Natural Proof Search for Classical Logic

Adam Lassiter

MComp (hons) Computer Science and Mathematics
The University of Bath
May 1, 2019

Natural Proof Search for Classical Logic

Submitted by: Adam Lassiter

Abstract

We investigate a natural algorithm for proof search within classical logic and bounds on the complexity class of such a search. We further examine natural optimisations to this algorithm and how they affect complexity bounds.

Acknowledgements

Herein is the conclusion to four years of university study and seven months of study on proof search and additive linear logic sequent proofs. Written in support of the final year of MComp (hons) Computer Science and Mathematics degree at University of Bath, it has led me through many ups and downs, constantly tackling new problems and potential solutions. For all the wonder and difficulties surrounding P versus NP problems, this points another finger the way of 'not equal' but further research and proof remains to be seen. Fortunately, I feel this provides a remarkable approach to a human-understandable and natural proof search, providing links between previous methods and may help build a deeper understanding.

I would like to express my gratitude to Dr. Willem Heijltjes for his enthusiasm, guidance and words of wisdom throughout, always bright-eyed at the prospect of new discoveries. I further extend my gratitude to all those who kept me motivated throughout, enduring my endless 'rubber-duck debugging' and guiding me through the finer points of English grammar.

Happy reading,

Adam Lassiter

Contents

Chapter 1

Introduction

This paper investigates a natural proof search introduced by Heijltjes & Hughes (2015). Given a formula composed of conjunctions and disjunctions of variables, it falls into one of three categories: provably true, satisfiably true/false, provably false. Satisfiably true or false terms are equivalent to those studied in the boolean satisfiability problem (SAT). Since the set of provably false formulae forms the complement to the set of satisfiably true formulae, with provably true similarly forming the complement to satisfiably false, the question of proof search and its complexity ties directly into problems of P versus NP. The motivating question was 'What is the complexity of the chosen proof search'.

The paper begins with a run-through of classical logic and sequent proofs before examining the proof search algorithm in question. The research entails studying the properties of this algorithm and implementation details that affect computational complexity, in particular a natural optimisation that gives great performance benefits for certain classes of formulae. Finally, the essence of the problem is dissected — how can the complexity of proof search be bounded by the structure of the formula?

A C implementation of the described algorithm complements the research and may be found at `https://gitlab.com/adamlassiter/petri-nets`. This is broken down into:

- An additive linear logic (ALL) formula parser with some additional inferences for some operators ($(a \implies b) \equiv (\neg a \vee b)$ etc.)
- An implementation of the coalescence algorithm using a derivative of petri nets, including a described optimisation on the complexity bound for certain formulae

- A system to backtrack over petri nets fired through coalescence to their equivalent sequent proofs — a bug remains present in this code where proofs are occasionally backtracked incorrectly

While this remains somewhat of a work in progress, the code as is provides consistent results and moderately good performance — when benchmarked against Naoyuki Tamura's Prolog sequent prover[1] saw a decrease in runtime of up to 25x for some formulae.

[1] `http://bach.istc.kobe-u.ac.jp/seqprover/`

Chapter 2

Classical Logic

Definition (Formulae)**.**
A *formula* within classical logic is constructed as follows:

$$\begin{array}{lcl} A, B, C & ::= & \top \mid \bot \mid a \mid \neg\, a \mid A \vee B \mid A \wedge B \\ \Gamma, \Delta, \Sigma & ::= & A \mid A, B \mid A, B, C \ldots \end{array}$$

where $\vee, \wedge$ are additive linear logic disjunction and conjunction respectively and Γ, Δ, Σ are contexts.

Example.
Consider the formulae constructed as follows:

$$A := a \vee b \quad B := \neg b \vee c \quad C := A \wedge B \equiv (a \vee b) \wedge (\neg b \vee c)$$

Definition (Sequent Proofs)**.**
Within *classical logic*, a *sequent proof* is constructed from the following rules:

$$\frac{}{\vdash \top}\ \top \qquad \frac{\vdash \Gamma, A}{\vdash \Gamma, A \vee B}\ \vee \qquad \frac{\vdash \Gamma}{\vdash \Gamma, A}\ w$$

$$\frac{}{\vdash a, \neg\, a}\ ax \qquad \frac{\vdash \Gamma, A \qquad \vdash \Gamma, B}{\vdash \Gamma, A \wedge B}\ \wedge \qquad \frac{\vdash \Gamma, A, A}{\vdash \Gamma, A}\ c$$

where A, B, C are formulae and Γ, Δ, Σ are sequents. A sequent proof provides, without context, a proof of its conclusion and each line of the proof represents a tautology.

Example.
Consider the formula $P := (a \vee \neg a) \wedge \top$. The sequent proof of P, written $\vdash P$, is constructed as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash a, \neg a}\,ax}{\vdash a \vee \neg a, \neg a}\,\vee R}{\vdash a \vee \neg a, a \vee \neg a}\,\vee R}{\vdash a \vee \neg a}\,c \qquad \dfrac{}{\vdash \top}\,\top}{\vdash (a \vee \neg a) \wedge \top}\,\wedge R$$

Remark.
Within the context of weakening and contraction, *additive* and *multiplicative* rules of linear logic are inter-derivable. In this case, the *additive* rules are used, with the effect of $\vee R, \wedge R$ maintaining the number of formulae in a seqent across derivation steps.

Definition (Derivations)**.**
Given *tops* $\Gamma_1 \dots \Gamma_n$ for the sequent proof $\vdash \Delta$, a *derivation* is a tree providing a proof of $\Gamma_1 \dots \Gamma_n \implies \Delta$.

A derivation is written as:

$$\begin{array}{c} \vdash \Gamma_1 \qquad \dots \qquad \vdash \Gamma_n \\ \hline\hline \vdash \Delta \end{array}\ \textit{[label]}$$

where the *label* describes which rules may be used within the derivation.

Corollary (Derivation Equivalence)**.**
A sequent proof is a derivation where all top derivations of the tree are $= \top, ax$. Equivalence of derivations may be weakly defined up to equivalence of leaves and conclusion.

