standard induction then yields the **Truth Lemma**: $M_\Delta \vDash \varphi^{[n]}$ iff $\varphi^{[n]} \in \Delta$

Completeness follows: any consistent staged set has a realisation.

Conservativity.
If reasoning is confined to a single stage, staged propositional logic collapses back to classical propositional logic. This means that staged logic is conservative over its classical counterpart: for any formula confined to a single stage, provability in staged logic coincides exactly with provability in classical logic; all theorems of the classical system are preserved, and no additional single-stage theorems appear.

6 Modal enrichment.

The same metatheoretic results extend smoothly to the modal operators $\Box$ and $\Diamond$. Since they are interpreted in the standard way over the $\geq$ relation on stages, they validate at least S4 (or S4.2 under directedness). Their soundness and completeness follow the familiar modal proofs, and they remain a conservative addition to the staged framework.

Thus staged propositional logic is a conservative, sound, and complete system. The staged structure matters only when paradox-prone constructions are attempted, which are automatically displaced into higher stages and so cannot close on themselves.

7 Staged Predicate Logic

The extension of staged logic from the propositional to the predicate level follows the same core idea: **every operation advances the stage**. In the propositional system we saw that each connective and each act of truth-ascription increments the index. At the predicate level, *quantification* is added to this list. This yields a system that is preserves the anti-paradox discipline: no formula can apply to, or quantify over, itself at the same stage.

3.1 Syntax

- **Terms.** Variables, constants, and function symbols are built up in the usual way. Terms themselves carry no stage index; it is formulas in which they appear that do.
- **Predication.** If **P** is a **k**-ary predicate symbol and $t_{1,} \ldots, t_k$ are terms, then

$$P(t_{1,} \ldots, t_k)^{[n+1]} \tag{3}$$

8 is a well-formed formula whenever the terms are interpreted at stage . Predication thus always raises the stage: a statement about objects available at stage lives at stage .

- **Quantifiers.** If $\varphi^{[n]}$ is a formula in which x may occur free, then both $(\forall x\varphi)^{[n+1]}$ and $(\exists x\varphi)^{[n+1]}$ are well formed formulas. Quantification therefore also increments the stage, in line with the rule that every operator forces a step up.

The syntax is thus exactly that of ordinary first-order logic, with the single addition that every construction is stratified by an explicit stage index. This simple change ensures that no formula can quantify over or predicate of itself at its own level.

8.0.1 Semantics

The semantics of staged predicate logic generalises directly from the propositional case. In Section 2 we used a **sequence of valuations** $V_{[0]}, V_{[1]}, V_{[2]} \ldots$, each extending the truth-assignments of the previous stage. At the predicate level we require a richer structure: **staged domains** and **staged interpretations** for predicate and function symbols.
A staged model is a sequence

$$M = \langle V_{[0]}, V_{[1]}, \ldots ; I_{[0]}, I_{[1]}, \ldots \rangle \tag{4}$$

where:

- $D_{[0]} \subseteq D_{[1]} \subseteq D_{[2]} \subseteq \cdots$ are expanding domains of objects, reflecting indefinite extensibility,
- for each predicate symbol P, $I_{[0]}(P) \subseteq D^k_{[n]}$ gives its extension at stage $[n]$,
- for each function symbol $f, I_{[0]}\,(f) : D^k_{[n]} \to D_{[n]}$ interprets it at that stage.

This is the natural extension of the propositional case: previously valuations of propositional letters were staged, we now stage both the **domain** and the **interpretations** of non-logical symbols.
Satisfaction is defined strictly stage-by-stage:

- **Atomic formulas.** $M, [n+1] \vDash P(t_1, \ldots, t_k)$ iff $\left([t_1]_{[n]} \cdots [t_k]_{[n]}\right) \in I_{[n]}(P)$
- **Universal quantifier.** $M, [n+1] \vDash \forall x \varphi$ iff for all $a \in D_{[n]}$, it is the case that $M, [n] \vDash \varphi(a/x)$
- **Existential quantifier.** $M, [n+1] \vDash \exists x \varphi$ iff there exists an $a \in D_{[n]}$, such that $M, [n] \vDash \varphi(a/x)$

Thus quantifiers at stage $[n+1]$ always range over the domain already fixed at stage $[n]$. As in the propositional case, truth at a given stage depends only on the resources available at the previous one, ensuring that no formula ever determines its own truth at the level it inhabits.

3.3 Meta-theory

The metatheory of staged predicate logic extends directly from the propositional case (§2.4). The proofs of soundness and completeness proceed in parallel, with only two additions:

1. Quantifiers are stage-advancing operators: $\forall x\varphi$ and $\exists x\varphi$ at $[n+1]$ range only over the domain fixed at stage $[n]$.
2. Witnesses for existentials. In the Henkin construction, whenever $(\exists x\varphi(x))^{[n+1]}$ appears, we introduce a fresh constant symbol c drawn from the stage-$[n]$ domain, and add $\varphi(c)^{[n+1]}$ to the set. In this way every existential claim at $[n+1]$ has an explicit 'witness' from stage $[n]$, ensuring that the staged canonical model realises all existentials.

With these additions, the canonical staged model construction and Truth Lemma go through unchanged. The same remarks apply with modal enrichment: $\Box$ and $\Diamond$ remain conservative over the staged framework, validating at least S4 (or S4.2 under directedness), and their soundness and completeness follow the standard modal proofs.

As before, the system is conservative over ordinary first-order logic when confined to a single stage, while the staging discipline provides the extra structure that blocks paradoxes.

8.0.2 Proof Theory of Staged Logic

The proof system mirrors the semantics: every logical operation both raises the formula stage and advances the derivation stage. This applies uniformly across the propositional and predicate levels. In particular, the rules for quantifiers extend the same discipline introduced in section 3: each act of generalisation or instantiation increments the stage, ensuring that nothing can be re-used at the level in which it was first formed.

We annotate each proof line as $\varphi^{[n@d]}$ = "formula φ has **formula stage** n and is available only from **derivation stage** d onward."

Two global constraints govern all inferences:

- **Formula stage:** object-level operator $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$, $\forall$, $\exists$), truth-ascription, predication) raises the formula stage by +1 relative to their inputs.
- **Derivation stage:** every inference outputs a conclusion whose derivation index is Max (inpiut-derivation-stages) + 1. No inference is permitted to lower either index.

8.1 Connectives

Conjunction.
From $\varphi^{[n@d]}$and $\psi^{[m@e]}$, infer $(\varphi^{[n@d]}\wedge\psi^{[m@e]})^{[Max(m,n)+1@Max(d,e)+1]}$
Conjunction Elimination:

$$(\varphi^{[n@d]}\wedge\psi^{[m@e]})^{[Max(m,n)+1@Max(d,e)+1]} \Longrightarrow \varphi^{[n@Max(d,e)+1]}$$

(and similarly for ψ).
Disjunction.
The same strategy applies to disjunction introduction and elimination, with the same stage/derivation lifts.

8.2 Implication. Implication

$\to$ Introduction. From a sub-derivation that assumes $\varphi^{[n@d]}$ and derives $\psi^{[m@e]}(d < e)$, discharge and infer:

$$(\varphi^{[n@d]}\to\psi^{[m@e]})^{[Max(m,n)+1@e+1]} \tag{5}$$

The conditional "remembers" the derivation stage d at which its antecedent must be provided.

8.3 $\to$-Elimination (see 4.2 Modus Ponens).

Negation.
From $(\varphi^{[n@d]}\to\perp^{[m@e]})^{[Max(m,n)+1@e+1]}$infer $(\neg(\varphi^{[n@d]})^{[n+1@e+2]})$
Note that the negation of $\varphi^{[n]}$is one construction stage advanced over the affirmation, and the derivation stage is one staged more advanced than the *reductio* from which it was derived.

8.4 Reductio ad absurdum.

8.5 From infer .

8.6 Biconditional. Biconditional

$$(\varphi^{[n@d]}\leftrightarrow\psi^{[m@e]})^{[Max(m,n)+1@Max(d,e)+1]} \tag{6}$$

Elimination yields *typed* conditionals:

$$(\varphi^{[n@d]} \to \psi^{[m@e]})^{[Max(m,n)+1@Max(d,e)+1]} \tag{7}$$

$$(\psi^{[m@e]} \to \varphi^{[n@d]})^{[Max(m,n)+1@Max(d,e)+1]} \tag{8}$$

Because the eliminants carry this input-stage requirement, you cannot feed-back a result born later in the same derivation.

8.6.1 Modus Ponens (MP)

From $\varphi^{[n@d]}$ and
$(\varphi^{[n@d]} \to \psi^{[m@e]})^{[Max(m,n)+1@e+1]}$,
infer $\psi^{[m@e+2]}$.
Note. The antecedent must occur at the same base stage as in the conditional. In addition, the derivation index must align: a conditional introduced at derivation stage d can only be applied to an antecedent available at d. An antecedent that first appears at a later derivation stage cannot be fed back into the earlier conditional. Where the derivation is a schema (or can be re-applied at later stages), one may indeed obtain further consequents – but always at strictly higher derivation indices. In particular, biconditionals never allow one to cycle back to the original antecedent at the same derivation stage. This blocks the looping step of Curry's paradox, which in the classical setting recycles an antecedent into its own conditional (see section 7.5).

9 ⊥-Elimination (Ex Falso)

From $\bot^{[n@d]}$ infer any $\varphi^{[n+1@d+1]}$.
Here absurdity at stage $[n]$ trivialises only at formula-stage $[n+1]$. This ensures consistency with the staged semantics: contradiction advances the object-stage and never allows stage-lowering.

10 Quantifiers

Quantifiers follow the same stage-shift principle as the propositional connectives.

- Universal quantifier. If one can derive $\varphi(x)^{[n]}$ for an arbitrary x drawn from the current domain, then one may infer $(\forall x\varphi(x))^{[n+1]}$. Instantiation is similarly confined: from $(\forall x\varphi(x))^{[n+1]}$one can obtain $\varphi(t)^{[n+1]}$ for any term t.

- Existential quantifier. From $\varphi(t)^{[n]}$ one may infer $(\exists x\varphi(x))^{[n+1]}$.Conversely, to reason from $(\exists x\varphi(x))^{[n+1]}$ one may temporarily assume some instance $\varphi(t)^{[n]}$ and derive a further conclusion, which is then discharged to yield that conclusion at stage $[n+1]$.

In every case, the quantifier step increments the stage. One cannot both generalise and immediately re-instantiate at the same level; there is always a one-step gap. This is precisely what blocks familiar paradoxes of unrestricted comprehension.

11 Relation to Type Theory

Staged logic may look superficially similar to Russellian type theory (Russell and Whitehead, 1925, introduction) since both forbid vicious selfapplication. But the mechanisms are very different.

Here are the key differences:

5.1 No hierarchy of kinds. In type theory, expressions are assigned *fixed* types; a highertype expression cannot apply to a lower one, and equality across types is awkward. In staged logic, by contrast, there is **no intrinsic restriction on predicates or terms**. Any predicate may, in principle, apply to any object. What is disciplined is not *what* may combine with what, but **how a given formula is formed**: every objectlevel operation ($\neg, \wedge, \vee, \rightarrow, \leftrightarrow$intro, $\forall, \exists$, predication, truthascription, etc) advances the **formula stage** by $+1$.

5.2 Where the hierarchy lives. The hierarchy in staged logic is therefore a hierarchy of **formula construction**, not of kinds or types. The same predicate symbol may occur at multiple stages; there is no need to proliferate type indices or to maintain a separate calculus for equality across types.

5.3 Biconditional and crossstage equivalence. In staged logic, biconditionals can express a mixedstage equivalence $(\varphi^{[n]} \leftrightarrow \psi^{[m]})^{[Max(m,n)+1]}$The latter is wellformed and lives at stage $[Max(m, n) + 1]$. Elimination produces *typed* conditionals whose antecedents must be supplied at the appropriate derivation stage. We do **not** block paradoxes by forbidding crossstage $\leftrightarrow$; we block them earlier, at definition/-formation (or in the case of Curry, by blocking looped back and forth *Modus Ponens* across the same biconditional).

Bottom line. Staged logic achieves the good of type theory (no samelevel selfapplication) without its rigidity. The discipline comes from the staged formation/semantics themselves, not from an externally imposed stratification of expressions.

12 Relation to Predicativism

Staged logic also resembles an important aspects of predicativism: it blocks definitions that would make an entity depend on itself *at the same stage.* But the resemblance is limited.

Predicativism proper. On the classical predicativist view (e.g. Poincaré, (1905, 1906, 1912), [29]), impredicative definitions are disallowed: no entity may be defined in terms of a totality that already contains it.

Staged permissiveness. Staged logic permits many impredicativelooking definitions because staging enforces the onestep separation that removes circularity. Example: let $CM(a, b)$ be the set of common multiples of a and b. In staged logic one may:

1. form $CM(a, b)$ at stage $[n + 1]$ (collecting multiples computed over stage$[n]$ arithmetic), and then

2. define the least common multiple $lcm(a, b)$ at stage $[n+2]$ as the $\leq$least element of $CM(a, b)$.

There is no circularity: the defining totality (CM) is fixed *one stage earlier* than the thing defined (lcm).

General pattern. Staged logic blocks only those definitions that *force samestage selfapplication.* Impredicative but noncircular constructions are admissible because the staged discipline guarantees the required separation.

13 Unified Solution for Paradoxes

We now show how this **one and the same staged mechanism** blocks a wide range of paradoxes. The common ingredient each paradox requires is *samestage* selfapplication (a set taking itself as member; a sentence asserting its own untruth; a diagonal code naming its own formula, etc.). Staged logic makes this structurally impossible: each objectlevel operation advances the stage by +1, so any attempted selfapplication is automatically displaced to a *higher* stage.

We use the following standing conventions, already in force in §2.4:

- Superscripts $[n]$ mark **formula stages**;

- If φ is at stage $[n]$, then $\textbf{True}(\varphi)$ is at $[n + 1]$, $\neg\textbf{True}(\varphi)$ at $[n + 2]$, etc.;

- Biconditionals are allowed cross-stage: $(\varphi^{[m]} \leftrightarrow \varphi^{[n]})^{[Max(m,n)+1]}$ with typed $\leftrightarrow$-elimination;

- In the proof theory, every inference raises the **derivation stage**; Modus Ponens is *typed* and cannot mix the wrong input stages, including the wrong derivation stage.

14 Paradoxes

14.1 Liar

The canonical Liar sentence is: $\mathbf{L} \equiv \neg\mathbf{True}(\ulcorner\mathbf{L}\urcorner)$.
In staged logic, let L be attempted at stage $[n]$. Then $\mathbf{True}(L^{[n]})$ exists only at $[n+1]$; $\neg$ $\mathbf{True}(L^{[n]})$ at $[n+2]$. So the wouldbe definition
$L^{[n]} \equiv (\neg\mathbf{True}(L^{[n]}))^{[n+2]}$
cannot be formed as a single stagewise wellformed formula (swff): the righthand side strictly lives at $[n+2]$, and so the LHS is strictly an $[n+2]$ formula. It is possible to rewrite it as:
$L^{[n+2]} \equiv (\neg\mathbf{True}(L^{[n]}))^{[n+2]}$
But then $L^{[n]}$ as it occurs in the RHS is stage-wise badly formed. There are two ways such a formula might be treated. Firstly, the RHS could be rejected outright as inadmissible. Secondly $L^{[n]}$ could be treated as lacking a truth-value. In such a case the predicate $(\mathbf{True}\left(L^{[n]}\right))^{[n+1]}$ would be false, and so $(\neg\mathbf{True}(L^{[n]}))^{[n+2]}$ would be true. In either case there would be no paradox.[5]

14.2 Strengthened Liar / Revenge

A typical revenge form says: "Every instance of this very schema is false." Formalising that requires quantifying over *all* instances, including the one being asserted. But $\forall$ raises stage by $+1$; so any universal quantification over instances sits a stage *above* those instances. The schema cannot capture itself at its own level. Even with modal enrichments (e.g. S4/S4.2 to talk about all future stages), the universal claim is always evaluated strictly *above* the instances it ranges over, and no loop closes.

14.3 Yablo

Let $S(1), S(2), \ldots$ be a sequence of sentences, where each $S(i)$ says: "all later $S(k)$ with $k > i$ are false."
In staged logic such sentences cannot all live at the same level. For example, suppose $S(i)$ is formed at stage $[n]$. The universal quantifier ("for all $k > i \ldots$") itself shifts us to stage $[n+1]$. Within its scope, the clause "$False(S(k))$" shifts again, since

[5] On this approach, a different Liar sentence $(\mathbf{False}\left(L^{[n]}\right))^{[n+1]}$ comes out true.

the truth-predicate always raises the stage. Thus the overall $S(i)$ sits strictly higher than each of the $S(k)$ it talks about.

The effect is that at any given stage $S(i)$ only finitely many of the $S(k < i)$ can even be formed at a time. Anyone insisting on a completed infinite totality of sentences will never be able to index any of them. Anyone accepting indexing one sentence at a time, will find that there is always a 'last' sentence that itself fails to refer to any 'subsequent' sentences, making it neither true nor false (there may be other evaluation methods for such cases, which will effect this analysis, but none will produce a paradox). If so then the next sentence will be false (since the former one was not). Similarly each 'prior' $S(i)$ is false at every stage, but there is never a liar-style collapse, because the sequence is spread out across higher and higher stages rather than circling back at a single level.

14.4 Curry (Semantic Version)

Classically: $\mathbf{C} \equiv \mathbf{(True(C)} \to \bot\mathbf{)}$ trivialises any theory with Modus Ponens.

Let C be a stage $[n]$ sentence. Then $(\mathbf{True}\left(C^{[n]}\right))$ sits at $[n + 1]$, and $\left(\mathbf{True}\left(C^{[n]}\right) \to \bot\right)$ at $[n + 2]$. The semantic version of Curry's paradox fails in much the same way that the Liar does. (Note: the removal of the predicate 'true' will still yield a stage imbalance and thus dissolve the paradox as with the liar).

14.5 Curry (Syntactic Version)

Here is the paradox in ordinary (unstaged) logic:

(1)	$C \leftrightarrow (C \to \bot)$	(Assumption)
(2)	C	(Assumption)
(3)	$C \to \bot$	(1,2 MP)
(4)	$\bot$	(2,3 MP)
(5)	$C \to \bot$	(2,4 CP)
(6)	C	(5,1 MP)
(7)	$\bot$	(5,6 MP)

The syntactic version does not define what the sentence δ is. The bicondional asserts the same truth conditions for both sides but does not define the two as being the identical entity.

Schematic Curry proceeds by setting $\varphi \leftrightarrow (\varphi \to \psi)$ and then using MP twice. In staged proof theory, the conditional introduced at $[n + 1]$ carries a tag requiring its antecedent at the corresponding derivation stage. But φ itself only becomes available a stage later. The antecedent is never of the *right type* for MP, so the derivation cannot run. To demonstrate this let us attempt to replicate the derivation in staged logic:

1. $(C^{[n@d]} \leftrightarrow \left(C^{[n@d]} \rightarrow \psi^{[m@e]}\right)^{[Max(m,n)+1@Max(d,e)+1]})^{[Max(m,n)+2@Max(d,e)+2]}$
2. $C^{[n@d]}$
3. $\left(C^{[n@d]} \rightarrow \psi^{[m@e]}\right)^{[Max(m,n)+1@Max(d,e)+2]}$
4. $\psi^{[m@Max(d,e)+2]}$
5. $\left(C^{[n@d]} \rightarrow \psi^{[m@e]}\right)^{[Max(m,n)+1@Max(d,e)+3]}$

   But now line 5 is not the same as the RHS of the assumption at line 1, and so there is no MP available to derive the LHS. Thus the paradox is blocked.

   Even if a case may be made that once a derivation has been made, it can in principle be re-made at every subsequent line, that still will not resurrect the paradox. It would mean that we could introduce a new version of the original biconditional written as:

6. $(C^{[n@d]} \leftrightarrow \left(C^{[n@d]} \rightarrow \psi^{[m@e]}\right)^{[Max(m,n)+1@Max(d,e)+1]})^{[Max(m,n)+2@Max(d,e)+3]}$

Now from (6) and (5) we might seek to use MP to conclude a version of $C^{[n]}$, but it would be: $C^{[n@Max(d,e)+4]}$ which would no longer allow us to deduce line 3. So even if it could be argued that there can be infinitely many successive versions of the biconditional, the result will be a spiral of deducing the higher staged right-hand side, and then the higher staged left hand side *ad infinitum*. Never does the process flatten to produce both sides simultaneously. The paradox is dissolved.

14.6 Russell's Paradox

Consider $R = \{x|x \notin x\}$. Interpreting membership as predication, $x \in y$ is $y(x)$, which is stage-raising. In staged comprehension, sets formed at $[n+1]$ may only collect objects available at $[n]$. Thus the extension defining R lives at $[n+1]$and ranges over stage$[n]$ sets. Consequently, $R \notin R$ for the simple reason that $R$is *not among* the stage$[n]$ sets it ranges over. Any test of $R \in R$ would itself occur at $[n+2]$ and cannot force a contradiction at $[n+1]$.
Once again the putative self-membership would require same-stage application and is therefore unavailable.

15 Set of All Sets (No Universal Set)

For the same reason, there is no set of *all* sets. At stage $[n+1]$ one can only collect the sets that are available at $[n]$. The collection thereby produced is not itself among those it collects. Indefinite extensibility is built in: every time a larger domain is formed, it sits strictly one stage higher.

15.1 Burali-Forti (Ordinal of All Ordinals)

The "ordinal of all ordinals at stage n" lives at $[n+1]$ and therefore is *not* among the ordinals at stage n. Again, there is no same-stage totality, and so there is no paradox.

15.2 Abstraction Paradoxes (Fregean Extensions; Basic Law V)

Abstraction operators (extensions, numbers, directions, etc.) are introduced one stage up from their bases. Two predicates may receive extensions at $[n+1]$; Basic Law Vstyle identifications compare those extensions *at that higher stage.* Because comprehension over concepts at $[n]$ never yields an extension *within* $[n]$, the Russell-type collapse is blocked. (In short: a staged BLV avoids the self-application that dooms the unstaged version (see section 8.5.)

15.2.1 Gödel Coding and Diagonalisation

Let $G(\varphi)$ denote the Gödel code of a formula φ. In staged semantics, coding is stageraising: codes for formulas at $[n]$ appear at earliest $[n+1]$ (as numerals or strings). The standard diagonal lemma attempts to form a sentence D that says of its *own* code that it has property P. But the self-substitution step produces a formula at $[n+2]$ referring to a code at $[n+1]$ for a formula at $[n]$. The would be equality of "the formula just written" with "the formula whose code is now named" cannot be maintained at one stage. What diagonalisation yields is an *openended hierarchy* of related formulas, not a fixed point.

15.2.2 Berry

Berry's paradox is (for example) "the least number not nameable in < 10 words". At any fixed stage of names that the Berry sentence itself refers to only has the names available up to that stage. Forming the set of numbers nameable with fewer than 10 words is stageraising; selecting the least such unnameable number is another stage further. The definition therefore does not close on itself.

15.2.3 Grelling

Grelling's paradox asks whether the property of being 'heterological' (i.e. the property fails to self-apply) is itself heterological. At the same level it could only be true if false and vice versa. But in staged logic predication is stageadvancing. The critical same-stage test (Heterological(heterological)) is simply not swff, unless each 'heterological' is fixed to a specific stage, in which case '$heterological^{[n]}$' refers to all the adjectives that fail to describe themselves. But it cannot refer to itself. It could be evaluated from the perspective of being (or failing to be) '$heterological^{[n+1]}$' but that would not be paradoxical.

16 Beyond Paradox: Mathematical Strength

Up to this point we have emphasised the defensive virtues of staged logic: it blocks the familiar paradoxes with a single principle. Yet the strength of any logical system is not measured only by the paradoxes it avoids, but also by the mathematics it supports. It might be thought that staged logic is not a weak fragment of its non-staged counterpart. In fact it turns out to be a framework robust enough to reconstruct large portions of arithmetic and analysis. This section sketches, with indicative arguments, how ordinary mathematics can be carried out within the staged setting.

17 Primitive Recursive Arithmetic (staged PRA)

The natural numbers can be generated stage by stage. Begin with $0^{[0]}$as the initial constant. The successor function is stage-advancing: if $n^{[n]}$ is available, then its successor is $S(n)^{[n+1]} = (n+1)^{[n+1]}$. So every numeral coincides with its stage index: $0^{[0]}, 1^{[1]}, 2^{[2]}, \ldots$

Primitive recursive functions are then defined inductively, always shifting one stage higher:

- **Addition.**
  Base: $(((n^{[n]} + 0^{[0]})^{[n+1]} = n^{[n]}))^{[n+2]}$
  Step: $((x^{[n]} + S(m^{[m]})^{[m+1]})^{[Max(m+1,n)+1]}) =$
  $(S\left(n^{[n]} + m^{[m]}\right)^{[Max(m+1,n)+1]})^{[Max(m+1,n)+2]})^{[Max(m+1,n)+3]}$

- **Multiplication.**
  Base: $((n^{[n]} \cdot 0^{[0]})^{[n+1]} = 0^{[0]})^{[n+2]}$
  Step: $(((n^{[n]} \cdot S(m^{[m]})^{[m+1]})^{[Max(m+1,n)+1]}) =$
  $((n^{[n]} \cdot m^{[m]})^{[((n^{[n]} \cdot 0^{[0]})^{[n+1]}]} + n^{[n]})^{[Max(m+1,n)+2]})^{[Max(m+1,n)+3]}$

- **Exponentiation.**
  Base: $(((n^{[n]})^{0^{[0]}})^{[n+1]} = 1^{[1]})^{[n+2]}$
  Step: $(((n^{[n]})^{(S(m^{[m]}))^{[m+1]}})^{[Max(n,m+1)+1]} =$
  $((((n^{[n]})^{m^{[m]}})^{Max(n.m)+1} \cdot n^{[n]})^{[Max(n,m)+2]})^{[Max(m,n)+3]}$

17.1 Pattern:

- Inner terms get their own stages.

- Every construction (successor, +, $\cdot$, exponentiation) raises the stage once.

- The outer equality raises the stage once more.

So each recursive clause lives strictly higher than the numerals it governs, but the content is exactly the familiar arithmetic law.
Each clause unfolds at the next stage, so no recursive definition ever closes on itself. Yet every primitive recursive function can still be defined. In this way, all of PRA is recoverable within the staged framework.

18 The Peano axioms

Once 0 and successor are in place, the classical Peano axioms can be restated in staged form. To illustrate how this is done, we write out one case in full (for the sake of brevity we will use presentational short-cuts for the others).
Non-zero successor. For any $n^{[n]}$
$(S(n)^{[n+1]} \neq 0^{[0]})^{[n+2]}$
Here $S(n)$ is formed at stage $[n+1]$; the atomic inequality is a new formula, so it lives at stage $[n+2]$.
Having seen the pattern, we introduce a shorthand convention**:**
Whenever we write an axiom schematically, e.g. $(x+0) = x$, or $\varphi(n) \rightarrow \varphi(S\,(n))$ it is to be read as if it was presented fully staged, i.e. each numeral k stands for $(k)^{[k]}$, each operation raises the stage by one, and the entire equality, or conditional does the same.
With this understood, we can state the staged version of the Peano Axioms:

1. Zero. $0^{[0]}\in \mathbb{N}$
2. Successor. If $n^{[n]}\in \mathbb{N}$, then $S(n^{[n]})^{[n+1]}\in \mathbb{N}$
3. Injectivity. If $(m+1)^{[m+1]} = (n+1)^{[n+1]}$ then $m^{[m]} = n^{[n]}$
4. 0 is not the successor of any number given above.
5. Induction: if $\varphi\left(0^{[0]}\right)$ holds, and for each $n^{[n]}$ $(\varphi(n^{[n]}) \to \varphi((n+1)^{[n+1]})$ holds (at stage $n+3$, then $\forall n^{[n]}\varphi(n^{[n]})$ holds at stage $n+4$.

The first four axioms are straightforward: since numbers are generated in strict succession, no collapse occurs. The induction axiom requires more care, because quantification itself is stage-advancing. Two complementary formulations are available. On the schema reading, induction is expressed as we have just presented it above: for each numeral at its own stage, the induction step guarantees the next numeral, so every instance of $\varphi(n^{[n]})$ eventually holds. On the modal reading, induction is captured by persistence across stages:

$$\varphi(0) \wedge \Box(\forall n(\varphi(n) \to \varphi(Sn)) \to \forall n\varphi(n)). \tag{9}$$

Both formulations respects the staged discipline (the universal quantifier is always evaluated one step above its instances), yet still permit induction across the entire domain. The modal form shows explicitly how the induction step 'propagates' through the hierarchy. Thus the full Peano axioms are derivable, and ordinary number theory is fully available.

19 Reverse mathematics

The programme of reverse mathematics (Friedman 1975, 1976; [8] Simpson 1984, 1988, 1998) analyses the strength of theorems relative to subsystems of second-order arithmetic. Its core tools are comprehension and induction axioms.

In staged logic, comprehension is automatically stratified: for any property $\varphi(x)$ at $[n]$, the set $\{x|\varphi(x)\}$ is introduced at $[n+1]$. This blocks Russell-type self-membership, but still supplies all the sets needed for ordinary mathematics. Induction axioms similarly shift one stage up.

Hence the basic systems of reverse mathematics can be mirrored:

- **RCA$_0$ (Recursive Comprehension Axiom):** staged comprehension for recursive predicates is unproblematic, since each recursion unfolds stage by stage.

- **WKL$_0$ (Weak König's Lemma):** the compactness arguments carry through once binary trees are staged.
- **ACA$_0$ (Arithmetical Comprehension):** comprehension for arithmetical predicates is staged, but remains consistent.

