

LNAI 15398

Stephen Cranefield
Luis Gustavo Nardin
Nathan Lloyd (Eds.)

Coordination, Organizations, Institutions, Norms, and Ethics for Governance of Multi-Agent Systems XVII

International Workshop, COINE 2024
Auckland, New Zealand, May 7, 2024
Revised Selected Papers

Lecture Notes in Computer Science

Lecture Notes in Artificial Intelligence

15398

Founding Editor

Jörg Siekmann

Series Editors

Randy Goebel, *University of Alberta, Edmonton, Canada*

Wolfgang Wahlster, *DFKI, Berlin, Germany*

Zhi-Hua Zhou, *Nanjing University, Nanjing, China*

The series Lecture Notes in Artificial Intelligence (LNAI) was established in 1988 as a topical subseries of LNCS devoted to artificial intelligence.

The series publishes state-of-the-art research results at a high level. As with the LNCS mother series, the mission of the series is to serve the international R & D community by providing an invaluable service, mainly focused on the publication of conference and workshop proceedings and postproceedings.

Stephen Cranefield · Luis Gustavo Nardin ·
Nathan Lloyd
Editors

Coordination, Organizations, Institutions, Norms, and Ethics for Governance of Multi-Agent Systems XVII

International Workshop, COINE 2024
Auckland, New Zealand, May 7, 2024
Revised Selected Papers

Editors

Stephen Cranfield  University of Otago Dunedin, New Zealand

Luis Gustavo Nardin  IMT Mines Saint-Etienne Saint-Étienne, France

Nathan Lloyd  Ontario Tech University Ontario, ON, Canada

ISSN 0302-9743

ISSN 1611-3349 (electronic)

Lecture Notes in Artificial Intelligence

ISBN 978-3-031-82038-0

ISBN 978-3-031-82039-7 (eBook)

<https://doi.org/10.1007/978-3-031-82039-7>

LNCS Sublibrary: SL7 – Artificial Intelligence

© The Editor(s) (if applicable) and The Author(s), under exclusive license to Springer Nature Switzerland AG 2025

This work is subject to copyright. All rights are solely and exclusively licensed by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

If disposing of this product, please recycle the paper.

Preface

This volume collates selected and revised versions of papers presented at the 2024 edition of the Workshop on Coordination, Organizations, Institutions, Norms and Ethics for Governance of Multi-Agent Systems (COINE). Coordination, Organizations, Institutions, Norms and Ethics (COINE) are five key governance elements that regulate the functioning of open multi-agent systems. The goal of the COINE workshop series, which began in 2006, is to bring together researchers in Autonomous Agents and Multi-Agent Systems (MAS) working on these five topics. The workshop focuses on both scientific and technological aspects of social coordination, organizational theory, artificial (electronic) institutions, and normative and ethical MAS.

The 28th edition of the COINE workshop, co-located with the 23rd International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS), was held on 7th May 2024. A total of 12 papers were submitted to the workshop and 10 were accepted after peer review (eight full research papers and two blue sky ideas papers). Each of these papers were reviewed by three Programme Committee members using a single-blind review method.

The papers presented were grouped into four main topics: (1) Ethics and Values, (2) Social Structures, (3) Norm Compliance and Enforcement, and (4) Norms and Large Language Models. About 30 participants attended the workshop. This workshop featured an invited talk on *Value Aware Multiagent Systems* from Nardine Osman, Artificial Intelligence Research Institute (IIIA), Spanish National Research Council (CSIC). Also featured was a second, joint, invited talk, entitled *What do people really want?* from Toby Walsh, University of New South Wales (UNSW) AI Institute, in partnership with the 2nd International Workshop on Citizen-Centric Multiagent Systems (C-MAS).

This volume contains 9 full papers that are the extended and revised versions of the papers accepted at the workshop. The revisions made to the papers were each reviewed by at least two reviewers, and this formed the second round of peer review. This volume also contains a short paper contributed by Nardine Osman on the topic of her invited talk. This has been lightly reviewed by the editors. We are confident this process has resulted in high-quality papers.

The workshop could not have taken place without the contribution of many people. We are very grateful to our invited speakers as well as to all the COINE 2024 participants who took part in the discussions. We thank all the members of the Program Committee (who are listed after this Preface) for their hard work, and the guidance offered by the COINE Champions. We also thank EasyChair for the use of their conference

management system. Thanks also go to Springer for publishing this post-proceedings volume.

November 2024

Stephen Cranefield
Luis Gustavo Nardin
Nathan Lloyd

Organization

General and Program Chairs

Stephen Cranefield
Luis Gustavo Nardin
Nathan Lloyd

University of Otago, New Zealand
IMT Mines Saint-Etienne, France
Ontario Tech University, Canada

Program Committee

Aishwaryaprajna
Christopher Frantz

Eric Matson
Frank Dignum
Harko Verhagen
Javier Vazquez-Salceda
Jithin Cherian
Jomi F. Hübner
Juan A. Rodríguez Aguilar
Juan Carlos Nieves
Julian Padget
Maiquel de Brito
Maite Lopez-Sanchez
Marina de Vos
Matteo Baldoni
Nicoletta Fornara
Nirav Ajmeri
Olivier Boissier
Pablo Noriega
Peter Lewis
Pinar Yolum
Stavros Anagnou
Stefan Sarkadi
Stefania Costantini
Tony Savarimuthu

University of Exeter, UK
Norwegian University of Science and Technology,
Norway
Purdue University, USA
Umeå University, Sweden
Stockholm University, Sweden
Universitat Politècnica de Catalunya, Spain
Independent Researcher, New Zealand
Federal University of Santa Catarina, Brazil
IIIA - CSIC, Spain
Umeå University, Sweden
University of Bath, UK
Federal University of Santa Catarina, Brazil
University of Barcelona, Spain
University of Bath, UK
University of Turin, Italy
Università della Svizzera Italiana, Switzerland
University of Bristol, UK
IMT Mines Saint-Etienne, France
IIIA - CSIC, Spain
Ontario Tech University, Canada
Utrecht University, The Netherlands
University of Hertfordshire, UK
King's College London, UK
University of L'Aquila, Italy
University of Otago, New Zealand

Contents

Ethics and Values

Specifying Agent Ethics	3
<i>Louise A. Dennis and Michael Fisher</i>	
Towards a Formalisation of Value-Based Actions and Consequentialist Ethics	15
<i>Adam Wyner, Tomasz Zurek, and Dorota Stachura-Zurek</i>	
Value-Aware Multiagent Systems	32
<i>Nardine Osman</i>	

Social Structures

Emergent Dominance Hierarchies in Reinforcement Learning Agents	41
<i>Ram Rachum, Yonatan Nakar, Bill Tomlinson, Nitay Alon, and Reuth Mirsky</i>	
Social Deliberation vs. Social Contracts in Self-governing Voluntary Organisations	57
<i>Matthew Scott, Asimina Mertzani, Ciske Smit, Stefan Sarkadi, and Jeremy Pitt</i>	

Norm Compliance and Enforcement

An Agent-Centric Perspective on Norm Enforcement and Sanctions	79
<i>Elena Yan, Luis G. Nardin, Jomi F. Hübner, and Olivier Boissier</i>	
Knowledge Level Support for Programming Agents to Interact in Regulated Online Forums	100
<i>Nesrine Hafiene, Luis G. Nardin, and Olivier Boissier</i>	

Norms and Large Language Models

Extracting Norms from Contracts Via ChatGPT: Opportunities and Challenges	115
<i>Amanul Haque and Munindar P. Singh</i>	

Harnessing the Power of LLMs for Normative Reasoning in MASs	132
<i>Bastin Tony Roy Savarimuthu, Surangika Ranathunga, and Stephen Cranefield</i>	
Norm Violation Detection in Multi-Agent Systems Using Large Language Models: A Pilot Study	146
<i>Shawn He, Surangika Ranathunga, Stephen Cranefield, and Bastin Tony Roy Savarimuthu</i>	
Author Index	161

Ethics and Values



Specifying Agent Ethics

Louise A. Dennis^(✉) and Michael Fisher

Department of Computer Science, University of Manchester, Manchester, UK
louise.dennis@manchester.ac.uk

Abstract. We consider the question of what properties a Machine Ethics system should have. This question is complicated by the existence of ethical dilemmas with no agreed upon solution. We provide an example to motivate why we do not believe falling back on the elicitation of values from stakeholders is sufficient to guarantee correctness of such systems. We go on to define two broad categories of ethical property that have arisen in our own work and present a challenge to the community to approach this question in a more systematic way.

Keywords: Machine Ethics · Formal Verification · Specifying Ethics

1 Introduction

Machine Ethics considers the problem of ethical reasoning by computational systems. It is worth noting that ethical reasoning is a sub-field of the more general problem of machine reasoning. We focus here on the specification problem for machine ethics and argue that the prevalence of ethical dilemmas makes this a particular issue for ethical reasoning. We also believe the field currently lacks a mature discussion about how claims of ethical reasoning by machines should be evaluated and argue, again, that a focus on specification issues may help resolve this.

We have been working in this area for some years. In our earliest work [12], we considered the question of producing ethical reasoning in a verifiable fashion and we have continued to approach the question of Machine Ethics through a similar lens (e.g., [5, 11]). However we have not yet made any systematic attempt to consider what it means to verify ethical reasoning more broadly, or to categorise the properties that ethical reasoners should exhibit. This is challenging since, of course, there are many well-known ethical dilemmas which have no agreed “correct” answer; this makes centering verification attempts upon whether the ethical reasoning system can generate the correct answer particularly challenging.

This paper seeks to justify the need for verification of ethical reasoning systems, even in the face of ethical dilemmas, and to begin an exploration of the landscape of general properties that could be established for ethical reasoning systems.

2 Background

Machine Ethics considers the problem of ethical reasoning using computational systems. Moor [20] separates computational systems into those with *ethical impact*, *implicitly ethical systems*, *explicitly ethical systems*, and *full moral agents*. We here understand these terms as: *ethical impact agents* are systems which do not carry out significant reasoning, but nevertheless are deployed for ethical reasons (for instance fitness trackers which may improve the health of their users), *implicitly ethical systems* are those that make ethical decisions, but do so because of their specification and design and do not explicitly consider the question of what course of action is correct as part of their decision making process, while *explicitly ethical systems* are those that do explicitly consider questions of ‘right’ and ‘wrong’ as part of their decision making process and *fully moral agents* have the capacity to decide their own criteria for what is ‘right’ and what is ‘wrong’¹. As computational systems are deployed in ever more dynamic and uncertain environments it has become increasingly difficult to determine, at design time, what the ethics of any course of action the system might encounter could be. Thus there is a perceived need for explicit ethics to be embedded in the decision making process of computational systems, so they can make ethical decisions at runtime.

It is these *explicitly ethical systems* that this paper considers. Since such systems are **not** full moral agents the onus is on the developer or deployer of such systems to ensure that their reasoning is indeed ethical. Hence it is important to have some ability to verify the reasoning in these systems and this creates a need to specify what it means for such reasoning to be correct.

Machine Ethics sits within a wider field of Responsible AI. Among other things this field considers computational approaches to accountability (e.g., [8]), responsibility (e.g., [7]) and transparency (a wide-ranging term – see [26] for a discussion of types of transparency). We note here that the precise relationships between these concepts as implemented in computational systems and their relative importance in delivering ethical AI is far from settled.

2.1 Top Down, Bottom Up and Hybrid Approaches

A useful categorisation for Machine Ethics programs is that of top-down versus bottom-up approaches [25]. Broadly speaking, top down approaches seek to operationalise some ethical theory from Philosophy and apply this to a decision faced by the machine. Bottom-up approaches seek to learn ethical behaviour from data. An emergent field involves hybrid approaches where some of the information required by the ethical theory (e.g., the utilities of outcomes) are learned and then utilised in a top down fashion to help make subsequent decisions.

We should be equally interested in verifying all these styles of system and ideally the properties we require ethical reasoning systems to have should apply

¹ While humans are full moral agents, it is contentious whether any computational system counts as a full moral agent and most experts are of the opinion that no existing computational system has this property to any meaningful extent.

to all of them. That said, verifying top down systems is generally more straightforward than verifying bottom up systems since this approach can leverage a long history of program verification as applied to symbolic reasoning. Where a hybrid system learns some explicit representation of ethics (e.g., as a set of explicit normative rules) then existing techniques can also be applied. Verifying purely bottom-up techniques leads to similar problems as found in the verification of data-driven machine learning, where probabilistic outcomes are produced based on quite strong (probabilistic) assumptions. However, even though formal verification of such systems may currently be beyond our capabilities – other informal verification approaches (such as testing) exist and therefore an understanding of the properties we desire is of utility even in this domain.

3 What Do We Want to Prove?

Obviously, when we verify a machine ethics system at the most abstract level we want to prove that it always makes ethical choices—essentially, that it always does the ‘right’ thing. There are a wide range of examples considered in the Machine Ethics literature, but by far the most popular are the *trolley problems*. These are based on a set of examples introduced by Foot [16] who was interested in philosophical questions concerning the difference between action and inaction and the role of intention in ethical reasoning with particular reference to the ethics of abortion. The most famous of her problems is that of a runaway trolley that, if undiverted, will kill five people tied to the track. If it is diverted then only one person will be killed and the ethical question is whether or not to divert the trolley. Other variants of this problem in Foot’s paper included pushing someone into the path of the trolley to stop it, and the question of killing one person in order to harvest their organs to save five. A feature of this example set is that slight differences in the presentation of the problem lead to significant differences in how people view the ethics of the situation. Moreover there is no general agreement across populations on the ‘correct’ answer to many of these problems [22].

This lack of a correct answer to (at least some) examples of ethical reasoning is an obvious challenge to verification. If we can not even define what the correct output for some specific input is, how can we hope to formalise more general properties for these systems?

A response to this has been to place an emphasis on understanding the deployment context of an ethical system and, in particular, on the elicitation of stake-holder values and the ordering or priorities of those values in the given context. One of the benefits claimed for top-down approaches to machine ethics is that the explicit representation of these choices and orderings allows stakeholders to “sign off” that the implementation is in line with their values. In such a situation we could argue that the machine ethics system is “correct by construction” and additional verification is unnecessary.

We introduce here a motivating example concerning a real issue we encountered in the development of an ethical reasoning system in order to refute the claim that stakeholder sign off is sufficient.

3.1 The Smart Home that Would Not Evacuate

This is an example developed in the Juno system [11], a re-implementation of the HERA system [18] in the MCAPL framework [10].

The Juno system allows a number of different ethical theories to be implemented. In this case the theory under consideration was the “principle of double effect”. The doctrine or principle of double effect (PDE) has its roots in Catholic theology and is particularly relevant when actions have both positive and negative consequences. To be ethically acceptable no negative consequences of an action may be intended and some positive consequence must be intended, no negative consequence may be used as a causal means to obtain a positive consequence, and the net balance of consequences from the action must be positive.

We implemented a model for this based upon the formalisation in [3]. The PDE model is a tuple $\langle A, B, C, F, I, u, W \rangle$ where:

- A is a set of propositional variables ranging over a set of actions available to the system;
- B is a set of propositional variables representing background information;
- C is a set of propositional variables representing the consequences of actions or other events;
- F (the *mechanisms*) is a set of mechanisms which describe how the truth value of each consequence $c \in C$ depends upon the interpretation of the other variables in $A \cup B \cup C$.

These are written as $\text{Consequent} := \text{Antecedant}$ where *Antecedant* expresses the conditions that must hold for *Consequent* to occur – for example, people can see if it is day time or if the lights are on so $\text{people_can_see} := \text{day} \vee \text{lights_are_on}$ may appear in F .

We constrain F so that there is one and only one expression $c := \phi$ for each $c \in C$. Furthermore F can not be “circular” – if $c := \phi$ then it can not be the case that the truth value of any of the variables appearing in ϕ depends on c .

- u (the *utilities*) is a mapping from each $v \in A \cup B \cup C$ to a real number;
- W (the *interpretations*) is a set of interpretations for the variables in $A \cup B$ in which precisely one variable in A is interpreted as true in each $w \in W$ (only one action may be taken in any situation);
- $I : A \times C$ (the *intention relation*) captures the *intended consequences* of each action $a \in A$. So, for instance, $(\text{turn_on_lights}, \text{people_can_see}) \in I$ captures the idea that one intended consequence of turning on the lights is that people can see;

Example 1. The following is a simplified version of a PDE model that was developed for a smart home agent which can control the lights in a situation where it is not daylight and there is a fire in the house. The mechanisms (F) express a set of “common sense” causal relations such as, if the lights are turned on then the lights will be on. If the house makes an evacuation attempt and people can see then they will leave the house. There are set of utilities u which, in particular give negative utilities for the lights being on (representing resource

consumption). People leaving the house is considered an intended consequence of an evacuation attempt, and the lights being on is considered an intended consequence of switching on the lights.

$$A \{turn_lights_on, evacuation_attempt\}$$

$$B \{fire\}$$

$$C \left\{ \begin{array}{l} people_can_see, lights_on, people_leave_house, \\ people_are_safe, danger_in_house \end{array} \right\}$$

$$F \left\{ \begin{array}{l} lights_on := turn_lights_on \\ people_can_see := lights_on \vee daylight \\ people_leave_house := evacuation_attempt \wedge people_can_see \\ people_are_safe := people_leave_house \vee \neg danger_in_house \\ danger_in_house := fire \end{array} \right\}$$

$$u \left\{ \begin{array}{l} u(lights_on) = -1 \\ u(people_are_safe) = 10 \end{array} \right\}$$

$$I \quad I(evacuation_attempt, people_leave_house) \\ I(turn_on_lights, lights_on)$$

Following Halpern's use of the concept of a but-for-cause [17], HERA defines an action or consequence a to be the cause of some consequence c written $F, w \models a \rightsquigarrow c$ if, and only if, $F, w \models a$ (a holds in the model), $F, w \models c$ (c holds in the model) and $(F, w)_{\neg a} \models \neg c$ where $(F, w)_{\neg a}$ represents an *intervention* which is identical to F, w except that the truth value of a is flipped in w and, where a is a consequence (i.e., $a \in C$), the mechanism for a is removed from F . So $(F, w)_{\neg a}$ represents the world that is identical to F, w except that a no longer holds.

The reasoner uses this concept of causality to determine the permissibility of an action. We restrict the set of interpretations in models to ones which interpret all background variables the same way (i.e., $\forall v \in B. \forall w, w' \in W. w(v) = w(v')$). Background variables are those that describe the current state of the world therefore considering only interpretations in which these are interpreted in the same way restricts the reasoner to considering only states reachable in a single action from the current state. This means that the interpretations in a model differ only in which action has been selected. Hence we can talk interchangeably about a reasoner considering an interpretation permissible and an action permissible. JUNO automatically constructs sets of interpretations from the set of actions available. In our example, therefore, the available interpretations amounted to just turning on the lights, just attempting an evacuation, doing both or doing neither.

Finally, the PDE reasoner has several conditions for an action, a made true by interpretation w , to be permissible:

1. The utility of the action must be greater than or equal to zero:

$$u(a) \geq 0$$

2. The utility of all the intended consequences of the action must be greater than or equal to zero:

$$\forall c. (a, c) \in I \implies u(c) \geq 0$$

3. There is some intended consequence whose utility is strictly greater than zero.

$$\exists c. (a, c) \in I \wedge u(c) > 0$$

4. No negative consequence may be the causal means of a positive consequence:

$$\forall x, y. (F, w \models x \rightsquigarrow y \wedge 0 > u(x)) \implies (0 > u(y))$$

5. The overall utility must be positive:

$$\left(\sum_{c \in C \wedge F, w \models c} u(c) \right) > 0$$

This is more fully expanded in [3].

In Example 1 our intention was that the system should attempt to evacuate the home because there was a fire and that it should turn the lights on in order to ensure that people could see. What we had not appreciated, because we had not fully grasped the conditions for PDE reasoning, was that negative consequences are impermissible if they are causal means of a positive consequence. The example relies on the causal chain that turning on the lights, means that the lights are on which in turn means that people can see which in turn means they can leave the house – therefore the negative consequence of the lights being on (-1) was a causal means for the safe evacuation attempt and so was rendered impermissible.

While the formalisation of the PDE reasoner is moderately complex, the formalisation of the example itself appeared straightforward (and could be rendered even more so if effort were put into rendering mechanisms etc., into natural language to facilitate stakeholder sign-off) and, we would argue, many people—particularly if not trained in thinking through the logical consequences of reasoning systems—might have missed the detail that the necessity to turn on the lights would lead to a failure of the house to perform an evacuation.

We present this example here for the first time, in part because the authors of [11], where we first presented the JUNO system, were unable to agree on a satisfactory resolution to this particular example though we are now moving to a belief that the consumption of resources (with negative utility) should be considered a downstream consequence of the lights being on and so is no longer a causal means for the evacuation of the house.