This is described in detail by Girard et al. (1989) as *morally equivalent.*

Example.
Considering the leaves $\vdash A, A$ and $\vdash B$ with the conclusion $\vdash A \wedge B$, the following proofs are morally equivalent:

$$\dfrac{\dfrac{\vdash A, A}{\vdash A}\,c \qquad \vdash B}{\vdash A \wedge B}\,\wedge$$

$$\dfrac{\dfrac{\dfrac{\vdash A, A \qquad \dfrac{\vdash B}{\vdash A, B}\,w}{\vdash A, A \wedge B}\,\wedge \qquad \dfrac{\vdash B}{\vdash B, A \wedge B}\,w}{\vdash A \wedge B, A \wedge B}\,\wedge}{\vdash A \wedge B}\,c$$

Note that the collection of leaves is a `Set`, so equivalence is up to existence of terms only and not number. Furthermore, equivalence of leaves is also defined up to equivalence of sequents, in particular equivalence up to commutativity of formulae $\vdash A, B \equiv \vdash B, A$. Equivalence may also be considered up to idempotency of formulae $\vdash A, A \equiv \vdash A$, but this interferes with the correctness of some definitions.

Definition (Additive Stratification)**.**
A proof tree is said to be *additively stratified* if $\vdash P$ is structured as follows:

$$\cfrac{\cfrac{\cfrac{\cfrac{}{\vdash A_1}\,\top, ax}{\vdash \Gamma_1}\,w \qquad \dots \qquad \cfrac{\cfrac{}{\vdash A_n}\,\top, ax}{\vdash \Gamma_n}\,w}{\vdash P \dots P}\,\wedge, \vee}{\vdash P}\,c$$

That is, the inferences made in an additively stratified proof are strictly ordered by:

1. Top/Axiomatic
2. Weakening
3. Conjunction/Disjunction
4. Contraction

Example 2.1.
The following proof tree is additively stratified:

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{}{\vdash a, \neg a}\,ax}{\vdash a \vee \neg a, \neg a}\,\vee}{\vdash a \vee \neg a, a \vee \neg a}\,\vee \qquad \cfrac{\cfrac{}{\vdash \top}\,\top}{\vdash \top, a \vee \neg a}\,w}{\vdash (a \vee \neg a) \wedge \top, a \vee \neg a}\,\wedge \qquad \cfrac{\cfrac{}{\vdash \top}\,\top}{\vdash \top, (a \vee \neg a) \wedge \top}\,w}{\vdash (a \vee \neg a) \wedge \top, (a \vee \neg a) \wedge \top}\,\wedge}{\vdash (a \vee \neg a) \wedge \top}\,c$$

Proposition (Stratification Equivalence)**.**
Given $\vdash A$, there exists a morally equivalent additively stratified proof of A.

Proof. For each instance of a weakening below another inference, there exists an equivalent subproof that is additively stratified:

$$\cfrac{\cfrac{\vdash \Gamma, A}{\vdash \Gamma, A \vee B}\,\vee}{\vdash \Gamma, A \vee B, \mathrm{C}}\,w \qquad \rightsquigarrow \qquad \cfrac{\cfrac{\vdash \Gamma, A}{\vdash \Gamma, A, C}\,w}{\vdash \Gamma, A \vee B, C}\,\vee$$

$$\dfrac{\dfrac{\vdash \Gamma, A \qquad \vdash \Gamma, B}{\vdash \Gamma, A \wedge B}\wedge}{\vdash \Gamma, A \wedge B, C}w \quad\rightsquigarrow\quad \dfrac{\dfrac{\vdash \Gamma, A}{\vdash \Gamma, A, C}w \qquad \dfrac{\vdash \Gamma, B}{\vdash \Gamma, B, C}w}{\vdash \Gamma, A \wedge B, C}\wedge$$

$$\dfrac{\dfrac{\vdash \Gamma, A, A}{\vdash \Gamma, A}c}{\vdash \Gamma, A, B}w \quad\rightsquigarrow\quad \dfrac{\dfrac{\vdash \Gamma, A, A}{\vdash \Gamma, A, A, B}w}{\vdash \Gamma, A, B}c$$

Similarly, for each instance of a contraction above another inference, there exists an equivalent subproof that is additively stratified:

$$\dfrac{\dfrac{\vdash \Gamma, A, A}{\vdash \Gamma, A}c}{\vdash \Gamma, A \vee B}\vee \quad\rightsquigarrow\quad \dfrac{\dfrac{\dfrac{\vdash \Gamma, A, A}{\vdash \Gamma, A \vee B, A}\vee}{\vdash \Gamma, A \vee B, A \vee B}\vee}{\vdash \Gamma, A \vee B}c$$

$$\dfrac{\dfrac{\vdash \Gamma, A, A}{\vdash \Gamma, A}c \qquad \vdash \Gamma, B}{\vdash \Gamma, A \wedge B}\wedge \quad\rightsquigarrow\quad \dfrac{\dfrac{\dfrac{\vdash \Gamma, A, A \qquad \dfrac{\vdash \Gamma, B}{\vdash \Gamma, A, B}w}{\vdash \Gamma, A, A \wedge B}\wedge \qquad \dfrac{\vdash \Gamma, B}{\vdash \Gamma, B, A \wedge B}w}{\vdash \Gamma, A \wedge B, A \wedge B}\wedge}{\vdash \Gamma, A \wedge B}c$$

By induction from the leaves downwards on a finite height tree, apply the associated rule to each pair of inferences of the form (c above *inf*). Any given $\vdash P$ may be rewritten:

$$\dfrac{\dfrac{\dfrac{}{\vdash A_1}\top, ax}{\vdash \Gamma_1}\wedge, \vee, w \qquad \dots \qquad \dfrac{\dfrac{}{\vdash A_n}\top, ax}{\vdash \Gamma_n}\wedge, \vee, w}{\vdash P}c$$