While a full staged reverse mathematics has not yet been developed, the architecture is already in place: each subsystem's comprehension principles translate into staged form without paradox. Thus the programme can be pursued without risking inconsistency.

19.1 Second-order completeness

In classical logic, second-order quantifiers cannot be captured by Henkin semantics without giving up categoricity; completeness theorems fail. In the staged setting, however, quantifiers range only over domains already fixed at the previous stage.
A staged Henkin construction proceeds as follows:

- At each stage $[n]$, add constants to witness existential claims about objects in $D_{[n]}$.
- At stage $[n+1]$, build maximally consistent sets that include second-order variables ranging only over subsets of $D_{[n]}$.
- Interpret quantifiers accordingly.

Because diagonalisation cannot close within a single stage (see section 7.10), the Gödelian construction that blocks second-order completeness is unavailable. One can therefore prove:
Theorem (sketch). Every consistent set of staged second-order formulas has a staged model.
Proof sketch: extend any consistent set to a maximally consistent one at each stage by adding witnesses; interpret domains as closed terms; define second-order quantifiers over subsets definable at the previous stage. By induction on stage, satisfaction matches derivability. Hence completeness holds.
This result highlights that staged logic is not weaker but in some respects stronger: it permits a completeness theorem precisely where the classical system does not.

20 Logicism rehabilitated

The downfall of Frege's programme was Basic Law V:

$$Ext(F) = Ext(G) \leftrightarrow \forall x(Fx \leftrightarrow Gx). \tag{10}$$

In unstaged form this leads directly to Russell's paradox, since one can form the extension of "$x \notin x$."

In staged logic, however, abstraction is stage-advancing. The extension of a predicate Fat $[n]$ is introduced only at $[n+1]$. Hence, if we attempt to form the extension of "$x \notin x$," the predicate itself lives at $[n]$, while its extension lives at $[n+1]$. There is no stage at which they coincide, and no paradox arises.

Thus staged Basic Law V is consistent. From it, one can reproduce Frege's definitions of the natural numbers:

- 0 as the extension of the empty concept.
- The successor of n as the extension of "concepts equinumerous with some concept containing n."

Because extensions are always formed a stage above their defining concepts, no circularity occurs. From these definitions, the Peano axioms follow. Arithmetic is therefore derivable from purely logical resources. In this sense, staged logic restores the constructive heart of Frege's logicism, while avoiding the collapse that destroyed it.

21 Summary

Staged logic is therefore not merely a defensive patch against paradox but a positive foundation for mathematics. Primitive recursive arithmetic and the Peano axioms are fully recoverable; a completeness theorem is likely to hold even for second-order staged logic; subsystems of reverse mathematics can be mirrored; and Frege's logicist programme can be rehabilitated. Each of these achievements stems from the same simple mechanism: every object-level operation advances the stage. What blocks paradox also permits mathematics to unfold safely.

22 Conclusion

Staged logic enforces one principle: every operation is stage advancing. This dissolves semantic paradoxes (liar, Curry), set-theoretic paradoxes (Russell, set-of-all-sets, Burali-Forti), abstraction paradoxes, and diagonalisation paradoxes uniformly.

It preserves classical reasoning where safe, blocks paradox where needed, and suggests natural modal enrichments for schemas. It unifies what once seemed disparate: all paradoxes as same-stage self-reference. In this respect, it continues Gabbay's tradition: systematic, context-sensitive frameworks, not *ad hoc* repairs. Staged logic provides a principled account of paradox, a unified logical framework, and a philosophical resource for indefinite extensibility.

References

[1] Jean Van Heijenoort B. Letter to Frege. pages 124–125, 1902.

[2] J Beall, M Glanzberg, Ripley, E Zalta, and N Uri Nodelman. Liar paradox. 2023.

[3] D Builes and J Wilson. 179:2199–2236, 2022.

[4] Cesare Burali-Forti. Una questione sui numeri transfiniti. 11:154–164, 1897.

[5] M Dummett. The philosophical significance of Gödel's theorem. pages 186–214, 1963.

[6] H Friedman. Some systems of second order arithmetic and their use. 1:235–242, 1974.

[7] H Friedman. Systems of second order arithmetic with restricted induction. i, ii. 41:557–559, 1975.

[8] H Friedman, S Simpson, and R Smith. Countable algebra and set existence axioms. 25:319–320, 1983.

[9] D Gabbay. Labelled deductive systems. 1996.

[10] D Gabbay. 1999.

[11] D Gabbay and F Guenther. Handbook of philosophical logic: Volume iv: Topics in the philosophy of language. 167, 1989.

[12] F Guenther. Semantics and the liar paradox. pages 617–706, 1989.

[13] Ø Linnebo. Pluralities and sets. 107:144–164, 2010.

[14] Ø Linnebo. The potential hierarchy of sets. 6:205–228, 2013.

[15] Ø Linnebo and S Shapiro. Actual and potential infinity. 53:160–191, 2018.

[16] H Poincare. Les math´ematiques et la logique. 13:813–835, 1905.

[17] H Poincare. Les math´ematiques et la logique, ii. 14:294–317, 1906.

[18] H Poincare. The logic of infinity. pages 45–64, 1912.

[19] G Priest. The structure of the paradoxes of self-reference. 103:25–34, 1994.

[20] B Russell and A Whitehead. On some difficulties in the theory of transfinite numbers and order types. 1906.

[21] B Russell and A Whitehead. Mathematical logic as based on a theory of types. 1908.

[22] B Russell and A Whitehead. pages 1925–1927, 1910.

[23] C Scambler. Can all things be counted? 50:1079–1106, 2021.

[24] L Shapiro and N Zalta. Curry's paradox. 2021.

[25] S Simpson. Which set axioms are needed to prove the cauchy/peano theorem for ordinary differential equations. 49:783–802, 1984.

[26] S Simpson. Partial realization of hilbert's program. 53:349–363, 1988.

[27] S Simpson. Subsystems of second order arithmetic. 1998.

[28] A Tarski. The semantic conception of truth: and the foundations of semantics. 4:341–376, 03 1944.

[29] H Weyl. 1917.

[30] W Woodin. The modal logic of arithmetic potentialism and the universal algorithm. 2018.

Received August 2025

On Kuroda Formula

Valentin Shehtman
Higher School of Modern Mathematics, MIPT, and National Research University Higher School of Economics, Moscow, Russia
vshehtman@gmail.com

Dmitry Shkatov
School of Computer Science and Applied Mathematics, University of the Witwatersrand, Johannesburg, South Africa
shkatov@gmail.com

Dmitrij Skvortsov
Russian Academy of Sciences, Moscow, Russia

Abstract

We give some characterizations of the class of intuitionistic predicate Kripke frames validating the Kuroda formula. We, first, prove that the Kuroda formula does not define a class of frames definable by a classical first-order sentence and, second, establish a criterion for countable frames to validate this formula.

1 Introduction

One of the first papers on completeness for intermediate predicate logics was Dov Gabbay's [1]. Theorem 2 from that paper proves that the intuitionistic predicate logic enriched with the Kuroda formula is complete with respect to a class of trees enjoying the McKinsey property: every point sees a maximal point. Later on, Dov Gabbay gave another prove showing that this logic is complete with respect to the class of all Kripke frames (not necessarily trees) with the McKinsey property [2, Chapter 3, Section 4]. The McKinsey property is elementary (i.e., first-order definable), but the class $\mathscr{KF}$ of all predicate Kripke frames validating the Kuroda formula is much larger.

In the present paper, we give some characterizations of the class $\mathscr{KF}$. First, we prove that $\mathscr{KF}$ is not elementary. Second, we give an explicit criterion for countable members of $\mathscr{KF}$.

With regret, we share the sad news that Dmitrij Skvortsov passed away when the work on this paper was nearly complete.

2 Preliminaries

2.1 Formulas and logics

We consider logics in a pure predicate language $\mathcal{L}$ containing the following symbols: countably many individual variables; for every $n \geqslant 0$, countably many n-ary predicate letters (nullary letters are also called proposition letters); the propositional constant $\bot$; the binary connectives $\wedge$, $\vee$, and $\to$; the quantifier symbols $\exists$ and $\forall$ (note that $\mathcal{L}$ does not contain individual constants, function symbols, or equality).

The definition of $\mathcal{L}$-formulas (or, simply, formulas) is standard. We use the standard abbreviations $\neg A := A \to \bot$ and $A \leftrightarrow B := (A \to B) \wedge (B \to A)$, and adopt the usual conventions on omitting parentheses. In what follows, the language $\mathcal{L}$ is identified with the set of its formulas. A formula is propositional if it contains no individual variables, and hence no predicate letters other than nullary and no quantifier symbols. A formula A is negative if $A = \neg B$, for some B.

The free variables of a formula are its parameters. The universal closure of a formula A, which may be assumed to be unique up to the enumeration of its parameters, is denoted by $\bar{\forall} A$. A formula without parameters is called a sentence. A theory is a set of sentences.

A superintuitionistic predicate logic, or simply logic, is a set of formulas that includes the intuitionistic predicate logic $\mathbf{QH}$ and is closed under Predicate Substitution, Modus Ponens, and Generalisation; thus, $\mathbf{QH}$ is the smallest superintuitionistic predicate logic. If $\mathbf{L}$ is a logic and A a formula, then $\mathbf{L} \vdash A$ means the same as $A \in \mathbf{L}$. The smallest logic that includes a logic $\mathbf{L}$ and a set Γ of formulas is denoted by $\mathbf{L} + \Gamma$; if A is a formula, we write $\mathbf{L} + A$ instead of $\mathbf{L} + \{A\}$. If $\mathbf{L}$ is a logic, and A and B are formulas, then we say that

- B is derivable from A in $\mathbf{L}$, and write $A \vdash_{\mathbf{L}} B$, if $\mathbf{L} + A \vdash B$;
- A and B are deductively equivalent in $\mathbf{L}$ if they are mutually derivable, i.e., if $A \vdash_{\mathbf{L}} B$ and $B \vdash_{\mathbf{L}} A$;
- B is simply derivable from A, and write $A \vdash B$, if B is derivable from A in $\mathbf{QH}$;
- A and B are simply deductively equivalent if they are mutually derivable, i.e., if $A \vdash B$ and $B \vdash A$.

It should be clear that, if A and B are deductively equivalent, then $A \in \mathbf{L}$ if and only if $B \in \mathbf{L}$, for every logic $\mathbf{L}$.

It is well known that the set of all logics forms a lattice with respect to the set-theoretic inclusion; we denote this lattice by $\mathfrak{L}$. Algebraically, the meet of $\mathbf{L}_1, \mathbf{L}_2 \in \mathfrak{L}$

is the logic $\mathbf{L}_1 \cap \mathbf{L}_2$, and the join of $\mathbf{L}_1, \mathbf{L}_2 \in \mathfrak{L}$ is the logic $\mathbf{L}_1 + \mathbf{L}_2$. The least element of $\mathfrak{L}$ is the logic **QH**; the greatest, the inconsistent logic $\mathcal{L}$ (the set of all formulas). The lattice $\mathfrak{L}$ is complete since it is closed under intersection.

In what follows, we will be referring to the following formulas and logics:

$$\begin{array}{lcll} CD & := & \forall x(P(x) \vee q) \to \forall x P(x) \vee q & \text{(the constant domain principle);} \\ EM & := & p \vee \neg p & \text{(the law of excluded middle);} \\ Z & := & (p \to q) \vee (q \to p) & \text{(Dummett's formula);} \\ \mathbf{QLC} & := & \mathbf{QH} + Z & \text{(Dummett's logic).} \end{array}$$

Definition 2.1. Let $n \geqslant 1$, let C be a propositional formula in proposition letters $p_1, \ldots, p_k$, let $P_1, \ldots, P_k$ be a list of distinct n-ary predicate letters, and let $\boldsymbol{x}$ be a fixed list of n distinct individual variables. An n-shift of C, denoted by C^n, is the formula obtained by replacing in C, for every $i \in \{1, \ldots, k\}$, each occurrence of p_i with the atomic formula $P_i(\boldsymbol{x})$.

2.2 Some properties of posets

We denote the cardinality of a set X by $|X|$.

A partially ordered set (a poset, for short) is a pair $(U, \leqslant)$ where U is a set and $\leqslant$ is a partial order on U. In this paper, all posets are non-empty (i.e., $U \neq \varnothing$). If $\leqslant$ is understood from the context, we may simply say that U is a poset. If $(U, \leqslant)$ is a poset and $x, y \in U$, then, as usual, we write $x < y$ to mean that $x \leqslant y$ and $x \neq y$. The set of maximal points of a poset U is denoted by $max[U]$. If U is a poset and $u \in U$, then the cone of u in U is the set $U \uparrow u := \{w \in U \mid u \leqslant w\}$; if U is understood from the context, we write $u{\uparrow}$ instead of $U \uparrow u$. A subset X of a poset U is upward-closed if it includes the cones of all its points, i.e., if $(\forall u \in X)\, u{\uparrow} \subseteq X$. We define

$$\begin{array}{rcl} X{\uparrow} & := & \{u \in U \mid (\exists x \in X)\, x \leqslant u\}; \\ X{\downarrow} & := & \{u \in U \mid (\exists x \in X)\, u \leqslant x\}. \end{array}$$

Note that X is upward-closed iff $X{\uparrow} = X$.

Definition 2.2. Let $(U, \leqslant)$ be a poset and $X, Y \subseteq U$. We say that X is cofinal for Y if $Y \subseteq X{\downarrow}$. If X is cofinal for U itself, i.e., if $U \subseteq X{\downarrow}$, then we say that X is a cofinal subset of U or that X is cofinal in U. The cofinality $cf[U]$ of a poset U is the minimal cardinality of a cofinal subset of U.

Note that, since U itself is cofinal in U, surely $cf[U] \leqslant |U|$.

Lemma 2.3. Let $(U, \leqslant)$ be a poset and $X, Y \subseteq U$.

(1) If X is cofinal in U and $U \neq \varnothing$, then $X \neq \varnothing$.

(2) If Y is upward-closed, then X is cofinal for Y iff $X \cap Y$ is a cofinal subset of Y.

Remark 2.4. If Y is not upward-closed, then Lemma 2.3 (2) in general does not hold.

Lemma 2.5. Let $(U, \leqslant)$ be a linearly ordered set, X an upward-closed subset of U, and Y a non-empty subset of U. Then, X is cofinal for Y iff $X \neq \varnothing$.

Definition 2.6. A poset $(U, \leqslant)$ satisfies the McKinsey property if the set $max[U]$ of its maximal points is cofinal in U, i.e., if $U \subseteq max[U]{\downarrow}$.

Lemma 2.7. Let $(U, \leqslant)$ be a poset and $X \subseteq U$.

(1) If X is a cofinal in U, then $max[U] \subseteq X$.

(2) If U satisfies the McKinsey property, then X is cofinal in U iff $max[U] \subseteq X$.

Lemma 2.8. Let U be a poset.

(1) $cf[U] \geqslant |max[U]|$.

(2) If U satisfies the McKinsey property, then $cf[U] = |max[U]|$.

Remark 2.9. The converse of Lemma 2.8 (2) in general does not hold.

Lemma 2.10. The intersection of a finite number of upward-closed cofinal subsets of a poset U is itself cofinal (and similarly for sets that are upward-closed and cofinal for some subset Y of U).

2.3 Constant domain Kripke semantics

Recall that a predicate Kripke frame with a constant domain is a triple $(U, \leqslant, D)$ where $(U, \leqslant)$ is a non-empty poset n(elements of U are called possible words) and D is a non-empty set (called the domain of individuals); the frame $(U, \leqslant, D)$ is called a frame over $(U, \leqslant\ )$ with the domain D.

For a non-empty set D, a D-sentence is an expression obtained from a predicate formula by replacing all its parameters with elements of D.[1]

[1] More precisely, with constants naming elements of D.

If F is a frame over $(U, \leqslant)$ with a domain D, then a valuation over F, or an F-valuation, is a map V sending each D-sentence A to upward-closed subsets of $(U, \leqslant)$ and satisfying the following conditions:[2]

$$\begin{array}{lcl} V(\bot) & = & \varnothing, \\ V(A \wedge B) & = & V(A) \cap V(B), \\ V(A \vee B) & = & V(A) \cup V(B), \\ V(A \to B) & = & \{u \in U \mid u{\uparrow} \cap V(A) \subseteq V(B)\}, \\ V(\forall x A(x)) & = & \bigcap \{V(A(d)) \mid d \in D\}, \\ V(\exists x A(x)) & = & \bigcup \{V(A(d)) \mid d \in D\}. \end{array}$$

These clauses imply that $V(\neg A) = \{u \in U \mid u{\uparrow} \cap V(A) = \varnothing\}$.

We can rewrite $u \in V(A)$ as $u \models A$ ('the truth of A at u') to see the usual inductive clauses for the intuitionistic constant domain Kripke semantics [3, Chapter 3]:

$$\begin{array}{lcl} u \not\models \bot; & & \\ u \models A \wedge B & \iff & v \models A \text{ and } v \models B; \\ u \models A \vee B & \iff & v \models A \text{ or } v \models B; \\ u \models A \to B & \iff & (\forall v \geqslant u)\,(v \models A \Rightarrow v \models B); \\ u \models \forall x A(x) & \iff & (\forall d \in D)\, u \models A(d); \\ u \models \exists x A(x) & \iff & (\exists d \in D)\, u \models A(d). \end{array}$$

These clauses imply that $u \models \neg A$ iff $\forall v \geqslant u\,(v \not\models A)$.

Let F be a predicate Kripke frame with a constant domain. An F-model is a pair $M = (F, V)$, where V is an F-valuation. We say that a formula A is true in a model M, or M-true, and write $M \models A$, if $V(\bar{\forall} A) = U$, i.e., if $V(A(\boldsymbol{d})) = U$, for every list $\boldsymbol{d}$ of constants from D corresponding to the parameters of A. We say that a formula A is valid on F, or F-valid, and write $F \models A$, if it is M-true in every F-model M. The set of all F-valid formulas is called the logic of a frame F and is denoted by $\mathbf{L}F$. It is well known that, for every frame F, the set $\mathbf{L}F$ is, indeed, a logic and that, moreover,

$$\bigcap \{\mathbf{L}F \mid F \text{ is a predicate frame with a constant domain}\} \quad = \quad \mathbf{QH} + \mathit{CD}.$$

The following simple lemma plays an important role in our considerations:

[2] These are the usual inductive clauses for truth in an algebraic structure over the Heyting algebra $H(U)$ of upward-closed subsets of a poset $(U, \leqslant)$ with a domain D; see [3, Chapter 4].

Lemma 2.11 (Double negation lemma). Let F be a frame over $(U, \leqslant\ )$ with the domain D, let V be an F-valuation, and A a D-sentence. Then,

(1) $V(\neg\neg A) = \{u \in U \mid V(A) \text{ is cofinal for } u{\uparrow}\}$.

(2) $V(\neg\neg A) = U$ iff $V(A)$ is cofinal in U.

Lemma 2.12. Let F be a frame with the domain D, let $M = (F, V)$ be an F-model, and let A be a formula. Then,

(1) $M \models \neg\neg A$ iff, for all lists $\boldsymbol{d}$ of elements of D, the sets $V(A(\boldsymbol{d}))$ are cofinal in U.

(2) $F \models \neg\neg A$ iff, for all F-valuations V and all lists $\boldsymbol{d}$ of elements of D, the sets $V(A(\boldsymbol{d}))$ are cofinal in U.

2.4 Elementarity

In what follows, $\Vdash$ denotes the classical validity, understood in the usual way.

To describe Kripke frames with constant domains classically, we fix the classical predicate language with two sorts of variables (one for worlds, denoted by $u, v, u_1, \ldots$, and one for individuals, denoted by $x, y, x_1, \ldots$), with equality applicable to pairs of variables of the same sort, and with a binary predicate symbol R restricted to variables for worlds. Every predicate Kripke frame F over $(U, \leqslant)$ with the domain D gives rise to the classical structure $\bar{F}$ with two domains, U and D, and with the binary predicate R on U defined so that

$$\bar{F} \Vdash R(u, v) \quad \text{iff} \quad u \leqslant v.$$

It should be clear that the classical theory T_{po} of the class of all structures of this type contains (and is axiomatized by) the usual axioms for partial orders together with equality axioms for both sorts of variables.[3]

An (intuitionistic predicate) formula A is called elementary (or first-order definable) w.r.t. to a class of frames $\mathscr{F}$ (within a fixed semantics) if there exists a classical theory $T \supseteq T_{po}$ such that, for every $F \in \mathscr{F}$,

$$F \models A \quad \Longleftrightarrow \quad \bar{F} \Vdash T,$$

in which case we say that the classical theory T corresponds to the formula A. If $\mathbf{L}$ a logic, we say that a formula A is elementary w.r.t. $\mathbf{L}$ if A is elementary w.r.t. the class $\{F \in \mathscr{F} \mid F \models \mathbf{L}\}$.

[3] Equality of worlds can be defined by $u = v := R(u, v) \wedge R(v, u)$, and is, therefore, strictly not needed.

3 Kuroda formula and Kuroda logic

3.1 Different axiomatizations of Kuroda logic

We will be using two deductively equivalent versions of the Kuroda formula:

$$\begin{array}{lcll} K & := & \neg\neg\forall x(P(x)\vee\neg P(x)) & (= \neg\neg\forall x EM^1), \\ DNS & := & \forall x\neg\neg P(x)\to\neg\neg\forall x P(x) & \text{(the double negation shift).} \end{array}$$

Since **QH** proves the converse of *DNS*, the formula *DNS* is **QH**-equivalent to the formula

$$DNS_{\leftrightarrow} \quad := \quad \forall x\neg\neg P(x)\leftrightarrow\neg\neg\forall x P(x).$$

Observation.

(1) Formulas K and *DNS* are deductively equivalent.

(2) For every $n \geqslant 1$, the formula *DNS* is deductively equivalent to

$$DNS_n \quad = \quad \forall x_1\ldots\forall x_n\neg\neg P^{(n)}(x_1,\ldots,x_n)\to\neg\neg\forall x_1\ldots\forall x_n P^{(n)}(x_1,\ldots,x_n)$$

(and, hence, to the formula obtained from DNS_n by replacing $\to$ with $\leftrightarrow$).

(3) For every $n \geqslant 1$, the formula K is deductively equivalent to

$$K_n \quad = \quad \neg\neg\forall x_1\ldots\forall x_n EM^n.$$

We call the logic

$$\mathbf{QHK} \quad := \quad \mathbf{QH}+K$$

the Kuroda logic.

The next proposition describes an extensive family of axiomatizations of **QHK**:

Proposition 3.0. Let C be a propositional formula axiomatizing the classical propositional logic **CL**, i.e., such that $\mathbf{H}+C=\mathbf{CL}$. Then, for every $n \geqslant 1$, the formula

$$K_n^C := \neg\neg\forall x_1\ldots\forall x_n\, C^n(x_1,\ldots,x_n)$$

is deductively equivalent to K.

3.2 Extremal properties of Kuroda logic

The Kuroda formula K had been discovered as a historically first example of a classically, but not intuitionistically, valid formula of the form $\neg A$, i.e, as a counterexample to the purported predicate counterpart of the celebrated Glivenko theorem. In fact, K is the strongest formula with the said property:

Claim 3.1. Let A be a formula such that $\neg A \in \mathbf{QCL} \setminus \mathbf{QH}$. Then, $K \vdash_{\mathbf{QH}} \neg A$.

This is an immediate consequence of the following:

Claim 3.2. Let A be a formula. Then, $\mathbf{QHK} \vdash \neg A$ if and only if $\mathbf{QCL} \vdash \neg A$.

We do not know whether the set $\mathbf{QCL} \setminus \mathbf{QH}$ (or its subset containing only formulas of the form $\neg A$) has the weakest formula. On the other hand, due to the disjunctive property of **QH**, this set cannot contain more than one minimal formula.

Claim 3.2 implies that **QHK** is the least (weakest) logic with the property we are considering:

Proposition 3.3. Let **L** is a predicate logic. Then, the following conditions are equivalent:

(i) $\mathbf{L} \vdash \neg A$ iff $\mathbf{QCL} \vdash \neg A$, for every formula A;

(ii) $\mathbf{QHK} \subseteq \mathbf{L} \subseteq \mathbf{QCL}$.

Proof. $(ii) \Rightarrow (i)$: In view of Claim 3.2, 'if' and 'only if' follow, respectively, from the first and the second inclusions of (ii).

$(i) \Rightarrow (ii)$: Since K is classically valid, it follows, by (i), that $K \in \mathbf{L}$; hence, $\mathbf{QHK} \subseteq \mathbf{L}$. Next, if the second inclusion does not hold, then there exists $A \in \mathbf{L} \setminus \mathbf{QCL}$. Hence, $\neg\neg A \in \mathbf{L} \setminus \mathbf{QCL}$, in contradiction with (i). □

3.3 Connections with the system of negative slices

The negative fragment $\mathbf{L}^{\neg}$ of a logic **L** is the set

$$\mathbf{L}^{\neg} := \{\neg A \mid \mathbf{L} \vdash \neg A\}.$$

The double negative fragment $\mathbf{L}^{\neg\neg}$ of a logic **L** is the set

$$\mathbf{L}^{\neg\neg} := \{\neg\neg A \mid \mathbf{L} \vdash \neg\neg A\}.$$

The fragments[4] $\mathbf{L}^{\neg}$ and $\mathbf{L}^{\neg\neg}$ determine each other, in the following sense:

$$\mathbf{L}^{\neg\neg} = \{\neg\neg A \in \mathcal{L} \mid \neg\neg A \in \mathbf{L}^{\neg}\} \quad \text{and} \quad \mathbf{L}^{\neg} = \{\neg A \in \mathcal{L} \mid \neg\neg\neg A \in \mathbf{L}^{\neg\neg}\}.$$

We say that logics $\mathbf{L}_1$ and $\mathbf{L}_2$ are negatively equivalent, and write $\mathbf{L}_1 \equiv_{Neg} \mathbf{L}_2$, if their negative (equivalently, double negative) fragments coincide:

$$\mathbf{L}_1 \equiv_{Neg} \mathbf{L}_2 \quad := \quad \mathbf{L}_1^{\neg} = \mathbf{L}_2^{\neg} \quad (\iff \mathbf{L}_1^{\neg\neg} = \mathbf{L}_2^{\neg\neg}).$$

It should be clear that the relation $\equiv_{Neg}$ is an equivalence on the structure $\mathfrak{L}$ of predicate logics. These equivalence classes form a system of slices in $\mathfrak{L}$, as defined in [7, Section 2] and [5, Section 3.2]. We call them negative slices. This system of negative slices was introduced and briefly discussed in [7, Section 3]; the reader can compare it to the system of positive slices introduced and studied in [5].

We denote the negative slice of a logic $\mathbf{L}$ by $[\mathbf{L}]_{Neg}$. It turns out that all negative slices are segments $\{\mathbf{L} : \mathbf{L}_1 \subseteq \mathbf{L} \subseteq \mathbf{L}_2\}$. On the one hand, the least logic of the slice $[\mathbf{L}]_{Neg}$ is $\mathbf{QH} + \mathbf{L}^{\neg}$, since the negative fragments of all logics from a given negative slice coincide. On the other hand, the greatest logic of a slice is the sum of all its logics. In general, we do not know a more satisfactory description of the greatest logic of an arbitrary negative slice.

Coming back to the Kuroda logic **QHK**, we see that Proposition 3.3 means that the negative slice $[\mathbf{QCL}]_{Neg}$ of the classical logic **QCL** is just the segment between **QHK** and **QCL**, i.e., the segment of all logics with the classical negative fragment. In other words, the Kuroda logic is the least logic with the classical negative fragment (the crucial characteristic property of the Kuroda logic). The negative fragments of some superclassical logics, i.e., extensions of **QCL**, were briefly described in [7, Section 2]; a more detailed description will be presented in the series of papers announced in [6].

3.4 Non-elementarity of the Kuroda formula

Proposition 3.4. The formula K is non-elementary w.r.t. the class of all Kripke frames with constant domains; moreover, it is non-elementary, in this semantics, w.r.t. the logic **QLC**, i.e., w.r.t. the class of Kripke frames with a constant domain over a linearly ordered set of possible worlds.

Proof. Suppose that K, and hence DNS, corresponds to a classical theory $T \supseteq T_{po}$. Let Θ_n (here, $n \geqslant 1$) be the theory obtained by adding to T the following formulas:

[4] These fragments were studied in [7], where they were denoted by, respectively, $Neg[\mathbf{L}]$ and $\mathbf{L}^{\neg\neg}$.

- the axiom of linearity $Lin := \forall u \forall v\,(R(u,v) \vee R(v,u))$;
- the axiom of seriality $Ser := \forall u \exists v\,(R(u,v) \wedge (u \neq v))$;
- the axiom postulating that $|D| \geqslant n$, i.e.,

$$E_n \quad := \quad \exists x_1 \ldots \exists x_n \bigwedge_{i \neq j} (x_i \neq x_j).$$

Also, let $\Theta := \bigcup\limits_{n \in \omega} \Theta_n$. First, we prove the following:

(I) For every $n \in \omega$, the theory Θ_n has a model over the set ω of natural numbers.