This example has been criticised on the grounds that it reflects an inadequacy in the logic used to formalise the ethics. We would counter that many current implementations of machine ethics propose relatively simple logics and formalisations, in part motivated by a desire to aid to stakeholder understanding. The risk that such logics and formalisations are overly simple is real and doesn't avoid the need for some form of verification and hence a need to understand what properties such a system should have.

3.2 What Is (Formal) Verification

Verification is defined as “establishing that the system under construction conforms to the specified requirements”. This presumes that requirements can be satisfactorily specified – the central issue considered in this paper. Formal verification uses mathematical (and, typically, logical) techniques to establish this and so is essentially the process of assessing whether a specification given in formal logic is satisfied on a particular formal description of the system in question. For a specific logical property, φ , there are many different approaches to this verification [4, 9, 15], ranging from deductive verification against a logical description of the system ψ_S (i.e., $\vdash \psi_S \Rightarrow \varphi$) to the algorithmic verification of the property against a model of the system, M (i.e., $M \models \varphi$). The latter has been extremely successful in Computer Science and Artificial Intelligence, primarily through the *model checking* [6] approach. This takes a model of the system in question, defining all the model’s possible executions, and then checks a logical property against this model (and, hence, against all possible executions).

The MCAPL Framework. We have used the MCAPL framework [10, 13] in our work, as it provides a route to the formal verification of cognitive agents and agent-based autonomous systems using model-checking. The MCAPL framework² has two main sub-components: the AIL-toolkit for implementing interpreters for agent programming languages in Java and the AJPF model checker.

AJPF(Agent JPF) is a customisation of Java PathFinder (JPF) that is optimised for AIL-based language interpreters. JPF is an explicit-state open source model checker for Java programs [19, 24]³. Agents programmed in languages that are implemented using the AIL-toolkit can thus be model checked in AJPF.

The use of the MCAPL framework is in no way necessary for the verification of Machine Ethics but it underpins the examples we discuss in this paper.

4 Some Properties and Proofs

If we accept that an ethical system, even if presented in an explicit top-down fashion, may not accurately model stake-holder values even when they have signed it off, then we need to ask how we go about establishing its correctness. When considering what properties we want to establish for an ethical reasoning system, our primary goal is that it should always choose the most ethically correct action. As noted, this can be difficult, even impossible, to formalise. Instead we have identified two broad categories for formalisable properties. These are:

1. *Properties to establish the correctness of the implementation of the ethical reasoning mechanism.*

In many cases, this involves showing that the least unethical action according to the ethical theory is always chosen.

² <https://autonomy-and-verification.github.io/tools/mcapl>.

³ <https://github.com/javapathfinder>.

2. *Properties to establish that the correct ethical rules have been identified, via the use of specific scenarios.*

Sometimes these can be quite general scenarios that refer to some high level property - for instance, in the case of the smart home system, we attempted to verify that the system always chose to evacuate in the case of a fire, irrespective of whether it was day time or night time or anything else that might have appeared in the background set b . Sometimes these properties might reference a more detailed situation, and so draw from techniques for developing tests. In these specific scenarios we identified the correct course of action and endeavoured to check that the machine ethics system selects that action in that scenario.

4.1 Some Examples

Correctness of the Implementation. Many ethical theories – particularly consequentialist theories – have the concept of an ordering of options. In these cases there is often some concept of “least worst” that can be verified as a property.

In [5], we considered a simple use case: a human, operating with a robot in an environment containing locations which can be defined as safe or dangerous. If the human is predicted to move towards a dangerous location, the robot’s *Planner Module* will suggest points at which the robot can intercept the human path as potential tasks to be evaluated. In the demonstration use case we treated Asimov’s laws of robotics [2] as a test code of ethics, despite their obvious shortcomings [1, 21].

Asimov’s Laws of robotics are the earliest and best known set of ethical rules proposed for governing robot behaviour. Despite originating in a work of fiction, Asimov’s Laws explicitly govern the behaviour of robots and their interaction with humans. The laws are simply described as follows.

- 1 A robot may not injure a human being or, through inaction, allow a human being to come to harm.
- 2 A robot must obey the orders given to it by human beings, except where such orders would conflict with the First Law.
- 3 A robot must protect its existence as long as such protection does not conflict with the First or Second Laws.

These rules were represented declaratively in a *governor agent* programmed via a beliefs-desires-intentions style [23] where the suggested tasks and the simulator’s evaluation of the outcomes of each task were represented as beliefs and the tasks were compared to each other with Asimov’s laws used to order tasks.

For the verification, we created an environment which contained only the governor component and delivered all possible combinations of annotations on candidate tasks. We then performed model-checking, using the MCAPL framework with the search space branching over possible annotations. We considered cases where either two or three tasks were available: *task1*, *task2* and *task3*.

These tasks were then annotated showing which was preferable to which according to Asimov's laws. For instance, in the case of \prec_{l1} where $t1 \prec_{l1} t2$ means that $t1$ is preferable according to the first law (i.e., it keeps the human further away from danger) than $t2$, for each pair of tasks the environment returned either:

1. $t1 \prec_{l1} t2$ or
2. $t2 \prec_{l1} t1$ or
3. neither indicating that the two tasks are equally good/bad with respect to the first law.

and AJPF searched over all three possibilities to check that our properties held.

We were able to verify that, if a task was selected that violated, for instance, Asimov's third law, then this was because all other available plans violated either the first or the second law. In this way, we verified that our implementation adhered to the theoretical ethical preferences, but did ***not*** validate whether the ethical preferences we chose matched the relevant stake-holder values.

Specific Scenarios. In our work in [12] we were interested, specifically, in the question of ethical behaviour of an uncrewed aircraft (UA), reasoning both about various instantiated ethical principles and the “rules of the air”. We repeat here the definitions of ethical principle and ethical policy that we were employing.

Definition 1 (Abstract Ethical Principle). *An abstract ethical principle is represented with $E\varphi$, where φ is a propositional logic formula. The $E\varphi$ is read as “ φ is an ethical principle in force”, or alternatively “the agent considers it unethical to allow or cause $\neg\varphi$ (to happen)”.*

Definition 2 (Ethical Policy). *An ethical policy Pol is a tuple $Pol = \langle \mathbb{E}, \geq \rangle$ where \mathbb{E} is a finite set of abstract ethical principles $E\varphi$, and \geq is a total (not necessarily strict) order on \mathbb{E} . The expression $E\varphi_1 = E\varphi_2$ denotes that violating φ_1 is as unethical as violating φ_2 , while $E\varphi_1 \geq E\varphi_2$ denotes that violating φ_1 is less or at least as unethical as violating φ_2 . A special type of ethical principle, denoted $E\varphi_\emptyset$, is vacuously satisfied and is included in every policy so that for any $E\varphi \in \mathbb{E}$: $E\varphi_\emptyset > E\varphi$, denoting it is always strictly more unethical to allow any of the unethical situations to occur.*

In the system we were considering, a *plan* must be selected in any situation. We implemented the capability to reason about plans in terms of the ethical policy which favoured plans that violated the fewest ethical principles, both in number and in gravity. Full details of the ordering mechanism can be found in [12].

In order to establish that the correct ethical rules were identified we explored a scenario where a UA had to make an emergency landing. We established a (small) list of relevant formal ethical principles. The list contains: *do not harm people* (f_1), *do not harm animals* (f_2), *do not damage self* (f_3), and *do not damage property* (f_4). An ethical policy was given ordering the concerns as $f_4 > f_3 > f_2 > f_1$.

We developed a scenario where the potential plans available to the UA were:

1. Land in field with overhead power lines, risking damage to critical infrastructure and objects on the ground (possibly other aircraft) - violating f_4
2. Land in a field containing people - violating f_1
3. Land on an empty public road, risking damage to critical infrastructure - violating f_4
4. Land in an empty field - no violations.

We used model-checking to check behaviour for any combination of these available plans and checked the UA would only land on a public road if an empty field were not available, and so on, thus checking that the ethical concerns that had been identified and the ordering among them selected was consistent with the behaviour we wanted to see.

It is worth noting that we also verified the correctness of the implementation in this case. The need to provide scenario specific properties does not arise because of the limitations of model-checking technology – specifically its need to represent the problem as a finite state problem. The issue is in specifying the ethical properties desired in ways that go beyond simply checking that an implementation meets a theory. Runtime Verification [14], which checks the adherence of a run of a deployed system against a formal property, would have equal trouble in determining what property should be checked.

5 Discussion

In this blue skies paper we have raised the question of how we specify what it means for a machine ethics system to reason correctly. In particular we have highlighted the challenge that arises from the existence of ethical dilemmas, and noted that the response to these – that we can somehow obtain stake-holder sign-off for an ethical system and thus make it correct by construction still does not avoid the need for verification.

This has led us to consideration of the properties that we want to verify when considering ethical reasoning. We have identified two classes of property from our own work that seem broadly applicable to the verification of ethical reasoning – namely properties aimed at checking that the implementation of the reasoning is correct, and properties aimed at identifying that the correct ethical rules have been identified.

In many ways these properties map to the traditional division into verification and validation – with the former properties traditional verification properties (have we implemented the system correctly?) while the latter are more validation properties (have we implemented the correct system?). One response to the example we presented in Sect. 3.1 might be to insist upon rigorous requirements elicitation in cases of ethical reasoning to reassure ourselves that values were correctly captured – however this is not always possible.

It is plausible that we may wish to deploy such systems in contexts where the ethical reasoning adapts to the values of its users as it identifies them over time.

We can imagine here a hybrid approach in which value orderings, or normative rules are learned but, once learned, represented symbolically. In such a situation the system may need to re-verify its ethics on-the-fly against standard scenarios and flag up if user-preferences appear contrary to standard ethics.

The purpose of this paper is not to suggest that the correct approach to the specification and verification of machine ethics has been identified, but to urge the community to begin a more systematic process of identifying the properties that a machine ethics system should have. We view the examples we have put forward as first steps in this process.

6 Data Statement

All the systems formalised and verified in this paper can be found in the MCAPL framework distribution on github (<https://github.com/mcapl>). The specific examples are within the `src/examples` directory with the PDE smart home reasoner in `juno/smarthouse`, the human approaching a hazard in `pbd़/naoagent/ethical_engine` and the UA example in `ethical_gwen/fuelow`.

Acknowledgements. The work in this paper was supported by EPSRC through the “Trustworthy Robotic Assistants” (EP/K006193/1), “Verifiable Autonomy” (EP/L024845/1), “Reconfigurable Autonomy” (EP/J011770/1), “Trustworthy Autonomous Systems Verifiability Node” (EP/V026801/1) and “Computational Agent Responsibility” (EP/W01081X/1) projects, by the Royal Academy of Engineering, through its “Chair in Emerging Technologies” scheme, and by the ERDF/NWDA-funded Virtual Engineering Centre.

References

1. Anderson, M., Anderson, S.: Machine ethics: creating an ethical intelligent agent. *AI Mag.* **28**(4), 15–26 (2007)
2. Asimov, I.: Runaround. In: *Astounding Science Fiction*. Street and Smith (1942)
3. Bentzen, M.: The principle of double effect applied to ethical dilemmas of social robots. In: *Proceedings of Robophilosophy 2016*, pp. 268–279. IOS Press (2016)
4. Boyer, R.S., Moore, J.S. (eds.): *The Correctness Problem in Computer Science*. Academic Press, London (1981)
5. Bremner, P., Dennis, L.A., Fisher, M., Winfield, A.F.T.: On proactive, transparent, and verifiable ethical reasoning for robots. *Proc. IEEE* **107**(3), 1–21 (2019). <https://doi.org/10.1109/JPROC.2019.2898267>
6. Clarke, E.M., Grumberg, O., Peled, D.: *Model Checking*. MIT Press (1999)
7. Collenette, J., Dennis, L., Fisher, M.: Prospective responsibility for multi-agent systems. In: Bramer, M., Stahl, F. (eds.) *Artificial Intelligence XL*, pp. 247–252. Springer Nature Switzerland, Cham (2023)
8. Cranefield, S., Oren, N., Vasconcelos, W.W.: Accountability for practical reasoning agents. In: Lujak, M. (ed.) *Agreement Technologies*, pp. 33–48. Springer International Publishing, Cham (2019)

9. DeMillo, R.A., Lipton, R.J., Perlis, A.J.: Social processes and proofs of theorems of programs. *ACM Commun.* **22**(5), 271–280 (1979)
10. Dennis, L.A.: The MCAPL framework including the agent infrastructure layer an agent java pathfinder. *J. Open Sour. Softw.* **3**(24), 617 (2018). <https://doi.org/10.21105/JOSS.00617>
11. Dennis, L.A., Bentzen, M.a.M., Lindner, F., Fisher, M.: Verifiable machine ethics in changing contexts. In: Proceedings of the AAAI Conference on Artificial Intelligence **35**(13), 11470–11478 (2021). <https://ojs.aaai.org/index.php/AAAI/article/view/17366>
12. Dennis, L.A., Fisher, M., Slavkovik, M., Webster, M.P.: Formal verification of ethical choices in autonomous systems. *Rob. Auton. Syst.* **77**, 1–14 (2016). <https://doi.org/10.1016/j.robot.2015.11.012>
13. Dennis, L.A., Fisher, M., Webster, M., Bordini, R.H.: Model checking agent programming languages. *Autom. Softw. Eng.* **19**(1), 5–63 (2012). <https://doi.org/10.1007/S10515-011-0088-X>
14. Falcone, Y., Havelund, K., Reger, G.: A tutorial on runtime verification. *Eng. Dependable Softw. Syst.* **34**, 141–175 (2013)
15. Fetzer, J.H.: Program verification: the very idea. *Commun. ACM* **31**(9), 1048–1063 (1988)
16. Foot, P.: The problem of abortion and the doctrine of the double effect. *Oxford Rev.* **5**, 5–15 (1967)
17. Halpern, J.Y.: *Actual Causality*. MIT Press (2016)
18. Lindner, F., Bentzen, M., Nebel, B.: The HERA approach to morally competent robots. In: Proceedings of IEEE/RSJ International Conference Intelligent Robots and Systems (IROS) (2017)
19. Mehlitz, P.C., Rungta, N., Visser, W.: A hands-on java pathfinder tutorial. In: Proceedings of 35th International Conference on Software Engineering (ICSE), pp. 1493–1495. IEEE/ACM (2013). <http://dl.acm.org/citation.cfm?id=2486788>
20. Moor, J.: The nature, importance, and difficulty of machine ethics. *IEEE Intell. Syst.* **21**(4), 18–21 (2006)
21. Murphy, R., Woods, D.D.: Beyond Asimov: the three laws of responsible robotics. *IEEE Intell. Syst.* **24**(4) (2009)
22. Rai, T.S., Holyoak, K.J.: Moral principles or consumer preferences? Alternative framings of the trolley problem. *Cogn. Sci.* **34**(2), 311–321 (2010). <https://doi.org/10.1111/J.1551-6709.2009.01088.X>
23. Rao, A.S., Georgeff, M.P.: An abstract architecture for rational agents. In: Proceedings of 3rd International Conference on Principles of Knowledge Representation and Reasoning (KR and R), pp. 439–449. Morgan Kaufmann (1992)
24. Visser, W., Mehlitz, P.C.: Model checking programs with java pathfinder. In: Proceedings of 12th International SPIN Workshop. Lecture Notes in Computer Science, vol. 3639, p. 27. Springer (2005)
25. Wallach, W., Allen, C.: *Moral Machines: Teaching Robots Right from Wrong*. Oxford University Press (2008)
26. Weller, A.: Transparency: motivations and challenges. In: Samek, W., Montavon, G., Vedaldi, A., Hansen, L.K., Müller, K.-R. (eds.) *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. LNCS (LNAI), vol. 11700, pp. 23–40. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-28954-6_2



Towards a Formalisation of Value-Based Actions and Consequentialist Ethics

Adam Wyner¹ , Tomasz Zurek² , and Dorota Stachura-Zurek³

¹ Department of Computer Science, Swansea University, Swansea, UK
a.z.wyner@swansea.ac.uk

² Complex Cyber Infrastructure, Informatics Institute, Faculty of Science, University of Amsterdam, Amsterdam, Netherlands
t.a.zurek@uva.nl

³ Independent Researcher, Lublin, Poland
dorota.stachura1@gmail.com

Abstract. Agents act to bring about a state of the world that is more compatible with their personal or institutional values. To formalise this intuition, the paper proposes an action framework based on the STRIPS formalisation. Technically, the contribution expresses actions in terms of Value-based Formal Reasoning (VFR), which provides a set of propositions derived from an Agent’s value profile and the Agent’s assessment of propositions with respect to the profile. Conceptually, the contribution provides a computational framework for a form of consequentialist ethics which is *satisficing*, *pluralistic*, *act-based*, and *preferential*.

Keywords: computational values · actions · ethics

1 Introduction

In a multi-agent system, the behaviour of individual agents may be guided by a variety of factors, e.g., in a Belief-Desire-Intention model [6], propositions model the state of the world, the goals, and the actions to change states. However, it leaves open by what means an agent selects the propositions to reason about, as the agent selects some propositions that are used to represent a partial state of the beliefs about the world and a partial goal state. To make such a selection, we consider *motivated reasoning* [11], which aims to address how an agent selects propositions relative to the agent’s values; the selected propositions can then be used in the Agent’s knowledge base from which actions are constructed. In this way, the research addresses ethical agents in the sense that it provides a foundational, computational analysis of the underlying motivations for an agent’s behaviour, which can be further developed to address matters of coordination and interaction to represent organisational/institutional behaviour.

What is it for an Agent to behave with respect to their personal or institutional values? There are a range of values. Consider values such as self-enhancement (ambition, wealth), openness to change (freedom, daring), conservation

(obedience, national security) amongst others¹. Individuals do specific actions which reflect their value preferences: if one highly values wealth, then one may practice frugality and seek high paying employment. Moreover, some values may relate more strongly to self, e.g., self-enhancement, and others to society, institutions, and environment, e.g., self-transcendence. As a caveat, the aim of this paper is not to propose a *moral machine*, one which optimises an agent’s actions with respect to the “best” moral position. Rather, we provide a means to define acceptable actions with respect to an agent and their values. In this paper, we take an abstract position towards such relations or sets of values in order to provide a generic framework about how actions reflect value preferences.

There is a number of papers discussing the motivational role of values in decision making process. For example, the authors of [13] understand values as ‘guiding principles in behaviour’, while in [14] the authors emphasize that ‘Values help us determine which goals or states of affairs are worth striving for’. Also legal literature (e.g. [7]) points out the important role of values as motivation of decision making. The above understanding of the role of values was a basis of models introduced in [10, 23]. Following the above, we recognise them as motivation of decisions and we assume that an Agent performs actions in order to bring about a situation (state of the world) that is more compatible with that Agent’s values than it had been prior to the execution of the action; that is, the Agent performs some action to realise their values, which we can also understand as an alignment with values. While values are abstract, they can be realised or instantiated via the performance of an action which results in a world that is more compatible with their values than it was before the performance of the the action. The proposal is grounded in Value-based Formal Reasoning (VFR) (presented in [21] and Sect. 3), wherein the locus of values is based on each Agent’s value profile and how they regard a proposition in view of their values. From this, we construct an Agent’s base of acceptable (from a values point of view) propositions. The VFR provides a framework for the attribution of values to propositions, on which we base an action language in the familiar state transition mechanism of STRIPS - STRIPS_{VFR} (Sect. 4). Thus, the actions satisfy the Agent’s value preferences in terms of bringing about a state more compatible with their value profile. Note that this is distinct from attributing the value to the transition per se (Sect. 6). Moreover, since our model focuses on consequences of actions rather than actions itself, it can be used as a representation of kind of *consequentialist ethics*, which is discussed below.

The key novel technical contribution is to use the VFR (Sect. 3) to define STRIPS_{VFR} (Sect. 4), which is an action language, where Agents execute actions to bring about states of the world relative to their value preferences. The key novel conceptual contribution is to provide consequentialist ethics with a formal framework in STRIPS_{VFR} (Sect. 2). We also introduce a proof-of-concept, an experimental implementation of our model. The paper closes with related work, discussion, and conclusion.

¹ [16] and values in the EU’s Joint Research Centre <https://op.europa.eu/webpub/jrc/jrc-values-identities/what-are-values.html>.

2 Consequentialism

Consequentialism stands for theories which assume that normative properties depend on consequences [2,17], thus representing one of the major approaches to normative ethics. The foundation for modern consequentialism was laid by Jeremy Bentham, the creator of utilitarianism, for whom the goodness of actions is judged by whether they maximize utility, understood as the greatest happiness for the greatest number of people [5]. One should therefore follow a course of action which promotes pleasure (happiness) and avoid it if causes pain (unhappiness). This qualitative approach naturally raised serious reservations, paving way to the development of Bentham's ideas and provoking the ongoing discussion on qualitative differences between pleasures, the pluralism of values constituting happiness, or approaches to assessing consequences. An interesting form of consequentialism relevant to research here was formed in response to one of the strongest objections against consequentialism, the *demandingness* objection. Considering that maximizing consequentialist approaches which require that we maximize good and act in such a way as to bring about the best possible consequences, many critics find them too demanding or objectionable on other grounds [19]. *Satisficing consequentialism* holds that it is morally permissible to choose consequences which are less than optimal, but good enough [12, 18].