Again, by induction from the root upwards on this partially stratified tree, apply the associated rule to each pair of inferences of the form (w below *inf*). $\vdash P$ may then be further rewritten:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash A_1}\top, ax}{\vdash \Gamma_1}w \qquad \dots \qquad \dfrac{\dfrac{}{\vdash A_n}\top, ax}{\vdash \Gamma_n}w}{\vdash P \dots P}\wedge, \vee}{\vdash P}c$$

$\square$

Example.
Similarly to Example 2.1, consider the two following morally equivalent proofs, with only the latter additively stratified.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash a, \neg a}\,ax}{\vdash a \vee \neg a, \neg a}\,\vee}{\vdash a \vee \neg a, a \vee \neg a}\,\vee}{\vdash a \vee \neg a}\,w \qquad \dfrac{}{\vdash \top}\,\top}{\vdash (a \vee \neg a) \wedge \top}\,\wedge$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash a, \neg a}\,ax}{\vdash a \vee \neg a, \neg a}\,\vee}{\vdash a \vee \neg a, a \vee \neg a}\,\vee \qquad \dfrac{\dfrac{}{\vdash \top}\,\top}{\vdash \top, a \vee \neg a}\,w}{\vdash (a \vee \neg a) \wedge \top, a \vee \neg a}\,\wedge \qquad \dfrac{\dfrac{}{\vdash \top}\,\top}{\vdash \top, (a \vee \neg a) \wedge \top}\,w}{\vdash (a \vee \neg a) \wedge \top, (a \vee \neg a) \wedge \top}\,\wedge}{\vdash (a \vee \neg a) \wedge \top}\,c$$

Note that moral equivalence here is without regard to number — there are two leaves $\vdash \top$ under additive stratification versus one otherwise. Furthermore, notice that the proof without additive stratification is shorter — this will be of importance when improving asymptotic performance.

Chapter 3

Coalescence

Definition (Petri Nets)**.**
For the purposes required here, a *petri net* $\mathcal{N}$ is $(\mathcal{P}, \mathcal{F})$ where $f \in \mathcal{F} : \mathcal{P}^m \times \mathcal{P}$. In particular, $\mathcal{P}$ is a set of places and $\mathcal{F}$ a set of flows or transitions. A *configuration* is a set $\mathcal{C} \subset \mathcal{P}$ of tokens in places.

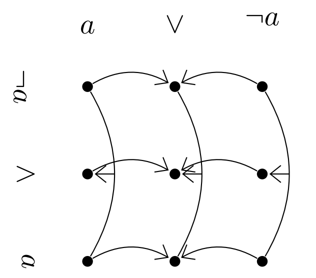
Notation from here follows the convention used by Lamarche & Straßburger (2005) of iterating over places in a formula and indexing by $i \in \mathcal{N}$. Given a formula in classical logic, conjunction or disjunction is encoded as:

$$\begin{aligned} P_1 \vee_2 Q_3 \quad &\mapsto \quad \{(P_1) \times \vee_2, (Q_3) \times \vee_2\} \\ P_1 \wedge_2 Q_3 \quad &\mapsto \quad \{(P_1, Q_3) \times \wedge_2\} \end{aligned}$$

where A, B are (not necessarily unique) subformulae in unique places iterated over by $\{1, 2, \ldots\}$. Subsequently, in an expression such as $A_1 \wedge_2 A_3$, a token on A_1 is unique to a token on A_3

This is then a direct continuation of the concepts explored by Heijltjes & Hughes (2015).

Example.
Consider the 2-d petri net representing the cross-product $a \vee \neg a \otimes \neg a \wedge a$ as follows:



Definition (Firing Petri Nets)**.**
Given a petri net $\mathcal{N}$ and configuration $\mathcal{C}$, a *firing* of the net $\mathcal{N}$ is a new configuration generated by application of a transition $f \in \mathcal{F}$ on m tokens $c_1 \ldots c_n \in \mathcal{C}$. In particular:

$$(\mathcal{N} = (\mathcal{P}, \mathcal{F}), \mathcal{C}) \mapsto (\mathcal{N}, (\mathcal{C} \cup f_{right}) \setminus f_{left})$$

for some $f = (f_{left}, f_{right}) \in \mathcal{F}$.

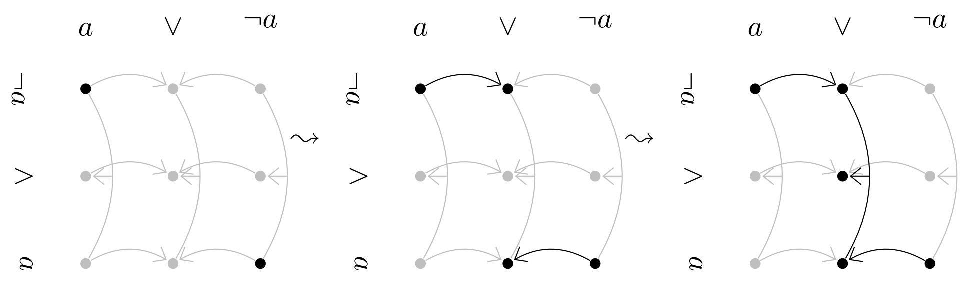
For the uses required here, a variant of firing is used instead called *spawning*. This generates new configurations in the same manner as firing, with one key difference:

$$(\mathcal{N} = (\mathcal{P}, \mathcal{F}), \mathcal{C}) \mapsto (\mathcal{N}, \mathcal{C} \cup f_{right})$$

for some $f = (f_{left}, f_{right}) \in \mathcal{F}$. That is, when a transition f is performed on tokens $x_1 \ldots_n$, these tokens remain present in the configuration in addition to the new token $f(x_1 \ldots x_n)$.

A petri net is said to be *exhaustively fired* if it is fired until there does not exist any such $f \in \mathcal{F}$ to fire.

Example.
Consider the 2d petri net representing $a \vee \neg a \otimes \neg a \wedge a$ with tokens at $\{(a, \neg a), (\neg a, a)\}$.