Let F_n be a Kripke frame over $(\omega, \leqslant)$ with the n-element domain $\{d_1, \ldots, d_n\}$. We show that $F_n \models DNS$, i.e., that $V(\forall x \neg\neg P(x)) \subseteq V(\neg\neg\forall x P(x))$, for every F_n-valuation V. Suppose that $u \in V(\forall x \neg\neg P(x))(= \bigcap\limits_{i=1}^{n} V(\neg\neg P(d_i)))$. Then, by Lemma 2.11 (1), for every $i \in \{1, \ldots, n\}$, the set $V(P(d_i))$ is cofinal for $u{\uparrow}$. Since $\varnothing$ is not cofinal for any non-empty subset of U, it follows that $V(P(d_i)) \neq \varnothing$, for every $i \in \{1, \ldots, n\}$. Choose $u_i \in V(P(d_i))$ and put $u_0 := \max\{u_1, \ldots, u_n\}$. Then, by upward closure, $u_0 \in \bigcap\limits_{i=1}^{n} V(P(d_i))(= V(\forall x P(x)))$. Since $V(\forall x P(x))$ is non-empty and ω is linear, it follows, by Lemma 2.5, that $V(\forall x P(x))$ is cofinal in U. Hence, by Lemma 2.11 (2), $V(\neg\neg\forall x P(x)) = U$. Thus, $F_n \models DNS$. Therefore, the classical structure $\bar{F}_n$ corresponding to F_n validates T, and hence Θ_n.

(II) It follows from (I), by Compactness theorem, that the theory Θ has a model; hence, by the Löwenheim–Skolem theorem, it has a countable model, say $\bar{F}$. To obtain a contradiction, we will show that $\bar{F} \not\Vdash T$. To that end, it suffices to prove the following:

(III) The Kripke frame F corresponding to $\bar{F}$ does not validate DNS.

First, note that the set of worlds U of the frame F is a denumerable chain; this chain cannot be finite due to seriality of R. Also note that, since $\bar{F} \Vdash E_n$ whenever $n \in \omega$, the domain D of F is also denumerable. Fix a bijection $\varphi\colon D \to U$ and define an F-valuation V as follows: for every $d \in D$,

$$V(P(d)) \quad := \quad \varphi(d){\uparrow} \setminus \{\varphi(d)\}. \qquad (*)$$

Due to seriality, for every $d \in D$, the set $V(P(d))$ is non-empty, and so, by Lemma 2.5, cofinal in U. Thus, by Lemma 2.11 (2), $V(\neg\neg P(d)) = U$, for every $d \in D$. Hence,

$$V(\forall x\, \neg\neg P(x)) \quad = \quad \bigcap_{d \in D} V(\neg\neg P(d)) \quad = \quad U.$$

On the other hand, the set $V(\forall x P(x))(= \bigcap\limits_{d\in D} V(P(d)))$ is empty since, for every $u \in U$, by $(*)$, $u \notin V(P(\varphi^{-1}(u)))$.
Hence, $V(\neg\forall x P(x)) = U$, and so $V(\neg\neg\forall x P(x)) = \varnothing$. Thus, $V(\mathit{DNS}) = \varnothing$, and so $F \not\models \mathit{DNS}$. □

We next observe that, in the statement of Proposition 3.4, the logic **QLC** cannot be replaced with logics of finite heights.

Recall that the height of a poset U, denoted by $h[U]$, is the supremum of cardinalities of the chains in U; if F is a predicate Kripke frame over a poset U, then we define $h[F] := h[U]$. Thus, over frames of a finite height, the formula K is valid, and so corresponds to the empty theory (hence, it is, trivially, elementary w.r.t. logics of finite heights).

We also recall the formulas P_h and P_h^+ corresponding to frames of finite height h; these formulas are defined by recursion (here, q_h and Q_h, for each $h \in \omega$, are distinct, respectively, nullary and unary predicate letters):

$$\begin{array}{lllllll} P_0 & := & \bot, & P_{h+1} & := & q_h \vee (q_h \to P_h); \\ P_0^+ & := & \bot, & P_{h+1}^+ & := & \forall x\, (Q_h(x) \vee (Q_h(x) \to P_h^+)). \end{array}$$

It is well known that, for every predicate Kripke frame F with constant or expanding domains,[5] for every $h \in \omega$,

$$F \models P_h \quad \Longleftrightarrow \quad F \models P_h^+ \quad \Longleftrightarrow \quad h[F] \leqslant h.$$

The formula K is valid on every Kripke frame F with expanding domains over a poset of a finite height. It is known (see [3, Theorem 6.3.8]) that every logic $\mathbf{QH} + P_h^+$ is complete in the semantics with expanding domains, i.e., for every $h \in \omega$,

$$\mathbf{QH} + P_h^+ \quad = \quad \bigcap (\mathbf{L}F : h[F] \leqslant h).$$

However, if $h \geqslant 2$, then the logic $\mathbf{QH} + P_h$ is Kripke incomplete, i.e., it is a proper sublogic of the logic $\mathbf{QH} + P_h^+$; this follows from the fact, proved by H. Ono [4], that $\mathbf{QH}+P_2 \not\vdash K$ and so, since $\mathbf{QH}+P_h \subseteq \mathbf{QH}+P_2$ whenever $h \geqslant 2$, also $\mathbf{QH}+P_h \not\vdash K$.[6]

Since K is valid on every Kripke frame of a finite height, it is elementary w.r.t. the logic $\mathbf{QH} + P_h^+$, for every $h \in \omega$. Since every frame validating P_h validates P_h^+, an analogous claim holds for weaker logics $\mathbf{QH} + P_h$. It should also be clear that K is elementary w.r.t. the logic $\mathbf{L}_\infty^+ := \bigcap\limits_{h\in\omega} (\mathbf{QH} + P_h^+)$ of all frames of a finite height.

[5] The semantics of expanding domains will be sketched out in Section 3.5.2; more details can be found in [3, Chapter 3].

[6] Note that $\mathbf{QH}+P_0^+ = \mathbf{QH}+P_0 = \mathcal{L}$ (the inconsistent logic) and $\mathbf{QH}+P_1^+ = \mathbf{QH}+P_1 = \mathbf{QCL}$ (the classical predicate logic); these logics are, clearly, Kripke complete.

3.5 On the validity of the Kuroda formula in Kripke semantics

3.5.1 Validity of K in the constant domain Kripke semantics

Definition 3.5. Let $(U, \leqslant)$ be a poset and $\mathcal{X}$ a family of subsets of U. We say that $\mathcal{X}$ is K-critical if there exists $u \in U$ such that

- every member of $\mathcal{X}$ is an upward-closed cofinal subset of $u{\uparrow}$;
- $\bigcap \mathcal{X}$ is not cofinal in $u{\uparrow}$.

If, additionally, $\bigcap \mathcal{X} = \varnothing$, then $\mathcal{X}$ is $K^{\varnothing}$-critical. If K-criticality of $\mathcal{X}$ is witnessed by $u \in U$, we say that $\mathcal{X}$ is K-critical for u. We say that a poset $(U, \leqslant)$ is K-critical if there exists a K-critical family in $(U, \leqslant)$.

Definition 3.6. Let $(U, \leqslant)$ be a poset. The K-cofinality of U, denoted by $cf_K[U]$, is defined as follows:

- if U is K-critical, then $cf_K[U]$ is the minimal cardinality of a K-critical family in $(U, \leqslant)$;
- otherwise, $cf_K[U] := \infty$ (we assume that ∞ is greater than any cardinal).

Lemma 3.7. Let $(U, \leqslant)$ be a poset. Then, $cf_K[U]$ equals the minimum cardinality of a $K^{\varnothing}$-critical family in $(U, \leqslant)$.

Lemma 3.8. A poset is not K-critical if and only if it does not contain $K^{\varnothing}$-critical families.

Theorem 3.9. Let F be a constant domain Kripke frame over a poset $(U, \leqslant)$ with the domain D. Then, F validates K if and only if $|D| < cf_K[U]$.

Proof. Suppose, first, that $F \not\models K$, and so $F \not\models DNS$. Then, there exist $u \in U$ and an F-valuation V such that $u \in V(\forall x \neg\neg P(x)) \setminus V(\neg\neg\forall x P(x))$.

Then, on the one hand, $u \in V(\neg\neg P(d))$, for every $d \in D$, i.e., by Lemma 2.11 (1) and Lemma 2.3(2), for every $d \in D$, the upward-closed set $X_d := V(P(d)) \cap u{\uparrow}$ is cofinal in $u{\uparrow}$. On the other hand, again by Lemma 2.11 (1), the set $V(\forall x P(x))$ ($= \bigcap\limits_{d \in D} V(P(d))$) is not cofinal for $u{\uparrow}$ in U, and so $\bigcap\limits_{d \in D} X_d$ is not cofinal in $u{\uparrow}$. Hence, the family $\mathcal{X} := \{X_d \mid d \in D\}$ is K-critical (for u) in $(U, \leqslant)$, and so $|D|(= |\mathcal{X}|) \geqslant cf_K[U]$.

Suppose, next, that $|D| \geqslant |\mathcal{X}| \geqslant cf_K[U]$. Then, $(U, \leqslant)$ contains a K-critical family $\mathcal{X}$ (say, for $u \in U$) with $|\mathcal{X}| \leqslant |D|$. Fix a surjection $\varphi : D \to \mathcal{X}$ and define an F-valuation V so that $V(P(d)) = \varphi(d)$, for every $d \in D$. Then, for every $d \in D$, the set $V(P(d))$ is a cofinal subset of $u\uparrow$, and so, by Lemma 2.11 (1), $(\forall d \in D)\, u \in V(\neg\neg P(d))$, i.e., $u \in \bigcap\limits_{d\in D} V(\neg\neg P(d))(\,=V(\forall x\neg\neg P(x)))$. On the other hand, since $\bigcap \mathcal{X}(\,=V(\forall x P(x)))$ is not cofinal in $u\uparrow$, if follows, by Lemma 2.11 (1), that $u \notin V(\neg\neg\forall x P(x))$. Hence, $u \in V(\forall x\neg\neg P(x)) \setminus V(\neg\neg\forall x P(x))$, and so $F \not\models DNS$. Hence, $F \not\models K$. □

Corollary 3.10. Let $(U, \leqslant)$ be a poset. Then, the following conditions are equivalent:

(i) Every Kripke frame with constant domains over $(U, \leqslant)$ validates K;

(ii) $cf_K[U] = \infty$ (i.e., if U is not K-critical).

Lemma 3.11. Every poset with the McKinsey property is not K-critical.

Lemma 3.12. The formula K is valid on every Kripke frame with constant domains over posets satisfying the McKinsey property.

In fact, Lemma 3.12 extends to the Kripke semantics with expanding domains—see Proposition 3.22.

Conjecture 3.13. A poset has the McKinsey property if and only if it is not K-critical.

Remark 3.14. Due to Lemma 2.10, $cf_K[U] \geqslant \aleph_0$ for every poset U.

Lemma 3.15. Let U be a linearly ordered poser without the greatest point. Then, $cf_K[U] = cf[U]$.

Remark 3.16. If U is a linear poset with the greatest point, then $cf[U] = 1$, by Lemma 2.5, while, due to Lemma 3.11, $cf_K[U] = \infty$.

Thus, Theorem 3.9 gives us the following:

Corollary 3.17. A Kripke frame F with constant domain over a linearly ordered U without the greatest point validates K if and only if $|D| < cf[U]$.

This claim gives us many linear frames with a constant domain validating K; e.g.,

Example 3.18. Let F be a frame over $U = \aleph_1$ with a denumerable domain D. Then $cf[U] = \aleph_1$. Therefore, F validates K.

Lemma 3.19. The formula K is valid on every Kripke frame with a finite constant domain.

Proof. Let F be a constant domain frame with the domain $\{d_1, \dots, d_m\}$ and let V be an F-valuation. Then,

$$V(\forall x \neg\neg P(x)) = \bigcap_{d=1}^{n} V(\neg\neg P(d_i)) = V(\neg\neg P(d_1)) \cap \ldots \cap V(\neg\neg P(d_n)),$$

and, likewise,

$$V(\neg\neg\forall x P(x)) = V(\neg\neg(P(d_1)) \cap \ldots \cap P(d_n)).$$

Since $\mathbf{H} \vdash \neg\neg p_1 \wedge \ldots \wedge \neg\neg p_n \leftrightarrow \neg\neg(p_1 \wedge \ldots \wedge p_n)$, it follows that $V(\forall x\neg\neg P(x)) = V(\neg\neg\forall x P(x))$, and so F validates DNS, and hence K. □

3.5.2 Expanding domains Kripke semantics

We assume that the reader is familiar with the expanding domains Kripke semantics (see, e.g., [3, Chapter 3]). However, to fix terminology and notation in a way consistent with our treatment of the constant domain semantics, we briefly recall the main differences between the two semantics.

In the expanding domains semantics, domains of worlds may differ. Hence, a predicate Kripke frame F with expanding domains, in addition to a non-empty poset $(U, \leqslant)$ and a non-empty domain D, has a system $\bar{D} := \{D_u \mid u \in U\}$ of domains satisfying the following conditions: first, for every $u \in U$, the set D_u is non-empty; second, $D_u \subseteq D_v \subseteq D$ whenever $u \leqslant v$. The set D_u is called the individual domain of a world $u \in U$, or a local domain of u. The global domain of a frame F is the set $D[F] := \bigcup\{D_u \mid u \in U\}$. Clearly, $D[F] \subseteq D$. Sometimes, it is assumed that $D[F] = D$; this restriction has no effect on the semantics since it does not affect truth or validity of formulas.

Let D be the global domain of a Kripke frame and let $d \in D$. The measure of existence of d is defined by $E(d) := \{u \in U \mid d \in D_u\}$. The measure of existence of a D-sentence A is defined by $E(A) := E(d_1) \cap \ldots \cap E(d_k)$, where $\{d_1, \dots, d_k\}$ is the set of constants from D occurring in A; thus,

$$\begin{array}{rcl} E(A) & = & \{u \in U \mid \text{ all constants occurring in } A \text{ belong to } D_u\} \\ & = & \{u \in U \mid A \text{ is a } D_u\text{-sentence}\}. \end{array}$$

If A and B are D-sentences, then

$$\begin{array}{rcll} E(\neg A) & = & E(A); & \\ E(A \circ B) & = & E(A) \cap E(B) & \text{if } \circ \in \{\wedge, \vee, \rightarrow\}; \\ E(A(d)) & = & E(\kappa x A(x)) \cap E(d) & \text{if } \kappa \in \{\forall, \exists\}. \end{array}$$

It should be clear that, if A is a sentence without constants from D, then $E(A) = U$.

If F is a frame with expanding domains, then an F-valuation is a map V sending each D-sentence A to an upward-closed subset of $E(A)$ and satisfying the following conditions:

$$\begin{array}{lcl} V(\bot) & = & \varnothing; \\ V(A \wedge B) & = & V(A) \cap V(B); \\ V(A \vee B) & = & V(A) \cup V(B); \\ V(A \to B) & = & \{u \in E(A \to B) \mid (u{\uparrow} \cap V(A)) \subseteq V(B)\}; \\ V(\forall x A(x)) & = & \{u \in E(\forall x A(x)) \mid u{\uparrow} \subseteq \bigcap\{V(A(d)) \mid d \in D\}\}; \\ V(\exists x A(x)) & = & E(\exists x A(x)) \cap \bigcup\{V(A(d)) \mid d \in D\}. \end{array}$$

We can write $u \in V(A)$ as $u \models A$ to see the usual inductive clauses for the intuitionistic Kripke semantics with expanding domains (see, e.g., [3, Section 3.2]) such as

$$\begin{array}{lcl} u \models A \to B & \iff & (\forall v \geqslant u)\,(v \models A \Rightarrow v \models B); \\ u \models \forall x A(x) & \iff & (\forall v \geqslant u)\,(\forall d \in D_v)\, v \models A(d); \\ u \models \exists x A(x) & \iff & (\exists d \in D_u)\, u \models A(d). \end{array}$$

An F-model is a pair $M = (F, V)$, where F is a frame and V is an F-valuation. A formula A is true in a model M over frame F if $V(\bar{\forall} A) = U$, i.e., if, for all lists $\boldsymbol{d} = (d_1, \ldots, d_k)$ of constants from D corresponding to parameters of A,

$$V(A(\boldsymbol{d})) = E(A(\boldsymbol{d})) = E(d_1) \cap \ldots \cap E(d_k).$$

We say that a formula A is valid on a Kripke frame with expanding domains F (or, simply, F-valid), and write $F \models A$, if A is true in every model over F. The set of all F-valid formulas is called the logic of the frame F and is denoted by $\mathbf{L}F$. It is well known that $\mathbf{L}F$ is, indeed, a predicate logic; moreover,

$$\bigcap(\mathbf{L}F \mid F \text{ is a frame with expanding domains}) \quad = \quad \mathbf{QH}.$$

Lemma 3.20. Let F be a Kripke frame (with expanding domains) over a poset U with the domain D, let V be an F-valuation, and let A be a D-sentence. Then,

(1) $V(\neg\neg A) = \{u \in E(A) \mid V(A) \text{ is cofinal for } u{\uparrow} \text{ (in } U)\}$.

(2) $V(\neg\neg A) = E(A)$ iff $V(A)$ is cofinal in $E(A)$.

Lemma 3.21. A sentence $\neg\neg A$ is valid on a Kripke frame (with expanding domains) F if, for every F-valuation V, the set $V(A)$ is cofinal in $E(A)$.

3.5.3 Criterion of validity of K on Kripke frames with expanding domains

It should be clear that the sentence $\forall x EM^1(x)$ is classically valid; therefore, it is valid at every maximal point of every Kripke frame with expanding domains. Hence, due to Lemma 3.21, we obtain the following well-known result (this result is tacitly assumed by Gabbay in [2, Chapter 3, Section 4]):

Proposition 3.22. The formula K is valid on every Kripke frame with expanding domains satisfying the McKinsey property.

Since every frame of a finite height satisfies the McKinsey property, we obtain the following:

Corollary 3.23. The formula K is valid on every Kripke frame of a finite height, and in particular every Kripke frame over a finite poset.

If $F = (U, \leqslant, \bar{D})$ is a frame with expanding domains and $u \in U$, then the cone of F at u is the frame $F \uparrow u$ obtained by restricting both $\leqslant$ and $\bar{D}$ to $u\!\uparrow$. Next, we define the set of cones of F that contain only worlds with a finite constant domain:

$$\mathit{fcd}[F] \quad := \quad \{u \in U \mid F \uparrow u \text{ is a frame with a finite constant domain}\}.$$

It should be clear that $\mathit{fcd}[F]$ is an upward-closed subset of U. By Lemma 3.19, for every $u \in \mathit{fcd}[F]$, the frame $F \uparrow u$ validates K.

Definition 3.24. A Kripke frame with expanding domains F over a poset U is an fcd-frame if the set $\mathit{max}[U] \cup \mathit{fcd}[F]$ is cofinal in U.

Due to Lemma 3.21 and Lemma 3.19, we obtain the following:

Proposition 3.25. Every *fcd*-frame validates K.

We now establish a criterion for the validity of K on Kripke frames with expanding domains over countable posets:

Theorem 3.26. Let F is a Kripke frame with expanding domains over a countable poset U. Then, F validates K if and only if it is an *fcd*-frame.

Proof. The 'if' part is given by Proposition 3.25, so it remains to prove the 'only if' part.

Let F be a frame with expanding domains over U that is not an *fcd*-frame; this means that the set $\mathit{max}[U] \cup \mathit{fcd}[F]$ is not cofinal in U. Hence, there exists $u_0 \in U$ such that $u_0\!\uparrow \cap\, (\mathit{max}[U] \cup \mathit{fcd}[F]) = \varnothing$. We now replace F with the frame $F \uparrow u_0$ and prove the following lemma about it; this will conclude the proof of the theorem. □

Lemma 3.27. Let F be a frame with the root u_0 over a denumerable poset U such that

$$\mathit{max}[U] \cup \mathit{fcd}[F] = \varnothing.$$

Then, there exists an F-valuation V falsifying K; moreover, $V(\forall x EM^1(x)) = \varnothing$ (and hence $V(\neg\forall x EM^1(x)) = U$ and $V(K) = \varnothing$).

Proof. We enumerate the set of worlds so that $U = \{u_i \mid i \in \omega\}$ (recall that u_0 is the root of U). For every $i \in \omega$ we choose, inductively, a world $u'_i \geqslant u_i$ and a fresh individual $d_i \in D_{u'_i}$ (i.e., $d_i \neq d_j$ whenever $j < i$). If, for some $u'_i \geqslant u_i$, the domain $D_{u'_i}$ is infinite, then we choose this u'_i and a fresh $d_i \in D_{u'_i}$. Second, if, for every $v \geqslant u$, the domain D_v is finite, then we choose an increasing chain $u_i < v_0 < \ldots < v_i$ so that $D_{v_0} \subset D_{v_1} \subset \ldots \subset D_{v_i}$ (since $\mathit{fcd}[F] = \varnothing$). Then, $|D_{v_i}| > i$, and we can put $u'_i := v_i$ and choose d_i in $D_{u'_i} = D_{v_i}$ that is distinct from $d_0, \ldots, d_{i-1}$.

Finally, we put $V(P(d_i) := u'_i{\uparrow} \setminus \{u'_i\}$, for every $i \in \omega$; the values $V(P(d))$ for all other $d \in D$ are immaterial, so we can make all of them empty. Clearly, all $V(P(d_i)$ are non-empty, since $\mathit{max}[U] = \varnothing$.

Now, since $V(P(d_i)$ are non-empty subsets of $u'_i{\uparrow}$, it follows that $u'_i \notin V(P(d_i))$ and $u'_i \notin V(\neg P(d_i))$. Hence, $u'_i \notin V(P(d_i) \vee \neg P(d_i))$, for every $i \in \omega$. Thus, by monotonicity, for every $i \in \omega$,

$$u_i \notin V(P(d_i) \vee \neg P(d_i)) = V(EM^1(d_i)).$$

Thus, $u_i \notin V(\forall x EM^1(x))$, for every $u_i \in U$, which means that $V(\forall x EM^1(x)) = \varnothing$.

□

Remark 3.28. Theorem 3.26 does not extend to the uncountable case: for a counterexample, consider the frame from Example 3.18, where both $\mathit{max}[U]$ and $\mathit{fcd}[F]$ are empty (and so non-cofinal).

It appears doubtful whether there exists a nice criterion characterizing the whole class $\mathscr{K}\mathscr{F}$.

References

[1] Dov Gabbay. Application of trees to intermediate logics. Journal of Symbolic Logic, 37:135–138, 1972.

[2] Dov Gabbay. Semantical Investigations in Heyting's Intuitionistic Logic. D. Reidel, 1981.

[3] Dov Gabbay, Valentin Shehtman, and Dmitrij Skvortsov. Quantification in Nonclassical Logic, Volume 1, volume 153 of Studies in Logic and the Foundations of Mathematics. Elsevier, 2009.

[4] Hiroakira Ono. Model extension theorem and Craig's interpolation theorem for intermediate predicate logics. Reports on Mathematical Logic, 15:41–58, 1983.

[5] Mikhail Rybakov, Dmitry Shkatov, and Dmitrij Skvortsov. On the system of positive slices in the structure of superintuitionistic predicate logics. In A. Ciabattoni, D. Gabelaia, and I. Sedlár, editors, Advances in Modal Logic, volume 15, pages 653–674. College Publications, 2024.

[6] Dmitrij Skvortsov. On finite domains based slices in the structure of superintuitionistic predicate logics, preview. Logical Investigations, 29(1):101–113, 2023.

[7] Dmitrij Skvortsov. On systems of slices in the structure of superintuitionistic predicate (or propositional) logics I. Pattern Recognition and Image Analysis, 33:511–516, 2023.

Received September 2025

A Unifying Framework for Probabilistic Argumentation

Gianvincenzo Alfano
Department of Informatics, Modeling, Electronics and System Engineering, University of Calabria, Italy

Mario A. Leiva and Gerardo I. Simari
Department of Computer Science and Engineering, Universidad Nacional del Sur (UNS) & Institute for Computer Science and Engineering (UNS-CONICET), Argentina

Abstract

Formal argumentation has attracted significant attention in the field of Knowledge Representation and Reasoning over the past two decades. Dung's Argumentation Framework (AF) has been extended in various directions, including approaches that incorporate quantified uncertainty regarding the existence of arguments and attacks. However, comparatively less effort has been devoted to integrating probabilistic reasoning into structured argumentation or other extensions of AF. In this paper, we introduce the Unified Probabilistic Argumentation Framework (UPAF), a general and expressive theoretical model capable of capturing a wide range of existing argumentation formalisms. UPAF can be equipped with an *environmental model* that assigns (not necessarily independent) probabilistic events to elements of the underlying argumentation structure, thus enabling reasoning under uncertainty. We demonstrate that UPAF can encode classical and extended forms of Dung's framework, as well as structured argumentation frameworks such as Assumption-Based Argumentation and Defeasible Logic Programming. Finally, we discuss the computational complexity of key reasoning tasks within UPAF instances.

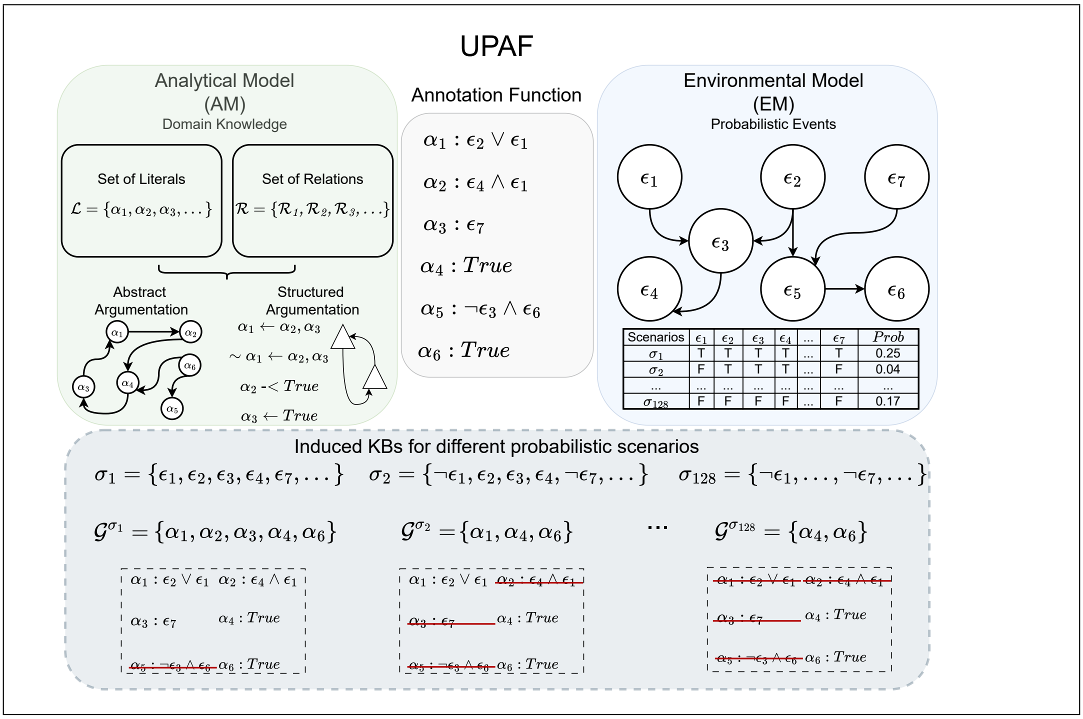
1 Introduction

People use argumentation to present and justify their views, persuade others, and logically derive conclusions—this process frequently occurs in contexts involving controversial information. Developing automated systems that can handle such information in a structured manner akin to human discussions is a significant challenge that has captivated the Artificial Intelligence community for decades. This has led to the creation of a vibrant research field known as formal argumentation [1], which has seen applications across a wide range of areas including legal reasoning [2, 3], decision support [4], electronic democracy [5], healthcare [6, 7], medical applications [8, 9], robotics [10], financial analysis [11], result explanation [12, 13], and the operation of multi-agent systems and social networks [14], among others.

Formal argumentation has attracted increasing attention for its intuitive nature and the fact that it tends to follow approaches to reasoning that are close to those applied by humans in formal reasoning tasks. Structured argumentation is focused on representing the inner structure of arguments, while abstract argumentation simply neglect the argument internals and simply focus on their relationships. There are several abstract and structured frameworks that have been studied and developed in depth in the past two decades or so, such as AF [15], Bipolar AF [16], SETAF [17], Assumption-based Argumentation (ABA) [18, 19], ASPIC+ [20], *Defeasible Logic Programming* (DeLP) [21, 22], *Deductive Argumentation* [23], and other rule-based systems like [24]. These frameworks offer principled approaches to knowledge representation and reasoning, making them well-suited for real-world applications where uncertainty and conflicting information are pervasive. By capturing the nuances of defeasible reasoning and handling incomplete or contradictory knowledge, these frameworks bridge the gap between formal logic and human-like reasoning, paving the way for intelligent systems to make more context-aware, robust, and human-compatible decisions.

In many applications and systems, the underlying knowledge bases gather information about a specific domain, and often contain contradictory data and uncertainty due to the diversity of sources and formats [25, 26]. To address these challenges, formalisms are used to represent knowledge, focusing on the ability to handle contradictory information and make inferences in situations of uncertainty. In this context, probability theory is one of the most developed approaches, where it starts with a set of possible scenarios representing the outcomes that an agent considers possible. The choice about the specific scenario to focus on typically reflects the agent's uncertainty. On the other hand, argumentation theory provides a logical and non-numerical approach to represent uncertainty, using arguments with premises and conclusions.