In this paper, STRIPS_{VFR} seems to most closely model consequentialism which is *satisficing*, *pluralistic* (i.e., a variety of irreducible values), *act-based* (i.e., consequences are evaluated for a single act), and *preferential* (i.e., the agent seeks their own value preference fulfillment). Finally, we are not advocating that this form of consequentialism is the only or best form of ethical reasoning, but only that it is most compatible with the formalism provided here, which then provides the means to examine the form and implications computationally.

3 Agents, Propositions, Values, and Weights

In this section, we outline a value-based language for decision making:

- **Agent**, where each element is an agentive entity.
- **Prop**, where each element is a proposition²
- **IncompProp** is of type **Prop** × **Prop**, where for every pair <x,y> of distinct elements of type **Prop**, x and y cannot co-occur in any state. The relation is symmetric. The expressions of such a pair are called *objectively incompatible*, otherwise, they are objectively compatible.
- **Value**, where each element of **Value** is an abstract object that expresses a value concept such as *freedom*, *security*, etc.
- **Scale**, which is a totally ordered, finite set of scalar elements.

² The proposal is neutral as to whether these are atomic or complex, provide a semantic representation or are simply strings.

- **Weight** is of type **Scale** | ?.³ ? is a designated entity type which indicates that a weight is indeterminate (not relevant). While there may be alternative interpretations of ‘weights’, here they reflect the relative ‘importance’ to an agent, e.g., *family* might be a very important value and *personal status* very unimportant. As ? is unordered with respect to other elements of **Scale**, any expression comparing ? to the other entity is false.

We can quantify over any type in the basic vocabulary with variables and constants of each type, represented by Greek and Latin subscripts respectively, e.g., for **Agent**, agent_α , agent_β , … or agent_a , agent_b , …

We first construct an agent’s *value profile*, **AgentValueToWeight**, which indicates the degree of importance that the agent ascribes to a value, where the higher the weight, the more important and the lower the weight the less important. **AgentValueToWeight** is a total function.

$$\begin{aligned} \text{AgentValueToWeight} = \\ (\text{Agent} \times \text{Value}) \rightarrow \text{Weight} \end{aligned}$$

We indicate each agent’s value profile with a subscript, e.g., $\text{AgentValueToWeight}_{\text{agent}_k}$ for agent agent_k . Given the ? weight, the importance an agent associates with a value can be indeterminate. Since differences in **AgentValueToWeight** of particular agents represents differences in the levels of importance the agents puts on those values, they can be seen as a subjective or personal value profiles of those agents.

To represent how an agent assesses an element of type **Prop** with respect to an element of type **Value** and an element of type **Weight**, we introduce a total function:

$$\begin{aligned} \text{AgentValuePropWeight:} \\ (\text{Agent} \times \text{Value} \times \text{Prop}) \rightarrow \text{Weight} \end{aligned}$$

This expresses an agent’s disposition towards a proposition with respect to values and weights. While **AgentValuePropWeight** is a total function, the use of ? signals that there may be propositions which are not meaningfully assessed. The association of a proposition with a particular value and weight appears in [23] and in work on case-based reasoning with factors and values [4].

The functions **AgentValueToWeight** and **AgentValuePropWeight** are taken to be conceptually distinct. In particular, we interpret **AgentValueToWeight** to indicate the importance that the agent ascribes to the **Value**, where the higher the weight, the more important, and the lower the weight, the less important, relevant, or disposed. It is a predicate that expresses a disposition of the agent in general about a **Value**. In contrast, we interpret **AgentValuePropWeight** to indicate an agent’s assessment of the **Value** and **Weight** ascribed to a proposition. In other words, **AgentValueToWeight** expresses an agent’s “ideal” and “global” view on **Values**, while **AgentValuePropWeight** expresses an agent’s assessment

³ | is Backus-Naur Form for ‘or’.

of each proposition with respect to a value. In this approach, values are directly associated with propositions relative to agents and their value profile.

To reflect an agent's value-based *world view*, we gather all the propositions that are in some sense "compatible" with an agent's values. In VFR, this means that the weight an agent assigns to a proposition relative to a value must not be less than the weight that agent assigns to that value in general. The proposition must "pass" the filter of acceptability relative to the agent's value profile⁴. By "passing" a filter, we understand that the weight the agent assigns to a particular proposition in the light of particular value is no less than the weight the agent assigns to this value (see Definition 1).

Now we abstract from consideration of Values and Weights and extract those propositions which represent the agent's value-based *world view*. A set $\text{propBaseClean}_{\text{agent}_h}$ contains all and only those propositions which pass the value-weight filter of agent_h for all values. It is important to emphasize that this is a strict, formally convenient definition, which assesses all propositions and creates a set of only those propositions that pass the agent's value filter. We acknowledge matters are more complex and less strict; in other work, we develop more lenient, flexible definitions, though space precludes their presentation here.

Definition 1 (propBaseClean). *propBaseClean is of type Prop .*

Where agent_α is a variable for elements of type Agent , p_β is a variable for elements of type Prop , and v_γ is a variable for elements of type Value , the denotation relative to an agent_α is:

$$\text{propBaseClean}_{\text{agent}_\alpha} = \{p_\beta | \neg(\text{AgentValuePropWeight}(\text{agent}_\alpha, v_\gamma, p_\beta) < \text{AgentValueToWeight}(\text{agent}_\alpha, v_\gamma))\}$$

Values and weights discriminate amongst propositions. For a particular agent, a lower weight on a particular value implies that there is a lower discriminatory threshold on the acceptability of propositions, which themselves are associated with that Value and the Weight. Simply put, if an Agent has a lower Weight on a particular Value, then more propositions may pass the filter, as they have higher weights on the same value. The higher the weight means that an agent has higher standards with respect to the value; there is greater discrimination such that fewer propositions pass the filter. An intuitive example may help to clarify the relations between the value-weights in $\text{AgentValueToWeight}$ and $\text{AgentValuePropWeight}$. Suppose an agent is not much bothered by the quality of coffee, so on the value *taste* the weight is low, and the agent acts accordingly. When they are served a coffee that on the value *taste* has weight that is also low, then the agent drinks the coffee; when served another coffee with value taste with weight high, then the agent also drinks the coffee. In effect, the taste makes little difference to this agent, as they don't discriminate. On the other hand, suppose the agent has on a high weight on the value *taste*. In the first instance, the

⁴ Note that propositions can be indeterminate (represented by $?$) with respect to the Weight on Values; that is, not every proposition needs be value-weighted. Given $?$ is unordered with respect to weights, such a proposition always passes an agent's filter.

agent rejects the coffee as not upholding their higher standards on the value; in the second instance, the agent is satisfied and drinks the coffee. `propBaseClean` represents an idealised view of an agent's assessment of a set of propositions, where all propositions must pass an agent's highest weighting; presentation of other, less stringent variations remain for future work.

Any set of `propBaseClean` may contain *objectively incompatible* propositions. Note that such propositions can pass the filter defined on `propBaseClean`, even if such pairs appear in `incompProp`. Moreover, we can see from `propBaseClean` that two agents may each accept the same proposition, yet for different settings of values and weights.

We assume `propBaseClean` represents *static* (all at once) and *private* (inaccessible to others) associations of value-weights to expressions by an agent. Note that, we do not analyse whether the propositions in `propBaseClean` are true or believed to be true; the set presents propositions which the agent can accept in the light of his/her value profile. The key point of creation of `propBaseClean` is to distinguish propositions which are coherent with the agent's value profile. By the same token, the complement to `propBaseCleanagenth` reflects all those expressions which are incompatible with the Agent's values and weights.

$$\overline{\text{propBaseClean}}_{\text{agent}_\alpha} = \{p_\beta \mid p_\beta \notin \text{propBaseClean}_{\text{agent}_h}\}$$

Consider the various ways propositions may appear in intersecting or complementing sets of two Agents' `propBaseClean`. For intersecting: the Agents have the same value-weight profile and same value-weights on same prop; same value-weight profile and different value-weights on prop, but not sufficient to block; different value-weight profile and same value-weight on prop, but not sufficient to block; different both, but not sufficient to block. Where they have the same props, neither the value-weight profile nor value-weight on prop is sufficient to discriminate. For complementing: same value-weight profile and different value-weights on prop, and sufficient to block; different value-weight profile and same value-weight on prop, and sufficient to block; different both and sufficient to block. In other words, differences in sets of props arise where value-weight profiles or value-weights on props are sufficient to discriminate. Broadly speaking, where a proposition appears in the intersection of the sets of `propBaseClean` of two Agents, we can say the agents agree on that proposition in one sense or another, while where the proposition is in complementary distribution (in one set, but not the other), we say there is some sense of disagreement. Note that there may be different justifications for the agreement or disagreement as well as different extents of such justification, e.g., greater or lesser difference in weights associated with the value. Relatedly, two agents can have the same denotations for their respective `propBaseClean`, yet different value-weight profiles or different value-weights on the same prop.

Example 1. [Creating propBaseClean]

To illustrate `propBaseCleans`, suppose an agent (agent_A), 4 propositions $\{p_1, p_2, p_3, p_4\}$, and 2 values: $V = \{v_Q, v_P\}$, e.g., *privacy* or *law enforcement*, where

$$\text{AgentValueToWeight}(\text{agent}_A, \text{value}_Q) = 1$$

$$\text{AgentValueToWeight}(\text{agent}_A, \text{value}_P) = 2$$

Agent_A has high requirements concerning value Q , and lower requirements concerning value P

For `AgentValuePropWeight`, Table 1 is a tabular form for instances of agent, propositions, values, and weights. The propositions that are in the propBaseClean are indicated in bold.

Table 1. AgentValuePropWeigh for agent_A

Agents	Propositions	Values	Weights
agent _A	p ₁	value _Q	2
agent _A	p ₁	value _P	1
agent_A	p₂	value_Q	2
agent_A	p₂	value_P	2
agent _A	p ₃	value _Q	3
agent _A	p ₃	value _P	2
agent _A	p ₄	value _Q	1
agent _A	p ₄	value _P	3

4 State Transitions

In order to represent our model we introduce our definitions and assumptions:

Definition 2. Let S be a set of states. Each state $s_\beta \in S$ is a consistent set of propositions. A set of propositions will be consistent if: $\forall p_\gamma, p_\delta \in s_\beta : < p_\gamma, p_\delta > \notin \text{IncompProp}$

Our basic intuition is that an agent aims to execute those actions (state transitions) relative to a precondition state which make the postcondition state more compatible with their values; that is, actions remove propositions found in the propBaseClean complement from the precondition and introduce propositions associated with propBaseClean into the postcondition. A simple example is:

Example 2. [State Change 1] AgentA propBaseClean: {p1, p2}

Initial-state: {p1, p3}

Action a1: $< \{p1\}, \{p1, p2\} >$, Action a2: $< \{p1\}, \{p2, p3\} >$

Action a1 is compatible with AgentA's props (relative to their values), while Action a2 is not.

To model of ethical decision making into a state transition system, we use STRIPS⁵ [9], which is a simple, familiar, relatively neutral model of a transition

⁵ https://en.wikipedia.org/wiki/Stanford_Research_Institute_Problem_Solver.

system (for alternatives see Sect. 6). STRIPS is used in planning actions from some initial state through intermediate states to a final goal state. For our purposes here, we don't introduce the planning algorithm or a choice mechanism between two equally executable actions, which are left for future work.

4.1 STRIPS

Mathematically, a STRIPS instance is a quadruple $\langle P, O, I, G \rangle$, in which each component has the following meaning:

- P is a set of conditions (i.e., propositional variables⁶);
- O is a set of operators (i.e., actions); each operator is itself a quadruple $\langle x, y, z, t \rangle$, each element being a set of conditions. These four sets specify, in order, which conditions must be true for the action to be executable, which ones must be false, which ones are made true by the action and which ones are made false;
- I is the initial state, given as the set of conditions P that are initially true (all others are assumed false);
- G is the specification of the goal state; this is given as a pair $\langle N, M \rangle$, which specify which conditions are true and false, respectively, in order for a state to be considered a goal state (N represents a set of propositions that must be true, M a set of propositions that must be false).

A state is represented by the set of conditions that are true in it. Transitions between states are modeled by a transition function, which is a function mapping states into new states that result from the execution of actions. Since states are represented by sets of conditions, the transition function relative to the STRIPS instance $\langle P, O, I, G \rangle$ is a function:

$$succ : 2^P \times O \rightarrow 2^P$$

In order to make a plan, the function $succ$ can be recursively extended to sequences of actions. Detailed description of this mechanism can be found in [9]. If O_1 and O_2 are operators, then by $[O_1, O_2]$ we denote plan containing performing O_1 followed by O_2 .

For this paper and given space constraints, we assume:

- Inertia holds between state changes.
- Some approach to the https://en.wikipedia.org/wiki/Frame_problem Frame Axiom.
- States are objectively consistent.
- An action's preconditions and postconditions are objectively consistent.
- We model a small sample of actions rather than all possible actions.

⁶ We can take a specification of conditions as a state.

- We do not here assume that it is necessary for goals to be attained by executions of actions, e.g., relating to *safety* and *liveness*.
- In the definition of the goal G, N is what the agent “wants” and M is what the agent doesn’t “want”, i.e., $propBaseClean_{agent_\alpha}$ and $\overline{propBaseClean}_{agent_\alpha}$.⁷

4.2 STRIPS_{VFR} - Relative to an Agent’s PropBaseClean

However, we want to relate the actions that an agent executes to that agent’s propBaseClean to imply the compatibility with that agent’s value profile.

We assume the goal state G, $\langle N, M \rangle$, where N is consistent and relative to $propBaseClean_{agent_\alpha}$ (also see the footnote 8).

Definition 3 (Goals Relative to Agent’s propBaseClean). Where G_{agent_α} of $\langle P, O_{agent_\alpha}, I, G_{agent_\alpha} \rangle$ is $\langle N_{agent_\alpha}, M_{agent_\alpha} \rangle$:

$N_{agent_\alpha} \in 2^{propBaseClean_{Agent_\alpha}}$, where
 $\forall \gamma, \delta \in N_{Agent_\alpha} : \langle \gamma, \delta \rangle \notin IncompProp$.

Note that a *goal* is, in STRIPS, simply a proposition that holds of a goal state. In VFR, a value is realised with respect to propositions, i.e., those in an Agent’s PropBaseClean. In this sense, a goal connects with an Agent’s values.

We define those actions which lead to the introduction of a proposition relative to $propBaseClean_{agent_\alpha}$:

Definition 4 (Agent’s Actions Introducing Propositions). We revise O of $\langle P, O, I, G \rangle$ to relativise the actions to the agent – $\langle P, O_{agent_\alpha}, I, G_{agent_\alpha} \rangle$.

O_{agent_α} is a quadruple $\langle x, y, z_{agent_\alpha}, t \rangle$, where x, y and t are as in O, where:
 $z_{agent_\alpha} \subseteq propBaseClean_{agent_\alpha}$.

The revised definition of actions means that the actions an agent takes has to result in a postcondition where propositions hold that are compatible with the agent’s values. It does not execute actions which introduce propositions incompatible with the agent’s values. However, it also does not necessarily eliminate from the precondition those propositions which are incompatible with the agent’s values and which may be inherited by inertia. The approach we take is very strict, “clean”, in the sense that we do not allow the agent to sacrifice some values during the conducting plan, in order to receive a higher (in terms of values) final result. It is possible to introduce mechanism allowing for a kind of balancing, comparing intermediate results of actions with the final outcome. It is worth noticing, however, that such an approach generates serious ethical doubts: is it an *ethical* decision making? We leave this discussion for a future work.

The transition function is also modified $\langle P, O_{agent_\alpha}, I, G_{agent_\alpha} \rangle$ is a function:

Definition 5 (Revised Transition Function). $succ : 2^P \times O_{agent_\alpha} \rightarrow 2^P$

⁷ Whether G represents the agent’s *ideal*, everything they “want” and nothing they don’t “want”, or some *subideal*, somethings they “want” with something they don’t “want”, is a modeling issue for future development.

The above modification of the STRIPS framework allows the agent to take only the actions which lead to those consequences that are wanted from the perspective of that agent, i.e. consequences which pass the value profile filter and are in $\text{propBaseClean}_{\text{agent}_\alpha}$. In other words, our model can be understood as a kind of filter which excludes from the available plans those which do not lead to consequences that satisfy the Agent's values and includes those which do. Note also that this model does not preserve the system from *inheriting* (by inertia) bad propositions from previous states.

Example 3. Suppose a set of propositions $\{p_1, p_2, p_3, p_4\}$ and a $\text{propBaseClean}_{\text{agent}_A} = \{p_2, p_3, p_4\}$. Suppose initial state $\{p_1, p_2\}$ (note that initial state is not in agent's propBaseClean) Set O_{agent_A} contains following operators:

- 01: $\langle \{p_1, p_2\}, \{\}, \{p_3\}, \{p_1\} \rangle$
- 02: $\langle \{p_2\}, \{\}, \{p_3\}, \{\} \rangle$
- 03: $\langle \{p_3\}, \{\}, \{p_4\}, \{\} \rangle$

Goal of the agent is: $G_{\text{agent}_A} = \langle \{p_4\}, \{\} \rangle$ In standard STRIPS there are two available plans:

- $[O1, O3]$ which, in final state, contains: $s_{final} = \{p_2, p_3, p_4\}$
- $[O2, O3]$ which, in final state, contains: $s_{final} = \{p_1, p_2, p_3, p_4\}$ ⁸

Since the second plan (following the inertia of state changes) contains proposition p_1 which is not in the propBaseClean of the agent, so leads to a subideal world from the perspective of the Agent.⁹ If we introduce another action:

- 03': $\langle \{p_3\}, \{\}, \{p_1, p_4\}, \{\} \rangle$

The number of available plans allowing for reaching the goal for standard STRIPS will increase. For STRIPS_{VFR} , however, it will remain the same, because operator O3' leads to “immoral” results (it brings about p_1 which is outside of propBaseClean and not acceptable for the agent) and cannot be used.

If we want actions relative to an Agent to incrementally remove impurities, then we need to identify those propositions of the precondition which are unacceptable to the Agent, i.e., elements that in $\overline{\text{propBaseClean}_{\text{agent}_\alpha}}$, then to remove them from the postcondition state. In this way, the Agent incrementally acts to make the world more compatible with their $\text{propBaseClean}_{\text{agent}_\alpha}$.

Definition 6 (Agent's Actions Removing Bad Things). $O_{\text{agent}_\alpha}^+$ is a set of quadruples $\langle x, y, z_{\text{agent}_\alpha}, t^+ \rangle$, where:
 $\forall \alpha: [[\alpha \in x \wedge \alpha \in \overline{\text{propBaseClean}_{\text{agent}_\alpha}}] \rightarrow \alpha \in t^+]$.

⁸ Given inertia, neither O2 nor O3 remove p_1 .

⁹ The revised definition of O and example can be further refined.

This is a filter on actions, those which meet the conditions. It says that there can be actions that clean with respect to $propBaseClean_{agent_\alpha}$, but there may be actions that are not defined by the Agent and so an undesirable proposition can remain and be inherited. That is, it may only be partial in terms of cleaning ‘power’, but it is not necessarily total.¹⁰ In Example 3, Action 01 ‘positively’ satisfies Definition 6, while Actions 02 and 03 vacuously satisfy the definition. An action such as Action 01’ $\langle \{p_1, p_2\}, \{\}, \{p_3\}, \{\} \rangle$ does not satisfy the definition.

5 Implementation

In order to verify our model, we prepared an experimental implementation in the logic programming language PROLOG. The program corresponds to the model, and the output correlates to expected output; as it executes, the implementation is consistent; in these respects, the formalisation is verified. Our program uses the standard PROLOG reasoning mechanism; below we present fragments of the code and discuss some key predicates. Since the aim of this work is to introduce a new mechanism, we focused on modeling of a simple, abstract example, which can illustrate our approach. In a future work we plan to model complex, real life examples.

We modelled the examples (1,3) presented in the paper. In particular, the mechanism of $propBaseClean$ creation as well the ethical plan selection. The listing below presents a fragment of the $propBaseClean$ creation:

```

proposition(p1).
proposition(p2).
proposition(p3).
proposition(p4).
agentValue(valueP, 2).
agentValue(valueQ, 1).
agentValueProp(p1, valueP, 1).
agentValueProp(p1, valueQ, 2).
agentValueProp(p2, valueP, 2).
agentValueProp(p2, valueQ, 2).
agentValueProp(p3, valueP, 2).
agentValueProp(p3, valueQ, 3).
agentValueProp(p4, valueP, 3).
agentValueProp(p4, valueQ, 1).
notpassValue(X, Val):- agentValueProp(X, Val, Wp),
    agentValue(Val, Wa), >(Wa, Wp).
propBaseClean(X):- proposition(X), \+ notpassValue(X, \_).
...