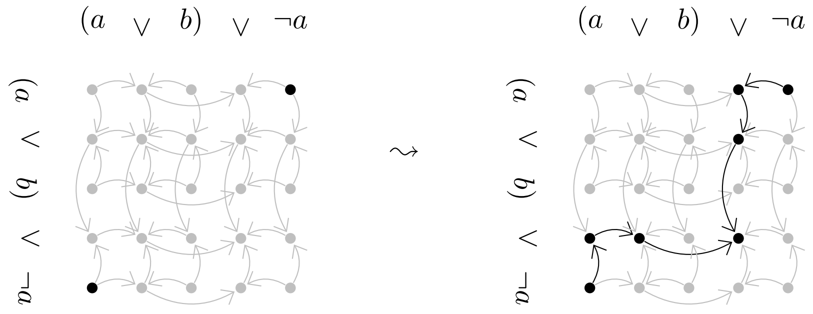


Note that, in the final step, a pair of tokens are required to perform the conjunction transition.

Definition (Coalescence)**.**
Given a formula P, the coalescence algorithm is as follows:

1. Set $n := 1$
2. Construct a n-dimensional petri net $\mathcal{N}$ as $\bigotimes_n P$ where each subformula represents a place and each conjunction and disjunction a flow
3. Construct a configuration C with a token at each place $p = (\ldots, a, \ldots, \neg a, \ldots)$ the intersection of a pair of tautological atoms or $(\ldots, \top, \ldots)$ an instance of *top*
4. Exhaustively fire the petri net $\mathcal{N}$ using the *spawning* method
5. If there exists a token in the configuration $\mathcal{C}^*$ at the root of the formula P, halt and return n
6. Otherwise, increment $n := n + 1$ and go to step 2

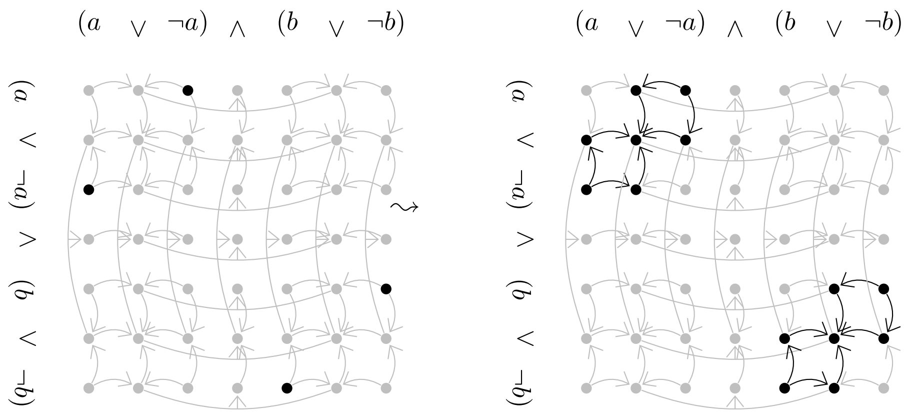
Example.
Consider the petri net proof for the formula $(a \vee b) \vee \neg a$, with a solution for $n = 2$. The net is initialised with tokens in all places satisfying either the ax-rule or $\top$-rule and fired exhaustively:



$(a \quad \vee \quad b) \quad \vee \quad \neg a$
$(a \quad \vee \quad b) \quad \vee \quad \neg a$
$\rightsquigarrow$
$(a \quad \vee \quad b) \quad \vee \quad \neg a$
$(a \quad \vee \quad b) \quad \vee \quad \neg a$

Remark.
Note the symmetry along the upper-left lower-right diagonal. In fact, this symmetry can be exploited in the general case, ignoring all tokens below (w.l.o.g.) the diagonal as they simply express the commutativity property $\vdash A, B \iff \vdash B, A$. If this is the case, the coalescence algorithm may halt early if the root of the formula is reached.

Example.
Consider the petri net proof for the formula $(a \vee \neg a) \wedge (b \vee \neg b)$, with a solution for $n = 3$. In an exhaustively fired net for $n = 2$, the proof looks as follows:


$(a \lor \neg a) \land (b \lor \neg b)$
$(a \lor \neg a) \land (b \lor \neg b)$
$(a \land \neg a) \lor (b \land \neg b)$
$(a \land \neg a) \lor (b \land \neg b)$
$\rightsquigarrow$

The coalescence algorithm fails to find a solution in 2 dimensions due to the lack of tokens in upper-right or lower-left quadrants. If there were tokens in such a place, the tree would be able to continue to grow towards the formula root at the centre of the net.

Remark.
Implementing firable petri nets is straightforward using an n-dimensional boolean array of places visited and a collection of n-tuples representing tokens. Within the context of coalescence proof search, this can be optimised in many ways, in this case a upper-triangular n-dimensional boolean matrix of places visited and a red-black tree of n-dimensional boolean array tokens can be used. This exploits most of the available symmetry in the structure of the nets as, in implementation, memory quickly becomes the limiting factor. Furthermore, this structure allows for $\mathcal{O}(\log n)$ insertion and removal, $\mathcal{O}(n)$ iteration and minimal memory.

Proposition (Coalescence Proof Search)**.**
The coalescence algorithm on P is exactly a proof search on P.

Proof. In particular, consider the additively stratified $\vdash P$ with $n-1$ contractions (should such a proof exist). The first configuration of tokens is precisely a proof all possible combinations of the *axiom* rule with $n-3$ other terms through the *weakening*. Each flow transition followed when fired is an application of either the $\lor$ or $\land$ rule. Finally, a token at the root of the formula is $\vdash P \ldots P$, with the *contraction* rule applied implicitly.

If there exists an additively stratified proof of P, the coalescence algorithm will find it. Since for every proof there exists an additively stratified proof, coalescence is precisely proof search. $\square$

Chapter 4

Dimensionality

Definition (Dimensionality)**.**
Given a formula P, the coalescence proof search produces a proof in an n-dimensional petri net. Equivalently, an additively stratified sequent proof $\vdash P$ has n terms at the bottom before any contractions are applied. The dimensionality of P is then defined $\dim(P) ::= n$.