Recently, there has been an increasing interest in extending abstract argumentation frameworks to manage uncertain information. This has been carried out by either considering quantified uncertainty about the existence of arguments and attacks, thus combining formal argumentation with probability theory. In fact, Probabilistic Argumentation [27] can be viewed as part of the several proposals that have been made in the last decades for extending reasoning tasks in AI frame-


UPAF
Analytical Model
(AM)
Domain Knowledge
Set of Literals
$\mathcal{L} = \{\alpha_1, \alpha_2, \alpha_3, \dots\}$
Set of Relations
$\mathcal{R} = \{\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3, \dots\}$
Abstract Argumentation
α_1 α_2 α_3 α_4 α_5 α_6
Structured Argumentation
$\alpha_1 \leftarrow \alpha_2, \alpha_3$
$\sim \alpha_1 \leftarrow \alpha_2, \alpha_3$
$\alpha_2 \text{ -< } True$
$\alpha_3 \leftarrow True$
Annotation Function
$\alpha_1 : \epsilon_2 \vee \epsilon_1$
$\alpha_2 : \epsilon_4 \wedge \epsilon_1$
$\alpha_3 : \epsilon_7$
$\alpha_4 : True$
$\alpha_5 : \neg\epsilon_3 \wedge \epsilon_6$
$\alpha_6 : True$
Environmental Model
(EM)
Probabilistic Events
ϵ_1 ϵ_2 ϵ_3 ϵ_4 ϵ_5 ϵ_6 ϵ_7
Induced KBs for different probabilistic scenarios
$\sigma_1 = \{\epsilon_1, \epsilon_2, \epsilon_3, \epsilon_4, \epsilon_7, \dots\}$
$\sigma_2 = \{\neg\epsilon_1, \epsilon_2, \epsilon_3, \epsilon_4, \neg\epsilon_7, \dots\}$
$\sigma_{128} = \{\neg\epsilon_1, \dots, \neg\epsilon_7, \dots\}$
$\mathcal{G}^{\sigma_1} = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_6\}$
$\mathcal{G}^{\sigma_2} = \{\alpha_1, \alpha_4, \alpha_6\}$
...
$\mathcal{G}^{\sigma_{128}} = \{\alpha_4, \alpha_6\}$
$\alpha_1 : \epsilon_2 \vee \epsilon_1$ $\alpha_2 : \epsilon_4 \wedge \epsilon_1$ $\alpha_3 : \epsilon_7$ $\alpha_4 : True$ ~~$\alpha_5 : \neg\epsilon_3 \wedge \epsilon_6$~~ $\alpha_6 : True$
$\alpha_1 : \epsilon_2 \vee \epsilon_1$ ~~$\alpha_2 : \epsilon_4 \wedge \epsilon_1$~~ ~~$\alpha_3 : \epsilon_7$~~ $\alpha_4 : True$ ~~$\alpha_5 : \neg\epsilon_3 \wedge \epsilon_6$~~ $\alpha_6 : True$
~~$\alpha_1 : \epsilon_2 \vee \epsilon_1$~~ ~~$\alpha_2 : \epsilon_4 \wedge \epsilon_1$~~ ~~$\alpha_3 : \epsilon_7$~~ $\alpha_4 : True$ ~~$\alpha_5 : \neg\epsilon_3 \wedge \epsilon_6$~~ $\alpha_6 : True$

Scenarios	ϵ_1	ϵ_2	ϵ_3	ϵ_4	...	ϵ_7	*Prob*
σ_1	T	T	T	T	...	T	0.25
σ_2	F	T	T	T	...	F	0.04
...	...	...	...	...	...	...	...
σ_{128}	F	F	F	F	...	F	0.17

Figure 1: An overview of the Unified Probabilistic Argumentation Framework (UPAF). The framework consists of an Analytical Model (AM), which formalizes the domain knowledge through literals and relations, and an Environmental Model (EM), which represents uncertainty via probabilistic events. An annotation function links elements from the AM to logical formulas over the events in the EM. Each probabilistic scenarios induces a specific knowledge base from the AM, which can then be used for reasoning.

works with probabilities. These include for instance Probabilistic SAT (PSAT) [28], Probabilistic Logic [29], Probabilistic Logic Programming [30], and Probabilistic Databases [31].

The proposed approach combines argumentation and probabilistic models. This allows modeling inconsistent, incomplete, and uncertain information, using the tolerance to inconsistency of argumentation to decide which information prevails in conflicting situations. An automated knowledge representation and reasoning system to support problems in complex, leveraging the capabilities of the two approaches mentioned, must provide certain core capabilities [32]. To mention some: (i) reasoning from evidence in a formal manner, (ii) modeling evidence about domain events

Scenario σ_i	event ϵ_1=`rainy`	event ϵ_2=`windy`	Probability $Pr(\sigma_i)$	Induced DeLP program $\mathcal{P}^{\sigma_i} = (\Pi^{\sigma_i}, \Delta^{\sigma_i})$	staus of `eat_fish`
σ_1	`true`	`true`	0.1	$\mathcal{P}^{\sigma_1} = (\emptyset, \{\delta_1, \delta_2\})$	U
σ_2	`true`	`false`	0.2	$\mathcal{P}^{\sigma_2} = (\{\pi_2\}, \{\delta_1, \delta_2\})$	U
σ_3	`false`	`true`	0.3	$\mathcal{P}^{\sigma_3} = \mathcal{P}^{\sigma_1}$	U
σ_4	`false`	`false`	0.4	$\mathcal{P}^{\sigma_4} = (\{\pi_1, \pi_3\}, \{\delta_1, \delta_2\})$	W

Table 1: Scenarios and probabilities for Example 1. W: warranted; U: undecided.

along with their associated level of uncertainty, (iii) modeling logical rules that allow the system to draw conclusions based on certain evidence and apply these rules iteratively, (iv) considering pieces of information that may not be compatible with each other, deciding which information is more relevant, and expressing the reason for the established difference, (v) presenting the actual state of the system based on the characteristics mentioned earlier and providing the user with the ability to understand how the system reaches that conclusion.

UPAF comprises two interrelated models: the *Environment Model* (EM) and the *Analytical Model* (AM). EM describes uncertain knowledge about the domain, subject to probabilistic events, while AM characterizes domain knowledge in terms of arguments and relationships among them. This duality is particularly useful for analyzing hypotheses (queries) explaining a phenomenon. The formalism employs argumentation in AM to model uncertain or incomplete knowledge and associates it with probabilistic events in EM through an annotation function. AM could include structured argumentation concepts, such as rules representing knowledge, each associated with EM events and their probability values. Inconsistencies, incompleteness, and uncertainty are modeled through the combination of argumentation and probabilistic models. Figure 1 provides an overview of the UPAF, depicting the components and semantics: AM models the domain knowledge, EM comprises probabilistic events, and associations connect AM elements with logical formulas derived from EM events. For each EM scenario, a subset of AM is satisfied based on associated formulas.

A well-known reasoning problem in argumentation is the so-called *acceptance* problem. Intuitively, the acceptance problem corresponds to computing the probability that a given goal element g of the AM is *accepted*. The following example illustrates this using the language of Defeasible Logic Programming with Presumptions for the AM.

Example 1. Consider a context where a person's ability to visit a restaurant (literal `can_go_restaurant`) depends on weather conditions (external independent events $\epsilon_1 = \texttt{rainy}$ and $\epsilon_2 = \texttt{windy}$) and the availability of an umbrella (literal `umbrella_available`). This is formally represented by the following conditional, fact, and strict rule, respectively:

$$\begin{array}{lcl} (\pi_1 = \texttt{can\_go\_restaurant} \leftarrow\ ) & : & \neg\texttt{rainy} \wedge \neg\texttt{windy} \\ (\pi_2 = \texttt{can\_go\_restaurant} \leftarrow \texttt{umbrella\_available}) & : & \texttt{rainy} \wedge \neg\texttt{windy} \end{array}$$

Moreover, umbrellas are typically available and usually the person does not eat fish (literal `eat_fish`). This is formally represented in the following non-conditional presumptions:

$$\begin{array}{lcl} (\delta_1 = \texttt{umbrella\_available} \prec\ ) & : & \texttt{true} \\ (\delta_2 = \sim\texttt{eat\_fish} \prec\ ) & : & \texttt{true} \end{array}$$

Finally, the user eats fish at the restaurant, but only if it hasn't rained, since this is when the fisherman have gone fishing. This is captured by the following strict, conditional rule:

$$(\pi_3 = \texttt{eat\_fish} \leftarrow \texttt{can\_go\_restaurant}) \quad : \quad \neg\texttt{rainy}$$

Given such a context, we would like to know whether the user will eat fish or not; towards this end, we can ask the system if literal `eat_fish` is *warranted.*

The model works with induced programs, which correspond to possible subsets of rules and facts induced by all the possible combinations of events, which we call *scenarios*, each with associated probabilities. The possible scenarios in this example are reported in the first four columns of Table 1, together with their associated probabilities. Thus, each of the four possible scenarios σ_i (with $i \in [1, 4]$), has a corresponding DeLP program $\mathcal{P}^{\sigma_i}$, each producing a possibly different warranted status for literal `eat_fish`, as shown in the last two columns of Table 1. The only scenario inducing a DeLP program where literal `eat_fish` is warranted is σ_4, where both events `rainy` and `windy` are false, which discards rule π_2. ■

UPAF offers a versatile approach to model diverse situations, accounting for inconsistency, uncertainty, and incompleteness in available information. Its capabilities make it suitable for reasoning about complex domains, exemplified by its application in cyber threat analysis in its instantiation with DeLP in the AM [33]. Key functionalities include establishing values for observed events in the system (observed evidence) and monitoring probabilities for a set of registered queries. In the context of cybersecurity, analysts can leverage this tool to log queries related to mitigation strategies and current attack techniques, influencing security alert levels and updating priorities for system administrators. Another significant feature involves performing counterfactual analysis, allowing users to specify desired scenarios and derive resulting probabilities, even in the absence of direct observational data and configurations of the events of the environmental model.

The primary goal of this work is to design a unique theoretical framework capable of integrating probabilistic reasoning into any argumentation formalism, be it abstract or structured.

Contributions The main contributions of this paper are as follows:

- Introduction of the *Unified Argumentation Framework* (UAF), an expressive model that brings together existing argumentation formalisms.
- Incorporation of an *environmental model* allowing the association of (not necessarily independent) probabilistic events to arguments and relationships, enabling reasoning under uncertainty. The resulting framework is called *Unified Probabilistic Argumentation Framework* (UPAF).
- Demonstration that UPAF subsumes classical and probabilistic versions of Dung's Argumentation Framework, as well as structured frameworks like Assumption-Based Argumentation (ABA) and Defeasible Logic Programming (DeLP).
- As a side result, to the best of our knowledge, this is the first piece of work extending ABA with probabilities.
- Analysis of the computational complexity of key reasoning tasks within well-known UPAF instances.

Organization The remainder of this paper is organized as follows. Section 2 reviews the necessary background on several abstract and structured argumentation formalisms. Section 3 introduces our main contribution, the Unified Probabilistic Argumentation Framework (UPAF), presenting its core components: the Analytical and the Environmental models, and the annotation function. Section 4 demonstrates how the proposed analytical model can instantiate well-known argumentation frameworks, while Section 5 shows how the full UPAF can encode their probabilistic counterparts. Section 6 analyzes the computational complexity of the main reasoning tasks. Finally, Section 7 discusses related work, and Section 8 concludes the paper and outlines future research directions.

2 Preliminaries

We briefly review the essential background for abstract Argumentation Frameworks (AFs) [34], and two main structured argumentation formalisms, namely DeLP [35] and ABA [36].

2.1 Abstract Argumentation Frameworks

An abstract argumentation framework consists of a set of arguments whose composition is left unspecified, and a set of conflicts between them [34].

Definition 1 (AF [34])**.** *An* abstract Argumentation Framework (*AF*) *is a pair* $\langle \mathsf{A}, \mathsf{R} \rangle$*, where* A *is a set of arguments and* $\mathsf{R} \subseteq \mathsf{A} \times \mathsf{A}$ *is an attack relation.*

Given an AF, in [34] a series of semantic notions are defined leading to the characterization of collectively acceptable sets of arguments. Given an AF $\langle \mathsf{A}, \mathsf{R} \rangle$ and a set $\mathbf{S} \subseteq \mathsf{A}$ of arguments, we say that $\mathbf{S}$ is *conflict-free* iff $\nexists\, a, b \in \mathbf{S}$ s.t. $(a, b) \in \mathsf{R}$. Moreover, $a \in \mathsf{A}$ is *acceptable w.r.t.* $\mathbf{S}$ iff $\forall b \in \mathsf{A}$ s.t. $(b, a) \in \mathsf{R}$, $\exists c \in \mathbf{S}$ s.t. $(c, b) \in \mathsf{R}$.

A set $\mathbf{S}$ is *admissible* iff it is conflict-free and $\forall a \in \mathbf{S}$, a is acceptable w.r.t. $\mathbf{S}$. By adding restrictions to the notion of admissibility, the complete ($\mathtt{co}$), preferred ($\mathtt{pr}$), stable ($\mathtt{st}$), and grounded ($\mathtt{gr}$) extensions of an *AF* are defined as follows. Given an AF $\langle \mathsf{A}, \mathsf{R} \rangle$ and a set $\mathbf{S} \subseteq \mathsf{A}$ of arguments, we say that $\mathbf{S}$ is a *complete extension* of *AF* iff it is admissible and $\forall a \in \mathsf{A}$, if a is acceptable w.r.t. $\mathbf{S}$, then $a \in \mathbf{S}$. Moreover, a complete extension od *AF* is said to be *preferred* (resp., *grounded*) iff it is maximal (resp., minimal) w.r.t. $\subseteq$. A preferred extension $\mathbf{S}$ is *stable* iff $\forall a \in \mathsf{A} \backslash \mathbf{S}$, $\exists\, b \in \mathbf{S}$ s.t. $(b, a) \in \mathsf{R}$.

For any AF $\langle \mathsf{A}, \mathsf{R} \rangle$ and semantics sem, $\mathsf{sem}(\langle \mathsf{A}, \mathsf{R} \rangle)$ denotes the set of sem-extensions of $\langle \mathsf{A}, \mathsf{R} \rangle$. For any AF $\langle \mathsf{A}, \mathsf{R} \rangle$, semantics sem, and argument $a \in \mathsf{A}$, we say that a is *credulously* (resp. *skeptically*) accepted (under semantics sem), denoted as $CA_{\mathsf{sem}}(a, \langle \mathsf{A}, \mathsf{R} \rangle)$ (resp., $SA_{\mathsf{sem}}(a, \langle \mathsf{A}, \mathsf{R} \rangle)$) if a belongs to at least one (resp., every) sem-extension of $\langle \mathsf{A}, \mathsf{R} \rangle$.

Example 2. The AF $\langle \mathsf{A}, \mathsf{R} \rangle = \langle \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\}, \{(\mathtt{a}, \mathtt{b}), (\mathtt{b}, \mathtt{a}), (\mathtt{a}, \mathtt{c}), (\mathtt{b}, \mathtt{c}), (\mathtt{c}, \mathtt{d}), (\mathtt{d}, \mathtt{c})\} \rangle$ has four complete extensions: $E_0 = \emptyset$, $E_1 = \{\mathtt{a}, \mathtt{d}\}$, $E_2 = \{\mathtt{b}, \mathtt{d}\}$ and $E_3 = \{\mathtt{d}\}$. That is, $\mathtt{co}(\langle \mathsf{A}, \mathsf{R} \rangle) = \{E_0, E_1, E_2, E_3\}$. E_0 is the grounded extension, E_1 and E_2 are stable (and preferred) extensions. Thus, $\mathtt{a}$, $\mathtt{b}$, $\mathtt{d}$ are credulously accepted for all semantics except the grounded; only $\mathtt{d}$ is skeptically accepted under stable and preferred semantics. ■

2.2 Defeasible Logic Programming

We review DeLP[1], a formalism combining Logic Programming with Defeasible Argumentation, and refer the reader to [35, 37] for a full presentation. DeLP allows for the creation of *arguments* that can "compete" with each other. In this competition, known as a *dialectical process*, one argument may defeat another based on a *comparison criterion* that determines the prevailing argument. Resulting from this process, the model will determine arguments that are *warranted* (those that are not *defeated* by other arguments)—the dialectical process is therefore a solid basis for providing a suitable explanation for the warrant status of an argument.

We assume the existence of a set **At** of atoms from which DeLP programs can be built. A literal is a ground atom $\alpha \in \mathbf{At}$ or its negation $\sim\alpha$, where symbol "$\sim$" represents strong negation; the formula $\sim\sim\alpha$ can be used for denoting α. We use *Lit* to denote the set of literals that can be obtained from the atoms in **At**; that is, we have $\mathit{Lit} = \mathbf{At} \cup \{\sim\alpha \mid \alpha \in \mathbf{At}\}$. In the following, we assume that literals used to define programs are taken from the set *Lit*.

A DeLP program $\mathcal{P}$ consists of *strict* and *defeasible* sets of rules defined using elements of **At** as follows. A strict rule is of the form $\alpha_0 \leftarrow \alpha_1, \ldots, \alpha_n$, where $\alpha_0, \alpha_1, \ldots, \alpha_n$ (with $n \geq 0$) are literals. A defeasible rule is of the form $\alpha_0 \prec \alpha_1, \ldots, \alpha_n$, where α_0, α_1, $\ldots$, α_n ($n \geq 0$) are literals. Given a strict or defeasible rule r, we use *head*(r) to denote α_0, and *body*(r) to denote the set of literals $\{\alpha_1, \ldots, \alpha_n\}$. Strict (resp., defeasible) rules with empty body will also be called *facts* (resp., *presumptions*). Intuitively, strict rules represent non-defeasible information, while defeasible rules represent tentative information (that is, information that can be used if nothing can be posed against it). Given a program $\mathcal{P}$, we will distinguish the subset Π of strict rules, Δ of defeasible rules, $\widehat{\Pi}$ of facts, and $\widehat{\Delta}$ of presumptions. We denote $\mathcal{P}$ as (Π, Δ) when needed. Moreover, we use $\mathit{Lit}_{\mathcal{P}}$ to denote the set of literals derived from the atoms occurring in a rule in $\mathcal{P}$. A set of rules is *contradictory* if and only if there exist derivations for two complementary literals α and $\sim\alpha$ from that set. We assume that Π is not contradictory—checking this can be done in PTIME. However, complementary literals can be derived from $\mathcal{P}$ when defeasible rules are used in the derivation. Two literals α and β are said to be *contradictory* if (i) neither $\Pi \cup \{\alpha\}$ nor $\Pi \cup \{\beta\}$ are contradictory, whereas (ii) $\Pi \cup \{\alpha, \beta\}$ is contradictory. Pairs of complementary literals are clearly contradictory. A set of literals is said to be contradictory if it contains two contradictory literals.

Given a DeLP program program $\mathcal{P} = (\Pi, \Delta)$ and a literal $\alpha \in \mathit{Lit}_{\mathcal{P}}$, a *(defeasible) derivation* for α w.r.t. $\mathcal{P}$ is a finite sequence $\alpha_1, \alpha_2, \ldots, \alpha_n = \alpha$ of literals such that

[1] We consider its extension DeLP with Presumptions (PreDeLP) [37], but for simplicity refer to it simply as "DeLP".

(i) each literal α_i is in the sequence because there exists a (strict or defeasible) rule $r \in \mathcal{P}$ with head α_i and body $\alpha_{i_1}, \alpha_{i_2}, \ldots, \alpha_{i_k}$ such that $i_j < i$ for all $j \in [1, k]$, and (ii) there do not exist two literals α_i and α_j such that $\alpha_j = {\sim}\alpha_i$. A derivation is said to be a *strict derivation* if only strict rules are used.

Example 3. Given the situation in Example 1, this knowledge can be modeled by means of a DeLP program $\mathcal{P} = (\Pi, \Delta)$, with:

$$\begin{aligned}\Pi = &\{\texttt{can\_go\_restaurant} \leftarrow \texttt{umbrella\_available},\\ &\texttt{eat\_fish} \leftarrow \texttt{can\_go\_restaurant}\}\\ \Delta = &\{\emptyset\}\\ \widehat{\Pi} = &\{\texttt{can\_go\_restaurant}\}\\ \widehat{\Delta} = &\{\texttt{umbrella\_available}, {\sim}\texttt{eat\_fish}\}.\end{aligned}$$ ■

For the treatment of contradictory knowledge, DeLP incorporates a defeasible argumentation formalism that allows the identification of the pieces of knowledge that are in conflict, and through the previously mentioned *dialectical process* decides which information prevails as warranted. The dialectical process involves the construction and evaluation of arguments that either support or interfere with a given query, building a *dialectical tree* in the process.

An argument for a literal α is a pair $\langle \mathcal{A}, \alpha \rangle$ where $(\mathcal{A} \subseteq \Delta)$ is a (possibly empty) set of defeasible elements that provides a minimal proof for α meeting the following requirements: $i)$ α is defeasibly derived from $\mathcal{A}$; $ii)$ $\Pi \cup \mathcal{A}$ is not contradictory; $iii)$ $\mathcal{A}$ is a minimal subset of Δ satisfying $i)$ and $ii)$.

Example 4. Given the DeLP program $\mathcal{P}$ of Example 3, some of the arguments we can obtain are the following:

$$\begin{aligned}\langle \mathcal{A}_1, \mathtt{c}_1 \rangle &= \langle \{\texttt{umbrella\_available} \prec\ \}, \texttt{can\_go\_restaurant} \rangle;\\ \langle \mathcal{A}_2, \mathtt{c}_2 \rangle &= \langle \{\texttt{umbrella\_available} \prec\ \}, \texttt{eat\_fish} \rangle;\\ \langle \mathcal{A}_3, \mathtt{c}_1 \rangle &= \langle \{\emptyset\}, \texttt{can\_go\_restaurant} \rangle;\\ \langle \mathcal{A}_4, \mathtt{c}_2 \rangle &= \langle \{\emptyset\}, \texttt{eat\_fish} \rangle\end{aligned}$$ ■

Given an argument $\langle \mathcal{A}, \alpha \rangle$, the literal α is called the conclusion supported by the argument, and $\mathcal{A}$ is the support of the argument. An argument $\langle \mathcal{B}, \beta \rangle$ is a subargument of $\langle \mathcal{A}, \alpha \rangle$ iff $\mathcal{B} \subseteq \mathcal{A}$. An argument $\langle \mathcal{A}, \alpha \rangle$ is said to be *presumptive* iff $\widehat{\mathcal{A}} \neq \emptyset$. Given argument $\langle \mathcal{A}_1, \alpha_1 \rangle$, counter-arguments are arguments that contradict it. Argument $\langle \mathcal{A}_2, \alpha_2 \rangle$ *counter-argues or attacks* $\langle \mathcal{A}_1, \alpha_1 \rangle$ at literal β iff there exists a subargument $\langle \mathcal{B}, \beta \rangle$ of $\langle \mathcal{A}_1, \alpha_1 \rangle$ s.t. set $\widehat{\mathcal{A}_1} \cup \widehat{\mathcal{A}_2} \cup \{\alpha_2, \beta\}$ is contradictory. A *proper defeater* of an argument $\langle \mathcal{A}, \alpha \rangle$ is a counterargument that, by some criterion, is considered to be better than $\langle \mathcal{A}, \alpha \rangle$; if the two are incomparable according to this criterion, the counterargument is said to be a *blocking* defeater. An important characteristic of DeLP is that the argument comparison criterion is modular, and thus the most appropriate criterion for the domain that is being represented can be selected; the default criterion used in original defeasible logic programming presentation is *generalized specificity* [38].

A literal α is *warranted* if there exists a non-defeated argument $\mathcal{A}$ supporting α. To establish if $\langle \mathcal{A}, \alpha \rangle$ is a non-defeated argument, *defeaters* for $\langle \mathcal{A}, \alpha \rangle$ are considered. Since there may be more than one defeater for a particular argument, many acceptable argumentation lines could arise from one argument, leading to a tree structure. This is called a *dialectical tree* because it represents an exhaustive dialectical analysis for the argument in its root; every node (except the root) represents a defeater of its parent, and leaves correspond to non-defeated arguments. Each path from the root to a leaf corresponds to a different acceptable argumentation line. A dialectical tree provides a structure for considering all possible acceptable argumentation lines that can be generated for deciding whether an argument is defeated.

Given a literal α and an argument $\langle \mathcal{A}, \alpha \rangle$ from a program $\mathcal{P}$, to decide whether α is warranted, every node in the tree is recursively marked as D (*defeated*) or U (*undefeated*, obtaining a marked dialectical tree $\mathcal{T}_{\mathcal{P}}(\mathcal{A})$: (1) all leaves in $\mathcal{T}_{\mathcal{P}}(\mathcal{A})$ are marked as "U"s; and (2) let $\mathcal{B}$ be an inner node of $\mathcal{T}_{\mathcal{P}}(\mathcal{A})$; then, $\mathcal{B}$ is marked as U *iff* every child of $\mathcal{B}$ is marked as D. Thus, node $\mathcal{B}$ is marked as D *iff* it has at least one child marked as U. Given an argument $\langle \mathcal{A}, \alpha \rangle$ obtained from $\mathcal{P}$, if the root of $\mathcal{T}_{\mathcal{P}}(\mathcal{A})$: is marked as U, then $\mathcal{T}_{\mathcal{P}}(\mathcal{A})$ *warrants* α, and α is *warranted* from $\mathcal{P}$. The DeLP interpreter takes a program $\mathcal{P}$ and a DeLP query literal α, and returns one of the following four possible answers: YES if α is warranted (W) from $\mathcal{P}$, NO if the complement of α (i.e., $\sim\alpha$) regarding strong negation is warranted from $\mathcal{P}$ ($\bar{\mathsf{W}}$), *undecided* (U) if neither α nor its complement $\sim\alpha$ are warranted from $\mathcal{P}$, or *unknown* if α is not in the language of the program $\mathcal{P}$. Hereafter, we will assume goal literals always belong to the language of $\mathcal{P}$.

2.3 Assumption-based Argumentation

We now review the basic notions underlying the prominent *Assumption-based Argumentation* (ABA) formalism [39]; the interested reader can find a thorough description of the formalism in [19].

ABA Semantics We assume the existence of a deductive system ($\mathtt{Lang}, \mathtt{Rules}$), where $\mathtt{Lang}$ is a formal language, i.e., a set of sentences, and $\mathtt{Rules}$ is a set of inference rules over $\mathtt{Lang}$. A rule $r \in \mathtt{Rules}$ has the form $a_0 \leftarrow a_1, \ldots, a_n$ with $a_i \in \mathtt{Lang}$. We denote the head of r with $\mathit{head}(r) = a_0$ and the (possibly empty) body of r with $body(r) = a_1, ..., a_n$. An ABA Framework (ABAF for short) is a tuple $\mathcal{D} = \Delta$, where ($\texttt{Lang}, \texttt{Rules}$) is a deductive system, $\texttt{Asm} \subseteq \texttt{Lang}$ a set of assumptions, and a contrary function $^{-} : \texttt{Asm} \rightarrow \texttt{Lang}$ [36, 19, 40, 41]. An ABAF is said to be *flat* iff assumptions may only appear in the rules' body (i.e., $a_0 \notin \texttt{Asm}$); this condition intuitively implies no assumptions can be inferred (i.e., they can only be assumed to be true or not). (though they are of restricted form, these admit several instances like logic programming and default logic [18]).

An atom $s \in \texttt{Lang}$ is tree-derivable from assumptions $S \subseteq \texttt{Asm}$ and rules $R \subseteq \texttt{Rules}$, denoted $S \vdash_R s$, if s can be derived from the set S of assumptions and the rules in R, i.e. there is a finite rooted labeled tree T such that the root is labeled with s, the set of labels for the leaves of T is equal to S or $S \cup \{\top\}$, and for every inner node v of T there is a rule $r \in R$ such that v is labeled with $head(r)$, the number of successors of v is $|body(r)|$ and every successor of v is labeled with a distinct $a \in body(r)$ or $\top$ if $body(r) = \emptyset$. Each tree-derivation $S \vdash_R s$ is an ABA argument. Moreover, $S \vdash_R s$ is a rule-induced argument iff $R \neq \emptyset$ and an assumption argument iff $R = \emptyset$. By $Th_{\mathcal{D}}(S) = \{s \in \texttt{Lang} \mid \exists S' \subseteq S : S' \vdash_R s\}$ we denote the set of all conclusions derivable from a set S in an ABAF $\mathcal{D}$. Note that $S \subseteq Th_{\mathcal{D}}(S)$ since, by definition, $\{a\} \vdash_{\emptyset} a$ for each assumption a. We let $\bar{S} = \{\bar{a} \mid a \in S\}$. A set $S \subseteq \texttt{Asm}$ attacks a set $S' \subseteq \texttt{Asm}$ if for some $a \in S'$ we have $\bar{a} \in Th_{\mathcal{D}}(S)$. Moreover, S defends S' iff S attacks each attacker S'' of S'. A set S is said to be *conflict-free* if it does not attack itself, and *admissible* if it is conflict-free and defends itself. With a little abuse of notation we say that S attacks a if S attacks the singleton $\{a\}$. An admissible set S of $\mathcal{D}$ is a complete extension of $\mathcal{D}$ iff S contains every assumption set it defends. Moreover a complete extension S of $\mathcal{D}$ is said to be grounded (resp., preferred and stable) iff S is $\subseteq$-minimal among all complete extensions of $\mathcal{D}$ (resp., $\subseteq$-maximal iff S attacks each $\{a\} \subseteq \texttt{Asm} \setminus S$).

Example 5. Consider the ABAF $\mathcal{D} = \langle \texttt{Lang}, \texttt{Rules}, \texttt{Asm}, ^{-} \rangle$ where $\texttt{Asm} = \{\texttt{a}, \texttt{b}, \texttt{c}, \texttt{d}\}$, $\texttt{Lang} = \texttt{Asm} \cup \{\texttt{x}, \texttt{a}', \texttt{b}', \texttt{c}', \texttt{d}'\}$, $\texttt{Rules} = \{r_1 : \texttt{x} \leftarrow \texttt{a}; r_2 : \texttt{c}' \leftarrow \texttt{b}, \texttt{x}; r_3 : \texttt{b}' \leftarrow \texttt{c}; r_4 : \texttt{d}' \leftarrow \texttt{a}, \texttt{b}\}$; and $\bar{y} = y'$ for any assumption $y \in \texttt{Asm}$.