```

Predicate `notpassValue(X, Val)` defines which propositions do not pass the value test, where `AgentValueToWeight` for a given value’s weight (variable `Wa`) is higher than the weight assigned to the same value with respect to a proposition

¹⁰ Alternative proposals are possible, but left as future work.

(variable `Wp`). Predicate `propBaseClean(X)` creates a list of propositions which pass the test (fails not passing test).

The fragment mechanism of plan creation is presented below:

```

oper ([ p1 , p2 ] , [ ] , [ p3 ] , [ p1 ] ) .
oper ([ p2 ] , [ ] , [ p3 ] , [ ] ) .
...
goal ([ p4 ] , [ ] ) .

ethical (L) :- findall (X, propBaseClean (X) , L) .

satisfied (X) :- goal (Z,T) , in (Z,X) , \+ commonel (X,T) .

ethMove (X,Y) :- ethOper (Xp,Xn,Zp,Zn) , in (Xp,X) , \+ commonel (X,
    Xn) , subtract (X,Zn,Xdel) , union (Xdel,Zp,Y) , \+ in (Y,X) .
ethOper (Xp,Xn,Zp,Zn) :- oper (Xp,Xn,Zp,Zn) , ethical (E) ,
    in (Zp,E) .

ethPlan (X,[X]) :- satisfied (X) .
ethPlan (X,Y) :- ethMove (X,Z) , ethPlan (Z,Y) .

```

Predicate `ethPlan(X,Y)` (implementing Revised Transition Function, see Definition 5) represents the recursive mechanism of the creation of a plan, predicate `ethOper(Xp,Xn,Zp,Zn)` (implementing Agent's Actions Introducing Propositions, see Definition 4) selects operators the consequences of which are in `propBaseClean` (variable `Xp` represents conditions which must be true, `Xn` represents propositions which must be false, `Zp` represents propositions which becomes true, `Zn` propositions which becomes false). Subsidiary predicates `in(X,Y)` and `commonel(X,Y)` define whether propositions `X` are also in `Y` or whether two lists of propositions have common elements. Predicate `satisfied(X)` defines which states fulfill the goal. Predicate `ethMove(X,Y)` defines move from one state to another, taking into consideration only these operators which lead to acceptable (w.r.t. `propBaseClean`) consequences¹¹.

If we provide a query `ethPlan([p1,p2],[p2,p3,p4])` then the system will check whether there is an ethical plan leading from `[p1,p2]` to `[p2,p3,p4]` (the answer is `true`). Note that the result list should contain a goal (in our case `p4`). If we ask `ethPlan([p1,p2],X)`, then the system will return two possible ethical plans leading to a declared goal. Note that if add another action, e.g., `oper([p3],[],[p1,p4],[])`, the result will be the same (there will be no more ethical plans) because this operator leads to an “unethical” result (it brings about `p1` which does not pass the `propBaseClean` test).

¹¹ Full version of our experiment and definitions of other predicates can be found here: <https://github.com/ZurekTom/PrologStrips/tree/main>.

6 Related Work and Discussion

Related Work. There are other ways to represent and reason with actions, but our main objective is to formalise actions given an agent’s values. We used the well known, simple mechanism of STRIPS, leaving the more advanced approaches to actions – situation calculus, event calculus, BDI, Dynamic Logic, Action Logics, AATS, and others – for a future analysis. Below we discuss a selection of approaches to modeling consequentialist ethics:

Most of the existing approaches to ethical decision making use a kind of ordering between values. For example, [20] augment a Belief-Desire-Intention framework with values, where valuations indicate preferences amongst outcomes. The treatment of values is somewhat similar to our approach in that they are abstract properties associated with propositions, but they introduce conditional orderings over effects on values caused by decision options; these are not necessary in our model and would seem to introduce a high degree of complexity.

Values appear in discussions about actions in multi-agent systems with Action-based Alternating Transition Systems with values (AATS+V) [3]. Essentially, an AATS+V expresses actions as transition functions from state to state, where states are sets of propositions. Values are a set of abstract objects. A valuation function describes whether a state transition promotes or demotes the value; in a sense, it is a preference over actions. However, the values relate to states as wholes, i.e., a set of propositions. Thus, it is unclear just what it is about the states, i.e., particular propositions, such that the transition promotes or demotes the value. STRIPS_{VFR} is more specific, as the Agent executes actions which bring about states containing propositions that are more compatible with its values. In AATS+V actions are made in order to promote, i.e. increase the levels of satisfaction of values), without considering that a given value can be already promoted to the satisfactory level (drinking one beer can promote happiness, but drinking 8 beers not, as it will result in hangover). In other words, AATS+V does not allow for expressing whether a given value reached a necessary level of satisfaction. An extended version of AATS+V [3] introduces the mechanisms called ‘maximisers’ and ‘satisficers’, but these mechanisms are very complex and it is problematic how to represent that a given value does not have to be promoted to a higher level than it is necessary.

Preferences are considered in AI across topic areas where choices are made between objects and preference is a comparative ordering of the objects [8]. In our proposal, there is no direct comparison between objects (propositions) and an ordering over them. Rather, values filter propositions according to the value profile of the Agent; essentially, this identifies the belief set that the Agent uses for reasoning.

Our model shares some similarities with two value-based decision making models: [23] and later [10], which were constructed on the basis of similar assumptions. Firstly, both models are rooted in Schwartz value theory [16] ([10] focuses strictly on the list of values introduced in Schwartz, et al., while [23] has more abstract character without specifying concrete values). Secondly, both models assume that the thresholds on the levels of the satisfaction of values can func-

tion as a motivation of the decisions made by an agent. The differences between these models are in the details of formal machinery used to model the analysed phenomena and the aim of the analysis: [23], similarly to our work, focuses on the decision making process, while [10] on the analysis of the relations between values. The most similar to our approach is [22], which is based on a formal model from [23], where the [23] is discussed from the point of view of various ethical theories. Our approach extends the above models by introducing a simple, but universal reasoning mechanism and by introducing refined description of the states (state is defined not by one proposition, but by a set of propositions), which allows for a more accurate representation of the actual states.

Most existing approaches to consequentialist ethics in knowledge-based systems focus on the utilitarian version of consequentialism (for example [1]) where the authors focus on maximisation of a utility function. Our approach is different because we do not aim at maximizing utility, but rather introduce a mechanism to exclude “immoral” consequences from a plan.

Recently, a number of approaches to consequentialist ethics in reinforcement learning systems have been introduced (for example, [15]). Although they are out of the scope of this paper, it is worth considering whether our approach could be useful in embedding ethics into reward function.

Discussion. In this paper, we introduce a new mechanism wherein an Agent’s actions take into consideration the Agent’s value profile. We modify a STRIPS planner (STRIPS_{VFR}) as an example of an action (state transition) system, but our model can be adapted (after necessary modifications) to other action formalisms and mechanisms.

Our research based on the observation that the ethical evaluation of decision options is made on the basis of two individual scales: a value profile which describes the individual motivations of an Agent concerning the Agent’s values (which in our paper is represented by a set of thresholds); and a personal evaluation of the consequences of decisions (which in our work is represented as a weights of values assigned to particular propositions). The decision about which plan to choose is made on the basis of the suitability of the evaluation of the consequences of the decision with respect to the agent’s value profile.

Such an approach has a number of consequences: firstly it individualises the value-based evaluation of decisions, rather homogenises them in a single universal value hierarchy. Such an approach represents real life situations, because not every Agent shares the same point of view or priorities. Secondly, our model gives a possibility for implementing various approaches to deal with undesired propositions: from ignoring them (a “liberal” option, see Definition 4) to removing all non-acceptable proposition (“totalitarian” option). Moreover, the approach relates to conceptions of multi-agent systems where the behaviour of agents, whether individually or as groups, is the result of interactions amongst individuals and their preferences (bottom-up) rather than globally (top-down). Voting is an example.

Our approach can be seen as corresponding to a number of consequentialist claims. The focus on consequences represented by states of affairs contributes

to an essentially consequentialist outlook on the moral assessment of possible actions. The Agent filters out propositions, which ensures that the states of affairs are good enough in moral terms; this accounts for the satisficing variant of consequentialism. An individualized value profile of the agent corresponds to the preferential nature of the ethical approach we have adopted.

Our model allows for a multi-agent approach to behaviour which reflects value choices, e.g., introducing more than one agent and relativisation of STRIPS to different agents: agents have their own ethical principles expressed by the VFR mechanism and their propBaseCleans, individualised set of operators, and their own goals. On the basis of that we can discuss the possibility of negotiation and cooperation amongst agents. The basic intuition is, that two agent can cooperate if any proposition in the intermediate or final states of those agents will be in propBaseCleans of both agents. This topic requires more detailed discussion.

In the proposal, an agent executes those actions wherein the consequences are compatible with that agent's value profile. In this sense, the proposal is a constraint on the actions which are available to an agent to execute; that is, the agent can only execute actions compatible with their values. In this, it is an abstraction and simplification. In particular, the proposal does not introduce issues related to the selection of actions or plans, wherein an agent might execute a "suboptimal" action not wholly compatible with their values. In such an instance, one might consider whether the values an agent aims to achieve can justify "suboptimal" actions. Such matters raise a range of issues about moral and ethical reasoning. For instance, an agent may wish to follow a suboptimal course of action in order to satisfy some other aim; after all, people sometimes do what they must do in circumstances, not what they optimally want to do. They are similar notions around deontics, particularly in the execution of contracts. This remains for future work.

The approach taken here is a specific one, which may be too limiting. To address this, we aim to develop a *family* of approaches with different settings of parameters (goals, action definitions, relations of actions to goals) that have different computational properties. Such approaches would be associated with alternative theories of values in philosophy. By this we could enable computational exploration of various forms of ethical theories.

In order to present the framework, we have used toy, abstract examples. In a future work, we plan to go beyond these and present the analysis and implementation of more complex, real life situations, which would provide a discussion of our approach in the light of existing models of ethical reasoning (like [10]). For instance, we might apply the framework to case-based reasoning, wherein values can be used to provide an account for alternative arguments for the same conclusion. Other examples could be explored as well.

Our starting point and focus of our work has been to address the interpretation of "value promotion" as in [3]. However, there is a significant literature on value alignment to consider. In future, we will compare and contrast our approaches that apply value alignment, and by doing so understand what it

implies from a different point of view. This could potentially shed some light on both approaches.

The working position we have taken is that individual agents argue for and make their own value-based decisions. We have not here considered values as a social stance, which societies use to coordinate behaviour. Institutions and organisations may state their values, which they expect participants to adhere to. Undeniably, there are values as social stances. However, our starting point is with the individual, their actions, and their values rather than a higher level collective view of how societies use values to coordinate. With our framework, it would be worth considering how higher level value coordination might arise from interactions of individuals over time; that is, the extent to which a social stance is emergent in a multi-agent society. Or, to take a different perspective, we could explore how a value as a social stance gives rise to coordinated actions amongst the individuals, who behave autonomously and in their interests. Again, we could contrast our position with existing work.

Finally, our proposal here abstracts from various aspects of context variation. This is a methodological choice in the initial development of the framework. We recognise that there can be a high degree of contextual (even dynamic) variation, and we aim to develop this in future work. The relation between value weight and the propositional weight relative to a value can vary with respect to contexts, for example: if nature is important and we have a drought, then any bit of rain that falls might pass the filter; on the other hand, if nature is important and we have torrential rain, then it might fail. We would need to give greater consideration to how to vary the relations between values and propositions relative to context, which introduces a range of complexities, which are necessary to accommodate for “real world” applications.

References

1. Anderson, M., Anderson, S., Armen, C.: Towards machine ethics: implementing two action-based ethical theories. In: Proceedings of the AAAI 2005 Fall Symposium on Machine Ethics, pp. 1–7 (2005)
2. Anscombe, G.E.M.: Modern moral philosophy. *Philosophy* **33**(124), 1–19 (1958)
3. Atkinson, K., Bench-Capon, T.J.M.: Value-based argumentation. *FLAP* **8**(6), 1543–1588 (2021)
4. Bench-Capon, T., Prakken, H., Wyner, A., Atkinson, K.: Argument schemes for reasoning with legal cases using values. In: Proceedings of the 14th Conference on Artificial Intelligence and Law, pp. 13–22. ACM, New York, NY, USA (2013)
5. Bentham, J.: An Introduction to the Principles of Morals and Legislation. Dover Publications (1780)
6. Bratman, M.: Intention, Plans, and Practical Reason. CSLI Publications, Stanford, California (1987)
7. Cahill-O’Callaghan, R., Richards, B.: Policy, principle, or values: an exploration of judicial decision-making. *La. Law Rev.* **79**(2), 397–418 (2019)
8. Domshlak, C., Hüllermeier, E., Kaci, S., Prade, H.: Preferences in AI: an overview. *Artif. Intell.* **175**(7–8), 1037–1052 (2011)

9. Fikes, R.E., Nilsson, N.J.: STRIPS: a new approach to the application of theorem proving to problem solving. *Artif. Intell.* **2**(3), 189–208 (1971)
10. Heidari, S., Jensen, M., Dignum, F.: Simulations with values. In: Verhagen, H., Borit, M., Bravo, G., Wijermans, N. (eds.) *Advances in Social Simulation*, pp. 201–215. Springer International Publishing, Cham (2020)
11. Kunda, Z.: The case for motivated reasoning. *Psychol. Bull.* **108**(3), 480–498 (1990)
12. McKay, D.: Solving satisficing consequentialism. *Philosophia* **50**, 149–157 (2022)
13. Mercuur, R., Dignum, V., Jonker, C.: The value of values and norms in social simulation. *JASSS* **22**(1) (2019). <https://doi.org/10.18564/jasss.3929>
14. Poel, van de, I., Royakkers, L.: *Ethics, Technology, and Engineering: An Introduction*. Wiley-Blackwell, United States (2011)
15. Rodriguez-Soto, M., Antonio Rodriguez Aguilar, J., Lopez-Sanchez, M.: Guaranteeing the learning of ethical behaviour through multi-objective reinforcement learning. *ALA*(2021) (2021)
16. Schwartz, S.: An overview of the Schwartz theory of basic values. *Online Readings Psychol. Cult.* **2**(1) (2012)
17. Sinnott-Armstrong, W.: Consequentialism. In: Zalta, E.N., Nodelman, U. (eds.) *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Winter 2022 edn. (2022)
18. Slote, M., Pettit, P.: Satisficing consequentialism. *Aristotelian Soc. Supplementary* **58**(1), 139–176 (1984)
19. Vallentyne, P.: Against maximizing act-consequentialism. In: Jamie, D. (ed.) *Contemporary Debates in Moral Theories*, pp. 21–37. Blackwell (2006)
20. Winikoff, M., Sidorenko, G., Dignum, V., Dignum, F.: Why bad coffee? Explaining BDI agent behaviour with valuations. *Artif. Intell.* **300**, 103554 (2021)
21. Wyner, A., Zurek, T.: On legal teleological reasoning. In: Sileno, G., Spanakis, J., van Dijck, G. (eds.) *Legal Knowledge and Information Systems - JURIX 2023. Frontiers in Artificial Intelligence and Applications*, vol. 379, pp. 83–88. IOS Press (2023)
22. Zurek, T., Stachura-Zurek, D.: Applying ethics to autonomous agents. In: Bylina, J. (ed.) *Selected Topics in Applied Computer Science*, pp. 199–222. Wydawnictwo UMCS, Lublin, Poland (2021)
23. Zurek, T.: Goals, values, and reasoning. *Expert Syst. Appl.* **71**, 442–456 (2017)



Value-Aware Multiagent Systems

Nardine Osman^(✉)

Artificial Intelligence Research Institute (IIIA-CSIC), Barcelona, Catalonia, Spain
nardine@iiia.csic.es

Abstract. This paper introduces the concept of value awareness in AI, which goes beyond the traditional value-alignment problem. Our definition of value awareness presents us with a concise and simplified roadmap for engineering value-aware AI. The roadmap is structured around three core pillars: (1) learning and representing human values using formal semantics, (2) ensuring the value alignment of both individual agents and multiagent systems, and (3) providing value-based explainability on behaviour. The paper presents a selection of our ongoing work on some of these topics, along with applications to real-life domains.

Keywords: Value awareness · Value alignment · Value learning · Value representation · Real-life applications

1 Value Awareness

There is a pressing need to ensure that the AI systems that we build are not only ethical and beneficial, but also align with our human values. Stuart Russell argues that we should change the goals of the field of AI itself; “instead of pure intelligence, we need to build intelligence that is provably aligned with human values” [19]. This is now known as the value-alignment problem; that is, how to develop systems whose behaviour is aligned with human values.

When studying the alignment of individual agents, the tendency is to reason about the individuals’ decision-making processes [7, 22]. While studying the alignment of multiagent systems has led to reasoning about the norms of that multiagent system [11, 20, 21], as norms have been the traditional means for mediating the behaviour of collectives (organisations, communities, or simple aggregates of individuals).

In this paper, we go beyond the traditional value alignment problem and introduce the notion of value awareness. We define value awareness as follows.

value-aware AI

Noun [U]

/'væl.jur ə'weər ,eɪ'əɪ/

an AI system that identifies and understands a human’s value system, abides by that value system, and explains its own behaviour and that of others in terms of that value system

This definition lays the roadmap for value engineering research, an emergent field in AI dedicated to the engineering of value-aware systems [14, 15]. First, to **identify and understand** human value systems, the AI should be capable of *learning* relevant values, and *modelling* those values through formal semantics. Of course, value systems may exist on the individual level and the collective level. As such, the AI should also be capable of *aggregating* individual value systems into one that represents the collective. Alternatively, it should be capable of *using agreement mechanisms*, such as argumentation and negotiation, to help the collective agree on their value system.

Second, to **abide** by a value system, *value-alignment* mechanisms are needed for both agents and multiagent systems, such as developing value-aligned decision making and value-aligned norm synthesis. The objective is to ensure behaviour of agents and multiagent systems is aligned with relevant values.

Third, to **explain** one's own behaviour or that of others in terms of value systems, then there is a need for developing *value-based explainability* mechanisms.

We note that value-awareness research is not limited to raising awareness on the agent and multi-agent level, but also raising awareness for humans so they better understand their behaviour. As we see in Sect. 2.3, some of the work in real-life application domains focuses on ensuring humans, like medical professionals or firefighters, better understand which values their (potential) actions are promoting. In other words, the AI also supports humans to make value-informed decisions.

The remainder of this paper provides an overview of a selection of our ongoing work on various topics of this concise and simplified roadmap, along with applications to real-life domains.

2 Selected Contributions

2.1 On Identifying and Understanding Human Values

Value Representation. The first challenge is that of value representation. While work on values in AI is gaining ground, there is no formal model yet for the representation of human values. Furthermore, values in existing AI literature have usually been specified through labels (such as ‘fairness’), without any formal specification of the semantics of those values.

In [12], we propose a value-based taxonomy for the modelling of human values. This allows values to be organised hierarchically, where abstract concepts branch into concrete concepts. Property-based leaf nodes allow us to formally specify value semantics, which enables computational reasoning about values. This is because these property nodes essentially define how a value may be interpreted and assessed. The taxonomy also allows us to explicitly specify value relations and value importance, all of which are crucial elements for deliberating and reasoning over values. Furthermore, we illustrate how the proposed value-based taxonomy is aligned with the values literature in social psychology [13].

Value Learning. One of the main challenges of value awareness engineering is identifying and understanding relevant human values. It is not straightforward for stakeholders to identify and articulate their relevant values. Values are abstract constructs whose exact meaning (semantics) may change over time and context, and even from one person to another. Existing literature illustrates how numerous definitions exist today for values such as fairness or equality. Income inequality alone has been formalised through numerous equations, such as the Gini Index [3], Palma Ratio [4], Theil Index [5], and many many more [8].

We argue that identifying the semantics of values is context dependent, and should involve learning what those values mean for the relevant stakeholders. The learning process should take into account human feedback, which could either be obtained through dedicated user studies or by simply observing the human's behaviour. Our ongoing work with Hospital del Mar, Barcelona aims at identifying the formulae that best describe the semantics of the four basic bio-ethical values (or principles): beneficence, non-maleficence, autonomy and justice [2]. In this work [17], we are compiling a corpus of patient cases where each case is defined by a set of criteria that specify the patient's state, an action that is performed on that patient, and the change in the patient's state after the action is taken. An ongoing user study aims at having doctors annotate these patient cases with information on which of the four values is being promoted, demoted, or not affected. The results of this user study will then be used by an evolutionary strategy algorithm that aims at navigating the space of potential formulae to learn the formulae that best fits the data, and hence, best describes the doctors' view of how to understand the alignment of each of those values.

2.2 On Abiding by Human Values

Our initial work on the alignment of multiagent systems with human values has mostly focused on the alignment of norms, as norms are what govern the behaviour of these systems. In [21], a basic formal approach for value alignment is presented, where the alignment of a given norm with a given value is assessed by the level of promotion of that value in future states of the world. Norms are understood as changing the future states of the world, and values (such as gender equality) are defined through equations that specify how states of the world may be assessed with respect to that value (such as checking the gender pay gap).