Example.
Borrowing the proof from Example 2.1, it has dimensionality 2 since the tree concludes with $\vdash P, P$ before any contractions.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash a, \neg a}\,ax}{\vdash a \vee \neg a, \neg a}\,\vee}{\vdash a \vee \neg a, a \vee \neg a}\,\vee \qquad \dfrac{\dfrac{}{\vdash \top}\,\top}{\vdash \top, a \vee \neg a}\,w}{\vdash (a \vee \neg a) \wedge \top, a \vee \neg a}\,\wedge \qquad \dfrac{\dfrac{}{\vdash \top}\,\top}{\vdash \top, (a \vee \neg a) \wedge \top}\,w}{\vdash (a \vee \neg a) \wedge \top, (a \vee \neg a) \wedge \top}\,\wedge}{\vdash (a \vee \neg a) \wedge \top}\,c$$

Definition 4.1 (Classes of Formulae)**.**
Let D^i be the subclasses of formulae defined as:

$$\begin{array}{lcl} D^1 & ::= & \top \mid D^1 \wedge D^1 \mid D^* \vee D^1 \\ D^2 & ::= & D^{i \geq 2} \vee D^2 \mid P \vee \neg P \mid \ldots \end{array}$$

where $D^* ::= \bigcup_{i \in \mathbb{N}} D_i$, such that D^n is the class of all formulae provable in n dimensions.

This definition will be revised later.

Remark (Satisfiability vs Provability)**.**
For any formula P, there exist four distinct classes: *true, false, satisfiably true, satisfiably false*, where satisfiable differs by finding a particular assignment of values to each variable. Coalescence searches for a proof of P in *true*, whereas the SAT problem addresses a proof of P in *true*.

Proposition (Bounds on Dimensionality)**.**
Given a formula P, the following provides a bound its dimensionality:

$$P \in D^n \implies n + 1 \leq vars(P)$$

Justification is left unaddressed here as the current construction does not emit a 'nice' proof.

Remark.
These methods do not lend themselves to 'nice' properties. It is possible that in the general case, *finding an exact value for the dimensionality of a given formula P may be equivalent to finding a proof of P*, but it remains to be proven. Instead, a natural optimisation is applied to the algorithm as follows, giving rise to a different set of classes of dimensionality and more pleasing properties.

Definition ($\top$-substitution)**.**
Given a sequent expressing a proof of Q, i.e. $\vdash Q, Q, \ldots Q$, the formula P may be equivalently expressed $P[Q := \top]$.

Example.
Consider the formula $P ::= (a \vee \neg a) \wedge (b \vee \neg b)$. The sequent proof $\vdash P$ is large and has dimensionality 3 as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdash a, a, \neg a}{\vdash a, (a \vee \neg a), \neg a} \quad \dfrac{\vdash a, \neg a, b}{\vdash a, \neg a, (b \vee \neg b)}}{\vdash a, \neg a, ((a \vee \neg a) \wedge (b \vee \neg b))}}{\vdash a, (a \vee \neg a), ((a \vee \neg a) \wedge (b \vee \neg b))} \quad \dfrac{\dfrac{\dfrac{\vdash a, \neg a, b}{\vdash a, \neg a, (b \vee \neg b)}}{\vdash a, (a \vee \neg a), (b \vee \neg b)} \quad \dfrac{\dfrac{\vdash a, b, \neg b}{\vdash a, b, (b \vee \neg b)}}{\vdash a, (b \vee \neg b), (b \vee \neg b)}}{\vdash a, ((a \vee \neg a) \wedge (b \vee \neg b)), (b \vee \neg b)}}{\vdash a, ((a \vee \neg a) \wedge (b \vee \neg b)), ((a \vee \neg a) \wedge (b \vee \neg b))}}{\vdash (a \vee \neg a), ((a \vee \neg a) \wedge (b \vee \neg b)), ((a \vee \neg a) \wedge (b \vee \neg b))} \quad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdash a, \neg a, b}{\vdash a, \neg a, (b \vee \neg b)}}{\vdash a, (a \vee \neg a), (b \vee \neg b)} \quad \dfrac{\dfrac{\vdash a, b, \neg b}{\vdash a, b, (b \vee \neg b)}}{\vdash a, (b \vee \neg b), (b \vee \neg b)}}{\vdash a, ((a \vee \neg a) \wedge (b \vee \neg b)), (b \vee \neg b)}}{\vdash (a \vee \neg a), ((a \vee \neg a) \wedge (b \vee \neg b)), (b \vee \neg b)} \quad \dfrac{\dfrac{\dfrac{\dfrac{\vdash a, b, \neg b}{\vdash a \vee \neg a, b, \neg b} \quad \dfrac{\vdash b, b, \neg b}{\vdash b \vee \neg b, b, \neg b}}{\vdash ((a \vee \neg a) \wedge (b \vee \neg b)), b, \neg b}}{\vdash ((a \vee \neg a) \wedge (b \vee \neg b)), b, (b \vee \neg b)}}{\vdash ((a \vee \neg a) \wedge (b \vee \neg b)), (b \vee \neg b), (b \vee \neg b)}}{\vdash ((a \vee \neg a) \wedge (b \vee \neg b)), ((a \vee \neg a) \wedge (b \vee \neg b)), (b \vee \neg b)}}{\vdash ((a \vee \neg a) \wedge (b \vee \neg b)), ((a \vee \neg a) \wedge (b \vee \neg b)), ((a \vee \neg a) \wedge (b \vee \neg b))}$$

Using substitutions of $\top$ for provable subformulae, the proof $\vdash P$ becomes much more manageable:

$$A := (a \vee \neg a) \qquad \dfrac{\dfrac{\vdash a, \neg a}{\vdash a, (a \vee \neg a)}}{\vdash (a \vee \neg a), (a \vee \neg a)}$$

$$B := (b \vee \neg b) \qquad \dfrac{\dfrac{\vdash b, \neg b}{\vdash b, (b \vee \neg b)}}{\vdash (b \vee \neg b), (b \vee \neg b)}$$

$$\dfrac{\dfrac{\vdash A, A \qquad \vdash A, B}{\vdash A, (A \wedge B)} \qquad \dfrac{\vdash A, B \qquad \vdash B, B}{\vdash B, (A \wedge B)}}{\vdash (A \wedge B), (A \wedge B)}$$

While the final step need only be performed with dimensionality 1, the prior steps require dimensionality 2, giving the full formula a dimensionality of 2.