The complete extensions are $S_1 = \{\texttt{a}\}$, $S_2 = \{\texttt{a}, \texttt{b}\}$, and $S_3 = \{\texttt{a}, \texttt{c}, \texttt{d}\}$, with S_1 being the grounded extension (as it is not attacked). Moreover, the sets S_2 and S_3 are also stable. Observe that the set S_2 attacks, by means of rules r_2 and r_4, the sets $\{\texttt{c}\}$ and $\{\texttt{d}\}$. ■

3 Unified Probabilistic Argumentation Frameworks

We now present the Unified Probabilistic Argumentation Framework (UPAF), a general and expressive theoretical model capable of capturing a wide range of existing argumentation formalisms. As briefly discussed in the introduction, UPAF is comprised of an *Analytical Model* (AM) that is able to represent an underlying KB containing contradictory information, an *Environmental Model* that assigns probabilistic (possibly non-independent) events to elements of the AM, and an annotation function that links the two.

Later, we will explore how UPAF can be seen as a meta-representation of several well-known abstract and structured argumentation formalisms.

3.1 The Analytical Model: General AFs

The Analytical Model (AM) must allow for competing ideas, so it must be able to represent contradictory information. We now define a unified argumentation model that can be used for this purpose.

Definition 1 (UAF). *A* Unified Argumentation Framework *(UAF) is an ordered tuple $\mathcal{G} = (\mathcal{L}, \mathcal{R})$, where $\mathcal{L}$ is a set of literals taken from a universal set of literals, and $\mathcal{R} = \{\mathcal{R}_1, \ldots, \mathcal{R}_n\}$ is a set of n relations of the form $\mathcal{R}_i : 2^{\mathcal{L}} \to (2^{\mathcal{L}} \cup \{\mathtt{true}, \mathtt{false}\})$.*

An *extension* of a UAF $\mathcal{G} = (\mathcal{L}, \mathcal{R})$ is a total function from $\mathcal{L}$ to truth values in $\{v_{in}, v_{un}, v_{out}\}$. A UAF semantics sem is a function that returns, for any UAF $\mathcal{G}$, the set of all its extensions. Given a UAF $\mathcal{G}$ and semantics sem, we denote with $\mathsf{sem}(\mathcal{G})$ the set of all the sem-extensions of $\mathcal{G}$. Note that this is meant to be a theoretical tool to define a generic structure—*as such, it will never be directly manipulated computationally.*

Intuitively, UAFs are designed to capture both abstract and structured formalisms. Any abstract argumentation framework [34] $\Lambda = \langle \mathsf{A}, \mathsf{R} \rangle$ can be seen as a UAF $\mathcal{G}_\Lambda = (\mathcal{L} = \mathsf{A}, \mathcal{R} = \{\mathsf{R}\})$, where $v_x = x$ and $\mathsf{sem} \in \{\mathtt{gr}, \mathtt{co}, \mathtt{pr}, \mathtt{st}\}$ denotes an AF semantics. UAF could also encode bipolar AF [16] (by adding $\mathcal{R}_2 = S$, where S is the support relation), as well as, moving to the structured argumentation, ABA and DeLP. For presentation purposes, a further discussion on this has been postponed to Section 4

Next, we define the acceptance problem, which is at the heart of the main reasoning problems in computational argumentation.

Definition 2 (UAF Acceptance Problem). *Given a UAF $\mathcal{G} = (\mathcal{L}, \mathcal{R})$, semantics sem, truth status $v \in \{v_{in}, v_{un}, v_{out}\}$, and goal literal $g \in \mathcal{L}$, we denote with $\mathsf{c\text{-}ACC}^v_{\mathsf{sem}}$ (resp., $\mathsf{s\text{-}ACC}^v_{\mathsf{sem}}$) the problem of determining whether any (resp., all) sem-extension for $\mathcal{G}$ contains $v(g)$.*

It is worth noting that, whenever $v = v_{in}$, $\mathsf{c\text{-}ACC}^v_{\mathsf{sem}}/\mathsf{c\text{-}ACC}^v_{\mathsf{sem}}$ is also known as credulous/skeptical acceptance problem. We will discuss the computational complexity of acceptance problem of several UAF instances in Section 6.

3.2 Environmental Model

The *environmental model* (EM) is meant to describe background knowledge in a leveraging probabilistic models; here, we assume that any probability distribution may be encoded, not necessarily making any simplifying assumptions such as independence. Differently from the AM, the EM must be consistent, that is there must exist a probability distribution over all possible states of the domain that satisfies all the constraints of the model, as well as the axioms of probability theory. We assume the existence of a set $\mathbf{E}$ of atoms from which the EM can be built. An event is a ground atom $\epsilon \in \mathbf{E}$ or its negation $\neg\epsilon$.

Definition 3 (Scenario)**.** *Given a set $E = \{\epsilon_1, \ldots, \epsilon_n\} \subseteq \mathbf{E}$ of n events $\epsilon_i \in \mathbf{E}$ and $S \subseteq 2^E$, a set $\sigma = S \cup \{\neg\epsilon \mid \epsilon \in E \setminus S\}$ a scenario of E.*

Hereafter, with a little abuse of notation, we often write $\sigma \in 2^E$. Logical formulas arise from the combination of atoms using the traditional connectives ($\wedge, \vee,$ *and* $\neg$). As usual, a scenario σ *satisfies* formula f, written $\sigma \models f$, iff:

- if $f = \epsilon$ is a unique event, then $\sigma \models f$ iff $\epsilon \in \sigma$;
- if $f = \neg f'$ then $\sigma \models f$ iff $\sigma \not\models f'$;
- if $f = f' \wedge f''$ then $\sigma \models f$ iff $\sigma \models f'$ and $\sigma \models f''$; and
- if $f = f' \vee f''$ then $\sigma \models f$ iff $\sigma \models f'$ or $\sigma \models f''$.

The environmental model can thus be defined simply as an encoding of a probability distribution over $2^{\mathbf{E}}$.

Definition 4 (Environmental model)**.** *Given a set $E = \{\epsilon_1, \ldots, \epsilon_n\} \subseteq \mathbf{E}$ of n events, an* Environmental Model *(EM) $\mathcal{E} : 2^E \to [0, 1]$ is any function returning a probability distribution over all possible scenarios $\sigma \in 2^E$, i.e. $\sum\limits_{\sigma \in 2^E} \mathcal{E}(\sigma) = 1$.*

Examples of probabilistic models that can be used are Bayesian networks (BNs) [42], Markov logic networks (MLNs) [43], extensions of first order logic such as Nilsson's probabilistic logic [44], or even ad-hoc constructions of $\mathcal{E}$ such as in the following example.

Example 6. The Environmental Model $\mathcal{E}$ of Example 1 is built over two events $E = \{\texttt{rainy}, \texttt{windy}\}$, and associates to the four scenarios σ_i (with $i \in [1, 4]$) the following probabilities:

- $\mathcal{E}(\sigma_1) = \texttt{0.1}$ with $\sigma_1 = \{\texttt{rainy}, \texttt{windy}\}$;
- $\mathcal{E}(\sigma_2) = \texttt{0.2}$ with $\sigma_2 = \{\texttt{rainy}, \neg\texttt{windy}\}$;

- $\mathcal{E}(\sigma_3) = 0.3$ with $\sigma_3 = \{\neg\texttt{rainy}, \texttt{windy}\}$; and
- $\mathcal{E}(\sigma_4) = 0.4$ with $\sigma_4 = \{\neg\texttt{rainy}, \neg\texttt{windy}\}$. ■

We are now ready to add probabilistic models to the unified framework introduced above.

3.3 Unified Probabilistic Argumentation Frameworks

We now introduce *Unified Probabilistic Argumentation Frameworks* (UPAF for short), which are able to deal with two kinds of uncertainty: *probabilistic uncertainty* and uncertainty arising from *underlying knowledge*, thus combining the environmental and analytical models presented before. Intuitively, every element of a UPAF $\mathcal{I}$ only holds in certain scenarios in the set $2^{\mathbf{E}}$, i.e., these elements are subject to probabilistic events. Each element of $\mathcal{I}$ is thus associated with a formula over $\mathbf{E}$ (using conjunction, disjunction, and negation, as usual) through the use of an *annotation function.*

Definition 5 (UPAF)**.** *A Unified Probabilistic Argumentation Framework is a tuple $\mathcal{I} = (\mathcal{G}, \mathcal{E}, \mathcal{F})$ where $\mathcal{G} = (\mathcal{L}, \mathcal{R})$ is a UAF (also called Analytical Model), $\mathcal{E} : 2^E \rightarrow [0, 1]$ is an Environmental Model, and $\mathcal{F}$ is an annotation function associating to elements in $\mathcal{L} \cup \{r \in \mathcal{R}_i \mid \mathcal{R}_i \in \mathcal{R}\}$ either i)* `true` *or ii) a propositional formula over elements in E.*

Hereafter, with a little abuse of notation, we simply write $\gamma \in \mathcal{G}$ to denote $\gamma \in \mathcal{L} \cup \{r \in \mathcal{R}_i \mid \mathcal{R}_i \in \mathcal{R}\}$. Given a UPAF $\mathcal{I} = (\mathcal{G}, \mathcal{E}, \mathcal{F})$ and a scenario σ, we denote with $\mathcal{G}^\sigma = \{\gamma \in \mathcal{G} \mid \sigma \models \mathcal{F}(\gamma)\}$ the UAF *induced by* σ—the UAF induced by a scenario is intuitively comprised of the elements whose annotations are satisfied in that scenario. It is worth noting that the same UAF can be induced from distinct scenarios.

The most direct way of considering consequences of GPAFs is thus to consider what happens in each scenario. That is, the acceptance status of a goal literal g depends on each induced UAF.

Definition 6 (UPAF Acceptance Problem)**.** *Let $\mathcal{I} = (\mathcal{G} = (\mathcal{L}, \mathcal{R}), \mathcal{E}, \mathcal{F})$ be a UPAF,* sem *a semantics, and $g \in \mathcal{L}$ a goal literal. The following is the acceptance interval of g w.r.t. $\mathcal{I}$:*

$$\left[g^\ell = \sum_{\sigma \in 2^E \, s.t. \ c\text{-}ACC^{v_{in}}_{\mathsf{sem}}(\mathcal{G}^\sigma, g) = \texttt{true}} \mathcal{E}(\sigma), \ g^u = 1 - \sum_{\sigma \in 2^E \, s.t. \ s\text{-}ACC(\mathcal{G}^\sigma, g)^{v_{out}}_{\mathsf{sem}} = \texttt{true}} \mathcal{E}(\sigma) \right].$$

That is, g^ℓ denotes the the sum of all the probabilities associated to scenarios inducing a UAF where g is credulously accepted. In contrast, $1-g^u$ denotes the sum of all the probabilities associated to scenarios inducing a UAF where g is skeptically rejected. Briefly put, $[g^\ell, g^u]$ provides a confidence interval characterizing the probability that literal g is accepted in $\mathcal{I}$—if this interval is wider than a single point, this means that there is uncertainty regarding the probabilities themselves, which can happen for instance in structured argumentation formalisms that contemplate an undecided status, as discussed below.

4 Instantiating the Analytical Model

In this section we show that several existing abstract and structured argumentation formalisms can be captured by means of the UAF introduced in the previous section.

4.1 Abstract Argumentation Frameworks

Definition 7. *Given an AF $\Lambda = \langle \mathsf{A}, \mathsf{R} \rangle$, we denote with $\mathcal{G}_\Lambda = (\mathcal{L} = \mathsf{A}, \mathcal{R} = \{\mathcal{R}_\mathsf{R} = \mathsf{R}\})$ the UAF obtained from Λ.*

Proposition 1. *Given an AF $\Lambda = \langle \mathsf{A}, \mathsf{R} \rangle$, semantics $\mathsf{sem} \in \{\mathtt{gr}, \mathtt{co}, \mathtt{st}, \mathtt{pr}\}$, and argument $g \in \mathsf{A}$, it holds that g is credulously (resp., skeptically) accepted in Λ under semantics sem iff $(\mathcal{G}_\Lambda, g)$ is a true instance of c-$\mathsf{ACC}^{v_{in}}_{\mathsf{sem}}$ (resp., s-$\mathsf{ACC}^{v_{in}}_{\mathsf{sem}}$).*

Dung's framework has been extended in many different ways, including the introduction of new kinds of interactions between arguments and/or attacks. We next show how existing formalisms can be modeled in UAF.

Bipolar Abstract Argumentation Frameworks Bipolar AF extends AF by means of an additional set $\mathsf{S} \subseteq \mathsf{A} \times \mathsf{A}$ of supports [16].

Definition 8. *Given a Bipolar AF $\Lambda = \langle \mathsf{A}, \mathsf{R}, \mathsf{S} \rangle$, we denote with $\mathcal{G}_\Lambda = (\mathcal{L} = \mathsf{A}, \mathcal{R} = \{\mathcal{R}_\mathsf{R} = \mathsf{R}, \mathcal{R}_\mathsf{S} = \mathsf{S}\})$ the UAF obtained from Λ.*

Proposition 2. *Given a BAF $\Lambda = \langle \mathsf{A}, \mathsf{R}, \mathsf{S} \rangle$, semantics $\mathsf{sem} \in \{\mathtt{gr}, \mathtt{co}, \mathtt{st}, \mathtt{pr}\}$, and argument $g \in \mathsf{A}$, it holds that g is credulously (resp., skeptically) accepted in Λ under semantics sem iff $(\mathcal{G}_\Lambda, g)$ is a true instance of c-$\mathsf{ACC}^{v_{in}}_{\mathsf{sem}}$ (resp., s-$\mathsf{ACC}^{v_{in}}_{\mathsf{sem}}$).*

Bipolar SETAF Bipolar SETAFs (BSAFs) extend BAF with collective attacks and supports. We next instantiate UAF with BSAF.

Definition 9. *Given a Bipolar SETAF* $\Lambda = \langle \mathsf{A}, \mathsf{R} \subseteq 2^{\mathsf{A}} \times \mathsf{A}, \mathsf{S} \subseteq 2^{\mathsf{A}} \times \mathsf{A} \rangle$*, we denote with* $\mathcal{G}_\Lambda = (\mathcal{L} = \mathsf{A}, \mathcal{R} = \{\mathcal{R}_{\mathsf{R}} = \mathsf{R}, \mathcal{R}_{\mathsf{S}} = \mathsf{S}\})$ *the UAF obtained from* Λ*.*

Proposition 3. *Given a BSAF* $\Lambda = \langle \mathsf{A}, \mathsf{R}, \mathsf{S} \rangle$*, semantics* $\mathsf{sem} \in \{\mathtt{gr}, \mathtt{co}, \mathtt{st}, \mathtt{pr}\}$*, and argument* $g \in \mathsf{A}$*, it holds that* g *is credulously (resp., skeptically) accepted in* Λ *under semantics* sem *iff* $(\mathcal{G}_\Lambda, g)$ *is a true instance of* $\textsf{c-ACC}^{v_{in}}_{\mathsf{sem}}$ *(resp.,* $\textsf{s-ACC}^{v_{in}}_{\mathsf{sem}}$*).*

Although we do not go into the details here, UAF can also be instantiated with higher-order AFs, where the AF is enriched with recursive attacks and supports [45, 46].

4.2 Defeasible Logic Programming

We now focus on showing how UAF can encode DeLP programs.

Definition 10. *Given a DeLP program* $\mathcal{P} = (\Pi, \Delta)$*, we denote with* $\mathcal{G}_\mathcal{P} = (\mathcal{L}_\mathcal{P}, \mathcal{R}_\mathcal{P})$ *the UAF obtained from* $\mathcal{P}$ *as follows:*

- $\mathcal{L}_\mathcal{P} = \{\alpha, {\sim}\alpha \mid \{\alpha, {\sim}\alpha\} \cap (\mathit{head}(r) \cup \mathit{body}(r)) \neq \emptyset \ \wedge r \in (\Pi \cup \Delta)\}$*;*
- $\mathcal{R} = \{\mathcal{R}_\Pi, \mathcal{R}_\Delta\}$*, where:*
  - $\mathcal{R}_\Pi = \{(\mathit{head}(r), \mathit{body}(r)) \mid r \in \Pi\}$
  - $\mathcal{R}_\Delta = \{(\mathit{head}(r), \mathit{body}(r)) \mid r \in \Delta\}$

Proposition 4. *Given a DeLP program* $\mathcal{P} = (\Pi, \Delta)$*, and goal literal* g*, it holds that the status of* g *is* W *(resp.,* $\bar{\mathsf{W}}$*,* U*), iff* $(\mathcal{G}_\mathcal{P}, g)$ *is a true instance of both* $\textsf{c-ACC}^{v}_{DeLP}$ *and* $\textsf{s-ACC}^{v}_{DeLP}$ *with* $v = v_{in}$ *(resp.,* $v = v_{out}$*,* $v = v_{un}$*).*

As the semantics of DeLP is unique-status (i.e., for each literal, a unique value is associated to it), we simply denote with ACC^v with $v = \mathsf{W}$ (resp., $v = \bar{\mathsf{W}}$, $v = \mathsf{U}$) the problem of determining whether the status of α is equal to v w.r.t. a DeLP program $\mathcal{P}$.

4.3 Assumption-based Argumentation

In this section, we show how UAF can encode the ABA framework.

Definition 11. *Given an ABAF* $\mathcal{D} = \langle \mathtt{Lang}, \mathtt{Rules}, \mathtt{Asm}, \bar{\ } \rangle$*, we denote with* $\mathcal{G}_\mathcal{D}$ $= (\mathcal{L}_\mathcal{D}, \mathcal{R}_\mathcal{D})$ *the UAF obtained from* $\mathcal{D}$ *as follows:*

- $\mathcal{L}_{\mathcal{D}} = \mathtt{Lang}$*;*
- $\mathcal{R}_{\mathcal{D}} = \{\mathcal{R}_{\mathtt{Asm}}, \mathcal{R}_{\mathtt{contr}}, \mathcal{R}_{\mathtt{Rules}}\}$*, where:*
  - $\mathcal{R}_{\mathtt{Asm}} = \{(a, \mathtt{true}) \mid a \in \mathtt{Asm}\}$
  - $\mathcal{R}_{\mathtt{contr}} = \{(a, \bar{}(a)) \mid a \in \mathtt{Asm}\}$
  - $\mathcal{R}_{\mathtt{Rules}} = \{(head(r), tail(r)) \mid r \in \mathtt{Rules}\}$

Proposition 5. *Given an ABAF* $\mathcal{D} = \langle \mathtt{Lang}, \mathtt{Rules}, \mathtt{Asm}, \bar{} \rangle$*, semantics* $\mathsf{sem} \in \{\mathtt{gr}, \mathtt{co}, \mathtt{st}, \mathtt{pr}\}$*, and goal literal* $g \in \mathtt{Lang}$*, it holds that* g *is credulously (resp., skeptically) accepted in* $\mathcal{D}$ *under semantics* sem *iff* $(\mathcal{G}_{\mathcal{D}}, g)$ *is a true instance of both* $\mathsf{c\text{-}ACC}^{v_{in}}_{\mathsf{sem}}$ *(resp.,* $\mathsf{s\text{-}ACC}^{v_{in}}_{\mathsf{sem}}$*).*

Having covered how UAF can encode a variety of argumentation formalisms, in the next section we extend the analysis to cover their probabilistic extensions with UPAF.

5 Probabilistic Argumentation Models as Instantiations of UPAF

As in the previous section, we start instantiating UPAF in Abstract Argumentation, and then move to the structured argumentation formalisms.

5.1 PrAF as a Special Instance of UPAF

Several works have approached the combination of abstract argumentation and probabilities. Among them, perhaps the the most popular is the so-called *constellations* approach [47, 48, 49, 50, 51], where alternative scenarios, called *possible worlds*, are associated with probabilities. In particular, in a *Probabilistic Argumentation Framework* (PrAF) [51, 52, 53, 54, 55] a probability distribution function (PDF) on the set of possible worlds is entailed by the probabilities that are associated with arguments and attacks.

A PrAF is a triple $\langle \mathsf{A}, \mathsf{R}, P \rangle$, where $\langle \mathsf{A}, \mathsf{R} \rangle$ is an AF and P is a function assigning a probability value to every argument in A, that is, $P : \mathsf{A} \to (0, 1]$. The semantics of a PrAF is given in terms of possible worlds (or scenarios using the terminology of UPAF), that is any AF $\langle \mathsf{A}', \mathsf{R}' \rangle$ with $\mathsf{A}' \subseteq \mathsf{A}$ and $\mathsf{R}' \subseteq \mathsf{A}' \times \mathsf{A}'$. We denote with $pw(\Delta)$ the set of all possible worlds of Δ. Each possible world $w \in pw(\Delta)$ has an associated probability defined as $P(w) = \prod_{a \in \mathsf{A}'} P(a) \times \prod_{a \in \mathsf{A} \setminus \mathsf{A}'} (1 - P(a))$. Relevant problems for PrAF are those concerning credulous and skeptical acceptance, defined as follows.

Definition 12. *Given an PrAF $\Delta = \langle \mathsf{A}, \mathsf{R}, P\rangle$, an argument $g \in \mathsf{A}$, and semantics $\mathit{sem} \in \{\mathtt{gr}, \mathtt{co}, \mathtt{st}, \mathtt{pr}\}$, the probability*

- *$\mathit{PrCA}_{\mathit{sem}}(\Delta, g)$ that g is credulously acceptable w.r.t. semantics sem is:*

$$\sum_{w \in pw(\Delta) \ s.t. \ \exists E \in \mathit{sem}(\Delta) \wedge g \in E} P(w);$$

- *$\mathit{PrSA}_{\mathit{sem}}(\Delta, g)$ that g is skeptically acceptable w.r.t. semantics sem is:*

$$\sum_{w \in pw(\Delta) \ s.t. \ \mathit{sem}(\Delta) \neq \emptyset \wedge \forall E \in \mathit{sem}(\Delta) \wedge g \in E} P(w).$$

PrAFs can be encoded in UPAF as follows.

Definition 13. *Given an PrAF $\Delta = \langle \mathsf{A}, \mathsf{R}, P\rangle$, we denote with $\mathcal{I}_\Delta = (\mathcal{G}_\Delta, \mathcal{E}, \mathcal{F})$ the UPAF obtained from Δ as follows:*

- *$\mathcal{G}_\Delta$ is the UAF obtained from AF $\langle \mathsf{A}, \mathsf{R}\rangle$;*
- *$\mathcal{E}$ is built over the set $E = \{\epsilon_a \mid a \in \mathsf{A}\}$ of independent events, such that: for any scenario $\sigma \in 2^E$, $\mathcal{E}(\sigma) = \prod_{\epsilon_a \in \sigma} P(a) \times \prod_{\epsilon_a \notin \sigma} 1 - P(a)$; and*
- *$\mathcal{F}$ associates, to each argument $a \in \mathsf{A}$ the simple formula ϵ_a.*

Proposition 6. *Given a PrAF $\Delta = \langle \mathsf{A}, \mathsf{R}, P\rangle$, semantics $\mathit{sem} \in \{\mathtt{gr}, \mathtt{co}, \mathtt{st}, \mathtt{pr}\}$, and argument $g \in \mathsf{A}$, it holds that $g^\ell = \mathit{PrCA}^{\mathit{sem}}(\Delta, g)$.*

It is worth noting that, altough we focused on probabilistic arguments only, the approach is easily generalizable to also model probabilistic attacks. Furthermore, the independence assumption could be relaxed by using a more general environmental model (e.g. Markov/Bayesian Networks).

5.2 Instantiating UPAF with DeLP3E

Defeasible Logic Programming with Presumptions and Probabilistic Environments (DeLP3E) [32] extends DeLP in the sense that every element of a DeLP program $\mathcal{P}$ only holds in certain scenarios in the set $2^{\mathbf{E}}$, i.e., these elements are subject to probabilistic events. Each element of $\mathcal{P}$ can thus be associated with a formula over $\mathbf{E}$.

Definition 14 (DeLP3E Program)**.** *A* DeLP3E program *is encoded as UPAF $\mathcal{I} = (\mathcal{P}, \mathcal{E}, \mathcal{F})$, where $\mathcal{P}$ is a DeLP program (an AM), $\mathcal{E} : 2^E \to [0, 1]$ is an EM, and $\mathcal{F}$ is the annotation function associating to elements in $\mathcal{P}$ either i)* `true` *or ii) a propositional formula built on E.*

Given DeLP3E program $\mathcal{I} = (\mathcal{P}, \mathcal{E}, \mathcal{F})$ and scenario σ, we denote with $\mathcal{P}^\sigma = (\{\gamma \in \Pi \mid \sigma \models \mathcal{F}(\gamma)\}, \{\gamma \in \Delta \mid \sigma \models \mathcal{F}(\gamma)\})$ the DeLP program *induced by σ*.

Example 7. The DeLP3E program of Example 1 is $\mathcal{I} = (\mathcal{P}, \mathcal{E}, \mathcal{F})$, where $\mathcal{P} = (\Pi = \{\pi_1, \pi_2, \pi_3\}, \Delta = \{\delta_1, \delta_2\})$ is the DeLP program of Example 3, $\mathcal{E}$ is the EM discussed in Example 6, and the annotation function $\mathcal{F}$ is as follows:

- $\mathcal{F}(\pi_1) = \neg\texttt{rainy} \wedge \neg\texttt{windy}$;
- $\mathcal{F}(\pi_2) = \texttt{rainy} \wedge \neg\texttt{windy}$;
- $\mathcal{F}(\pi_3) = \neg\texttt{rainy}$.
- $\mathcal{F}(\delta_1) = \texttt{true}$;
- $\mathcal{F}(\delta_2) = \texttt{true}$;

Moreover, the induced DeLP programs w.r.t. each scenario σ_i of $\mathcal{I}$ (with $i \in [1, 4]$) are the following:

- $\mathcal{P}^{\sigma_1} = (\emptyset, \{\delta_1, \delta_2\})$ with $\sigma_1 = \{\texttt{rainy}, \texttt{windy}\}$;
- $\mathcal{P}^{\sigma_2} = (\{\pi_2\}, \{\delta_1, \delta_2\})$ with $\sigma_2 = \{\texttt{rainy}, \neg\texttt{windy}\}$;
- $\mathcal{P}^{\sigma_3} = \mathcal{P}^{\sigma_1}$ with $\sigma_3 = \{\neg\texttt{rainy}, \texttt{windy}\}$; and
- $\mathcal{P}^{\sigma_4} = (\{\pi_1, \pi_3\}, \{\delta_1, \delta_2\})$ with $\sigma_4 = \{\neg\texttt{rainy}, \neg\texttt{windy}\}$.

Note that it is possible for the same DeLP program to be induced from distinct scenarios (e.g., scenarios σ_1 and σ_3). ■

The semantics of DeLP3E, or the warranting status of a literal α, depends on each induced DeLP program (obtained by all scenarios $\sigma \in 2^{\{\texttt{rainy},\texttt{windy}\}}$). This is captured in the following definition, that is a special casee of Definition 6.

Definition 15 (DeLP3E Acceptance Problem)**.** *Given a DeLP3E program $\mathcal{I} = (\mathcal{P}, \mathcal{E}, \mathcal{F})$, DeLP semantics* sem*=DeLP for the Analytical Model, and a goal literal $\alpha \in \mathcal{P}$, the following is the acceptance interval (also called warranted interval) of α w.r.t. $\mathcal{I}$:*

$$\left[\alpha^\ell = \sum_{\sigma \in 2^E s.t.\ \mathsf{ACC}^{\mathsf{W}}(\mathcal{P}^\sigma, \alpha)} \mathcal{E}(\sigma), \quad \alpha^u = 1 - \sum_{\sigma \in 2^E s.t.\ \mathsf{ACC}^{\bar{\mathsf{W}}}(\mathcal{P}^\sigma, \alpha)} \mathcal{E}(\sigma)\right].$$

That is, α^ℓ denotes the the sum of all the probabilities associated with scenarios inducing a DeLP program that warrants literal α. On the other hand, $1 - \alpha^u$ denotes the sum of all the probabilities associated with scenarios inducing a DeLP program that warrants the complement of α (i.e., $\sim\!\alpha$). As before, $[\alpha^\ell, \alpha^u]$ provides bounds for the (uncertain) probabilities with which literal α is warranted in $\mathcal{I}$ under semantics sem.

Example 8. Considering the DeLP3E $\mathcal{I}$ of Example 8 and the literal $\alpha = \texttt{eat\_fish}$, we have that $[\alpha^\ell, \alpha^u] = [\mathcal{E}(\sigma_4), 1] = [0.4, 1]$. ■

5.3 Instatinating UPAF with ABA

In this section, we present *Probabilistic Assumption-Based Argumentation* (PrABA), that is a restriction of UPAF where ABA is used as Analytical Model.