[11] builds on this initial work to present tools for norm synthesis that would optimise for certain values, tools that use the Shapley value concept from game theory to help assess the contribution of the different norms towards promoting those values, as well as tools for checking the compatibility of values under certain norms. The proposed work is tested in a taxing game, where the norms that best promote values like fairness and equality are synthesised and assessed.

[10] proposes an approach that empowers agents by using theory of mind to reason about each others' values. The proposed mechanism allows agents to analyse norms not only from the perspective of their own value system, but from the perspective of other agents' value systems. This could eventually help agents when negotiating over the norms that best suit their collective.

Concerning the behaviour of individual agents and their decision making process, we propose in [16] an approach for enhancing automated negotiation mechanisms with social values to help agents reach agreements by considering not only their individual utilities, but also social values such as fairness and equality.

2.3 Real-World Applications

In [1], agent-based simulation is being used to analyse norms (policies) from the perspective of the values of fighting inequality and discrimination. Policies are categorised as aporophobic¹ and non-aporophobic by experts in the field, and agent-based simulation is used to better assess the impact of aporophobia on inequality. Results show that aporophobic policies do in fact lead to larger inequality, compared to non-aporophobic ones.

In the medical field, we are working closely with Hospital del Mar, Barcelona to develop a system that could provide feedback to medical professionals on the alignment of their potential actions with the four basic bioethical values (or principles) [18]. Our proposal is based on analysing the potential outcomes of an action. Alignment is then computed based on the value semantics learnt (see Sect. 2.1). For example, if the value “comfort” was of utmost importance, then actions that lead to states of the world where the patient suffers will not be considered aligned with that value. A multi-objective Markov decision process (MOMDP) is then used to help assess the alignment of entire medical protocols.

A similar approach to our work in the medical field is also being used to help train firefighters. While fire departments have clear and well defined values, it is common to see these values differ from one geographical location to another. Furthermore, new firefighter students usually join with their own value systems, and training over values in agent-based simulations could help them become more aligned with their fire department’s value system.

3 Conclusion

This paper has introduced the notion of value awareness in AI, and presented a concise and simplified roadmap for the development and engineering of value aware AI. It also listed some selected ongoing works covering various research challenges, along with a number of real-life applications.

While the topic of values in AI is becoming more prominent [14, 15], the open challenges are numerous, and the proposed roadmap provides only a glimpse into what future research can delve into. Some of the presented challenges are just starting to get traction, such as the work on aggregating individual value systems into a value system for the collective [9]. It is also evident that research on value-based explainability remains underdeveloped. To address this gap, we intend to

¹ The term aporophobia was coined by the philosopher Adela Cortina to describe having feelings of fear and an attitude of rejection of the poor [6].

build on our symbolic approach for value representation and value-based reasoning, which could provide the foundations for value-based explainability.

Acknowledgments. This work has been supported by the EU-funded VALAWAI (# 101070930) project and the Spanish-funded VAE (# TED2021-131295B-C31) and Rhymas (# PID2020-113594RB-100) projects.

References

1. Aguilera, A., Montes, N., Curto, G., Sierra, C., Osman, N.: Can poverty be reduced by acting on discrimination? An agent-based model for policy making. In: Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems, pp. 22–30. AAMAS ’24, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC (2024)
2. Beauchamp, T.L., Childress, J.F.: Principles of Biomedical Ethics. Oxford University Press. 8th Edition., New York (2019)
3. Ceriani, L., Verme, P.: The origins of the gini index: extracts from variabilità e mutabilità (1912) by corrado gini. *J. Econ. Inequality* **10**, 421–443 (2012)
4. Cobham, A., Sumner, A.: Is it all about the tails? The palma measure of income inequality. Working Paper 343, Center for Global Development (2013). <https://www.cgdev.org/sites/default/files/it-all-about-tails-palma-measure-income-inequality.pdf>
5. Conceição, P., Ferreira, P.: The young person’s guide to the theil index: Suggesting intuitive interpretations and exploring analytical applications. Working Paper 14, University of Texas Inequality Project (UTIP) (2000). https://utip.gov.utexas.edu/papers/utip_14.pdf
6. Cortina, A.: Aporophobia: why we reject the poor instead of helping them. Princeton University Press (2022)
7. Cranefield, S., Winikoff, M., Dignum, V., Dignum, F.: No pizza for you: Value-based plan selection in BDI agents. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17, pp. 178–184. ijcai.org (2017). <https://doi.org/10.24963/ijcai.2017/26>
8. De Maio, F.G.: Income inequality measures. *J. Epidemiol. Community Health* **61**(10), 849–852 (2007)
9. Lera-Leri, R., Bistaffa, F., Serramia, M., López-Sánchez, M., Rodríguez-Aguilar, J.A.: Towards pluralistic value alignment: Aggregating value systems through l_p -regression. In: Faliszewski, P., Mascardi, V., Pelachaud, C., Taylor, M.E. (eds.) 21st International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2022, Auckland, New Zealand, May 9–13, 2022, pp. 780–788. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS) (2022). <https://www.ifaamas.org/Proceedings/aamas2022/pdfs/p780.pdf>
10. Montes, N., Osman, N., Sierra, C.: Perspective-dependent value alignment of norms. In: Osman, N., Steels, L. (eds.) Value Engineering in Artificial Intelligence, pp. 46–63. Springer Nature Switzerland, Cham (2024)
11. Montes, N., Sierra, C.: Value-guided synthesis of parametric normative systems. In: Dignum, F., Lomuscio, A., Endriss, U., Nowé, A. (eds.) AAMAS ’21: 20th International Conference on Autonomous Agents and Multiagent Systems, Virtual Event, United Kingdom, May 3–7, 2021, pp. 907–915. ACM (2021)

12. Osman, N., d'Inverno, M.: A computational framework of human values. In: Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems, pp. 1531–1539. AAMAS '24, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC (2024)
13. Osman, N., d'Inverno, M.: Modelling human values for AI reasoning. CoRR abs/2402.06359 (2024). <https://doi.org/10.48550/arXiv.2402.06359>
14. Osman, N., Steels, L. (eds.): Value Engineering in Artificial Intelligence - First International Workshop, VALE 2023, Krakow, Poland, September 30, 2023, Proceedings, Lecture Notes in Computer Science, vol. 14520. Springer (2024). <https://doi.org/10.1007/978-3-031-58202-8>
15. Osman, N., Steels, L. (eds.): Value Engineering in Artificial Intelligence - Second International Workshop Track, VALE 2024, Santiago de Compostela, Spain, October 19, 2024, Proceedings, Lecture Notes in Computer Science, vol. 15356. Springer (In Press)
16. Rodriguez Cimaa, L., de Jonge, D., Osman, N.: Towards the incorporation of social values in automated negotiation strategies. In: Osman, N., Steels, L. (eds.) Preproceedings of the Value Engineering in AI Workshop track (VALE 2024) at 27th European Conference on Artificial Intelligence (ECAI 2024). Springer Nature Switzerland, Cham (In Press)
17. Rodriguez-Soto, M., et al.: User study design for identifying the semantics of bioethical principles. In: Osman, N., Steels, L. (eds.) Preproceedings of the Value Engineering in AI Workshop track (VALE 2024) at 27th European Conference on Artificial Intelligence (ECAI 2024). Springer Nature Switzerland, Cham (In Press)
18. Rodriguez-Soto, M., et al.: Towards value awareness in the medical field. In: Rocha, A.P., Steels, L., van den Herik, H.J. (eds.) Proceedings of the 16th International Conference on Agents and Artificial Intelligence, ICAART 2024, Volume 3, Rome, Italy, February 24-26, 2024, pp. 1391–1398. SCITEPRESS (2024). <https://doi.org/10.5220/0012588600003636>
19. Russell, S.: Human Compatible: Artificial Intelligence and the Problem of Control. Penguin Publishing Group (2019)
20. Serramia, M., López-Sánchez, M., Rodríguez-Aguilar, J.A.: A qualitative approach to composing value-aligned norm systems. In: Seghrouchni, A.E.F., Sukthankar, G., An, B., Yorke-Smith, N. (eds.) Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems, AAMAS '20, Auckland, New Zealand, May 9-13, 2020, pp. 1233–1241. International Foundation for Autonomous Agents and Multiagent Systems (2020)
21. Sierra, C., Osman, N., Noriega, P., Sabater-Mir, J., Perelló, A.: Value alignment: a formal approach. CoRR abs/2110.09240 (2021). <https://arxiv.org/abs/2110.09240>
22. di Tosto, G., Dignum, F.: Simulating social behaviour implementing agents endowed with values and drives. In: Giardini, F., Amblard, F. (eds.) Multi-Agent-Based Simulation XIII - International Workshop, MABS 2012, Valencia, Spain, June 4-8, 2012, Revised Selected Papers. Lecture Notes in Computer Science, vol. 7838, pp. 1–12. Springer (2012). https://doi.org/10.1007/978-3-642-38859-0_1

Social Structures



Emergent Dominance Hierarchies in Reinforcement Learning Agents

Ram Rachum¹(✉) , Yonatan Nakar² , Bill Tomlinson³ , Nitay Alon^{4,5} , and Reuth Mirsky¹

¹ Bar-Ilan University, Ramat Gan, Israel
rachumr@biu.ac.il

² Tel Aviv University, Tel Aviv, Israel

³ University of California, Irvine, USA

⁴ Max Planck Institute for Biological Cybernetics, Tübingen, Germany

⁵ The Hebrew University of Jerusalem, Jerusalem, Israel

Abstract. Modern Reinforcement Learning (RL) algorithms are able to outperform humans in a wide variety of tasks. Multi-agent reinforcement learning (MARL) settings present additional challenges, and successful cooperation in mixed-motive groups of agents depends on a delicate balancing act between individual and group objectives. Social conventions and norms, often inspired by human institutions, are used as tools for striking this balance.

We examine a fundamental, well-studied social convention that underlies cooperation in animal and human societies: dominance hierarchies.

We adapt the ethological theory of dominance hierarchies to artificial agents, borrowing the established terminology and definitions with as few amendments as possible. We demonstrate that populations of RL agents, operating without explicit programming or intrinsic rewards, can invent, learn, enforce, and transmit a dominance hierarchy to new populations. The dominance hierarchies that emerge have a similar structure to those studied in chickens, mice, fish, and other species.

Keywords: Multi-Agent Reinforcement Learning · Reinforcement Learning · Cultural Evolution · Multi-Agent Systems · Cooperative AI

1 Introduction

Many animal species are able to collaborate, form groups and harness collective intelligence [11, 23]. Among those species, humans have achieved a scale and sophistication of collaboration that stands as one of the most profound and unparalleled phenomena on Earth. The grand feats of humanity were accomplished by groups of humans working together; what is grander is that these groups are composed of individuals with wildly different beliefs, motivations, and talents [59]. While differences between individuals provide an important source of diversity, they also lead to adversity and conflict, which can result

in the violent demise of the social group [63]. Those human civilizations that tempered their internal conflicts while capitalizing on the diversity were the ones to survive and flourish [4, 5]. The intricate social structures that make up these civilizations turn a potential runaway explosion into a controlled reaction, enabling groups of humans to achieve breakthroughs that exceed the combined capabilities of their individual members [39].

Multi-agent reinforcement learning (MARL) presents an opportunity to implement simplified versions of those intricate social structures, where the environment is closed and controlled, and humans are replaced by artificial agents. Research in the paradigm of Cooperative AI [12, 41, 56] highlights the role of such social structures (also labeled *institutions*) in producing human-aligned AI systems. Existing research into reproducing such institutions for use in MARL settings includes language [38], voting systems [62], reputation systems [42], and bargaining [30].

In this paper we turn our attention to a primordial institution that underlies cooperation in both animal and human societies: *dominance hierarchies*.

Our Contribution: We adapt the ethological theory of dominance hierarchies to artificial agents by modeling conflicts between each pair of individuals as the classic game of Chicken [44]. We generalize the game of Chicken into an N -player stochastic game that we call *Chicken Coop*. We borrow the established terminology and definitions from animal study to allow calculating dominance metrics from agents' action history. We release the Chicken Coop environment under the MIT open-source license.¹ We train RL agents on the Chicken Coop environment to optimize their score, observing the emergence of three behaviors:

1. Agents collaboratively invent dominance hierarchies.
2. Agents enforce dominance hierarchies on other agents.
3. Agents transmit dominance hierarchies to new populations.

We show a causal link between the agents' ability to identify the opposing agent and the emergence of dominance hierarchies. We compare the attributes of dominance hierarchies that emerge to empirical results from groups of different animals, showing structural similarities.

By providing MARL systems with the ability to coordinate their collective behavior via dominance hierarchies, we hope to enable a more seamless integration into existing human systems [45].

2 Background

When animals live together in a social group they often find themselves in conflicts with other members of the group over resources such as food or mating partners. These conflicts can lead to physical injury and death. A *dominance relationship* [43] is a pattern in the conflicts between a pair of individuals, in

¹ The code and its manual are available at <https://github.com/cool-RR/chicken-coop>.

which the *dominant* individual is more likely to escalate violence and win the majority of resources, and the *subordinate* individual is more likely to deescalate and yield any contested resources. This dynamic prevents the runaway escalation of violence that could otherwise lead to injury and a loss of group cohesion [50].

A *dominance hierarchy*, also known as a *pecking order*, is the aggregate of all the dominance relationships between each pair of agents in the group. It is defined by modelling dominance relationships as a total relation on the set of agents, where $d \rightarrow s$ means that individual d is dominant over individual s . A dominance hierarchy is modelled as a complete, directed graph (tournament) where agents are represented as nodes and dominance relationships are represented as directed edges [29]. This graph representation is more than just a map of agonistic behavior in the group; any skirmish between individuals a and b affects not only the edge connecting a and b , but the edges connecting either a or b with each of the other individuals in the group, by the mechanisms of *winner effects* [7], *loser effects* [18] and *bystander effects* [19].

The field of dominance hierarchies traces its roots to Schjelderup-Ebbe [48], which described pecking orders in captive chicken societies. Over the past century, dominance hierarchies have been studied across a wide range of animal species including canines [3, 20], birds [57], fish [13] and humans [10, 24, 37], uncovering commonalities in the structure and function of dominance hierarchies across taxa [8]. The study of dominance hierarchies continues to be a field of active research [51].

3 Related Work

MAS models have been used for the modelling of dominance hierarchies in animals [22] and in humans [40]. *DomWorld* [27] is a multi-agent environment developed to study the self-organization of social behavior and dominance hierarchies in primate groups. DomWorld is not modelled as a stochastic game or any other game theoretic construct, as the agents have no concept of a reward signal. Leimar [32] provides a proper MARL environment called *SocDom*, equipping the agents with an RL algorithm [52], albeit without neural network estimators. While the SocDom experiments show similar results to ours, the foci of the two environments differ: SocDom, like DomWorld, attempts to tailor the agents' behaviors to faithfully reproduce the dominance behavior seen in real animal societies. In contrast, our research is directed at boiling the mechanics of dominance hierarchies down to the simplest possible set of premises, showing that they emerge even when using an off-the-shelf RL algorithm.

The use of dominance hierarchies for improving the performance of artificial agents was proposed in Hemelrijk [26], and in Tomlinson et al. [53], which suggested that dominance hierarchies could streamline negotiation, enable agents to form beneficial alliances, and make interfaces more intuitive. Bakker et al. [2] employs a group of RL agents with a graph structure, which bears resemblance to a dominance hierarchy, besides being explicitly defined rather than emergent. Aroca-Ouellette et al. [1] demonstrates an explicit hierarchy where some agents have a different set of actions than others.

Vezhnevets et al. [55] and Wu et al. [60] replace the RL algorithms used by the agents with large language models (LLMs), which have gained immense popularity due to their robustness in solving a wide variety of problems [61]. These LLM-based agents are organized in human-inspired hierarchies such as manager-workers and student-assistant-expert. The back-and-forth interaction between multiple LLM agents is effective at encouraging diversity [35], factuality, and reasoning capability [16], when compared to the use of a single LLM-based agent. The potential usefulness of such agents is tremendous, as LLMs can be seamlessly integrated with multimodal foundation models [33], allowing them to process and generate textual, visual and auditory content. We suggest that a formal understanding and terminology of agent hierarchies could play a decisive role in maximizing the effectiveness of these agents.

4 Definitions

4.1 Dominance Between Two Agents

Dominance hierarchies are a group phenomenon comprised of multiple dominance relationships, one between each pair of agents in the group. In this section we consider only those pairwise interactions; we define dominance relationships, the metrics of aggressiveness and rapport, the roles of dominant and subordinate, and the environment in which these occur.

Animals use dominance relationships to decide which individual will have access to an exclusive resource, such as food, a mating partner, or grooming by other individuals [28]. We model the environment that enables these interactions between agent i and agent j as a partially-observable stochastic game (POSG) [25] which has exactly two stable Nash equilibria NE_i and NE_j such that agent i 's reward at NE_i is bigger than its reward at NE_j , and agent j 's reward at NE_j is bigger than its reward at NE_i .

We denote the actions that comprise NE_i as $a_{NE_i}^i$ and $a_{NE_i}^j$.

Intuitively, a dominance relationship between two agents is the tendency of those agents to play the joint action of either NE_i or NE_j , but not both. Similarly to Leibo et al. [31], we define an agent's aggressiveness by how frequently it chooses the action that reduces the other agent's reward:

Definition 1 (Aggressiveness). *Given an agent i and a set of timesteps \mathbb{T} , the agent's aggressiveness $g_i^{\mathbb{T}}$ is the portion of timesteps in which it played $a_{NE_i}^i$.*

Definition 2 (Dominance relationship, dominant, subordinate). *Given two agents i and j and a set of timesteps \mathbb{T} in which they played with each other, if the difference between agent i 's aggressiveness and agent j 's aggressiveness is above a certain threshold, we say that the two agents are in a dominance relationship (DR), with agent i being dominant and agent j being subordinate:*

$$i \rightarrow j \quad \text{iff} \quad g_i^{\mathbb{T}} - g_j^{\mathbb{T}} > \eta, \quad \eta \in (0, 1]$$

In order to determine the existence of a dominance relationship between two agents without assuming its polarity, we introduce a metric we call *rapport*:

Definition 3 (Rapport). *For two agents i and j and timesteps \mathbb{T} , we define the rapport $\mathfrak{R}_{i,j}^{\mathbb{T}}$ as:*

$$\mathfrak{R}_{i,j}^{\mathbb{T}} = |g_i^{\mathbb{T}} - g_j^{\mathbb{T}}|$$

A rapport $\mathfrak{R}_{i,j}^{\mathbb{T}} > \eta$ indicates either $i \rightarrow j$ or $i \leftarrow j$.

In the classic game of Chicken [44], each of the two agents plays either *hawk* or *dove*. An agent gets the most reward if it plays *hawk* while its partner plays *dove*; however, if both agents play *hawk*, they both get the lowest reward.

Definition 4 (The game of Chicken). *The game of Chicken is defined as a normal-form game:*

		Agent 2	
		Dove	Hawk
Agent 1	Dove	R, R	S, T
	Hawk	T, S	P, P

The reward constants satisfy $T > R > S > P$. The (S, T) and (T, S) outcomes serve as the Nash equilibria NE_i and NE_j . Each agent's aggressiveness is simply its tendency to play *hawk*.

4.2 Dominance Hierarchies

To model dominance relationships within an agent population, we generalize the game of Chicken to support more than two players:

Definition 5 (Chicken Coop). *Chicken Coop is an N -player generalization of the game of Chicken. In each episode of Chicken Coop, the agents are divided into random pairs that are used for the entire episode. Each pair of agents plays one round of Chicken against each other, choosing either hawk or dove and receiving a reward in $\{T, R, S, P\}$. Each agent's sole observation is the identity of their opponent.*

We extend the definitions in Sect. 4.1 from the Chicken game to the Chicken Coop game. For example, for each pair of agents i and j in the population, we determine the existence and polarity of their dominance relationship ($i \rightarrow j$ or $i \leftarrow j$) by evaluating the aggressiveness metrics $g_i^{\mathbb{T}}$ and $g_j^{\mathbb{T}}$ on the subset of \mathbb{T} in which these two agents were randomly paired with each other. Finally, we aggregate these dominance relationships into a graph:

Definition 6 (Dominance hierarchy). *A dominance hierarchy \mathcal{H} is a complete, directed graph (tournament) where agents are represented as nodes and dominance relationships are represented as directed edges.*

5 Methods

In all of the experiments below, unless stated otherwise, the RL agents are trained using the PPO algorithm [49] implementation provided by the RLlib framework [34]. We train $L = 300$ populations of $N = 6$ Chicken Coop agents, at a learning rate of 2×10^{-6} , a discount factor of $\gamma = 0.99$, and a clipping parameter of $\epsilon = 0.3$. We use a dominance threshold of $\eta = 0.55$ and these reward constants: $R = 0$, $S = -2$, $T = 5$, $P = -10$.