Remark.
It is now important to reexamine the coalescence algorithm.

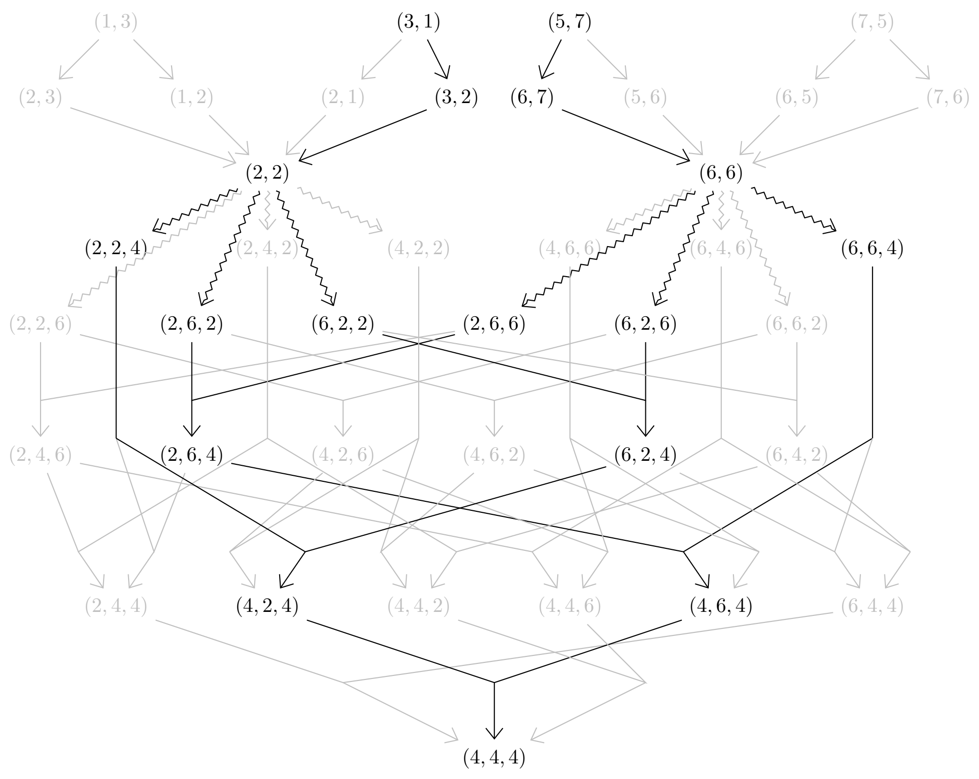
Storing a token as a `Tuple` gives an inefficient algorithm due to the implicit ordering. By commutativity, given a token $(a_1 \dots a_n)$, there exist up to $n!$ permutations of a_i's and thus $n!$ equivalences per token.

Instead, order is abstracted over and a token is stored as a `Multiset`. This gives the storage optimisation exploiting the symmetry of petri nets that expresses commutativity of terms in a sequent. There are now only n equivalences per token.

This may be taken a step further. Consider a token stored as a `Set` which expresses idempotency of terms in a sequent $\vdash P, P \equiv \vdash P$. This forms a canonical representation of tokens — the only token equivalent to $T \in$ `Set` is T itself.

Examples.
Consider again the formula $(a \vee \neg a) \wedge (b \vee \neg b)$, with a proofs as follow, but instead tracing the values of tokens.

Where tokens are member of the category `Tuple`, so as to demonstrate the exponential blow-up, both the 'minimal' (black) and 'saturated' (grey) proofs are as follows:



where $\rightsquigarrow$ is an application of weakening. This is notable as this will be the point coalescence will reach for $n := 2$ and a suitably intelligent method for extending dimensions will maintain the proof search so far. Both 'minimal' and 'saturated' proofs are included as each represent the deterministic versus non-deterministic searches — proof search versus validation.

Where tokens are now members of `Multiset`, the proof of the full formula follows:

$$\begin{tikzcd}
 & <1,3> \arrow[dl] \arrow[dr] & & & & <5,7> \arrow[dl] \arrow[dr] & \\
<2,3> \arrow[dr] & & <1,2> \arrow[dl] & & <6,7> \arrow[dr] & & <5,6> \arrow[dl] \\
 & <2,2> \arrow[dl, squiggly] \arrow[dr, squiggly] & & & & <6,6> \arrow[dl, squiggly] \arrow[dr, squiggly] & \\
<2,2,4> \arrow[ddrr] & & <2,2,6> \arrow[dr, dash] & & <2,6,6> \arrow[dl] & & <4,6,6> \arrow[ddll] \\
 & & & <2,4,6> \arrow[dl, dash] \arrow[dr, dash] & & & \\
 & & <2,4,4> \arrow[dr, dash] & & <4,4,6> \arrow[dl] & & \\
 & & & <4,4,4> & & &
\end{tikzcd}$$

The above remains tedious and unnatural. Proofs of each of $a \vee \neg a$ and $b \vee \neg b$ are constructed in two steps, while the remaining steps are to prove the conjunction of these (proven) subformulae. Instead, consider the tokens as members of `Set` as follows:

$$\begin{tikzcd}
 & \{1,3\} \arrow[dl] \arrow[dr] & & & & \{5,7\} \arrow[dl] \arrow[dr] & \\
\{2,3\} \arrow[dr] & & \{1,2\} \arrow[dl] & & \{6,7\} \arrow[dr] & & \{5,6\} \arrow[dl] \\
 & \{2\} \arrow[drr, dash] & & & & \{6\} \arrow[dll] & \\
 & & & \{4\} & & &
\end{tikzcd}$$

This is finally deemed a 'satisfying' proof — dimensionality is extracted as the largest set required in the proof, a nicer algebra of dimensionality may be constructed and there is some essence of justification for the correctness of this $\top$-substitution — all through changing the datatype of tokens.