Definition 16 (PrABA Program)**.** *A* Probabilistic Assumption-Based Argumentation *(PrABA) program can be encoded in UPAF as $\mathcal{I} = (\mathcal{G}_\mathcal{D}, \mathcal{E}, \mathcal{F})$ where $\mathcal{G}_\mathcal{D} = (\mathcal{L}_\mathcal{D}, \mathcal{R}_\mathcal{D})$ is the UAF obtained from an ABAF $\mathcal{D} = \langle \mathtt{Lang}, \mathtt{Rules}, \mathtt{Asm}, ^{-}\rangle$, $\mathcal{E} : 2^E \to [0,1]$ is an EM, and $\mathcal{F}$ is the annotation function associating to elements in $(\mathcal{L}_\mathcal{D} \cup \mathcal{R}_{\mathtt{Asm}} \cup \mathcal{R}_{\mathtt{contr}} \cup \mathcal{R}_{\mathtt{Rules}})$ either i)* `true` *or ii) a propositional formula built over elements of E.*

Given a PrABA program $\mathcal{I} = (\mathcal{G}_\mathcal{D}, \mathcal{E}, \mathcal{F})$ and a scenario σ, the ABAF induced by σ (that is, the ABAF obtained from $\mathcal{D}$ by removing annotated elements that are not entailed from the scenario σ) is $\mathcal{D}^\sigma = \langle \mathtt{Lang}^\sigma, \mathtt{Rules}^\sigma, \mathtt{Asm}^\sigma, ^{-}\rangle$ where:

- $\mathtt{Lang}^\sigma = \{x \in \mathtt{Lang} \mid \sigma \models \mathcal{F}(x)\}$;
- $\mathtt{Rules}^\sigma = \{r \in \mathtt{Rules} \mid \sigma \models \mathcal{F}((head(r), tail(r))\}$;
- $\mathtt{Asm}^\sigma = \{x \in \mathtt{Asm} \mid \sigma \models \mathcal{F}(x, \mathtt{true})\}$

Example 9. *Consider the PrABA $\mathcal{I} = (\mathcal{G}_\mathcal{D}, \mathcal{E}, \mathcal{F})$ where:*

- *$\mathcal{D} = \langle \mathtt{Lang}, \mathtt{Rules}, \mathtt{Asm}, ^{-}\rangle$ is the ABAF of Example 5.*
- *$\mathcal{E}$ is an Environmental Model built over two events $E = \{\epsilon_1, \epsilon_2\}$, and associates to the four scenarios σ_i (with $i \in [1,4]$) the following probabilities:*
  - $\mathcal{E}(\sigma_1 = \{\epsilon_1, \epsilon_2\}) = 0.1$; $\mathcal{E}(\sigma_2 = \{\epsilon_1, \neg\epsilon_2\}) = 0.2$
  - $\mathcal{E}(\sigma_3 = \{\neg\epsilon_1, \epsilon_2\}) = 0.3$ $\mathcal{E}(\sigma_4 = \{\neg\epsilon_1, \neg\epsilon_2\}) = 0.4$.
- *the function $\mathcal{F}$ annotates all elements with* `true` *except from $\mathcal{F}(\mathtt{a}) = \neg\epsilon_1 \wedge \neg\epsilon_2$, $\mathcal{F}(r_4) = \epsilon_1$*

Thus, there are three distinct induced ABAFs. Among them, considering scenario $\sigma_1 = \{\neg\epsilon_1, \neg\epsilon_2\}$ having probability $\mathcal{E}(\sigma_1) = 0.3$, the induced ABAF is $\mathcal{D}^\sigma = \langle \mathtt{Lang}, \mathtt{Rules} \setminus \{r_4\}, \mathtt{Asm}, ^{-}\rangle$. ■

The semantics of PrABA is defined on each induced ABAF (obtained by all possible scenarios). This is captured in the following definition, that is a special instance of Definition 6.

	UAF instances					UPAF instances	
sem	AF	BAF	BSAF	DeLP	ABA	PrAF	DeLP3E
gr	P [34]	P [56]	P [57, 58]	-	P [59, 60, 61]	FP$^{\#P}$-h [62]	
co	NP-c[63]	NP-c[56]	NP-c[57, 58]	-	NP-c[59, 60, 61]	FP$^{\#P}$-h [62]	
st	NP-c[63]	NP-c[56]	NP-c[57, 58]	-	NP-c[59, 60, 61]	FP$^{\#P}$-h [62]	
pr	NP-c[63, 64]	NP-c[56]	NP-c[57, 58]	-	NP-c[59, 60, 61]	FP$^{\#P}$-h [62]	
d	-	-	-	coNP-h[65]	-	-	#P-h [66]

Table 2: Complexity results for $\mathsf{c}\text{-}\mathsf{ACC}^{v_{in}}_{\mathsf{sem}}$ problems in UAF (columns 2-6) and UPAF (columns 7-8) instances. For complexity class $\mathcal{C}$, $\mathcal{C}$-c (resp., $\mathcal{C}$-h) stands for $\mathcal{C}$-complete (resp., $\mathcal{C}$-hard). For ABA we refer to the Logic Programming fragment. $\mathtt{d}$ stands for DeLP semantics. For each result, we also report the respective references.

Definition 17 (PrABA Acceptance Problem). *Given a PrABA $\mathcal{I} = (\mathcal{G}_{\mathcal{D}} = (\mathcal{L}_{\mathcal{D}}, \mathcal{R}_{\mathcal{D}}), \mathcal{E}, \mathcal{F})$, semantics $\mathsf{sem} \in \{\mathtt{gr}, \mathtt{co}, \mathtt{st}, \mathtt{pr}\}$, and a goal literal $g \in \mathcal{L}_{\mathcal{D}}$, the following is the acceptance interval of g w.r.t. $\mathcal{I}$:*

$$\left[g^{\ell} = \sum_{\sigma \in 2^E \, s.t. \; \mathsf{c}\text{-}\mathsf{ACC}^{v_{in}}_{\mathsf{sem}}(\mathcal{D}^{\sigma}, g)} \mathcal{E}(\sigma), \quad \alpha^u = 1 - \sum_{\sigma \in 2^E \, s.t. \; \mathsf{s}\text{-}\mathsf{ACC}^{v_{out}}_{\mathsf{sem}}(\mathcal{D}^{\sigma}, g)} \mathcal{E}(\sigma) \right].$$

That is, g^{ℓ} denotes the the sum of all the probabilities associated to scenarios inducing an ABAF where g is credulously accepted. In contrast, $1 - g^u$ denotes the sum of all the probabilities associated to scenarios inducing an ABAF where the g is skeptically rejected. Again, the $[g^{\ell}, g^u]$ interval quantifies the probability with which goal literal g is accepted in $\mathcal{I}$ under semantics sem.

6 Complexity of UPAF

In this section, we discuss the computational complexity of the acceptance problem in the above-presented UPAF instances. In particular, we focus on the problem $\mathsf{c}\text{-}\mathsf{ACC}^{v_{in}}_{\mathsf{sem}}$ as it is the one that has received the most attention, often referred to as the (probabilistic) credulous acceptance problem. Before discussing the computational complexity, we define the relationship between DeLP3E and DeLP.

Proposition 7. *Given a DeLP program $\mathcal{P}$, and literal $\alpha \in \mathcal{P}$, it holds that α is warranted (resp., unwarranted, undecided) in $\mathcal{P}$ iff $[1, 1]$ (resp., $[0, 0]$, $[0, 1]$) is the warranted interval of α w.r.t. $\mathcal{I}_{\mathcal{P}} = (\mathcal{P}, \mathcal{E}, \mathcal{F})$, where $\mathcal{E} : \{\emptyset\} \to [0, 1]$, and $\mathcal{F}(\gamma) = \mathtt{true}$ for any $\gamma \in \mathcal{P}$.*

Proof. First observe that $2^E = \{\emptyset\}$, and thus the only scenario is $\sigma = \emptyset$. As $\mathcal{E}$ induces a probability distribution over 2^E, we have that $\mathcal{E}(\sigma) = 1$. Thus, $\mathcal{P}^{\sigma} = \mathcal{P}$.

Moreover, from Definition 15, we have that $\alpha^\ell = 1$ whenever α is warranted in $\mathcal{P}$. Similarly, $\alpha^u = 0$ whenever $\neg\alpha$ is warranted in $\mathcal{P}$. □

Thus, DeLP can be seen as a special case of DeLP3E where the EM part is not taken into account. The complexity of the problem of deciding the warrant status of a literal α in a DeLP program $\mathcal{P}$, has been shown to be coNP-hard in [67]). Thus, by exploiting the result of Proposition 7, it follows that $\mathsf{c\text{-}ACC}^{v_{in}}_{\mathsf{sem}}$ for DeLP3E is coNP-hard too. Moreover, it has been shown that the complexity $\mathsf{c\text{-}ACC}^{v_{in}}_{\mathsf{sem}}$ in DeLP3E remains intractable (#P-hard) even when assuming that either (i) $\mathsf{c\text{-}ACC}^{v_{in}}_{\mathsf{sem}}$ can be solved in polynomial time, or (ii) that probabilities among all possible scenarios of the EM (i.e., $\mathcal{E}(\sigma_i)$) can be computed in polynomial time [66]. The fact that computing the warranted interval of literals is hard suggests that a brute force procedure to compute the warranted interval $[\alpha^\ell, \alpha^u]$ (consists of computing the warranted status of α in each DeLP program $\mathcal{P}^\sigma$ induced by any scenario σ) would be infeasible.

In Table 2 we also report complexity results for $\mathsf{c\text{-}ACC}^{v_{in}}_{\mathsf{sem}}$ problems in UAF (columns 2-6) and UPAF (columns 7-8) instances. We also report the respective references for each result. From the results reported in Table 2 we can conclude that the complexity of all the considered frameworks lies within the same level of the polynomial hierarchy when probabilities are not taken into account. For UPAF instances, i.e., when probabilities are allowed, the acceptance problem in DeLP3E is $\#P$-hard, while in Probabilistic AF under independence and constellation approach it is $FP^{\#P}$-hard. Moreover, as ABA could encode AF, a many-to-one reduction from PrAF acceptance to PrABA acceptance can be provided. Thus, as a consequence, acceptance in PrABA is still $FP^{\#P}$-hard.

7 Related Work

Structured argumentation differs from abstract argumentation in the use of a formal language for the representation of knowledge and the prescription of rules governing the construction of arguments and counterarguments from this knowledge. A structured argument is thereby characterized by the systematic explicitness of its premises and claims, underpinned by a formal definition of the relationship that exists between the premises and the claim. In [40], four frameworks that consider the structure of arguments are presented; two of them—ASPIC+ [20] and ABA [36, 19]—build the set of all possible arguments from the knowledge base and then rely on using one of the possible Dung (abstract) semantics to decide on the acceptance of arguments. The other two—Logic-Based Deductive Argumentation [23] and DeLP [22]—only build the arguments involved in answering the query. The latter two frameworks exhibit several differences [40]; among them is the base logic used as a knowledge representation language: [23] relies on propositional logic, requiring a theorem prover to solve queries, while DeLP [22] adopts an extension of logic programming, which is on the other hand a computational framework.

For a better understanding of the differences among the above-mentioned frameworks, we refer the interested reader to [68], where a variant of DeLP program using the grounded semantics is also discussed. An important distinction between DeLP and the other three frameworks, which significantly affects the resolution of a query, rests on how attacks between arguments are described. DeLP considers two forms of defeat: *proper* and *blocking*; the former is akin to Dung's form of defeat, called *attack* in [34], whereas the latter behaves differently since the two arguments that are part of the blocking defeat relation, attacker and attackee, are defeated. Certainly, this scenario can be represented within Dung's framework as a mutual attack. However, the DeLP program mechanism prohibits the consecutive application of two blocking defeaters because, in a well-structured dialogue, introducing another blocking defeater becomes redundant as the initial two have already been refuted. Additionally, in the pursuit of the answers essential to the query, various dialogical aspects are considered, enhancing the reasoning process by disallowing common dialogical fallacies. These attributes have been incorporated into the advancement of a game-based semantics [69, 70].

The combination of probabilistic theory with structured argumentation has been explored in various prior works, ranging from early probabilistic argumentation systems to approaches based on possibilistic and probabilistic logic [71, 50, 72, 51].

8 Conclusions and Future Work

In this work, we have introduced the Unified Probabilistic Argumentation Framework (UPAF), an expressive theoretical model designed to capture a wide range of existing argumentation formalisms. We demonstrate the UPAF's strength lies in its two-component architecture: an Analytical Model (AM) to represent domain-knowledge and its conflicts, and an Environmental Model (EM) to assign probabilistic events to elements of the argumentative structure. We have formally established UPAF's capability to encode classical and extended versions of Dung's framework, as well as structured formalisms like Assumption-Based Argumentation (ABA) and Defeasible Logic Programming (DeLP). Additionally, we analyzed the computational complexity of key reasoning tasks, laying the groundwork for their future algorithmic study. As a first direction for future work, we plan to implement UPAF instances. Moreover, given the high computational cost of computing the exact (acceptance) interval, as future work, we plan to investigate techniques that allow us to approximate, with an acceptable value, that exact interval.

References

[1] D. Gabbay, M. Giacomin, G. R. Simari, M. Thimm (Eds.), Handbook of Formal Argumentation, College Public., 2021.

[2] P. M. Dung, P. M. Thang, N. D. Hung, Modular argumentation for modelling legal doctrines of performance relief, Argument Comput. 1 (1) (2010) 47–69.

[3] K. Atkinson, T. J. M. Bench-Capon, Argumentation schemes in AI and law, Argument & Computation 12 (3) (2021) 417–434.

[4] L. Amgoud, H. Prade, Using arguments for making and explaining decisions, Artificial Intelligence 173 (3-4) (2009) 413–436.

[5] T. J. M. Bench-Capon, K. Atkinson, A. Z. Wyner, Using argumentation to structure e-participation in policy making, Transactions on Large-Scale Data and Knowledge-Centered Systems 18 (2015) 1–29.

[6] M. Snaith, R. Ø. Nielsen, S. R. Kotnis, A. Pease, Ethical challenges in argumentation and dialogue in a healthcare context, Argument & Computation 12 (2) (2021) 249–264.

[7] K. Cyras, T. Oliveira, A. Karamlou, F. Toni, Assumption-based argumentation with preferences and goals for patient-centric reasoning with interacting clinical guidelines, Argument Comput. 12 (2) (2021) 149–189.

[8] N. Kökciyan, I. Sassoon, E. Sklar, S. Modgil, S. Parsons, Applying metalevel argumentation frameworks to support medical decision making, IEEE Intelligent Systems 36 (2) (2021) 64–71.

[9] R. Craven, F. Toni, C. Cadar, A. Hadad, M. Williams, Efficient argumentation for medical decision-making, in: Principles of Knowledge Representation and Reasoning: Proceedings of the Thirteenth International Conference (KR), 2012.

[10] X. Fan, S. Liu, H. Zhang, C. Leung, C. Miao, Explained activity recognition with computational assumption-based argumentation, in: Proceedings of ECAI Conference, Vol. 285, IOS Press, 2016, pp. 1590–1591.

[11] A. Pazienza, D. Grossi, F. Grasso, R. Palmieri, M. Zito, S. Ferilli, An abstract argumentation approach for the prediction of analysts' recommendations following earnings conference calls, Intelligenza Artificiale 13 (2) (2019) 173–188.

[12] M. E. B. Brarda, L. H. Tamargo, A. J. García, Using argumentation to obtain and explain results in a decision support system, IEEE Intelligent Systems 36 (2) (2021) 36–42.

[13] G. Alfano, M. Calautti, S. Greco, F. Parisi, I. Trubitsyna, Explainable acceptance in probabilistic abstract argumentation: Complexity and approximation, in: Proc. of KR, 2020, pp. 33–43.

[14] N. Kökciyan, N. Yaglikci, P. Yolum, An argumentation approach for resolving privacy disputes in online social networks, ACM Transactions on Internet Technology 17 (3)

(2017) 27:1–27:22.

[15] P. M. Dung, On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games, Artif. Intell. 77 (2) (1995) 321–358. doi:10.1016/0004-3702(94)00041-X.
URL https://doi.org/10.1016/0004-3702(94)00041-X

[16] L. Amgoud, C. Cayrol, M. Lagasquie-Schiex, P. Livet, On bipolarity in argumentation frameworks, Int. J. Intell. Syst. 23 (10) (2008) 1062–1093.

[17] S. H. Nielsen, S. Parsons, A generalization of dung's abstract framework for argumentation: Arguing with sets of attacking arguments, in: Proc. of ArgMAS, Vol. 4766 of Lecture Notes in Computer Science, 2006, pp. 54–73.

[18] A. Bondarenko, P. M. Dung, R. A. Kowalski, F. Toni, An abstract, argumentation-theoretic approach to default reasoning, Artif. Intell. 93 (1997) 63–101.

[19] F. Toni, A tutorial on assumption-based argumentation, Argument & Computation 5 (1) (2014) 89–117.

[20] S. Modgil, H. Prakken, The ASPIC$^+$ framework for structured argumentation: A tutorial, Argument & Computation 5 (1) (2014) 31–62.

[21] A. J. García, G. R. Simari, Defeasible logic programming: An argumentative approach, Theory Pract. Log. Program. 4 (1-2) (2004) 95–138.

[22] A. J. García, G. R. Simari, Defeasible logic programming: DeLP-servers, contextual queries, and explanations for answers, Argument & Computation 5 (1) (2014) 63–88.

[23] P. Besnard, A. Hunter, Constructing argument graphs with deductive arguments: A tutorial, Argument & Computation 5 (1) (2014) 5–30.

[24] L. Amgoud, F. Nouioua, An argumentation system for defeasible reasoning, Int. J. Approx. Reason. 85 (2017) 1–20.

[25] J. Y. Halpern, Reasoning about uncertainty, MIT Press, 2005.

[26] S. Parsons, Qualitative methods for reasoning under uncertainty, MIT Press, 2001.

[27] A. Hunter, S. Polberg, N. Potyka, T. Rienstra, M. Thimm, Probabilistic argumentation: A survey, Handbook of Formal Argumentation 2 (2021) 397–441.

[28] G. F. Georgakopoulos, D. J. Kavvadias, C. H. Papadimitriou, Probabilistic satisfiability, J. Complex. 4 (1) (1988) 1–11.

[29] N. J. Nilsson, Probabilistic logic revisited, Artif. Intell. 59 (1-2) (1993) 39–42.

[30] F. Riguzzi, T. Swift, A survey of probabilistic logic programming, in: M. Kifer, Y. A. Liu (Eds.), Declarative Logic Programming: Theory, Systems, and Applications, ACM / Morgan & Claypool, 2018, pp. 185–228.

[31] D. Suciu, D. Olteanu, C. Ré, C. Koch, Probabilistic Databases, Synthesis Lectures on Data Management, Morgan & Claypool Publishers, 2011.

[32] P. Shakarian, G. I. Simari, G. Moores, D. Paulo, S. Parsons, M. A. Falappa, A. Aleali, Belief revision in structured probabilistic argumentation - model and application to cyber security, Ann. Math. Artif. Intell. 78 (3-4) (2016) 259–301. doi:10.1007/S10472-015-9483-5.
URL https://doi.org/10.1007/s10472-015-9483-5

[33] P. Shakarian, G. I. Simari, G. Moores, S. Parsons, M. A. Falappa, An argumentation-based framework to address the attribution problem in cyber-warfare, in: Proc. Cyber-Security, ASE, 2014.

[34] P. M. Dung, On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games, Artif. Intell. 77 (2) (1995) 321–358.

[35] A. J. García, G. R. Simari, Defeasible logic programming: An argumentative approach, TPLP 4 (1-2) (2004) 95–138. doi:10.1017/S1471068403001674.
URL https://doi.org/10.1017/S1471068403001674

[36] A. Bondarenko, F. Toni, R. A. Kowalski, An assumption-based framework for non-monotonic reasoning, in: Proceedings of the Second International Workshop on Logic Programming and Non-monotonic Reasoning (LPNMR), 1993, pp. 171–189.

[37] M. V. Martinez, A. J. García, G. R. Simari, On the use of presumptions in structured defeasible reasoning, in: B. Verheij, S. Szeider, S. Woltran (Eds.), Proc. COMMA, Vol. 245, IOS Press, 2012, pp. 185–196.

[38] F. Stolzenburg, A. J. García, C. I. Chesñevar, G. R. Simari, Computing generalized specificity, Journal of Applied Non-Classical Logics 13 (1) (2003) 87–113.

[39] K. Cyras, X. Fan, C. Schulz, F. Toni, Assumption-based argumentation: Disputes, explanations, preferences, IFCoLog Journal of Logics and Their Applications 4 (8) (2017) 2407.

[40] P. Besnard, A. J. Garcia, A. Hunter, S. Modgil, H. Prakken, G. R. Simari, F. Toni, Introduction to structured argumentation, Argument & Computation – Special Issue: Tutorials on Structured Argumentation 5 (1) (2014) 1–4.

[41] R. Craven, F. Toni, Argument graphs and assumption-based argumentation, Artif. Intell. 233 (2016) 1–59.

[42] J. Pearl, Bayesian networks (1988).

[43] M. Richardson, P. Domingos, Markov logic networks, Machine learning 62 (1-2) (2006) 107–136.

[44] N. J. Nilsson, Probabilistic logic, Artif. Intell. 28 (1) (1986) 71–87.

[45] S. Gottifredi, A. Cohen, A. J. García, G. R. Simari, Characterizing acceptability semantics of argumentation frameworks with recursive attack and support relations, Artif. Intell. 262 (2018) 336–368.

[46] G. Alfano, A. Cohen, S. Gottifredi, S. Greco, F. Parisi, G. R. Simari, Credulous acceptance in high-order argumentation frameworks with necessities: An incremental approach, Artif. Intell. 333 (2024) 104159.

[47] P. M. Dung, P. M. Thang, Towards (probabilistic) argumentation for jury-based dispute resolution, in: Proc. of Int. Conf. on Computational Models of Argument (COMMA), 2010, pp. 171–182.

[48] T. Rienstra, Towards a probabilistic Dung-style argumentation system, in: Proc. of Int. Conf. on Agreement Technologies (AT), 2012, pp. 138–152.

[49] D. Doder, S. Woltran, Probabilistic argumentation frameworks-A logical approach, in:

Proc. of Int. Conf. on Scalable Uncertainty Management (SUM), 2014, pp. 134–147.

[50] A. Hunter, Some foundations for probabilistic abstract argumentation, in: B. Verheij, S. Szeider, S. Woltran (Eds.), Computational Models of Argument - Proceedings of COMMA 2012, Vienna, Austria, September 10-12, 2012, Vol. 245 of Frontiers in Artificial Intelligence and Applications, IOS Press, 2012, pp. 117–128. `doi:10.3233/978-1-61499-111-3-117`.
URL `https://doi.org/10.3233/978-1-61499-111-3-117`

[51] H. Li, N. Oren, T. J. Norman, Probabilistic argumentation frameworks, in: S. Modgil, N. Oren, F. Toni (Eds.), Theorie and Applications of Formal Argumentation - First International Workshop, TAFA 2011. Barcelona, Spain, July 16-17, 2011, Revised Selected Papers, Vol. 7132 of Lecture Notes in Computer Science, Springer, 2011, pp. 1–16. `doi:10.1007/978-3-642-29184-5\_1`.
URL `https://doi.org/10.1007/978-3-642-29184-5_1`

[52] B. Fazzinga, S. Flesca, F. Parisi, On the complexity of probabilistic abstract argumentation frameworks, ACM Trans. Comput. Log. 16 (3) (2015) 22.

[53] N. Potyka, A polynomial-time fragment of epistemic probabilistic argumentation, Int. J. Approx. Reason. 115 (2019) 265–289.

[54] R. Riveret, N. Oren, G. Sartor, A probabilistic deontic argumentation framework, Int. J. Approx. Reason. 126 (2020) 249–271.

[55] P. Dondio, Toward a computational analysis of probabilistic argumentation frameworks, Cybern. Syst. 45 (3) (2014) 254–278.

[56] W. Dvorák, P. E. Dunne, Computational problems in formal argumentation and their complexity, FLAP 4 (8) (2017).

[57] M. Berthold, A. Rapberger, M. Ulbricht, Capturing non-flat assumption-based argumentation with bipolar setafs, in: Proceedings of the 21st International Conference on Principles of Knowledge Representation and Reasoning, KR, 2024.

[58] W. Dvořák, M. König, S. Woltran, Parameterized complexity of abstract argumentation with collective attacks, Argument & Computation 16 (1) (2025) 19462174251319186.

[59] Y. Dimopoulos, B. Nebel, F. Toni, On the computational complexity of assumption-based argumentation for default reasoning, Artif. Intell. 141 (1/2) (2002) 57–78.

[60] T. Lehtonen, J. P. Wallner, M. Järvisalo, Declarative algorithms and complexity results for assumption-based argumentation, J. Artif. Intell. Res. 71 (2021) 265–318.

[61] K. Cyras, Q. Heinrich, F. Toni, Computational complexity of flat and generic assumption-based argumentation, with and without probabilities, Artif. Intell. 293 (2021) 103449.

[62] B. Fazzinga, S. Flesca, F. Furfaro, Credulous and skeptical acceptability in probabilistic abstract argumentation: complexity results, Intelligenza Artificiale 12 (2) (2018) 181–191. `doi:10.3233/IA-180041`.
URL `https://doi.org/10.3233/IA-180041`

[63] Y. Dimopoulos, A. Torres, Graph theoretical structures in logic programs and default theories, Theor. Comput. Sci. 170 (1-2) (1996) 209–244.

[64] P. E. Dunne, T. J. M. Bench-Capon, Coherence in finite argument systems, Artif. Intell. 141 (1/2) (2002) 187–203.

[65] G. Alfano, S. Greco, F. Parisi, G. I. Simari, G. R. Simari, Incremental computation for structured argumentation over dynamic delp knowledge bases, Artif. Intell. 300 (2021) 103553.

[66] G. I. Simari, P. Shakarian, M. A. Falappa, A quantitative approach to belief revision in structured probabilistic argumentation, Ann. Math. Artif. Intell. 76 (3-4) (2016) 375–408. doi:10.1007/s10472-015-9476-4.
URL https://doi.org/10.1007/s10472-015-9476-4

[67] G. Alfano, S. Greco, F. Parisi, G. I. Simari, G. R. Simari, Incremental computation for structured argumentation over dynamic delp knowledge bases, Artificial Intelligence 300 (2021) 103553. doi:10.1016/j.artint.2021.103553.
URL https://www.sciencedirect.com/science/article/pii/S0004370221001041

[68] A. J. García, H. Prakken, G. R. Simari, A comparative study of some central notions of ASPIC$^+$ and DeLP, Theory Pract. Log. Program. 20 (3) (2020) 358–390.

[69] I. D. Viglizzo, F. A. Tohmé, G. R. Simari, The foundations of DeLP: defeating relations, games and truth values, Ann. Math. Artif. Intell. 57 (2) (2009) 181–204.

[70] Y. Soto, A. Cohen, C. A. D. Deagustini, M. V. Martinez, G. I. Simari, A Principle-based Framework for Analyzing Dialogue Game-based Semantics, in: Proc. KR (To Appear – available at: https://bit.ly/KR25-Soto-CR), 2025.

[71] M. Thimm, A probabilistic semantics for abstract argumentation, in: L. D. Raedt, C. Bessiere, D. Dubois, P. Doherty, P. Frasconi, F. Heintz, P. J. F. Lucas (Eds.), ECAI 2012 - 20th European Conference on Artificial Intelligence. Including Prestigious Applications of Artificial Intelligence (PAIS-2012) System Demonstrations Track, Montpellier, France, August 27-31 , 2012, Vol. 242 of Frontiers in Artificial Intelligence and Applications, IOS Press, 2012, pp. 750–755. doi:10.3233/978-1-61499-098-7-750.
URL https://doi.org/10.3233/978-1-61499-098-7-750

[72] B. Fazzinga, S. Flesca, F. Parisi, On the complexity of probabilistic abstract argumentation, in: F. Rossi (Ed.), IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013, IJCAI/AAAI, 2013, pp. 898–904.
URL http://www.aaai.org/ocs/index.php/IJCAI/IJCAI13/paper/view/6563

Received August 2025

Introspective Revision of Structured Belief Bases with Argumentative Inference

Marcelo A. Falappa Alejandro J. García
Guillermo R. Simari
Department of Computer Science and Engineering
Universidad Nacional del Sur
Institute of Computer Science and Engineering
Consejo Nacional de Investigaciones Científicas y Técnicas
Bahía Blanca - ARGENTINA [*]

Abstract

This work addresses the interaction between two significant areas of research in Knowledge Representation and Reasoning: *Belief Revision* and *Argumentation*. Both areas focus on determining the validity of the beliefs within an agent's knowledge base. In belief revision, the emphasis is on modifying beliefs to sustain a coherent knowledge base for the agent. This process generally involves incorporating new information while eliminating existing beliefs. Conversely, argumentation research focuses on assessing the epistemic state of the beliefs within that knowledge base. The decision to accept a belief is made after evaluating all arguments both in favor of and against the claim. Each area contributes tools that model human commonsense reasoning as applied in everyday situations. In recent years, efforts have been made to formalize the interaction between these two areas by applying belief revision strategies to argumentative systems and utilizing formal argumentation techniques to guide change processes in belief revision. One such approach employs stratified belief bases, enhanced with argumentative inference mechanisms. Here, a definition is proposed for the introspective revision of stratified belief bases. The concept involves the introduction of change revision operators that can be applied to a stratified belief base. These operators are designed to either eliminate conflicting arguments or modify certain beliefs to accommodate new beliefs, resulting in a change in the status of specific arguments.

[*] This work is partially supported by Universidad Nacional del Sur (UNS, PGI 24/N055) and Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET, PIP 11220220100171CO) from Argentina.