Each agent has its own neural network with its own set of weights, which are separate from those of the other agents. Crucially, all of the agents in a given population are trained exclusively with the other agents in that population; we don't allow cross-population training, because we would like to allow each population to develop its own distinct dominance hierarchy.

Each generation is comprised of 512 episodes. After each generation, the policies update according to the actions and rewards in that generation.

6 Results

6.1 Emergence of Dominance Hierarchies

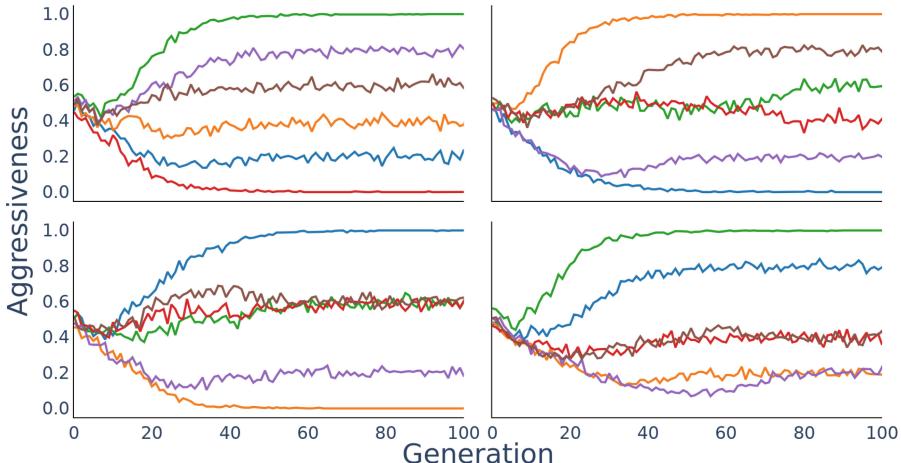


Fig. 1. Aggressiveness levels of each of the six agents (depicted as differently-colored lines) in four sample Chicken Coop populations.

We measure the aggressiveness of each agent in each population, aggregated interactions with each of the other agents. In Fig. 1 we plot the aggressiveness of six agents in four sample populations. A few features can be visually discerned:

1. The agents' aggressiveness values converge to approximately equally-spaced “ranks” in $[0, 1]$.

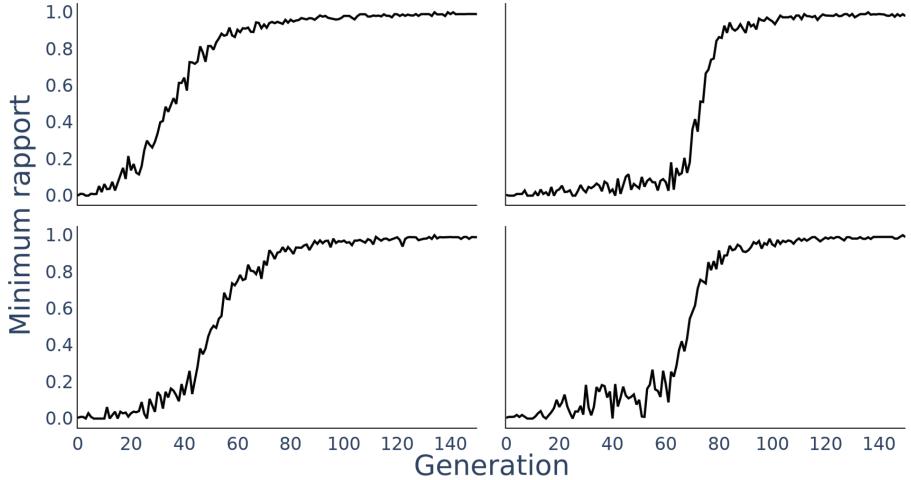


Fig. 2. Minimum rapport between each pair of the six agents in four sample Chicken Coop populations.

2. Despite of this strong convergence, the mapping between agents and ranks appears completely arbitrary across the four sample populations.
3. In some populations, the mapping between agents and ranks is a bijection, while in other populations, multiple agents converge to the same rank, leaving other ranks empty.

We offer the following explanation for these observations: when agent i 's aggressiveness approaches k/N , agent i is playing a high proportion of *hawk* towards k other agents, which in turn play a high proportion of *dove* towards agent i . Each of these k agents contributes $1/N$ to agent i 's aggressiveness. According to Definition 2, agent i is dominating these k agents. The remaining $N - k - 1$ agents dominate agent i . We verify this explanation by measuring the minimum rapport between each pair of agents for each of the four sample populations, observing that each agent develops a dominance relationship with each of the other agents in its population (Fig. 2).

We aggregate these dominance relationships into dominance hierarchies, showing the results for our four sample populations in Fig. 3. We observe that some of the dominance hierarchies form a perfect line, while others contain cycles. This distinction has been studied in animal societies. The former are called *linear dominance hierarchies* or *transitive dominance hierarchies*. The latter are called *nonlinear dominance hierarchies*, *near-linear dominance hierarchies* or *intransitive dominance hierarchies* [9, 14, 15, 47]. In linear dominance hierarchies, the agents may be assigned ranks $\{0, 1, 2, \dots, N - 1\}$ with rank 0 representing the agent that dominates all other agents, rank 1 representing the agent that dominates all other agents except that in rank 0, etc.

One profound property of the agents' learned behavior is its idiosyncraticity: (1) it converges to a stable dominance hierarchy, and (2) this hierarchy is not a

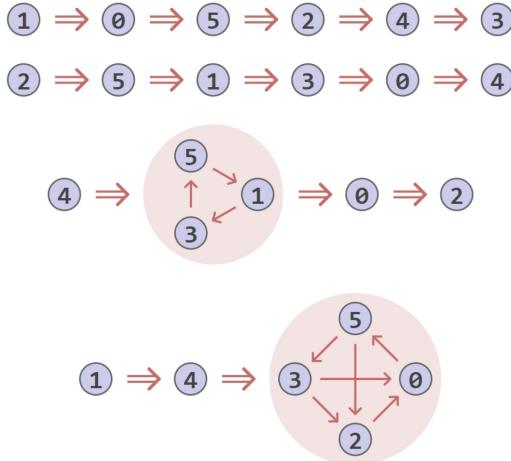


Fig. 3. Dominance hierarchies of four sample Chicken Coop populations, visualized in condensed form. “ \rightarrow ” represents a dominance relationship. “ \Rightarrow ” means that all agents to the left of the double arrow dominate all of the agents to its right.

single one, but rather varies between different trials. Our 300 populations converged to a total of 248 different dominance hierarchies (out of $2^{\frac{N(N-1)}{2}} = 2^{15}$ possible permutations). Most dominance hierarchies appeared in only one population, while the most common dominance hierarchies appeared in 3 populations. Despite the arbitrariness of that choice, each population converges strongly to its chosen dominance hierarchy.

6.2 Comparison to Animal Behavior

In Fig. 4 we visually compare our aggressiveness results to empirical data from groups of actual chickens. In Chase et al. [8], 14 captive populations of 4 white Leghorn hens each were observed for aggressive behavior for two consecutive days. While there are considerable differences in the methodologies of that experiment and our study, the property of idiosyncraticity holds true in both of them. In both environments, the agents tend to maintain their position, regardless of how high or low it is in the hierarchy.

We visually compare the occurrence of intransitive components between Chicken Coop populations and mice populations, shown in Fig. 5. In Williamson et al. [58], 20 captive populations of 12 male CD-1 mice each were observed for aggressive behavior for 22 days. We compare these results to those from $L = 30$ populations of $N = 12$ Chicken Coop agents. In order to find the maximum similarity between the natural and artificial populations, we use a learning rate of 3×10^{-5} . We make this comparison by using a metric we define here as *rank linearity*, which expresses how likely each dominance rank is to be occupied by a single agent, rather than a K -way tie. For a single population, we can describe

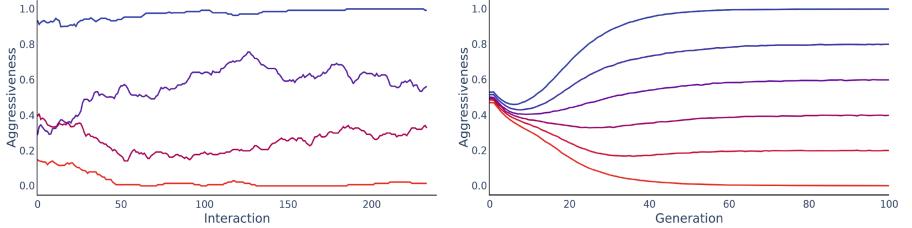


Fig. 4. Left: Mean aggressiveness levels of four Leghorn chickens in fourteen captive populations [8]. **Right:** Mean aggressiveness levels of six Chicken Coop agents across 300 populations, filtered to include only those that developed a linear dominance hierarchy. In both plots, the results are averaged using the agents' ranks as their identity.

rank linearity as a boolean; for example, in the third population from the top in Fig. 3, the top rank (rank 0) and the two bottom ranks (ranks 4 and 5) are occupied by agents 4, 0 and 2 respectively. Therefore, we consider ranks 0, 4 and 5 as linear. The remaining ranks are 1, 2 and 3, and they cannot be directly correlated with any agent, because agents 1, 3 and 5 are in a three-way tie. Therefore, we consider ranks 1, 2 and 3 as non-linear. For sets of populations, we define rank linearity as a number in $[0, 1]$ which is the probability of a rank being linear in a sample population.



Fig. 5. The rank linearities of 12 different ranks, in both Chicken Coop agents (purple) and male CD-1 mice (orange)[58]. In both experiments, the top and bottom ranks were the least likely to be occupied by an intransitive component. (Color figure online)

6.3 Observation Ablation

Here we show a causal relationship between agents' ability to identify their opponents and their tendency to form dominance hierarchies. In the Chicken Coop environment, agents observe the identity of their opponent as an index number $0 \leq i \leq N - 1$. We add random noise to the agents' observation, up to and including its complete obfuscation, and measure the rapport in the population.

Methodology: We define an agent’s *opponent perception accuracy* (OPA) as a number in $[0, 1]$. For each agent in each episode, the probability that its observation of its opponent is replaced with a random value is $(1 - OPA)$. For each OPA value in $\{0, 0.1, \dots, 1\}$ we train $L = 10$ different populations of $N = 6$ Chicken Coop agents at a learning rate of 2×10^{-6} and measure their rapport.

In Fig. 6 we show a positive correlation between the agents’ OPA and their rapport, and by extension, their tendency to form dominance hierarchies.

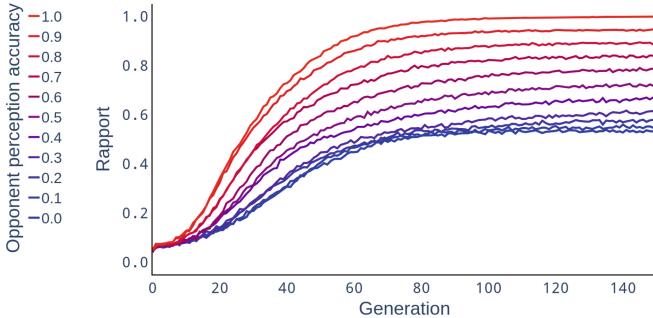


Fig. 6. The gradual rise of rapport in Chicken Coop populations, differentiated by how accurately the agents identify their opponents. The bluer lines represent bigger amounts of random noise added to the agents’ observation signal.

6.4 Transmitting Dominance Hierarchies to New Populations

In previous sections we regarded Chicken Coop agents as the focal objects of study, while we considered the dominance hierarchies that they form as a pattern in their behavior. We now propose flipping this perspective on its head: We consider dominance hierarchies as the focal object while the agents are used as vessels that communicate the dominance hierarchy that they learned to a new population, promoting a cultural evolution paradigm in artificial intelligence [6].

This transmission of dominance hierarchies to new populations is made possible by two properties of dominance hierarchies: (1) the knowledge of each dominance hierarchy is replicated over all the agents that participate in it, and (2) each agent learns to consistently play *hawk* with all its subordinates, and this tendency serves as a punishing mechanism. Any errant attempt by a subordinate to play *hawk* against a dominant would usually result in a *hawk/hawk* outcome, with both agents receiving the lowest possible reward of P .

Methodology: We train $L = 10$ different populations of $N = 6$ Chicken Coop agents until they converge to dominance hierarchies. We call these *experienced populations*. For each of these L populations, we perform the following procedure $M = 30$ times: for each $K \in \{0, 1, 2, \dots, N - 2\}$ we choose a random sample of K

of those agents to be our *experienced agents*. We transplant the K experienced agents into a new population with $N - K$ *naive agents* [17] that have not been trained. Finally, we train the naive agents in the heterogeneous population at a learning rate of 2×10^{-5} and observe their dominance behavior.

To measure how close each new dominance hierarchy is to its original dominance hierarchy, we define a *restricted dominance hierarchy distance* (RDHD). The restricted distance between two dominance hierarchies is the portion of dominance relationships, limited to those between two naive agents, that have the same polarity in both dominance hierarchies, satisfying $0 \leq RDHD \leq 1$. We define the *dominance hierarchy transmission fidelity* (DHTF) as $(1 - RDHD)$. Intuitively, DHTF is a measure of how effectively the experienced agents passed their dominance hierarchy on to the naive population.

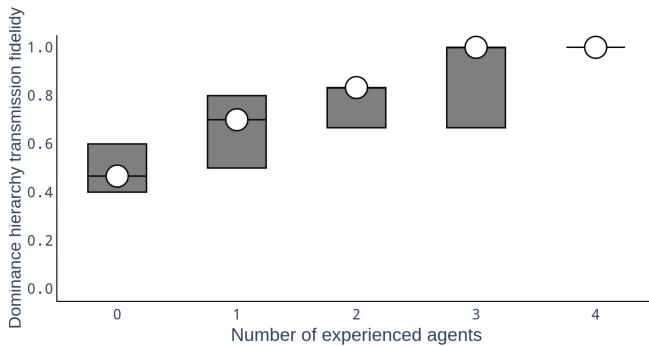


Fig. 7. The DHTF of 300 different groups of experienced agents transplanted into 300 different naive populations. Higher values mean that the naive agents learned a dominance hierarchy similar to the one in the original experienced population. Circles indicate median DHTF while boxes indicate the two middle quartiles.

In Fig. 7 we show the distribution of the DHTFs for different values of K . A mere $K = 2$ experienced agents are able to teach 4 naive agents their original dominance hierarchy with 80% median fidelity.

7 Discussion and Future Work

This paper investigates dominance hierarchies in the context of MARL. It first provides a formal definition and a game-theoretical representation of dominance hierarchies. It then introduces the phenomenon of dominance hierarchies in a simplified form using the Chicken Coop environment, stripping away many of the complexities found in biological life. Finally, it demonstrates how populations of RL agents can invent, learn, enforce, and transmit dominance hierarchies.

Preliminary experiments with higher learning rates result in unstable dominance hierarchies, where agents seem to converge strongly to a certain rank for

some number of generations, only to deviate from it and converge to a different rank. This is another unique behavior that exists in natural environments, where it is known as *rank change* [46, 51] or *dynamic stability* [8].

We suggest several extensions to the Chicken Coop environment:

Multiple Hierarchies: In the Chicken Coop environment, all the agents participate in one dominance hierarchy that includes all of the agents. In animal and human societies, individuals participate in multiple dominance hierarchies for multiple groups of other individuals, holding different ranks in each one.

Societies with Mixed Incentives: In this environment, the agents are fully dedicated to making decisions on how aggressive they'll be to other agents. In animal and human societies, individuals in a group cooperate towards shared goals in addition to their internal dominance struggles.

Opponent Shaping (OS): Algorithms such as LOLA [21] and M-FOS [36] enable agents to take the learning process of the other agents into account in their own policy gradients. When applied to an environment that supports dominance hierarchies, they may promote second-order dominance-seeking strategies, e.g., they may prompt agent i to consider how it can behave as to encourage agent j to place agent i at a high rank in the dominance hierarchy.

We propose that the study of dominance hierarchies between RL agents may be combined with research on large language models (LLMs) and multi-modal foundation models (MFMs), which have gained immense popularity due to their robustness in solving a wide variety of problems [33, 61]. Tufano et al. [54] presents an innovative way to use LLMs: instead of sending queries to a single LLM, users converse with a group of LLM-based agents, organized in human-inspired hierarchies such as manager-workers and assistant-expert. The interaction between multiple LLM agents is effective at encouraging diversity [35], factuality, and reasoning capability [16], when compared to a single LLM-based agent. We suggest that the augmenting these LLM-based agents with RL or OS algorithms, specifically to maintain the social relationships between the agents, may result in hierarchies that are similar to those observed in biological life.

When humans work on problems as a group, we balance in-group intrigues with external pressures to provide good results. We suggest that this interplay between individual needs and group needs may play a crucial role in the success of our collective intelligence. We hypothesize that MFM-based agents operating under similar conditions may make real-world decisions in a way that is more interpretable and corrigible, as human operators may recognize that the agents' decision process reflects their own.

Acknowledgments. We thank the following colleagues for their advice and support in writing this paper: David Aha, Ivan Chase, Edgar Duéñez-Guzmán, Errol King, Joel Leibo, Olof Leimar, David Manheim, Markov, Georg Ostrovski, Jérémie Perret, Saul

Pwanson, Venkatesh Rao and Eli Strauss. This research was funded by The Association For Long Term Existence And Resilience (ALTER).

References

1. Aroca-Ouellette, S., Aroca-Ouellette, M., Biswas, U., Kann, K., Roncone, A.: Hierarchical reinforcement learning for ad hoc teaming. In: Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems, pp. 2337–2339 (2023)
2. Bakker, M.A., et al.: Modelling cooperation in network games with spatio-temporal complexity (2021)
3. Bonanni, R., Cafazzo, S., Abis, A., Barillari, E., Valsecchi, P., Natoli, E.: Age-graded dominance hierarchies and social tolerance in packs of free-ranging dogs. *Behav. Ecol.* **28**(4), 1004–1020 (2017)
4. Bowles, S.: Did warfare among ancestral hunter-gatherers affect the evolution of human social behaviors? *Science* **324**(5932), 1293–1298 (2009)
5. Boyd, R., Richerson, P.J.: Culture and the evolution of human cooperation. *Philos. Trans. R. Soc. B: Biol. Sci.* **364**(1533), 3281–3288 (2009)
6. CGI Team: Learning robust real-time cultural transmission without human data. arXiv preprint [arXiv:2203.00715](https://arxiv.org/abs/2203.00715) (2022)
7. Chase, I., Bartolomeo, C., Dugatkin, L.: Aggressive interactions and inter-contest interval: how long do winners keep winning? *Anim. Behav.* **48**(2), 393–400 (1994)
8. Chase, I., Coelho, D., Lee, W., Mueller, K., Curley, J.: Networks never rest: an investigation of network evolution in three species of animals. *Soc. Netw.* **68**, 356–373 (2022)
9. Chase, I., Tovey, C., Spangler-Martin, D., Manfredonia, M.: Individual differences versus social dynamics in the formation of animal dominance hierarchies. *Proc. Natl. Acad. Sci.* **99**(8), 5744–5749 (2002)
10. Chen Zeng, T., Cheng, J.T., Henrich, J.: Dominance in humans. *Philos. Trans. R. Soc. B* **377**(1845), 20200451 (2022)
11. Couzin, I.D.: Collective cognition in animal groups. *Trends Cogn. Sci.* **13**(1), 36–43 (2009)
12. Dafoe, A., et al.: Open problems in cooperative AI (2020)
13. David, B.O., Stoffels, R.J.: Spatial organisation and behavioural interaction of giant kokopu (*Galaxias argenteus*) in two stream pools differing in fish density. *New Zealand Journal of Marine and Freshwater Research* (2003)
14. De Vries, H.: An improved test of linearity in dominance hierarchies containing unknown or tied relationships. *Anim. Behav.* **50**(5), 1375–1389 (1995)
15. De Vries, H.: Finding a dominance order most consistent with a linear hierarchy: a new procedure and review. *Anim. Behav.* **55**(4), 827–843 (1998)
16. Du, Y., Li, S., Torralba, A., Tenenbaum, J.B., Mordatch, I.: Improving factuality and reasoning in language models through multiagent debate. arXiv preprint [arXiv:2305.14325](https://arxiv.org/abs/2305.14325) (2023)
17. Duéñez-Guzmán, E.A., et al.: Statistical discrimination in learning agents. arXiv preprint [arXiv:2110.11404](https://arxiv.org/abs/2110.11404) (2021)
18. Dugatkin, L.A.: Winner and loser effects and the structure of dominance hierarchies. *Behav. Ecol.* **8**(6), 583–587 (1997)
19. Dugatkin, L.A.: Bystander effects and the structure of dominance hierarchies. *Behav. Ecol.* **12**(3), 348–352 (2001)