Remark 4.2.
In essence, allowing for efficient substitution of a $\top$ term for proven subformulae allows the coalescence algorithm to run without restricting the output to additively stratified proofs. A proof may be reconstructed including substitutions, where each subproof substituted is itself additively stratified, but now there are contractions for the conclusion of a subproof on top of weakenings for the axioms of a superproof.

Definition (Classes of Formulae)**.**

With this new concept of coalescence, dimensionality and proof search, the definitions in 4.1 are revisited and revised. Let $\mathcal{D}^i$ be defined as the subclasses of formulae:

$$
\begin{array}{lll}
\mathcal{D}^1 & ::= & \top \,|\, \mathcal{D}^1 \wedge \mathcal{D}^1 \,|\, \mathcal{D}^* \vee \mathcal{D}^1 \\
\mathcal{D}^2 & ::= & P \vee \neg P \,|\, \mathcal{D}^2 \vee \mathcal{D}^{i\geq 2} \,|\, \mathcal{D}^2 \wedge \mathcal{D}^{i\leq 2} \\
\mathcal{D}^3 & ::= & (P \wedge Q) \vee (P \wedge \neg Q) \vee (\neg P \wedge Q) \vee (\neg P \wedge \neg Q) \,|\, \mathcal{D}^3 \vee \mathcal{D}^{i\geq 3} \,|\, \mathcal{D}^3 \wedge \mathcal{D}^{i\leq 3}
\end{array}
$$

where $\mathcal{D}^* ::= \bigcup_{i\in\mathbb{N}} \mathcal{D}^i$, $P \in \mathcal{D}^*$, such that $\mathcal{D}^n$ is the class of all formulae provable in exactly n dimensions.

Corollary 4.3.
For $P \in \mathcal{D}^i, Q \in \mathcal{D}^j$, the following statements hold:

$$P \vee Q \in \mathcal{D}^{\min i,j} \qquad P \wedge Q \in \mathcal{D}^{\max i,j}$$

In fact, for $P \in \mathcal{D}^n$ in v variables, $n \leq v+1$.

Proposition 4.4 (Dimensionality Bounds of Disjunctive Normal Forms)**.**
Let DNF^n denote the set of all formulae in n variables in disjunctive normal form. That is, $P \in \mathit{DNF}^n \implies P = \bigvee_{j=1..2^n}(\bigwedge_{i=1..n} a_i)$ the disjunction of 2^n conjuctions, each representing an 'assignment' of boolean values to n variables. Then the following statement holds:

$$P \in \mathit{DNF}^n \implies P \in \mathcal{D}^{n+1}$$

Proof. By induction on the number of variables n.

Let the base case be $n := 1$ and the proof is trivial — the sequent proof $\vdash a \vee \neg a$ requires 2 dimensions and the hypothesis holds.

Suppose the inductive hypothesis holds up to $n-1$ variables — in i dimensions where $i \leq n$ and all DNF-terms in v variables where $i-1 \equiv v < n$ such that:

$$\forall\, i \leq n \,:\, P \in \mathit{DNF}^i \implies P \in \mathcal{D}^i$$

It is now sufficient to show the hypothesis continues to hold for up to n variables:

$$\forall\, Q \in \mathit{DNF}^n : Q \in \mathcal{D}^{n+1}$$

Let a_n be fresh in P and $a_1 \ldots a_{n-1}$ be used, then the disjunctive normal form expression P is in the class:

$$P ::= * \,|\, P_j \vee P \qquad P_j ::= * \,|\, a_i \,|\, \neg a_i \,|\, P_j \wedge P_j$$

Write $P ::= \bigvee_j (P_j ::= \bigwedge_j a_{ij})$ for some $P_j\ \forall\, j \in 1 \dots n$ where P_j is purely a conjunction of an 'assignment' a_{ij} of variables. Now construct Q as follows:

$$Q \equiv Q_l \vee Q_r := (\bigvee_i P_i \wedge a_n) \vee (\bigvee_i P_i \wedge \neg a_n) \in \mathit{DNF}^n$$

where Q is the simplest DNF^n-term extension of P. Each P_i is of course not provable by itself, therefore not provable up to n dimensions, but up to n dimensions produces sequents of n terms. Similarly, Q is therefore not provable in n dimensions.

Since Q not provable in n dimensions, extend the proof to $n+1$ dimensions — on the left, the leaf sequents of Q_l may be extended from $\vdash \Gamma$ atomic terms to $\vdash \Gamma, \neg a$ giving $n+1$ dimensionality as required. Now the conclusion of this proof subtree may be up to:

$$\cfrac{\overline{\overline{\vdash Q_l \dots_n Q_l, \neg a_n}}}{\vdash Q \dots_n Q, \neg a_n}$$

Similarly, this may be applied to Q_r but instead 'solving' for each $a_{i<n}$ yielding $\forall\, i < n$:

$$\cfrac{\overline{\overline{\vdash Q_r \dots_n Q_r, a_i}}}{\vdash Q \dots_n Q, a_i}$$

Reducing across all subtrees by the $\wedge$ rule yields $\vdash Q \dots_n Q, P_j \wedge a_n$ for some assignment P_j. Finally, the $\vee$ rule may be applied up the formula syntax tree to yield $\vdash Q \dots_n Q, Q \equiv \vdash Q \dots_{n+1} Q \implies Q \in \mathcal{D}^{n+1}$, where $Q \in \mathit{DNF}^n$ as required. $\square$

Remark 4.5.
This proof omits the subtleties and specifics of deriving each assignment of P_j with respect to the two cases of $a_n, \neg a_n$. This is considered tedious and does not aid to a clear representation of the intermediate steps but remains to be formally proven, should such a thing be desirable.

Proposition.
Building on the above, the dimensionality of any given formula can be predicted by examining the most variables in a disjunctive normal form subformula:

$$P \in \mathcal{D}^{n+1} \iff \exists\, Q \in P : Q \in DNF^n$$

Proof. It can be trivially seen that by combining the results in Corollary 4.3 and Proposition 4.4 that the statement is true.