Vol. 12 No. 6 2025
Journal of Applied Logics — IfCoLog Journal of Logics and their Applications

1 Introduction

Belief Revision is a dynamic process in which an agent reevaluates and adjusts its collection of stored beliefs in response to new information that challenges the current understanding, driving the agent to reflect, adapt, and possibly reshape its beliefs to maintain coherence, consistency, and accuracy in its knowledge base. The investigation of how agents revise their beliefs has been carried out in Philosophy, seeking to clarify various aspects of the very idea of changing beliefs, and in Logic, attempting to formalize different perspectives and delve into the intricacies of how agents adjust their beliefs in light of new evidence. Computer Science offers a new perspective, with the possibility of implementing intelligent agents within the domain of Artificial Intelligence, where the principles of belief revision find practical applications and innovations. This offers a fresh view on specific aspects of knowledge representation and reasoning, as well as databases. *Argumentation* primarily focuses on the process of deciding which claims supported by arguments that are obtained by reasoning from accepted premises can be accepted. This process develops in the context of disagreement, where the dynamics of support and interference among competing arguments are examined. When autonomous agents are involved, Belief Revision refers to the process by which an agent updates its beliefs when new information is received from other agents or changes in the environment are detected, aiming to restore the consistency of the agent's knowledge base. On the other hand, Argumentation focuses on how an agent derives its current beliefs from a potentially incomplete or inconsistent knowledge base, while ensuring that these beliefs remain consistent.

Comparing Argumentation and Belief Revision using the standard AGM approach is a tempting strategy that promises to unlock significant insights into both frameworks. This possible approach would not only enhance the understanding of their complexities but also reveal the fundamental principles that shape these theories, offering a valuable perspective. The AGM formalization of belief change, through *partial meet contractions*, and its variants: *safe contractions* [2], *epistemic entrenchment* [27], *sphere systems* [28], and *kernel contractions* [29], has been highly influential in the field of belief change and the dynamics of knowledge. No other research has had as significant an impact on the development of belief revision theory. The AGM postulates [1, 25] provide a clear framework that clarifies fundamental concepts; in particular, the use of beliefs sets, *i.e.*, deductively closed sets of formulas, is one of the basic elements of the standard AGM theory.

However, the use of belief sets is not effective as a formal method for developing an argumentation theory because the basic reasoning steps are abstracted away eliminating the crucial distinction between explicit and implicit beliefs. Identifying the implicit beliefs represented by an argument is the main focus of a formal argumentation theory that analyzes the structure of arguments. Therefore, the elements present in the formalisms used for belief base revision [31, 22] and in iterated epistemic change [13, 33, 34, 14] provide a richer and more suitable foundation for our research, allowing us to obtain deeper insights into the change processes at all levels.

The relationship between belief revision and argumentation was examined by Doyle in his influential work on reason maintenance [15]. This analysis was later expanded in another publication [16], where the similarities and differences between the two concepts were explored in depth. In this work, Doyle contrasted the *foundations* approach with the *coherence* approach. The paper by Doyle on *Truth Maintenance System*, or *TMS*, is an example of the first, and AGM theory provides a grounding of the last. Doyle's analysis underscores the parallels observed in the approaches, suggesting that a more productive perspective on the issue is to concentrate on the significant and fundamental similarities rather than on the seemingly minor differences that may exist. While going further into the philosophical debate surrounding these two approaches to representing and revising beliefs is beyond the scope of this paper, it is nevertheless worth observing the ways in which these perspectives have mutually enriched each other. Each viewpoint brings its own unique insights, creating a dynamic interplay that deepens our understanding of belief systems [26].

Another research effort [37], is dedicated to uncovering the complex connections and relationships between belief revision and nonmonotonic inference. This route was further pursued in subsequent works, particularly in [35], where the focus shifted to the dynamic nature of iterative change operations. This perspective is encapsulated in the concept known as *Belief Revision as Defeasible Inference* (BRDI) [17], which seeks to reveal the complexities of how beliefs evolve in response to new information. In analyzing both areas, a significant observation is that belief revision theories primarily operate at an abstract level. These theories are characterized by a set of postulates that outline the criteria for *what* is deemed a valid inference. Essentially, they provide a framework for evaluating the soundness of belief changes without identifying the processes involved, leaving open the question of how the *construction* of a concrete revision operator is, in fact, realized. On the other hand, argumentation provides an approach that concentrates on the methods through which conclusions are derived. It emphasizes the importance of clarifying the reasoning behind beliefs, offering detailed explanations and justifications for why one belief is favored over another. This focus not only enhances the transparency of reasoning processes but also facilitates a deeper understanding of the complexities surrounding belief formation and revision.

A method for integrating belief revision and argumentation was introduced in [23], offering a framework for improving our understanding of these interconnected concepts. The knowledge representation language introduced offers the possibility of introducing *undefeasible* beliefs, represented as K, and *defeasible* beliefs, represented as Δ. This framework proposes the reuse of rejected, undefeated conditionals from K by modifying the relationship between their components. An undefeasible conditional $\alpha \rightarrow \beta$, which represents a strict relationship between its antecedent and consequent, is not just discarded from K during a change. Instead, it is transformed into a *defeasible conditional* $\alpha \rightsquigarrow \beta$ and stored in the set Δ of defeasible beliefs.

A stratified belief base is a knowledge base where beliefs stored in it are assigned specific values, generating strata such that each stratum contains all the beliefs that share the same evaluation. The concept of value in this work will remain an abstract notion, providing a natural way to label each stratum. The set of labels is assumed to have a total order; furthermore, and characterizing our approach, each stratum is required to be internally consistent. It is important to note that there may be direct inconsistencies between the information stored in different layers. Inconsistency among potential beliefs will be addressed by the reasoning mechanism we introduce, resulting in a consistent set of actual beliefs. A different, paraconsistent approach was taken by Benferhat *et al* in [8, 9, 10],and we will further discuss this contribution later (see Section 5 for brief details.)

A stratified belief base offers a solid foundation for an intelligent agent's epistemic state, similar to structured belief bases [24]. This epistemic state will consist of explicit beliefs, meaning those stored in the stratified belief base, and implicit beliefs, which can be inferred from the explicit beliefs. To derive implicit beliefs from such bases, we will use an argumentation system approach based on Dung's style semantics [18]. Since each stratum must be consistent, a mechanism must be developed to revise stratified belief bases when conflicting information is attempted to be added to the same stratum. The decision to maintain consistency within each stratum forces a rational agent resolve conflicts before drawing inferences from its stratified epistemic base. Our goal is to broaden the concept of one-level reuse of beliefs as demonstrated in [23] by developing a multi-level framework. In this framework, beliefs that are discarded from one level during the revision process will serve as input for revising the next, less valued level. Beliefs are not entirely abandoned but can instead be maintained in a weakened state. We will illustrate how this reuse is significant for the formation of beliefs and how argumentative inferences are altered following changes.

This paper joins the research effort aimed to understand the connections between Belief Revision and Argumentation by offering the following contributions. We will propose a new kind of change operator, called introspective revision, that makes some specific change in a stratified belief base. We will propose a set of postulates for this new operator, a possible construction, a we will show the behaviour of this operator giving several examples and adding a proposition.

1.1 Overview

This article is organized as follows. Section 2 contains some preliminaries and notation about stratified belief bases. Section 3 presents a mechanism of warranted inference through argumentation. Section 4 introduces the idea of introspective revision and some postulates. Finally, Section 5 provides concluding remarks and outlines directions for future work.

2 Stratified Belief Bases: Preliminaries and Notation

In classical belief revision models, some sentences that are inconsistent with the epistemic input are fully discarded. This decision seems to be going against the idea that is good to preserve as much old information as possible. This criterion is known as the *principle of minimal change* [25] in the related literature. The multi-level reuse of beliefs offers the opportunity of taking those sentences that are eliminated by the revision process in one stratum and using them to revise the next less valued stratum. This reuse mechanism avoids the complete loss of that information, offering the advantage of having a *dynamic classification of beliefs*, that is, beliefs are dynamically classified using their value. The process of dynamic classification has been frequently used in the evolution of human knowledge. For instance, the belief establishing that all metals are solid under normal conditions of pressure and temperature was maximally reliable, *i.e.*, having the highest value, for centuries. However, at some point in history, it was discovered that mercury is a metal that is in liquid state under afore said conditions, and the discovery effected a change in the belief describing that property of metals. Many other beliefs have changed in similar manner throughout the evolution of Science and undoubtedly will continue happening.

In this paper we will adopt a propositional language $\mathcal{L}$ with a complete set of boolean connectives: $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$. Formulæ in $\mathcal{L}$ will be denoted by lowercase Greek letters: $\alpha, \beta, \delta, \dots, \omega$. Sets of sentences in $\mathcal{L}$ will be denoted by uppercase Latin letters: $A, B, C, \dots, Z$. The symbol $\top$ represents a tautology or *truth*, and the symbol $\perp$ represents a contradiction or *falsum*. The characters γ and σ will be reserved to represent selection and incision functions for change operators respectively. We use a consequence operator $Cn : 2^{\mathcal{L}} \longrightarrow 2^{\mathcal{L}}$ that takes sets of sentences in $\mathcal{L}$ and produces new sets of sentences in $\mathcal{L}$. This operator satisfies *inclusion* ($A \subseteq Cn(A)$), *idempotence* ($Cn(A) = Cn(Cn(A))$), and *monotony* (if $A \subseteq B$ then

$Cn(A) \subseteq Cn(B)$). It is assumed that the consequence operator includes classical consequences and verifies the standard properties of *supraclassicality* (if α can be derived from A by deduction in classical logic, then $\alpha \in Cn(A)$), *deduction* ($\beta \in Cn(A \cup \{\alpha\})$ if and only if $(\alpha \rightarrow \beta) \in Cn(A)$) and *compactness* (if $\alpha \in Cn(A)$ then $\alpha \in Cn(A')$ for some finite subset A' of A). In general, we will write $\alpha \in Cn(A)$ as $A \vdash \alpha$.

In our approach, an agent's epistemic state will be represented as a stratified belief base [21, 20, 19]. In such base, each stratum will contain a set of sentences with the same value and each stratum will be assumed consistent. This decision contrast with other approaches, such as [8, 9] where a form of argumentation is used as a general inference mechanism from a (potentially) inconsistent stratified belief base rendering a paraconsistent approach.

Definition 1. $\Sigma = (\Sigma_0, \ldots, \Sigma_n)$ *is a* stratified belief base (SBB) *if and only if* $\Sigma_i \subseteq \mathcal{L}$ *and* Σ_i *is finite and consistent for all* $0 \leq i \leq n$*, and all the sentences in* Σ_i *have the same value. The strata are considered to be ordered in the following way: the beliefs of* Σ_j *have a higher assigned value than the ones in* Σ_i *when* $j < i$*. Sometimes, slightly abusing the notation,* $x \in \Sigma$ *will mean that* $x \in \Sigma_i$ *for some* $\Sigma_i \in \Sigma$*,* $0 \leq i \leq n$*.*

It seems a good design decision for a rational agent, faced with contradictory beliefs of different levels, should opt to believe only the stronger of them disregarding the weaker. Such agent will reserve the first stratum Σ_0 for proper beliefs, while the rest of the layers refer to information that: (i) it has assigned a lower level than the information in Σ_0, (ii) it is believed only when it does not contradict beliefs that are in Σ_0 or that are contained in a stratum of higher level, and (iii) even when it is believed, it will remain more provisional than any belief in Σ_0. Therefore, in our approach, stratified belief bases are largely composed of information that is not fully believed, and the stratification is produced by the perceived value of that information. This characterization differs from other approaches that consider that beliefs have a certain strength or entrenchment [27, 36, 40].

It is important to note that our approach differs from the research presented in [8, 9]. In these works, our abstract notion of value is grounded as reliability and each stratum can be inconsistent. Moreover, the reliability assigned to the elements in the belief base cannot be modified. Our approach considers consistent strata and will allow to change the level of beliefs as a consequence of a revision.

3 Warranted inference through argumentation

Stratified belief bases (sbb) are collections of sentences with different values attached, possibly containing multiple copies of sentences in different strata, each stratum being consistent, and the union of all the strata could be inconsistent, *i.e.*, sentences in one stratum can contradict sentences in a different stratum. To make good use of this information in selecting the implicit beliefs, a mechanism with the ability of prioritizing the beliefs with higher value and with the capability of solving conflicts is needed; argumentation frameworks were developed for that purpose.

In this work, we introduce a formalism that defines an argumentation framework that allows to entail warranted conclusions from a stratified belief base. The use of argumentation will introduce the possibility of pondering arguments for and against a given conclusion and extending the analysis to interference occurring in the intermediate steps of the reasoning process, *i.e.*, at the level of sub-arguments.

3.1 Arguments from stratified belief bases

Arguments are pieces of reasoning that support conclusions from evidence. They can have intermediate reasoning steps, and hence, can be structured with sub-arguments. Each sub-argument provides support for an intermediate conclusion. In the literature, structured arguments are assumed to be consistent, in the sense that an argument does not contain two sub-arguments that support contradictory conclusions. In other words, the set of formulas used in the reasoning to infer the conclusion is assumed to be consistent [41, 38, 11, 12]. Thus, an argument cannot be in conflict with itself, *i.e.*, cannot be self-defeating, providing a consistent explanation for the conclusion it supports.

Our definition of an argument is adapted from the notion of argument structure introduced in [41] and widely accepted in many formalizations that consider how arguments are built [39, 7]. The intuitions behind these definitions are simple. An argument for a conclusion α is a minimal set of consistent formulas from the stratified belief base Σ that entails the conclusion. Furthermore, pondering the entity that builds the argument, lets consider an agent whose beliefs are represented as Σ. For this agent, Σ_0 contains its more valued beliefs, therefore it is assumed that this stratum represents a set of indisputable beliefs; they provide the foundation for the other accepted beliefs. Following the same conceptualization, no argument should contradict the sentences in Σ_0. The definitions and remarks below formalize the above mentioned intuitions.

Definition 2. *Let $\Sigma = (\Sigma_0, \ldots, \Sigma_n)$ be a stratified belief base and α a sentence. A set of sentences A is an* argument *for α from Σ, denoted $\langle A, \alpha\rangle_\Sigma$ or simply $\langle A, \alpha\rangle$,* *if:*

1. $A \subseteq (\Sigma_0 \cup \ldots \cup \Sigma_n)$.

2. $A \cup \Sigma_0$ *is consistent.*

3. $\alpha \in Cn(A \cup \Sigma_0)$.

4. *There is no* $X \subset A$ *that satisfies conditions (2) and (3).*

Given the argument $\langle A, \alpha \rangle_\Sigma$*, the sentence* α *is called the* conclusion *of the argument, and the set* A *is called the* support *of* α*. Sometimes, when no confusion could arise, we will simplify the notation for arguments as* $\langle A, \alpha \rangle$*.*

Definition 3. *Let* Σ *be a stratified belief base. An argument* $\langle B, \beta \rangle_\Sigma$ *is a* sub-argument *of* $\langle A, \alpha \rangle_\Sigma$*, if* $B \subseteq A$*.*

Observation 1. *Given an argument* $\langle A, \alpha \rangle_\Sigma$ *built from a stratified belief base* Σ*, the condition 2 of Definition 2 determines that* $\neg\alpha \notin Cn(\Sigma_0)$*.*

Observation 2. *Let* Σ *be a stratified belief base, and let* $\langle A, \alpha \rangle_\Sigma$ *be an argument from* Σ *with sub-arguments* $\langle B, \beta \rangle_\Sigma$*, and* $\langle C, \gamma \rangle_\Sigma$*. Since* A *is consistent (Def. 2),* $B \cup C$ *is a consistent set of formulas.*

Example 1. *Consider the stratified belief base* $\Sigma = (\Sigma_0, \Sigma_1, \Sigma_2, \Sigma_3)$*, where:*

$$\Sigma_0 = \left\{ \begin{array}{c} k \\ h \\ h \rightarrow j \end{array} \right\} \quad \Sigma_1 = \left\{ \begin{array}{c} b \rightarrow c \\ e \rightarrow \neg b \\ h \rightarrow b \\ k \rightarrow \neg f \end{array} \right\} \quad \Sigma_2 = \left\{ \begin{array}{c} e \\ a \rightarrow b \\ f \rightarrow \neg c \\ h \rightarrow f \end{array} \right\} \quad \Sigma_3 = \left\{ \begin{array}{c} a \\ j \end{array} \right\}$$

The following arguments are only some of the arguments that can be obtained from Σ*:*

$\langle \{k\}, k \rangle$	
$\langle \{h\}, h \rangle$	
$\langle \{e\}, e \rangle$	
$\langle A_0, \neg e \rangle$	$A_0 = \{a, a \rightarrow b, e \rightarrow \neg b\}$
$\langle A_1, c \rangle$	$A_1 = \{a, a \rightarrow b, b \rightarrow c\}$
$\langle A_2, b \rangle$	$A_2 = \{a, a \rightarrow b\}$
$\langle A_3, \neg b \rangle$	$A_3 = \{e, e \rightarrow \neg b\}$
$\langle A_4, \neg c \rangle$	$A_4 = \{h, h \rightarrow f, f \rightarrow \neg c\}$
$\langle A_5, f \rangle$	$A_5 = \{h, h \rightarrow f\}$
$\langle A_6, b \rangle$	$A_6 = \{h, h \rightarrow b\}$
$\langle A_7, \neg f \rangle$	$A_7 = \{k, k \rightarrow \neg f\}$

As an example, note that $\langle A_2, b \rangle$ *is a sub-argument of both* $\langle A_1, c \rangle$ *and* $\langle A_0, \neg e \rangle$*, and that* $\langle A_5, f \rangle$ *is a sub-argument of* $\langle A_4, \neg c \rangle$*.*

Since arguments can contain sentences of different strata, the comparison of arguments will take account of such information. The value of a set of sentences will depend on the stratum the sentences belong to. Arguments will have a higher value if they are built with sentences of a stratum with a lower index. Since a sentence can belong to more of one stratum, the stratum with lower index (greatest value) should be chosen. Hence, the *index* of a sentence β in Σ, denoted $\mathit{index}_{\Sigma}(\beta)$, will be the index corresponding to the lowest indexed stratum – and hence most valued – in Σ to which β belongs to.

Definition 4. *Let $\Sigma = (\Sigma_0, \ldots, \Sigma_n)$ be a stratified belief base such that $\beta \in (\Sigma_0 \cup \ldots \cup \Sigma_n)$. Then, $\mathit{index}_{\Sigma}(\beta) = i$ if and only if $\beta \in \Sigma_i$ and there is no Σ_j $(j < i)$ such that $\beta \in \Sigma_j$. We say that i is the* stratum *of β.*

The stratum of an argument is therefore defined by the index of its weakest sentence.

Definition 5. *Let Σ be a stratified belief base and $A = \{x_1, x_2, \ldots, x_m\}$ be a set of sentences in Σ. The* stratum of the set A *is*

$$\mathit{str}_{\Sigma}(A) = \max\{\mathit{index}_{\Sigma}(x_1), \mathit{index}_{\Sigma}(x_2), \ldots, \mathit{index}_{\Sigma}(x_m)\}$$

Example 2. *Consider the Σ in Example 1, then:*

$\mathit{index}_{\Sigma}(k) = 0$, $\mathit{index}_{\Sigma}(j) = 3$, $\mathit{index}_{\Sigma}(h \to b) = 1$,
$\mathit{str}_{\Sigma}(\langle\{k\}, k\rangle) = 0$, $\mathit{str}_{\Sigma}(\langle\{h\}, h\rangle) = 0$, $\mathit{str}_{\Sigma}(\langle\{e\}, e\rangle) = 2$,
$\mathit{str}_{\Sigma}(\langle\{j\}, j\rangle) = 3$, $\mathit{str}_{\Sigma}(\langle\{h, h \to j\}, j\rangle) = 0$, $\mathit{str}_{\Sigma}(\langle A_1, c\rangle) = 3$,
$\mathit{str}_{\Sigma}(\langle A_2, b\rangle) = 3$, $\mathit{str}_{\Sigma}(\langle A_3, \neg b\rangle) = 2$, $\mathit{str}_{\Sigma}(\langle A_4, \neg c\rangle) = 2$,
$\mathit{str}_{\Sigma}(\langle A_5, f\rangle) = 2$, $\mathit{str}_{\Sigma}(\langle A_6, b\rangle) = 1$, $\mathit{str}_{\Sigma}(\langle A_7, \neg f\rangle) = 1$

Next, based on $\mathit{str}_{\Sigma}(\,\cdot\,)$, we introduce the order "$\succeq_{\Sigma}$" in the set of arguments that can be built from Σ. Recall that a lower number represents more value.

Definition 6. *Let $\langle B, \beta\rangle_{\Sigma}$ and $\langle A, \alpha\rangle_{\Sigma}$ be two arguments from a stratified belief base Σ. We will say that $\langle B, \beta\rangle_{\Sigma}$ is* as least as good *than $\langle A, \alpha\rangle_{\Sigma}$, denoted as $\langle B, \beta\rangle_{\Sigma} \succeq_{\Sigma} \langle A, \alpha\rangle_{\Sigma}$, if and only if $\mathit{str}_{\Sigma}(B) \leq \mathit{str}_{\Sigma}(A)$.*

The following proposition establishes that a sub-argument cannot be weaker than any of its super-arguments [19].

Proposition 1. *Let Σ be a stratified belief base, if $\langle B, \beta\rangle_{\Sigma}$ is a sub-argument of $\langle A, \alpha\rangle_{\Sigma}$ then $\langle B, \beta\rangle_{\Sigma} \succeq_{\Sigma} \langle A, \alpha\rangle_{\Sigma}$.*

3.2 Conflict and Defeat relations

Two arguments that support contradictory conclusions clearly interfere with each other; this will be considered as a relation of *conflict*. For instance, the arguments $\langle A_2, b\rangle$ and $\langle A_3, \neg b\rangle$ from Example 1 are in conflict. Furthermore, given two arguments A and B that are in conflict, such as $B \succeq_\Sigma A$, it is natural to establish that the interference of B over A is successful; this introduces the notion of *defeat*.

Defeat can be effected in different ways. In presenting the idea, we have introduced a particular type of defeat where the attack is aimed at the conclusion, this form of defeat is called *rebuttal*; other kinds of defeat exists leading to a richer relation. When the attack is indirect, *i.e.*, being aimed at an intermediate step in the reasoning, the defeat is called *undercut*. In this work, undercuts will be accomplished by the attack of one argument to a sub-argument of another argument, that is, it is a rebuttal over a sub-argument of the attacked argument. There is another form of attack, called *assumption attack*. Given that we will not use assumptions in our formalism, this latter form of attack will not appear in our presentation. The formal definitions of conflict and defeat are introduced next.

Definition 7. *Let Σ be a stratified belief base. Two arguments $\langle A, \alpha\rangle_\Sigma$ and $\langle B, \beta\rangle_\Sigma$ are* in conflict*, if $\Sigma_0 \cup \{\alpha, \beta\}$ is an inconsistent set.*

We would like to discuss the particular use of the stratum Σ_0 as context to analyze the inconsistency of the conclusions of two arguments. The Σ_0 stratum is a set of consistent beliefs with maximum value and any argument, by its construction, should be consistent with it. Since each conclusion by itself is consistent with Σ_0, the conclusions must be contradictory with each other, possibly through the use of an appropriated subset of the beliefs in Σ_0. The example below shows the general case.

Example 3. *Consider the stratified belief base $\Sigma = (\Sigma_0, \Sigma_1)$, where:*

$$\Sigma_0 = \left\{ \begin{array}{c} x \\ y \\ a \rightarrow c \\ b \rightarrow \neg c \end{array} \right\} \quad \Sigma_1 = \left\{ \begin{array}{c} x \rightarrow a \\ y \rightarrow b \end{array} \right\}$$

The arguments $\langle A, a\rangle$ with $A = \{x, x \rightarrow a\}$ and $\langle B, b\rangle$ with $B = \{y, y \rightarrow b\}$ are in conflict even though their conclusions are not in direct contradiction.

Definition 8. *Let Σ be a stratified belief base. An argument $\langle B, \beta\rangle_\Sigma$* defeats *$\langle A, \alpha\rangle_\Sigma$, if there exists a sub-argument $\langle C, \gamma\rangle_\Sigma$ of $\langle A, \alpha\rangle_\Sigma$, such that the following two conditions hold:*

- *$\langle B, \beta\rangle_\Sigma$ and $\langle C, \gamma\rangle_\Sigma$ are in conflict, and*
- *$\langle B, \beta\rangle_\Sigma \succeq_\Sigma \langle C, \gamma\rangle_\Sigma$.*

We will say that $\langle B, \beta\rangle_\Sigma$ defeats $\langle A, \alpha\rangle_\Sigma$ at sub-argument $\langle C, \gamma\rangle_\Sigma$.

Example 4. *Consider the stratified belief base of Example 1. Argument $\langle A_3, \neg b\rangle$ defeats $\langle A_1, c\rangle$ (at the sub-argument $\langle A_2, b\rangle$), and $\langle A_6, b\rangle$ defeats $\langle A_3, \neg b\rangle$. Argument $\langle A_4, \neg c\rangle$ defeats $\langle A_1, c\rangle$ and $\langle A_7, \neg f\rangle$ defeats $\langle A_4, \neg c\rangle$ (at the sub-argument $\langle A_5, f\rangle$).*

Suppose the existence of two arguments $\langle C, \gamma\rangle_\Sigma$ and $\langle A, \alpha\rangle_\Sigma$ that are in conflict. Suppose that $str_\Sigma(C) = str_\Sigma(A)$. Then it holds that $\langle C, \gamma\rangle_\Sigma \succeq_\Sigma \langle A, \alpha\rangle_\Sigma$ and also that $\langle A, \alpha\rangle_\Sigma \succeq_\Sigma \langle C, \gamma\rangle_\Sigma$. Hence, in this particular case, it holds that $\langle C, \gamma\rangle_\Sigma$ defeats $\langle A, \alpha\rangle_\Sigma$ and $\langle A, \alpha\rangle_\Sigma$ defeats $\langle C, \gamma\rangle_\Sigma$.

As a consequence of the way the comparison criterion "$\succeq_\Sigma$" is defined, if B defeats A, then B will also defeat every super-argument of A, that is, B also defeats every argument C such that A is a sub-argument of C (see Proposition 1) [19].

Observation 3. *Let Σ be a stratified belief base. If $\langle B, \beta\rangle_\Sigma$ is a sub-argument of $\langle A, \alpha\rangle_\Sigma$ and $\langle C, \gamma\rangle_\Sigma$ is a sub-argument of $\langle A, \alpha\rangle_\Sigma$, then $\langle B, \beta\rangle_\Sigma$ cannot defeat $\langle C, \gamma\rangle_\Sigma$ and vice versa. (see Observation 2).*

Proposition 2. *Let $\langle A, \alpha\rangle_\Sigma$ be an argument. If $A \subseteq \Sigma_0$ then there is no argument $\langle B, \beta\rangle$ such that $\langle B, \beta\rangle_\Sigma$ defeats $\langle A, \alpha\rangle_\Sigma$.*

Corollary 1. *Given an argument $\langle A, \alpha\rangle_\Sigma$. If $str_\Sigma(A) = 0$ then there exists no argument that can defeat $\langle A, \alpha\rangle_\Sigma$.*

3.3 Argumentative Inference

The argumentation formalism introduced above allows us to build the set of arguments that will be involved in the process of deciding what the implicit beliefs are. In argumentation terminology this is the set of beliefs supported by warranted arguments. For obtaining these beliefs, we will make use of the semantics defined for abstract argumentation frameworks.

An argumentation framework [18] is a pair $\langle \mathfrak{Args}, \mathbf{R}\rangle$, where $\mathfrak{Args}$ is a finite set of arguments and $\mathbf{R}$ is a binary relation between arguments such that $\mathbf{R} \subseteq \mathfrak{Args} \times \mathfrak{Args}$. The notation $(A, B) \in \mathbf{R}$ (or, equivalently, $A\,\mathbf{R}\,B$) means that A *attacks* B. It is interesting to note that the Dung's relation called attack in his original paper corresponds to the notion of defeat introduced in this work, *i.e.*, in Dung's formalism every attack was successful, therefore we will consistently use defeat instead of attack.

An argumentation semantics is the formal definition of the argument evaluation process leading to decide which arguments are able to survive the attacks defined in the framework. These "survivors" will be considered as being able to support their conclusions. The research has produced two different ways of carrying out the evaluation, namely extension-based and labeling-based argumentation semantics. The first one provides a declarative definition, meanwhile the second one is procedural in nature. An extension is a subset of arguments contained in the framework, and the extension-based approach specifies how to obtain the subsets that form the set of extensions. Every extension contains a set of arguments that together can be acceptable in the context of the attack relation. The labeling-based approach provides a way of assigning a label to each argument in the framework choosing that label from an appropriated set, such as $\{in, out, undecided\}$. The assignment of labels yields a set of labelings that correspond to the extensions found through the declarative method.

Dung [18] introduces several argumentation semantics that provide a way of evaluating the status of the arguments in the framework constructing extensions, *e.g.,* complete, grounded, stable, and preferred semantics. Other semantics have been proposed after the initial definition, *e.g.,* stage, semi-stable, ideal, $CF2$, and prudent semantics. For further details on recent developments see [4].

3.4 Semantic and Epistemic Model

Given a stratified belief base Σ and using the argumentation formalism defined above, an argumentation framework $\langle \mathfrak{Args}, \mathbf{R} \rangle$ can be obtained, and some *skeptical semantics* can be applied in order to obtain warranted conclusions. An argument will be skeptically *warranted* if and only if it is warranted in all the extensions produced by the chosen semantics. Grounded semantics is an example of skeptical semantics, and we will use it in the example below.

Example 5. *Consider again Example 1. The argumentation framework* $\langle \mathfrak{Args}, \boldsymbol{R} \rangle$ *can be obtained, where* $\{A_0, A_1, A_2, A_3, A_4, A_5, A_6, A_7\} \subseteq \mathfrak{Args}$ *and* $\boldsymbol{R}$ *contains, for instance, the following tuples:*

$$\{(A_3, A_2), (A_3, A_1), (A_4, A_1), (A_7, A_4), (A_7, A_5), (A_6, A_3)\}$$

If we choose to use grounded semantics [18], the arguments A_1*,* A_2*,* A_6 *and* A_7 *are warranted, and so all of* b, c*, and* $\neg f$ *can be inferred from our stratified belief base, among others.*

Note that the particular semantics chosen for obtaining the warranted arguments is a decision that will affect the behavior of the system. For more details regarding different aspects of skeptical semantics in abstract argumentation frameworks see [3, 5]. The reader may obtain a general perspective on argumentation in general and argumentation semantics in particular in [7], or in [11, 39] where deeper and detailed accounts are presented.