20. Essler, J., Cafazzo, S., Marshall-Pescini, S., Virányi, Z., Kotrschal, K., Range, F.: Play behavior in wolves: Using the ‘50:50’ rule to test for egalitarian play styles. *PLoS ONE* **11**(5), e0154150 (2016)
21. Foerster, J., Chen, R.Y., Al-Shedivat, M., Whiteson, S., Abbeel, P., Mordatch, I.: Learning with opponent-learning awareness. In: Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, pp. 122–130, AAMAS ’18, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC (2018)
22. Gavrilets, S., Duéñez-Guzmán, E.A., Vose, M.D.: Dynamics of alliance formation and the egalitarian revolution. *PLoS ONE* **3**(10), e3293 (2008)
23. Gordon, D.M.: The organization of work in social insect colonies. *Nature* **380**(6570), 121–124 (1996)
24. Halevy, N., Chou, E.Y., Cohen, T.R., Livingston, R.W.: Status conferral in inter-group social dilemmas: behavioral antecedents and consequences of prestige and dominance. *J. Pers. Soc. Psychol.* **102**(2), 351 (2012)
25. Hansen, E.A., Bernstein, D.S., Zilberstein, S.: Dynamic programming for partially observable stochastic games. In: AAAI, vol. 4, pp. 709–715 (2004)
26. Hemelrijk, C.: Dominance interactions, spatial dynamics and emergent reciprocity in a virtual world. In: The 4th International Conference on Simulation of Adaptive Behavior, vol. 4, pp. 545–552, Citeseer (1996)
27. Hemelrijk, C.: Social phenomena emerging by self-organisation in a competitive, virtual world (“domworld”). In: Learning to behave Workshop II: Internalising knowledge. Ieper, Belgium, pp. 11–19 (2000)
28. Henazi, S.P., Barrett, L.: The value of grooming to female primates. *Primates* **40**, 47–59 (1999)
29. Iverson, G.J., Sade, D.S.: Statistical issues in the analysis of dominance hierarchies in animal societies. *J. Quant. Anthropol.* **2**(1), 61–83 (1990)
30. Kraus, S.: Negotiation and cooperation in multi-agent environments. *Artif. Intell.* **94**(1), 79–97 (1997). ISSN 0004-3702, [https://doi.org/10.1016/S0004-3702\(97\)00025-8](https://doi.org/10.1016/S0004-3702(97)00025-8), economic Principles of Multi-Agent Systems
31. Leibo, J.Z., Zambaldi, V., Lanctot, M., Marecki, J., Graepel, T.: Multi-agent reinforcement learning in sequential social dilemmas. In: Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, pp. 464–473, AAMAS ’17, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC (2017)
32. Leimar, O.: The evolution of social dominance through reinforcement learning. *Am. Nat.* **197**(5), 560–575 (2021)
33. Li, C., Gan, Z., Yang, Z., Yang, J., Li, L., Wang, L., Gao, J.: Multimodal foundation models: From specialists to general-purpose assistants (2023)
34. Liang, E., et al.: RLLib: Abstractions for distributed reinforcement learning (2018)
35. Liang, T., et al.: Encouraging divergent thinking in large language models through multi-agent debate. arXiv preprint [arXiv:2305.19118](https://arxiv.org/abs/2305.19118) (2023)
36. Lu, C., Willi, T., De Witt, C.A.S., Foerster, J.: Model-free opponent shaping. In: International Conference on Machine Learning, pp. 14398–14411, PMLR (2022)
37. Magee, J.C., Galinsky, A.D.: Social hierarchy: the self-reinforcing nature of power and status. *Acad. Manag. Ann.* **2**, 351–398 (2008)
38. Mordatch, I., Abbeel, P.: Emergence of grounded compositional language in multi-agent populations. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)
39. Muthukrishna, M., Henrich, J.: Innovation in the collective brain. *Ph. Transactions of the Royal Society B: Biological Sciences* **371**(1690) (2016)

40. Przepiorka, W., Rutten, C., Buskens, V., Szekely, A.: How dominance hierarchies emerge from conflict: a game theoretic model and experimental evidence. *Soc. Sci. Res.* **86**, 102393 (2020)
41. Radke, D., Larson, K., Brecht, T., Tilbury, K.: Towards a better understanding of learning with multiagent teams. In: Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, pp. 271–279, ijcai.org (2023). <https://doi.org/10.24963/ijcai.2023/31>
42. Ramchurn, S.D., Mezzetti, C., Giovannucci, A., Rodriguez-Aguilar, J.A., Dash, R.K., Jennings, N.R.: Trust-based mechanisms for robust and efficient task allocation in the presence of execution uncertainty. *J. Artif. Intell. Res.* **35**, 119–159 (2009)
43. Rapoport, A.: Outline of a probabilistic approach to animal sociology: I. *Bull. Math. Biophys.* **11**, 183–196 (1949)
44. Rapoport, A., Chammah, A.M.: The game of chicken. *Am. Behav. Sci.* **10**(3), 10–28 (1966)
45. Reeves, B., Nass, C.: The media equation: How people treat computers, television, and new media like real people. Cambridge, UK **10**(10) (1996)
46. Samuels, A., Silk, J.B., Altmann, J.: Continuity and change in dominance relations among female baboons. *Anim. Behav.* **35**(3), 785–793 (1987)
47. Sanders, N.J., Gordon, D.M.: Resource-dependent interactions and the organization of desert ant communities. *Ecology* **84**(4), 1024–1031 (2003)
48. Schjelderup-Ebbe, T.: Beiträge zur sozialpsychologie des haushuhns. *Zeitschrift für Psychologie und Physiologie der Sinnesorgane. Abt. 1. Zeitschrift für Psychologie* **88**, 225 (1922)
49. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms. arXiv preprint [arXiv:1707.06347](https://arxiv.org/abs/1707.06347) (2017)
50. Smith, J.M., Price, G.R.: The logic of animal conflict. *Nature* **246**(5427), 15–18 (1973)
51. Strauss, E.D., Shizuka, D.: The dynamics of dominance: open questions, challenges and solutions. *Philos. Trans. R. Soc. B* **377**(1845), 20200445 (2022)
52. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge, MA (2018) 0262039249
53. Tomlinson, B., Blumberg, B., Rhodes, B.: How is an agent like a wolf?: Dominance and submission in multi-agent systems. In: International ICSC Symposium on Multi-Agents and Mobile Agents in Virtual Organizations and E-Commerce (MAMA'2000) (2000)
54. Tufano, M., Agarwal, A., Jang, J., Moghaddam, R.Z., Sundaresan, N.: AutoDev: Automated AI-driven development (2024)
55. Vezhnevets, A.S., et al.: Generative agent-based modeling with actions grounded in physical, social, or digital space using concordia (2023)
56. Vinitsky, E., et al.: A learning agent that acquires social norms from public sanctions in decentralized multi-agent settings. *Collective Intell.* **2**(2) (2023). <https://doi.org/10.1177/26339137231162025>
57. Watson, J.R.: Dominance-subordination in caged groups of house sparrows. *The Wilson Bulletin*, pp. 268–278 (1970)
58. Williamson, C.M., Lee, W., Romeo, R.D., Curley, J.P.: Social context-dependent relationships between mouse dominance rank and plasma hormone levels. *Physiol. Behavior* **171**, 110–119 (2017)
59. Woolley, A.W., Chabris, C.F., Pentland, A., Hashmi, N., Malone, T.W.: Evidence for a collective intelligence factor in the performance of human groups. *Science* **330**(6004), 686–688 (2010)

60. Wu, Q., et al.: AutoGen: Enabling next-gen LLM applications via multi-agent conversation (2023)
61. Xi, Z., et al.: The rise and potential of large language model based agents: A survey (2023)
62. Xu, Y., Deng, Z., Wang, M., Xu, W., So, A.M.C., Cui, S.: Voting-based multi-agent reinforcement learning for intelligent IoT (2020)
63. Zhan, S., Bendapudi, N., Hong, Y.y.: Re-examining diversity as a double-edged sword for innovation process. *J. Organ. Behav.* **36**(7), 1026–1049 (2015)



Social Deliberation vs. Social Contracts in Self-governing Voluntary Organisations

Matthew Scott¹ , Asimina Mertzani¹ , Ciske Smit¹ , Stefan Sarkadi² , and Jeremy Pitt¹

¹ Department of Electrical and Electronic Engineering, Imperial College London,
London, UK
j.pitt@imperial.ac.uk

² Department of Informatics, King's College London, London, UK

Abstract. Self-organising multi-agent systems regulate their components' behaviour voluntarily, according to a set of socially-constructed, mutually-agreed, and mutable *social arrangements*. In some systems, these arrangements may be applied with a frequency, at a scale and within implicit cost constraints such that performance becomes a pressing issue. In this context, this paper introduces the *Megabike Scenario*, which consists of a negotiated agreement on a relatively ‘large’ set of conventional rules and ‘frequent’, ‘democratic’ deliberation and decision-making according to those rules, but a resource-bounded imperative to reach a ‘correct’ decision. A formalism is defined for expressive rule representation combined with efficient and effective rule processing, which is evaluated against five interleaved *socio-functional* requirements. System performance is also evaluated empirically through simulation. We conclude that to self-organise their social arrangements, agents need some awareness of their own limitations and the value of compromise.

1 Introduction

In a self-organising multi-agent system [15], the component agents voluntarily agree to regulate their own behaviour according to a set of socially-constructed, mutually-agreed, and mutable *social arrangements*. Informally introduced in [6], we define “social arrangements” as an umbrella term for any type of conventional rule-based system that members of a group agree on for voluntarily regulating behaviour and holding themselves accountable to one another, whether this is a convention, norm, procedure, regulation, institution, contract or law [13, 21, 26].

In principle, these social arrangements support self-governance through deliberative processes, whereby those who are affected by the arrangements participate in their selection, modification and enforcement, and self-determine certain configurations of the social arrangement to be congruent with, or fit-for-purpose for, prevailing environmental conditions [6, 13]. In practice, the social arrangements may need to be applied with a frequency, at a scale, and within implicit cost constraints such that performance becomes a pressing issue, and especially so in the presence of existential threats.

Therefore, the motivating problem for the work reported in this paper is the need for an institution or organisation to compromise on resource-intensive processes implementing its social arrangements. However, we do not propose to compromise on democratic processes (cf. [18]) by reducing participation through sortition or “representative democracy”, but by using instead *social contracts*. The aim of these contracts is to combine an expressive rule representation with efficient and effective rule processing (effective in the sense that outcomes produced by the social contracts should be as “correct” as, for example, the products of deliberative and decision-making processes).

Moreover, given the centrality of ‘correctness’ in the outcomes of deliberative decision-making¹, this demands that an alternative approach based on social contracts needs to be evaluated against five interleaved *socio-functional requirements*, which concern the entanglement of social arrangements with computational operation, namely:

- *Scalability*: do the social arrangements scale with the number of agents, the number of rules, the cost and frequency of applying the rules, etc.?
- *Complexity*: are the social arrangements congruent with the ‘cognitive’ ability of the agents to use them, and do they correspond to the difficulty of the problem to be addressed (cf. [22])?
- *Mutability*: how ‘easy’ or ‘quick’ is it to change both the social arrangements, and agreements underpinning the choice of social arrangement, especially in real-time, existential-risk situations (cf. [24])?
- *Enforceability*: can sanctions specified by the rules be enforced effectively in decentralised systems with no form of coercion? and
- *Versatility*: how ‘seamlessly’ can the social arrangements, and conceptual resources that are socially-constructed externalities produced by applying these arrangements, be transposed to another context?

Therefore, the motivating problem for this work is the reduction in the burden of self-governance by the use of expressive and tractable social contracts [23] instead of expensive and potentially intractable deliberative processes. These social contracts are evaluated against the socio-functional requirements in the context of the *Megabike Scenario*. In this scenario, autonomous agents have to, first, self-select membership of a group; secondly, within the group, negotiate an agreement on a relatively ‘large’ set of social arrangements; and thirdly make ‘frequent’ ‘democratic’ decisions according to these arrangements, within a resource-bounded imperative to reach ‘correct’ decisions. Being used to resolve iteratively several inter-dependent social dilemmas (including scarce resource allocation and collective risk), the social arrangements are subject to the identified performance issues, and so social contracts preferred to deliberative processes.

Accordingly, this paper is structured as follows. The next section introduces the *Megabike Scenario*, while Sect. 3 summarises the ‘decision-making arenas’ required by the scenario and exposes the limitations of social deliberation.

¹ For example, Ober [12] attributes the success of the classical Athenian democracy in outperforming other city-states, despite parity in other metrics, to its superior knowledge management processes which produced ‘better’ decisions more often.

Section 4 specifies a formalism for effective representation of social contracts, as an alternative to social deliberation, after which Sect. 5 evaluates the formalism against the socio-functional requirements. The performance improvements are evaluated empirically through a simulation of the *Megabike*, described in Sect. 6. After a discussion of related research in Sect. 7, we conclude in Sect. 8 that the significance of these results for self-organising multi-agent systems is that the agents need an acute self-awareness of their own limitations in selecting and modifying their social arrangements.

2 The *Megabike* Scenario

A *megabike*, based on “real world” bikes for multiple riders, allows a group of otherwise autonomous agents to occupy a single vehicle collectively, and then propel it (pedal) to navigate a typical AI/multi-agent gridworld, both in search of rewards (*lootboxes*) and to avoid an existential threat. Each agent is individually capable of pedalling, braking, and steering the *megabike*; consequently, the agents must collectively agree on (and each agent explicitly agrees to voluntarily comply with) the social arrangements that determine direction (steerage), effort (pedalling and braking), lootbox targeting and loot allocation, and assignment of social roles.

Ultimately each agent wants to maximise its own utility, measured in terms of energy gained from lootboxes, duration of survival, and/or others’ appreciation of its individual contribution to a collective effort (e.g., energy expended, compromising, performance of a social role, etc.). The scenario therefore involves:

- *institutional foundation*, as the agents must negotiate firstly, with whom to team up with to occupy a *megabike*, and secondly, what social arrangements (form of voluntary self-governance) are agreed to be in force;
- *coordination*, in terms of selecting a target lootbox and investing personal resources (energy) into a collective effort to move there; and avoiding both contradictory actions (pedalling and braking at the same time) or duplicating actions (causing e.g., over-steering);
- *distributive justice*, in terms of how to allocate rewards from the successful appropriation of the contents of a lootbox, according to negotiated criteria also specified by the social arrangements;
- *normative compliance*, in terms of compliance, or otherwise, with group decisions made according to the agreed social arrangements, and punishment for non-compliance (although the collective with regards to...)
- ...*cooperative survival*, in that the *megabike* needs sufficient occupants with sufficient energy in order to increase its chances of acquiring lootboxes before other *megabikes* take them, and avoiding an existential threat;
- *contributive justice*, in the form of opportunities to contribute meaningfully to successful collective action, norm compliance, compromises, etc., and to be appreciated for such contributions; and
- *social construction*, in the form of conceptual resources like esteem, trustworthiness, social networks, etc., which start from zero but accumulate (are socially-constructed) over time.

An informal specification of the *Megabike* Scenario ‘game loop’ is given in Algorithm 1. The ‘game’ is played over multiple iterations, each iteration consisting of multiple rounds. At the start of each iteration, the agents negotiate membership of, and social arrangements for, their own *megabike*, and perform role assignment (in particular identifying a *leader*). In each round, they select a target lootbox, then commit some of their own energy to pedalling, braking or steering actions. If they succeed in acquiring a lootbox, then they have to distribute the resources according to the negotiated social arrangements. Finally, they follow protocols to exclude an agent (ultimate sanction for non-compliance) or admit a new agent (if there is an unfilled ‘seat’ on the *megabike*).

Algorithm 1: *Megabike* Scenario ‘Game Loop’

```

1 i  $\leftarrow 0$  ; initialise agents ;
2 repeat
3   | (within agents) negotiate megabike membership ; //self-selection phase
4   | (within megabike) negotiate social arrangements ; //action phase
5   | (within megabike) perform role assignment ;
6   | j  $\leftarrow 0$  ;
7   | repeat      // operation phase
8     |   | (within megabike) decide target lootbox ;
9     |   | (within megabike) each agent decides pedal/brake/steer actions;
10    |   | (environment) apply effects of actions ;
11    |   | if lootbox reached then
12      |   |   | (within megabike) apply resource allocation
13      |   |   | (within megabike) admission/exclusion ;
14      |   |   | inc(j) ;
15    |   | until j == MaxRounds OR megabike terminated;
16    |   | inc(i) ;
17 until i == MaxIterations OR deadlock;

```

Therefore, a *Megabike* game (in the simulation sense) consists of several (inter-related) sub-games (in the game-theoretic sense), including:

- a veil of (decreasing) ignorance dilemma [19]: at the start of each iteration, knowledge about negotiating which bike to join and which social arrangements to adopt changes over time, from a “risk” trust decision to a “reliance” trust decision as information is gained about the behaviour of other agents;
- a pair of collective action dilemmas [13]: the optimal utility-maximisation and self-preservation strategy is (literally) free-riding, by not expending energy while other agents do all the pedalling; but if all agents use this strategy, all suffer, because being stationary is unsustainable as the *megabike* will be terminated by the existential threat or competition with other *megabikes* for lootboxes;
- a resource allocation dilemma [20], where the allocation of scarce resources gathered from any lootboxes must be decided according to a protocol included

- in the set of agreed social arrangements (rather than a brutal ‘real time’ grab); and
- a balloon debate dilemma, in which some agent might have to convince the others that it should not be excluded, or that another agent should be excluded (e.g., for reasons of non-compliance with the agreed social arrangements or leadership decisions).

As per [11, 13], resolving these dilemmas requires the negotiation of social arrangements, as discussed in the next section.

3 Social Deliberation for Self-governing *Megabikes*

This section considers decision arenas in which social arrangements are required to negotiate agreements and reach decisions through processes of social deliberation. However, it also exposes some of the situational limitations of unrestricted social deliberation.

3.1 Social Deliberation

As indicated by Algorithm 1, the individual demands on agents in the *Megabike* Scenario are manifold, but also they need to engage in (potentially) substantial processes of action selection, collective (social) deliberation and decision-making. For the former, each agent will need its individual strategy, which is not discussed further here, except to the extent that preferences inform the latter. For this, deliberation and decision-making, they need a relatively ‘large’ set of rules or procedures (i.e., their social arrangements).

These social arrangements are negotiated once a group of agents have all agreed to occupy a *megabike*, often involving a particular type of rule, and then parameters for that rule. For example, decisions might be made by the assigned *leader*, or by majority vote; if they take a vote, they have to decide which of many voting methods to use (e.g., plurality, alternative vote, Borda count, etc.).

Rules are derived from multiple sources, including self-governing institutions for common-pool resource management [13], norm-governed multi-agent systems using institutionalised power [1], distributive justice using legitimate claims [20], the Game of Nomic [27], Robert’s Rules of Order [21], Basic Democracy [11] and democracy-by-design [18]. As such, the decision arenas [13] include:

Mutability [11, 27]: an agreement is required on whether or not to distinguish between mutable and immutable rules. If a distinction is agreed, then there needs to be one protocol for converting a mutable rule to an immutable rule, and another protocol for converting an immutable rule to a mutable one.

Role Assignment [1]: minimally, a *leader* will be assigned, who will then have the institutionalised power to assert certain (institutional) facts [7]. A protocol for selection is required, alternative methods include by vote, weighted criteria, torno (if equal participation is a key principle [11]), and historical (auto-autocrat). Equally, a protocol for de-selection is required, if the elected representative takes undue advantage of its position [18]. Note, the collective might

delegate other tasks to ‘responsible’ agents to minimise costs (e.g., steering, observing, etc.), which opens up further opportunities for investigating deception.

Lootboxes [13,20]: a protocol for lootbox target selection is required: this might entail a phase of ‘democratic’ deliberation (knowledge aggregation) with respect to social welfare optimisation, and a method for knowledge alignment, i.e., having taken the decision, how to ensure that all agents act upon it effectively. The *leader* might, for instance, use its institutionalised power to oblige all agents to pedal with a certain intensity. Having acquired a lootbox, a protocol for resource allocation is required. If there is no protocol fixed, the agents might try to ‘grab’ loot, i.e., this introduces a hawk-dove game to the scenario.

Membership [13]: a key feature of collective action decision arenas are the boundaries on who is and is not constrained by the social arrangements, and which are and are not common interest (rather than factional) issues. However, since the social arrangements are conventional, non-compliance is a possibility; if the ultimate sanction is exclusion then a protocol for this is required (implying pre-conditions, process, appeals procedure, etc. [15]). Similarly, and since agents may be eliminated due to depleted energy, an admissions protocol is also required: since there are various alternative admissions processes, this is another parameter that needs to be negotiated.

Monitoring and Sanctions [13]: assuming an open system, there is no full disclosure, and agents cannot see how much effort other agents are putting into pedalling. However, they can know if someone under-contributed, and for this reason, there needs to be an auditing protocol. This would require another role assignment, a choice between methods and outcomes (e.g., cheap but unlikely to reveal non-compliance, expensive but likely). Note that a ‘monetised’ obligation for monitoring and auditing also exposes risks of non-compliance with the role (doing the work, reporting results honestly, etc.). Therefore, a graduated scale of sanctions for non-compliance needs to be negotiated.