($\Longleftarrow$) The reverse implication is given exactly by Proposition 4.4.

($\Longrightarrow$) Suppose $P \in \mathcal{D}^{n+1}$. Then, one of the following is true:

1. $P \equiv (L \wedge R)$ where there exist proofs for each of L, R and by Corollary 4.3, without loss of generality, $L \in \mathcal{D}^{n+1}$. Then, the proof recurses on L.

2. $P \equiv (L \vee R)$ where there exist proofs for each of L, R and by Corollary 4.3 $L, R \in \mathcal{D}^{n+1}$. Then, the proof recurses on both L and R.

3. $P \equiv (L \vee R)$ where there do not exist proofs for both L, R. Therefore, if there is a proof of P, then $P \in \mathit{DNF}$ and $P \in \mathcal{D}^{n+1} \implies P \in \mathit{DNF}^n$ as required.

$\square$

Remarks.
Renne (2004) and Otten & Kreitz (1995) and Bibel (1981) and Andrews (1981)

Chapter 5

Conclusion

Building on the work done by Heijltjes & Hughes (2015), proof search in classical logic through additive linear logic (ALL) was further investigated, with particular focus on the complexity of search. The process investigated, called *coalescence*, is a top-down proof search from axiom links through to the conclusion. This method is promising as it boasts great efficiency for ALL proof search and has a natural transformation to sequent calculus proofs.

Sequent Calculus and ALL

A classical formula can be proved by an n-dimensional additive proof, for some n dependant upon the formula. Some simple classes for formulae were proposed — boolean constants *only* are 1-dimensional, normal additive proofs are 2-dimensional amongst others etc. It was proven that this idea is consistent through *additive stratification* of the sequent calculus — that is, any sequent proof may be 'rearranged' up to the order of rules applied, in particular with *weakening* at the top and *contraction* at the bottom. Coalescence is then exactly (additively stratified) proof search.

Coalescence

A proof search is then constructed through use of 'natural' deductions. Using a system analogous to *petri nets*, construct a net through the n-dimensional cross product of places in a formula and associated transitions across the net. Each token in the net begins at an axiom and transitions are exhaustively applied up the formula's syntax tree — or down an equivalent sequent proof tree. A place is a coordinate in the n-dimensional grid representing a context of n places in the formula that is provable. The process either halts when the root of the formula syntax tree is reached and a proof is constructed, or restarts in a higher dimension when applicable transitions are exhausted and the dimensionality must be increased. The dimensionality of a proof is then the dimensionality of our grid when the root is reached.

Motivation

Clearly complexity scales with dimensionality and the motivating question is then: *'What dimension is sufficient for a given formula?'*. In essence, this gives an upper bound for complexity of proof search — specifically, $\mathcal{O}(m^n)$ where m is the length of the formula and n the dimension of the proof.

Some Examples

The definition of dimensionality and construction of proofs was found to not yield results as expected — for example, $(a \vee \neg a) \wedge (b \vee \neg b)$ would suggest a dimension of 2 since each component may be proven in 2 dimensions and the conjunction should be trivial. Instead, unsatisfyingly, this increases dimension quickly and unreasonably, with the aforementioned case yielding dimensionality 3 and linear growth for subsequent additional cases of $(x \vee \neg x)$ appended through conjunction.

Solution

To solve this issue, the search algorithm was generalised over the properties of sequents — namely, idempotency and commutativity of terms in a context. This is analogous to some notion of applying conjunctions 'diagonally' and achieved through switching from `tuple` or `multiset` links to `set` links. This can be seen as more familiar proof search territory.

There exist various other implementation trade-offs: dense/sparse representation, high-performance data structures and assorted other ad-hoc optimisations. These were not touched upon here as they do not affect the complexity of the search in any meaningful way.

Further Examples

The optimisation through generalising over sequents yielded favourable results for conjunctive normal form (CNF) formulae but maintained poor performance for disjunctive normal form (DNF). A simple algebra of classes, with conjunction and disjunction of proven formulae equivalent to maximum and minimum functions of left and right subformulae dimensionality, was constructed and provided bounds on dimensionality for the conjunction or disjunction of any *provable* subformulae. Finally, a generalised bound for any formula was hypothesised with the 'essence' of the associated proof provided — dimensionality is equivalent to one more than the most number of variables in any DNF subformula.

Ongoing Work

Further study is required as to how coalescence proof search relates to traditional proof search methods, but it is expected to be similar to either the 'connections' or 'matrix' method — see Renne (2004), Bibel (1981), Otten & Kreitz (1995) and Andrews (1981).

Progress is still to be made as to DNF formulae — generalisation over associativity of ALL terms (the only remaining algebraic property of terms not addressed) and automatic construction of subformulae (allowing for a better description of DNF formulae) may hold the key to a more natural computation.

The proof of Proposition 4.4 remains incomplete and future work may see this amended.

Chapter 6

References

Andrews, P. B. (1981), ‘Theorem proving via general matings’, *Journal of the ACM (JACM)* **28**(2), 193–214.

Bibel, W. (1981), ‘On matrices with connections’, *J. ACM* **28**(4), 633–645.

Girard, J.-Y., Taylor, P. & Lafont, Y. (1989), *Proofs and types*, Vol. 7, Cambridge university press Cambridge.

Heijltjes, W. & Hughes, D. J. (2015), Complexity bounds for sum-product logic via additive proof nets and petri nets, *in* ‘2015 30th Annual ACM/IEEE Symposium on Logic in Computer Science’, IEEE, pp. 80–91.

Lamarche, F. & Straßburger, L. (2005), Naming proofs in classical propositional logic, *in* ‘International Conference on Typed Lambda Calculi and Applications’, Springer, pp. 246–261.

Otten, J. & Kreitz, C. (1995), A connection based proof method for intuitionistic logic, *in* ‘International Workshop on Theorem Proving with Analytic Tableaux and Related Methods’, Springer, pp. 122–137.

Renne, B. (2004), ‘Tableaux for the logic of proofs’, *CUNY PhD Program in Computer Science, Technical Report TR-2004001, New York* pp. 148–158.