Given the definition of warrant, we are now in a position to determine what the epistemic attitude of a Σ is with respect to a sentence α:

- **In/Accepted**: when α is warranted by Σ.
- **Out/Rejected**: when $\neg\alpha$ is warranted by Σ.
- **Undecided**: when none of the above cases occur.

Thus, we can determine the conclusions that are accepted, rejected, or undecided in a Σ. Given a stratified belief base Σ, we define the following sets:

- Accepted(Σ) is the set of sentences accepted in Σ.
- Rejected(Σ) is the set of sentences rejected in Σ.
- Undecided(Σ) is the set of sentences undecided in Σ.

4 Introspective Revision

Belief revision systems are supposed to operate in dynamic environments with constant change. While the most prominent belief revision models, such as the operators of partial meet contraction/revision [1] or the operators of erasure/updating [32], do not allow iterative changes, it is impossible to think of real-world applications of these systems that would allow for a single change. For this reason, multiple iterative change systems have been developed, including...

Analyzing the existing literature, there are not many systems for changing stratified belief bases. In [19] is proposed a multiple revision operator that takes a stratified belief base Σ, a consistent set A and a level i and it produces a new stratified belief base Σ' where A is prioritized revised at the level i. Given that the formalism presented here presents a new epistemic model and contains a powerful argument-based inference mechanism, we believe it is appropriate to define a new form of change, which we call *introspective revision.*

Assuming that a real system can support tens, hundreds, or thousands of changes, it is possible for a stratified database to grow in both the number of beliefs and the number of strata. This growth may lead to an excess of information, affecting the quality of the conclusions obtained. Therefore, the introspective belief revision will be an operation similar to the consolidation operator proposed by Hansson in [30]. Rather than removing inconsistencies from a singular belief bases, the intention is to remove potential disagreements or conflicts in sentences of different strata that are part of arguments, mainly affecting the undecided sentences of the stratified belief base.

4.1 Postulates

We will note $\Sigma\odot$ for the introspective revision of Σ. We propose the following postulates for introspective revision:

Inclusion: $\Sigma\odot \subseteq \Sigma$.

Positive Preservation: $\text{Accepted}(\Sigma) \subseteq \text{Accepted}(\Sigma\odot)$.

Negative Preservation: $\text{Rejected}(\Sigma) \subseteq \text{Rejected}(\Sigma\odot)$.

Vacuity: If $\text{Undecided}(\Sigma) = \varnothing$ then $\Sigma\odot = \Sigma$.

Weak Coherence: $\text{Undecided}(\Sigma\odot) \subseteq \text{Undecided}(\Sigma)$.

Strong Coherence: $\text{Undecided}(\Sigma\odot) = \varnothing$.

Core Retainment: If $\alpha \in \Sigma$ and $\alpha \notin \Sigma\odot$ then there exits some argument $\langle B, \beta\rangle_\Sigma$ such that $\alpha \in B$ and $\beta \in \text{Undecided}(\Sigma)$.

Inclusion states that introspective revision just eliminates sentences. Nevertheless, *Positive* and *Negative Preservation* states the contrary, that is, accepted and rejected conclusions should not decrease. *Vacuit*y states that introspective revision has no effect if there are no undecided conclusions. Weak and Strong Coherence are special ways of consistency. In a stratified belief base, each stratum is consistent, but inconsistencies may exist among sentences from different strata. Since an argument can be constructed with sentences from all strata, *Weak Coherences* means that inconsistencies will only be eliminated between some arguments in conflict where there is no warranted argument. *Strong Coherences* means that inconsistencies will only be eliminated between all arguments in conflict where there is no warranted argument. *Core Retainment* means that a sentence α is removed in an introspective revision if α is part of and argument $\langle B, \beta\rangle_\Sigma$ such that $\beta \in \text{Undecided}(\Sigma)$. Although the inclusion postulate states that introspective revision only eliminates sentences, note that eliminating sentences eliminates conflicting arguments and hence, that may increase the number of accepted or rejected sentences in a stratified belief base.

4.2 Construction

We will now present a construction for an introspective revision operator. Every stratum of a stratified belief base is consistent by definition; therefore, if there is a conflict, it must involve sentences from different strata.

Example 6. *Consider the stratified belief base $\Sigma = (\Sigma_0, \Sigma_1, \Sigma_2)$, where:*

$$\Sigma_0 = \left\{ \begin{array}{c} d \\ e \\ a \rightarrow c \\ b \rightarrow \neg c \end{array} \right\} \quad \Sigma_1 = \left\{ \begin{array}{c} x \\ y \rightarrow b \end{array} \right\} \quad \Sigma_2 = \left\{ \begin{array}{c} x \rightarrow a \\ y \end{array} \right\}$$

The arguments $\langle A, a \rangle$, with $A = \{x, x \rightarrow a\}$, and $\langle B, b \rangle$, with $B = \{y, y \rightarrow b\}$, are in conflict and they are built with sentences of Σ_1 and Σ_2.

Since Σ_0 contains the more valued beliefs, we will consider an introspective revision operator that will not modify beliefs in this stratum. From Definitions 1 (Stratified Belief Base) and 7 (Conflict Between Arguments), it is easy to show that:

- Every stratum Σ_i is consistent.
- If there are two arguments A and B in conflict, they are built by sentences of at least two different strata i and j, with $0 < i$, $0 < j$, and $i \neq j$.

What are we trying to do with introspective revision? To eliminate incomparable conflicting arguments, that is arguments where is not possible to establish a preference. In [29], Hansson introduced the concept of the α-kernel, which is defined as the collection of minimal subsets from an arbitrary set of sentences that imply α. Building on the definition of kernels, we can informally define an introspective revision operator through the following sequence of actions:

1. Considering the union of all strata in the SBB, find in that set the $\perp$-kernels.
2. Eliminate those $\perp$-kernels that include incomparable arguments.
3. Cut each one of the remaining $\perp$-kernels with an incision function, protecting Σ_0.

Definition 9. *Let be Σ a stratified belief base and α be a sentence. Then we define the α-*extended-kernels *of Σ, noted as $\Sigma \perp\!\!\!\perp \alpha$, to the set of sets K such that:*

1. $K \subseteq \cup_{i>0} \Sigma_i$.

2. $K \cup \Sigma_0 \vdash \alpha$.

3. There is no K' *such that* $K' \subset K$ *and* $K' \cup \Sigma_0 \vdash \alpha$.

Every set K *is an* α*-extended-kernel if it is a minimal subset of* $\cup_{i>0}\Sigma_i$ *such that* K *with* Σ_0 *implies* α.

If $\alpha = \perp$ then a $\perp$-extended-kernel is a minimal inconsistent subset of $\cup_{i>0}\Sigma_i$ with Σ_0. The index of each sentence is important for determining the cut of each kernel. Therefore, the subset of each kernel will be identified by the index of the stratum to which each sentence belongs.

Example 7. *Consider the stratified belief base* $\Sigma = (\Sigma_0, \Sigma_1, \Sigma_2)$*, where:*

$$\Sigma_0 = \left\{ \begin{array}{c} s \\ s \rightarrow t \end{array} \right\} \quad \Sigma_1 = \left\{ \begin{array}{c} p \\ p \rightarrow q \\ t \rightarrow v \end{array} \right\} \quad \Sigma_2 = \left\{ \begin{array}{c} r \\ r \rightarrow \neg q \\ s \rightarrow \neg v \end{array} \right\}$$

Then, there are two $\perp$*-extended-kernels:*

- $\{p, p \rightarrow q\}_1 \cup \{r, r \rightarrow \neg q\}_2$ *meaning that* $\{p, p \rightarrow q\}$ *belongs to stratum* 1 *and* $\{r, r \rightarrow \neg q\}$ *belongs to stratum* 2.
- $\{t \rightarrow v\}_1 \cup \{s \rightarrow \neg v\}_2$ *meaning that* $\{t \rightarrow v\}$ *belongs to stratum* 1 *and* $\{s \rightarrow \neg v\}$ *belongs to stratum* 2.

Therefore:

$$\Sigma \perp\!\!\!\perp \perp = \{\{p, p \rightarrow q\}_1 \cup \{r, r \rightarrow \neg q\}_2, \{t \rightarrow v\}_1 \cup \{s \rightarrow \neg v\}_2\}$$

Example 8. *Consider the stratified belief base* $\Sigma = (\Sigma_0, \Sigma_1, \Sigma_2)$*, where:*

$$\Sigma_0 = \left\{ \begin{array}{c} p \rightarrow q \\ r \rightarrow \neg q \end{array} \right\} \quad \Sigma_1 = \left\{ \begin{array}{c} s \\ s \rightarrow v \end{array} \right\} \quad \Sigma_2 = \left\{ \begin{array}{c} p \\ r \\ r \rightarrow \neg v \end{array} \right\}$$

Then, there are two $\perp$*-extended-kernels:*

- $\{p, r\}_2$ *meaning that* $\{p, r\}$ *belongs to stratum* 2.
- $\{s, s \rightarrow v\}_1 \cup \{r, r \rightarrow \neg v\}_2$ *meaning that* $\{s, s \rightarrow v\}$ *belongs to stratum* 1 *and* $\{r, r \rightarrow \neg v\}$ *belongs to stratum* 2.

Therefore:

$$\Sigma \perp\!\!\!\perp \bot = \{\{p, r\}_2, \{s, s \rightarrow v\}_1 \cup \{r, r \rightarrow \neg v\}_2\}$$

The next thing to do is to filter out only those $\bot$-generalized-kernels that contain uncomparable conflicting arguments.

Definition 10. *Let be Σ a stratified belief base, α be a sentence, and $\Sigma \perp\!\!\!\perp \bot$ be the set of $\bot$-extended-kernels. Then ∇ is a* filter *if it is a function that selects those $\bot$-extended-kernels that contain incomparable conflicting arguments, that is:*

1. $\nabla(\Sigma \perp\!\!\!\perp \bot) \subseteq \Sigma \perp\!\!\!\perp \bot$.
2. *$X \in \nabla(\Sigma \perp\!\!\!\perp \bot)$ if and only if X contains two incomparable arguments in conflict.*

Definition 11. *Let be Σ a stratified belief base, α be a sentence, and $\Sigma \perp\!\!\!\perp \alpha$ be the set of extended-kernels. Then σ is an* incision function *for $\Sigma \perp\!\!\!\perp \alpha$ if and only if:*

1. $\sigma(\Sigma \perp\!\!\!\perp \alpha) \subseteq \bigcup(\Sigma \perp\!\!\!\perp \alpha)$.
2. *If $X \in \Sigma \perp\!\!\!\perp \alpha$ then $X \cap (\sigma(\Sigma \perp\!\!\!\perp \alpha)) \neq \emptyset$.*

Definition 12. *Let be Σ a stratified belief base, α be a sentence, and $\Sigma \perp\!\!\!\perp \alpha$ be the set of extended-kernels. Then σ is an* fair incision function *for $\Sigma \perp\!\!\!\perp \alpha$ if it is an incision function that selects the sentences with the highest index in each extended-kernel.*

Example 9. *Consider the stratified belief base of the Example 8. There are the following sets of arguments in conflict:*

- *$\langle \{p\}, q \rangle_\Sigma$ and $\langle \{r\}, \neg q \rangle_\Sigma$ such that $str_\Sigma(\{p\}) = str_\Sigma(\{r\})$, that is, none is better than the other.*
- *$\langle \{s, s \rightarrow v\}, v \rangle_\Sigma$ and $\langle \{r, r \rightarrow \neg v\}, \neg v \rangle_\Sigma$ such that $\{s, s \rightarrow v\} \succeq_\Sigma \{r, r \rightarrow \neg v\}$.*

Analyzing these conflicting arguments, we find that the first two are incomparable and therefore indicating that the extended-kernel that includes will be filtered and it should be removed. Then, a fair incision function will select p and r since both have the highest index. Therefore, $\nabla(\Sigma \perp\!\!\!\perp \bot) = \{\{p, r\}_2\}$ and a fair incision function will select both sentences in the one $\bot$-extended kernel.

Definition 13. *Let be Σ a stratified belief base, α be a sentence, $\Sigma \perp\!\!\!\perp \bot$ be the set of $\bot$-extended-kernels, ∇ be a filter function and σ be a fair incision function. Then, the* introspective kernel revision *of Σ, denoted by $\Sigma \odot$ is defined as:*

$$\Sigma \odot = \Sigma \setminus \sigma(\nabla(\Sigma \perp\!\!\!\perp \bot))$$

Example 10. *Consider the stratified belief base of the Example 8. Besides there are two $\bot$-extended kernels, the one to be eliminated is $\{p, r\}_2$. Since the sentences have the same index, all of them are eliminated, as if we forgot about them [6]. Then, the result of introspective kernel revision is:*

$$\Sigma_0 = \left\{ \begin{array}{c} p \to q \\ r \to \neg q \end{array} \right\} \quad \Sigma_1 = \left\{ \begin{array}{c} s \\ s \to v \end{array} \right\} \quad \Sigma_2 = \left\{ \begin{array}{c} r \to \neg v \end{array} \right\}$$

Now we will present a new and complete example.

Example 11. *Consider the following stratified belief base $\Sigma = (\Sigma_0, \Sigma_1, \Sigma_2, \Sigma_3)$:*

$$\Sigma_0 = \left\{ \begin{array}{c} p \to q \\ r \to \neg q \end{array} \right\} \quad \Sigma_1 = \left\{ \begin{array}{c} s \\ s \to v \end{array} \right\} \quad \Sigma_2 = \left\{ \begin{array}{c} p \\ r \\ r \to \neg v \end{array} \right\} \quad \Sigma_3 = \left\{ \begin{array}{c} p \\ q \to \neg t \end{array} \right\}$$

Then, there are three $\bot$-extended-kernels:

- $\Sigma_A = \{p, r\}_2$ *meaning that* $\{p, r\}$ *belongs to stratum* 2.
- $\Sigma_B = \{r\}_2 \cup \{p\}_3$ *meaning that* $\{r\}$ *belongs to stratum* 2 *and* $\{p\}$ *belongs to stratum* 3.
- $\Sigma_C = \{s, s \to v\}_1 \cup \{r, r \to \neg v\}_2$ *meaning that* $\{s, s \to v\}$ *belongs to stratum* 1 *and* $\{r, r \to \neg v\}$ *belongs to stratum* 2.

There are the following sets of arguments in conflict:

- $\langle \{p\}, q \rangle_\Sigma$ *and* $\langle \{r\}, \neg q \rangle_\Sigma \in \Sigma_A$ *with* $str_\Sigma(2)$ *such that none is better than the other.*
- $\langle \{p\}, q \rangle_\Sigma$ *and* $\langle \{r\}, \neg q \rangle_\Sigma \in \Sigma_B$ *such that* $\{r\} \succeq_\Sigma \{q\}$.
- $\langle \{s, s \to v\}, v \rangle_\Sigma$ *and* $\langle \{r, r \to \neg v\}, \neg v \rangle_\Sigma \in \Sigma_C$ *such that* $\{s, s \to v\} \succeq_\Sigma \{r, r \to \neg v\}$.

From the three $\bot$-extended kernels ($\Sigma_A, \Sigma_B, \Sigma_C$), Σ_A is the one with incomparable arguments; therefore, it will be selected by the filter function. Then, assuming a fair incision function a possible result of the introspective revision $\Sigma \odot = (\Sigma'_0, \Sigma'_1, \Sigma'_2, \Sigma'_3)$ will be:

$$\Sigma'_0 = \left\{ \begin{array}{c} p \to q \\ r \to \neg q \end{array} \right\} \quad \Sigma'_1 = \left\{ \begin{array}{c} s \\ s \to v \end{array} \right\} \quad \Sigma'_2 = \left\{ \begin{array}{c} r \to \neg v \end{array} \right\} \quad \Sigma'_3 = \left\{ \begin{array}{c} p \\ q \to \neg t \end{array} \right\}$$

Note that the arguments $\langle\{p\}, q\rangle_\Sigma$ and $\langle\{r\}, \neg q\rangle_\Sigma$ are in conflict in Σ and there is no conclusion about q: it is neither accepted nor rejected. However, q is accepted in $\Sigma\odot$ because there exists the argument $\langle\{p\}, q\rangle_{\Sigma\odot}$.

Proposition 3. *The operator of* instrospective kernel revision *satisfies* inclusion*,* positive preservation*,* negative preservation*,* vacuity*,* strong coherence *and* core retainment*.*

Proof*. Consider de Definition 13 of the operator of Introspective Kernel Revision:*

- *Inclusion:* $\Sigma\odot \subseteq \Sigma$*. Trivial by definition:*

- *Positive Preservation:* $\mathrm{Accepted}(\Sigma) \subseteq \mathrm{Accepted}(\Sigma\odot)$*. Let* α *be a sentence such that* α *is accepted in* Σ*. Then, there is an warranted argument* A *for* α *in* Σ*. If* A *is warranted then there is no argument* B *in conflict with* A *such that* A *and* B *are incomparable. Therefore, the filter function will not select those* $\perp$*-extended-kernels containing* A *and therefore* α *will be accepted in* $\Sigma\odot$*.*

- *Negative Preservation:* $\mathrm{Rejected}(\Sigma) \subseteq \mathrm{Rejected}(\Sigma\odot)$*. Let* α *be a sentence such that* α *is rejected in* Σ*. Then, there is an warranted argument* A *for* $\neg\alpha$ *in* Σ*. If* A *is warranted then there is no argument* B *in conflict with* A *such that* A *and* B *are incomparable. Therefore, the filter function will not select those* $\perp$*-extended-kernels containing* A *and therefore* α *will be rejected in* $\Sigma\odot$*.*

- *Vacuitty: If* $\mathrm{Undecided}(\Sigma) = \varnothing$ *then* $\Sigma\odot = \Sigma$*. Trivial by definition.*

- *Strong Coherence:* $\mathrm{Undecided}(\Sigma\odot) = \varnothing$*. Suppose that there exists some literal* α *such that* $\alpha \in \mathrm{Undecided}(\Sigma\odot)$*. Therefore, there exists some argument* A *for* α *such that* A *is not warranted. Then, there exists some* $\perp$*-extended kernel in* $\Sigma\odot$*, for instance* H*, such that* $H \vdash \perp$ *and* $A \subseteq H$*. But this is absurd because all the* $\perp$*-extended-kernels of* $\Sigma\odot$ *are* $\perp$*-extended-kernels of* Σ *and they are selected by the filter function and therefore they are cut by the incision function.*

- *Core Retainment: If* $\alpha \in \Sigma$ *and* $\alpha \notin \Sigma\odot$ *then there exits some argument* $\langle B, \beta\rangle_\Sigma$ *such that* $\alpha \in B$ *and* $\beta \in \mathrm{Undecided}(\Sigma)$*. If* $\alpha \in \Sigma$ *and* $\alpha \notin \Sigma\odot$ *then* α *is cut by the incision function and* α *belongs to some minimal inconsistent set* H *that is a* $\perp$*-extended-kernel filtered by the filter function. Then,* H *contains incomparable conflicting arguments, for instance,* $\langle A, \beta\rangle_\Sigma$ *and* $\langle B, \delta\rangle_\Sigma$*,* $\beta, \delta \in \mathrm{Undecided}(\Sigma)$ *and* $\alpha \in A$ *or* $\alpha \in B$*.*

5 Conclusion and Future Work

To our knowledge, there are very few similar works that combine knowledge representation using stratified belief bases, an argumentative inference mechanism, and knowledge dynamics. The formalism we have introduced leads to a process known as *introspective revision*. This process involves a systematic examination of various layers (strata) of beliefs to identify any that may obstruct or create uncertainty in answering specific questions.

Another key aspect of an introspective revision is that no additional information or preferences are required to make the change effective. Change involves identifying conflicts and working to resolve them by leveraging the strength of beliefs, as indicated by the index. By definition, every stratum is consistent. Therefore, each $\perp$-extended kernel, which is a minimally inconsistent set, must consist of two or more strata. To resolve contradictions, it is enough to remove the sentences belonging to the weakest stratum, which is the one with the highest index. This approach ensures rational change operations since no $\perp$-extended kernel is eliminated in the process. This outcome would occur with a more extreme operation, such as a complete meet contraction (by falsum).

Following this line of research, we intend to establish comprehensive representation theorems for the operator we have introduced. This will involve an exploration into the development of constructions based on remainder sets, which will be key in that work. As a result of these investigations, we can speculate on the formulation of some form of introspective partial meet revision operators. These operators will not only improve our theoretical understanding but also provide practical frameworks for applications in various decision-making processes.

Furthermore, we will undertake a comprehensive analysis and comparative study between the introspective partial meet revision operators and the introspective kernel revision methods. Our primary goal will be to carefully examine the arguments introduced in both approaches. We will consider both the accepted conclusions and those that have been rejected. This will enable a deeper understanding of the strengths and weaknesses inherent in these revision strategies.

References

[1] Carlos Alchourrón, Peter Gärdenfors, and David Makinson. On the Logic of Theory Change: Partial Meet Contraction and Revision Functions. *The Journal of Symbolic Logic*, 50:510–530, 1985.

[2] Carlos Alchourrón and David Makinson. On the Logic of Theory Change: Safe Contraction. *Studia Logica*, 44:405–422, 1985.

[3] Pietro Baroni and Massimiliano Giacomin. Comparing argumentation semantics with respect to skepticism. In *9th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty, (ECSQARU 2007), Hammamet, Tunisia*, pages 210–221, 2007.

[4] Pietro Baroni and Massimiliano Giacomin. Semantics of abstract argument systems. In Iyad Rahwan and Guillermo Ricardo Simari, editors, *Argumentation in Artificial Intelligence*, pages 24–44. Springer, 2009.

[5] Pietro Baroni and Massimiliano Giacomin. Skepticism relations for comparing argumentation semantics. *International Journal Approximate Reasoning*, 50(6):854–866, 2009.

[6] Ringo Baumann, Dov M. Gabbay, and Odinaldo Rodrigues. Forgetting an argument. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI, The Tenth AAAI Symposium EAAI 2020, USA*, pages 2750–2757. AAAI Press, 2020.

[7] Trevor J. M. Bench-Capon and Paul E. Dunne. Argumentation in artificial intelligence. *Artificial Intelligence*, 171(10-15):619–641, 2007.

[8] S. Benferhat, D. Dubois, and H. Prade. Argumentative inference in uncertain and inconsistent knowledge bases. In D. Heckerman and E. H. Mamdani, editors, *Proceedings of Ninth Annual Conference on Uncertainty in Artificial Intelligence, UAI 1993*, pages 411–419. Morgan Kaufmann, 1993.

[9] Salem Benferhat, Didier Dubois, and Henri Prade. How to infer from inconsistent beliefs without revising. In *Proceedings of IJCAI 1995*, pages 1449–1455, 1995.

[10] Salem Benferhat, Souhila Kaci, Daniel Le Berre, and Mary-Anne Williams. Weakening conflicting information for iterated revision and knowledge integration. *Artificial Intelligence Journal*, 153(1-2):339–371, 2004.

[11] Philippe Besnard and Anthony Hunter. A logic-based theory of deductive arguments. *Artificial Intelligence*, 128(1-2):203–235, 2001.

[12] Philippe Besnard and Anthony Hunter. Argumentation based on classical logic. In Iyad Rahwan and Guillermo Ricardo Simari, editors, *Argumentation in Artificial Intelligence*, pages 133–152. Springer, 2009.

[13] Adnan Darwiche and Judea Pearl. On the Logic of Iterated Belief Revision. *Artificial Intelligence*, 89:1–29, 1997.

[14] James Delgrande, Didier Dubois, and Jérôme Lang. Iterated revision and prioritized merging. In *In Proceedings of KR 2006*, pages 210–220. AAAI Press, 2006.

[15] Jon Doyle. A truth maintenance system. *Artificial Intelligence*, 12:231–272, 1979.

[16] Jon Doyle. Reason Maintenance and Belief Revision: Foundations versus Coherence Theories. In P. Gärdenfors, editor, *Belief Revision*, pages 29–51. Cambridge University Press, 1992.

[17] Didier Dubois. Three scenarios for the revision of epistemic states. In *Proceedings of NMR 2006*, 2006.

[18] Phan Minh Dung. On the Acceptability of Arguments and its Fundamental Role in

Nonmonotonic Reasoning, Logic Programming and n-Person Games. *Artificial Intelligence*, 77(2):321–358, 1995.

[19] Marcelo A. Falappa, Alejandro Javier García, Gabriele Kern-Isberner, and Guillermo Ricardo Simari. Stratified Belief Bases Revision with Argumentative Inference. *Journal of Philosophical Logic*, 42(1):161–193, 2013.

[20] Marcelo A. Falappa, Alejandro Javier García, and Guillermo Ricardo Simari. A Set of Operations for Stratified Belief Bases. In Christoph Beierle, Gerhard Brewka, and Matthias Thimm, editors, *Computational Models of Rationality, Essays dedicated to Gabriele Kern-Isberner on the occasion of her 60th birthday*, pages 223–242. College Publications, 2016.

[21] Marcelo A. Falappa, Alejandro Javier García, and Guillermo Ricardo Simari. Merging operators on stratified belief bases equipped with argumentative inference. *Journal of Applied Non Classical Logics*, 33(3-4):387–420, 2023.

[22] Marcelo Alejandro Falappa, Eduardo Leopoldo Fermé, and Gabriele Kern-Isberner. On the Logic of Theory Change: Relations between Incision and Selection Functions. In *Proceedings of ECAI-2006*, pages 402–406, 2006.

[23] Marcelo Alejandro Falappa, Gabriele Kern-Isberner, and Guillermo Ricardo Simari. Belief Revision, Explanations and Defeasible Reasoning. *Artificial Intelligence Journal*, 141:1–28, 2002.

[24] Dov M. Gabbay and Odinaldo Rodrigues. Structured belief bases: A practical approach to prioritised base revision. In Dov M. Gabbay, Rudolf Kruse, Andreas Nonnengart, and Hans Jürgen Ohlbach, editors, *Qualitative and Quantitative Practical Reasoning, Proceedings of First International Joint Conference on Qualitative and Quantitative Practical Reasoning ECSQARU-FAPR'97*, volume 1244 of *Lecture Notes in Computer Science*, pages 267–281. Springer, 1997.

[25] Peter Gärdenfors. *Knowledge in Flux: Modelling the Dynamics of Epistemic States.* The MIT Press, Bradford Books, Cambridge, Massachusetts, 1988.

[26] Peter Gärdenfors. The Dynamics of Belief Systems: Foundations vs.Coherence Theories. *Revue Internationale of Philosophie*, 44:24–46, 1990.

[27] Peter Gärdenfors and David Makinson. Revisions of Knowledge Systems using Epistemic Entrenchment. *Second Conference on Theoretical Aspects of Reasoning About Knowledge*, pages 83–95, 1988.

[28] Adam Grove. Two Modellings for Theory Change. *The Journal of Philosophical Logic*, 17:157–170, 1988.

[29] Sven Ove Hansson. Kernel Contraction. *The Journal of Symbolic Logic*, 59:845–859, 1994.

[30] Sven Ove Hansson. Semi-Revision. *Journal of Applied Non-Classical Logic*, 7:151–175, 1997.

[31] Sven Ove Hansson. *A Textbook of Belief Dynamics: Theory Change and Database Updating.* Kluwer Academic Publishers, 1999.

[32] Hirofumi Katsuno and Alberto O. Mendelzon. On the difference between updating a

knowledge base and revising it. In James F. Allen, Richard Fikes, and Erik Sandewall, editors, *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR'91). Cambridge, MA, USA, April 22-25, 1991*, pages 387–394. Morgan Kaufmann, 1991.

[33] Gabriele Kern-Isberner. Postulates for conditional belief revision. In *Proceedings of Sixteenth International Joint Conference on Artificial Intelligence (IJCAI 1999)*, pages 186–191. Morgan Kaufmann, 1999.

[34] Gabriele Kern-Isberner. A thorough axiomatization of a principle of conditional preservation in belief revision. *Annals of Mathematics and Artificial Intelligence*, 40(1-2):127–164, 2004.

[35] Gabriele Kern-Isberner. Linking iterated belief change operations to nonmonotonic reasoning. In G. Brewka and J. Lang, editors, *Proceedings of 11th International Conference on Knowledge Representation and Reasoning, (KR 2008)*, pages 166–176, Menlo Park, CA, 2008. AAAI Press.

[36] Sten Lindström and Wlodzimierz Rabinowicz. Epistemic Entrenchment with Incomparabilities and Relational Belief Revision. In André Fuhrmann and Michael Morreau, editors, *The Logic of Theory Change*, volume 465 of *Lecture Notes in Computer Science*, pages 93–126. Springer, 1989.

[37] David Makinson and Peter Gärdenfors. Relations between the Logic of Theory Change and Nonmonotonic Logic. *Lecture Notes in Computer Science*, 465:183–205, 1991.

[38] John L. Pollock. Justification and defeat. *Artificial Intelligence*, 67:377–407, 1994.

[39] Iyad Rahwan and Guillermo Ricardo Simari. *Argumentation in Artificial Intelligence.* Springer, 2009.

[40] Hans Rott. Preferential Belief Change using Generalized Epistemic Entrenchment. *The Journal of Logic, Language and Information*, 1:45–78, 1992.

[41] Guillermo R. Simari and Ron P. Loui. A mathematical treatment of defeasible reasoning and its implementation. *Artificial Intelligence*, 53:125–157, 1992.

Received August 2025