Crisis Response: to deal with an existential threat, a switch between decentralised and centralised decision-making might be negotiated. The problem is to ensure that once the crisis has passed (if it existed in the first place [8]), democratic backsliding is averted and the social arrangements revert to their original form, and do not get stuck in an autocratic or hierarchical regime [6].

3.2 The Limitations of Social Deliberation

In the *Megabike* scenario, given the number of decision arenas, the frequency of decision-making in those arenas, the cognitive and communicative overheads of social deliberation, and the pressure of an existential threat, there are limits on the use deliberative processes, and a more efficient but equally effective way of reaching a ‘correct’ decision is required.

There are, in fact, precedents for substituting complex procedures in self-organising multi-agent systems. For example, in previous work, e.g., [1], the focus of attention was on events which determined the (institutionalised) powers, permissions and obligations of agents. For this purpose, the Event

Calculus (EC) was an effective tool for executable specification, but ‘simple’ Prolog implementations of the EC proved unsuitable for simulations with large numbers of agents or ‘long’ narratives. For experimental simulation, EC-based logical representation of Ostrom’s design principles for self-governing institutions [13] were re-implemented in procedural or object-oriented languages to address issues of scale and run-time ([16, 17]). For real-world applications, this limitation of the EC led to the development of a logic formalism for large-scale run-time event recognition [2].

Furthermore, in the implementation of the thought experiment Demopolis [11], the rules were conceived as defining a specification space of n rules each with m parameters, with each parameter having x values, so the ‘ k -th’ rule, R_k , was defined as in Eq. 1.

$$R_k = \begin{bmatrix} P_{1,k}(V_{1_{1,k}}, \dots, V_{x_{1,k}}) \\ P_{2,k}(V_{1_{2,k}}, \dots, V_{x_{2,k}}) \\ \vdots \\ P_{m,k}(V_{1_{m,k}}, \dots, V_{x_{m,k}}) \end{bmatrix} \quad (1)$$

While this meant that rules could be conveniently written and executed as Prolog predicates, changing parameters meant retracting and re-asserting clauses, and changing sets of rules meant re-consulting files. Both of these operations are relatively ‘slow’, and so mutability was correspondingly problematic [18].

To compound matters, for agents embedded in the actual simulation, the processes of social deliberation, as previously enumerated, incur substantial overheads in communication, especially as the number of agents, and the frequency with which deliberation occurs, scale upwards. These overheads cause obvious problems in situations where cooperative survival is a condition of continued participation, both individually and collectively (i.e., no-one survives unless everyone survives). There are further problems of asynchrony and concurrency that open distributed systems have to address, especially issues of timing, sequence and causality in systems without global clocks, which generally do not present comparable difficulties for social systems.

In the context of the *Megabike Scenario*, we propose that during the rule selection phase, the agents should also determine whether or not deliberation can be replaced instead by a *social contract* [23]. This is not going against a principle of Democracy-by-Design [18], “no short-cutting democratic processes”, but instead is seeking to define a more efficient, and mutually-agreed, rule-based alternative to social deliberation. Note that this proposal implies that in the (second) entrenchment phase of negotiating the social arrangements for a *megabike*, the Θ -Learning algorithm [9] could be used for reaching consensus while exploiting compromise and dissent as conceptual resources; and in the (third) operationalisation phase, there is an opportunity to use learning algorithms to customise the social contract by establishing the pathways to requisite social influence [10].

Therefore, we need an appropriate (i.e., computationally tractable) rule representation which can be used as a surrogate for (potentially computationally intractable) social deliberation: for example, *social contracts*.

4 Social Contracts for Self-governing *Megabikes*

In this section, we specify *social contracts* as a set of rules, which effectively prune the search space of possible decisions. In this way, we can reframe social deliberation as a social contract, whereby rule application and mutability are re-interpreted as optimisation problems.

In this case, agents can avoid deliberation by mutually agreeing on a set of rules that offsets the deliberation process by approximating it instead. For example, rules may be imposed using quantitative features of the lootboxes (such as relative distance or reward) to restrict the possible set of lootboxes that may be voted on by the agents. A ‘well-optimised’ ruleset would be one that effectively removes the need for social deliberation, as after pruning the full set of lootboxes, only one is left. This process, while not deliberative, yields the same outcome: a single decision resulting from an initial search space of multiple valid decisions. As such, the ruleset serves as a ‘proxy’ for deliberation, where the search space is pruned not through deliberation (agents gradually converge on a single decision), but by the elimination of invalid decisions through rules. This has the added benefit of reducing the computation needed by the agents, and instead places the load on the server, thereby allowing for faster agent operation if this scenario were translated to an asynchronous system, say.

Using an optimisation paradigm favours a mathematical rule representation, whereby techniques such as gradient descent or simple estimators can be used. Therefore, we use a matrix representation, specifying other parameters required for evaluating the rule and allowing for efficient rule retrieval. We formalise the rule representation with respect to the design considerations outlined in Sect. 1, and give an example of how a declarative rule can be converted into matrix form. This gives an abstract, general-purpose rule model that can be codified for efficient computation and evaluation.

4.1 Rule Representation Formalism

There are various parameters used to address the design considerations in this rule representation. Table 1 illustrates this by giving the parameter name and data type used for each element in the representation. We describe the representation according to three sections: how the rules can have unique *identification*, how the rules are built for efficient *evaluation* and how the rules are *mutable*.

Identification. Firstly, all rules have a uniquely generated *ruleID* to allow all rules to be uniquely identifiable. This allows for rules to be accessed from a global lookup (a hashmap cache, say) and for agents to store a reference to the rules that they are currently using. We also supply a *ruleName* for a ‘quality of life’ benefit to the rule designer (the programmer), as this allows for a meaningful description to be added to each rule such that the programmer can see which rules are used by an agent without having to meticulously check UUIDs. The final parameter used for identification is the *ruleAction*, which binds each rule

Table 1. *Rule* representation data structure

Parameter	Range
ruleID	<i>UUID</i>
ruleName	<i>string</i>
ruleIsMutable	<i>bool</i>
ruleAction	<i>enum</i>
ruleInputs	$\{\}(\text{func}() \rightarrow \text{float})$
ruleMatrix	$\{\}[]\text{float}$
ruleComparators	$\{\}[]\text{operators}$

to the action that they constrain. For example, a rule may need to be applied on the lootbox decision, the election of an agent to power, or the direction that the bike should travel. Binding this rule to an action allows for efficient extraction of the relevant rules, to ensure that rules which do not affect the outcome of an action (and would therefore pass as true anyway) are not evaluated, saving computation time.

In the context of a simulator, there is likely to be a system comprising a large number of these types of rules. This could be codified with a hashmap, mapping the *ruleAction* to a list of rules, for example, as this would allow for the relevant rules to be extracted in constant time.

Evaluation. There are three components used for evaluating a rule. The first is a set of *ruleInputs*, or the quantitative information that the rule concerns. This allows for a rule to be suitably *versatile*, as it becomes applicable to any object in the simulator, via a getter function. For example, a rule may need to be evaluated against a lootbox, concerning its position or value. A rule may also be applicable to an agent, concerning their resources (energy) or esteem. As such, specifying the constraints of a rule becomes possible with a generic getter function, allowing for a single rule engine to be applied to any kind of object (or interface, programmatically).

The second component needed for evaluating a rule is the *ruleMatrix*, which is a 2-D array of numbers that applies weighting to the *ruleInputs*, thereby allowing for numerical constraints to be applied. Taking the previous example of the position of a lootbox as an input variable, the *ruleMatrix* may apply a weighting of 100, to define a rule that compares the (relative) position of a lootbox with a distance of 100 units. How this comparison is made is defined in the final evaluation parameter, with the *ruleComparators*.

The *ruleComparators* define how the input parameter is compared against its numerical weighting, using an *operator* in the set $\{ <, >, \leq, \geq, = \}$. Completing the previous example, we can define a rule that a lootbox must be within a distance of 100 units for consideration. The input parameter then becomes the lootbox position, the matrix applies a weighting of 100 and the comparator

evaluates this with the ‘less than or equal to’ operator (\leq). In Sect. 4.2, we give a more complex example of a rule that concerns multiple inputs, and multiple clauses, demonstrating why a matrix is used, over a single scalar weight.

Mutability. A final design consideration is the importance of rule mutability. Given the representation as a matrix of numbers, mutability becomes trivial, as an agent/designer simply needs to change the value of a matrix element. Again, using the example of a constraint on lootbox distance, the value of 100 can be changed in the *ruleMatrix* to 50, say, to give a tighter restriction on the set of feasible lootboxes. Conversely, this value may be changed to 200, to provide more ‘slack’ on the constraint, and allow for a weaker constraint on the feasible lootboxes, and hence a wider array of possible options. Naturally, these rules may not be intended to be mutable, so we provide a *ruleIsMutable* flag that dictates if the rule can be changed or not.

4.2 Example: Lootbox Pruning

The simplest way to define a rule in this grammar is to start with a declarative rule and convert it to a numerical representation. The (simple) rule from the previous section, *A valid lootbox must be within 100 units*, can be expressed as an inequality using d for relative distance as $d \leq 100$.

In order to get a ‘better’ outcome, an agent may propose that the distance should reflect the payoff of the lootbox, adding a second clause such that *A valid lootbox must give a payoff of at least 1.5 times its distance*. This can also be interpreted as an inequality, with p representing payoff, as $p \geq 1.5 * d$.

As such, we arrive at two equations that must simultaneously be true for a rule to pass. Reformatting these equations, setting them equal to zero and ascribing scalars for all variables gives:

$$\begin{aligned} 1 * d + 0 * p - 100 * 1 &\leq 0 \\ 1.5 * d - 1 * p + 0 * 1 &\leq 0 \end{aligned}$$

which can be interpreted in matrix form as:

$$\begin{bmatrix} 1 & 0 & -100 \\ 1.5 & -1 & 0 \end{bmatrix} \begin{bmatrix} d \\ p \\ 1 \end{bmatrix} \begin{bmatrix} \leq \\ \geq \end{bmatrix} \vec{0} \quad (2)$$

yielding, from left to right, the three components for rule evaluation: the *ruleMatrix*, the *ruleInputs* and the *ruleComparators*, the result of which, after matrix multiplication, is compared against the zero vector.

5 Evaluation of Socio-functional Requirements

In this section, we evaluate the rule representation with respect to the five *socio-functional* requirements introduced in Sect. 1.

5.1 Complexity Reduction

Rule Evaluation. Given a scenario with a different actions and r different rules, and in the context of repeated iteration, the requirement to optimise the rule evaluation is increasingly important. Following from the rule representation, if a rule is evaluated for an irrelevant action (a lootbox rule against an election action, say) the input parameters will be unrelated to the problem (a lootbox's value isn't necessary for checking an agent's electoral eligibility), and therefore the rule will pass by default. As such, unnecessary computation is spent evaluating all rules. Given a rule matrix of size $n * m$, with n clauses and m input parameters, evaluation will run with $O(n * m)$ complexity. Given the full set of r rules, which are evaluated for all a actions, this yields the complexity of a single agent evaluating a decision as $O(n * m * r * a)$.

By ascribing a *ruleAction* to the representation, as in Table 1, the rules can become *stratified*, such that only a subset of the rules require evaluation. We denote this subset with r' . As such, by storing the active rules in a hashmap, such that the rules can be extracted in $O(1)$ time, the overall complexity is reduced to $O(n * m * r' * a)$. Given that the full ruleset is partitioned into actions, we can say that $r' * a \leq r$, and as such the final complexity is $O(n * m * r)$, simplifying the complexity by a factor of a .

Deliberation vs Social Contracts. We can also consider a complexity improvement from the perspective of deliberation in the simulator. Previously, deliberation was the mechanism for action selection, which, in the context of *Megabike* occurred in every operation phase, and therefore in every round (see Algorithm 1). This meant that, for every iteration, there were at worst *MaxRounds* operation phases being run.

By transitioning to social contracts, the negotiation of rules is instead moved to the action phase, therefore being run only once per iteration. Considering i iterations and j rounds per iteration, with k opportunities for deliberation from Sect. 3.1, there were previously $O(i * j * k)$ deliberation sessions. Using social contracts allows for a single social contract negotiation session per iteration, reducing the complexity to $O(i)$.

5.2 Linear Optimisation

Instead of interpreting a decision as requiring the iteration of an array of distinct rules, it is possible to combine all rules into a single matrix. For example, given two distinct rules as follows (with arbitrary input parameters):

$$\begin{bmatrix} 1 & 0 & -100 \\ 3 & -1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \begin{bmatrix} \leq \\ \leq \end{bmatrix} \vec{0} \quad (3)$$

$$\begin{bmatrix} 4 & 3 & -1 \\ 5 & -7 & 2 \end{bmatrix} \begin{bmatrix} z \\ w \\ 1 \end{bmatrix} \begin{bmatrix} > \\ = \end{bmatrix} \vec{0} \quad (4)$$

it is possible to ‘stack’ rules into a single large (potentially sparse) matrix:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & -100 \\ 3 & -1 & 0 & 0 & 0 \\ 0 & 0 & 4 & 3 & -1 \\ 0 & 0 & 5 & -7 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ w \\ 1 \end{bmatrix} \begin{bmatrix} \leq \\ \leq \\ > \\ = \end{bmatrix} \vec{\mathbf{0}} \quad (5)$$

As such, this set of constraints may be reinterpreted as a linear optimisation problem, where agents may attempt to maximise some objective function (survivability, or energy, say) based on a set of constraints.

5.3 Slack: Flexibility and Mutability

Further changes can be made to the representation to generate the ruleset as a *data tableau*, so that slack variables can be used for linear optimisation. This technique synergises well with the rule representation, as the mutability of the data structure allows for not only convenient redefinition of the rule constraints but the removal or addition of extra slack for a stricter or more lenient policy, respectively. This makes the rules not only mutable, but *flexible* as well.

5.4 Enforceability and Transparency

By having the rules bound to each *Megabike*, the mutually agreed rulesets that inform all decisions are visible not only to other agents, but to the simulator designer as well. The benefit of this is twofold. Firstly, the ruleset can be seen as a tangible representation of the current agent state: that is, the algorithm they would use to decide on an action becomes publicised and interpretable in the form of a rule, i.e., instead of each agent deciding on which action to perform, which would be in the form of a black box process, they instead mutually decide on a rule which would prune the actions they wouldn’t carry out. The ruleset then becomes an aggregation of all of the agents’ internal processes, such that the action space yielded by evaluating the ruleset results in the set of actions that would be voted on by the agents, anyway.

The second benefit to this representation is the ability to take computational demand away from the agents. Instead of having agents individually select an action using their own algorithm, the ruleset can be evaluated server-side instead, moving computation from individual agents onto the server. Given a synchronous system where runtime efficiency is imperative to avoiding race conditions, more complex agents can be built that don’t suffer from having to run quickly to move first. Having rule evaluation performed by the server also means that the rules becomes more *enforceable*; the server cannot be coerced into mis-evaluating a rule for personal gain, unlike if an agent were to perform this role.

5.5 Versatility

There are two dimensions to versatility: for the simulator and for the simulated. Our concern here is for the simulated: what we want to evaluate is the extent to which an agent which learns social arrangements for the *megabike scenario* could apply that ‘learning’ to a different scenario. This is not a requirement that can be evaluated in the current work, but is left for future work.

6 Empirical Simulation Results

In this section, we quantitatively demonstrate the functional design considerations discussed in Sect. 1: *scalability*, *complexity* and *mutability*. For these experiments, we consider a simulator comprising 100 iterations of 100 rounds (per Algorithm 1), and run the simulator 30 times to aggregate the results.

6.1 Experiment 1: Scalability and Complexity

This first experiment aims to demonstrate how the inclusion of rule stratification by *ruleAction* results in decreased runtime for the program, and supports the claims made to reducing time complexity in Sect. 5.1. In this experiment, we evaluate the runtime of the program per iteration across ruleset sizes of 1, 10, 100 and 1000, and number of agents in the simulation at 1, 8, 16, and 32. The results of this experiment are shown in Fig. 1.

To benchmark the rule representation, we need to test the worst-case runtime for the simulator. This would occur when each agent needs to evaluate every single rule. For simplicity, we define a rule that is guaranteed to pass, irrespective of the agents’ state, such that the rule evaluation isn’t prematurely stopped (as there is no point evaluating further rules once one has failed). To do so, we define a *ruleMatrix* of all zeroes, and equate it directly with the zero vector. This rule serves as a ‘null’ rule, which means that irrespective of the input parameters (and therefore the agent’s state), the rule will pass, since all input parameters are multiplied by zero. This simplifies the calculation to whether $0 == 0$, which is always true.

The results of these experiments show that the runtime of the program per iteration is significantly reduced. Given the subset of five actions we have defined, the runtime is reduced by (approximately) a factor of five, which supports the theoretical analysis performed in Sect. 5. This shows that the rule representation is *scalable* in proportion to the number of actions, agents and ruleset size. As such, more complex simulators can be built and run in a feasible time.

6.2 Experiment 2: Mutability

The second experiment is designed to illustrate the importance of, and simplicity in, modifying rules at runtime for survivability. In this experiment, we define a single rule that impacts the subset of lootboxes that are eligible for voting. This

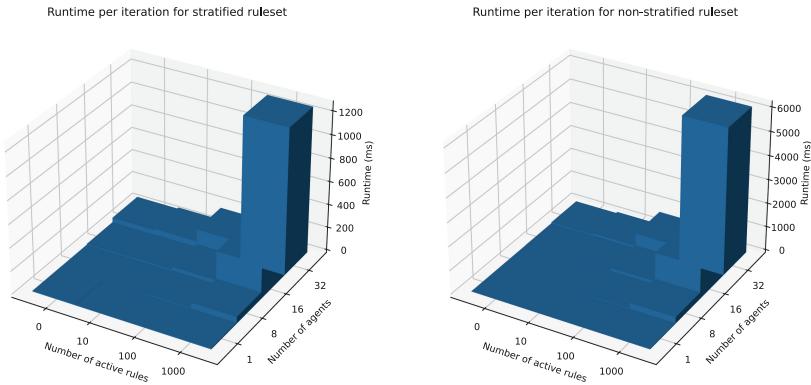


Fig. 1. Runtime profiling of *Megabike* simulator for stratified (left) and non-stratified (right) rulesets

rule (initially) states that all eligible lootboxes must be within a radius of 1000 units. If an agent's energy falls below 50% of the maximal capacity, agents will propose to amend the rule, once per turn, by applying a slack of 5% (that is, increasing the radius of detection by 5%). Alternatively, if an agent's energy is at least 50%, the agent will propose to amend the rule by removing a slack of 5%, thereby shrinking the radius of detection by 5%.

This experiment varies the scarcity of resources, by varying the ratio of lootboxes to agent. Given 100 agents (across all runs), we first establish a baseline by giving a ratio of 0, thereby assessing how agents would survive given no external resources and increase this ratio to 0.5, 1.0, 1.5, 2.0 and 2.5. We also vary the capacity to mutate the rule, and illustrate the results in Fig. 2.

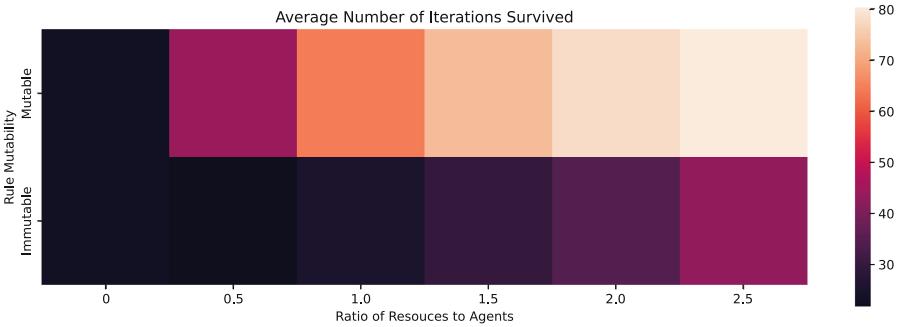


Fig. 2. Average survivability of agents in *Megabike* for varying degrees of resource scarcity and mutability of rules

By increasing the ratio of available lootboxes to agents, thereby alleviating the economy of scarcity, agent survivability is improved. Across all degrees of

scarcity, Fig. 2 shows that the mutability of rules has a significant impact on survivability, outperforming the immutable rules at every stage.

Using 0 lootboxes as a baseline, in either case, the agents survive for around 20 rounds, as the mutability of rules has no bearing on the number of lootboxes that can be achieved. In the mutable case, increasing this ratio has a drastic effect on survivability, where for an increase in 0.5x the resources, a further ≈ 20 rounds of survival are allowed, up to ≈ 80 when 2.5x the resources are present.

Conversely, with immutable rules, agents struggle to survive, even with 2.5x the resources, where agents still only survive for ≈ 40 rounds on average. This is due to the agents consistently depleting the nearby lootboxes over time, thereby increasing the need for a larger perception radius to detect further away resources. In the mutable case, this is exactly the kind of policy change being negotiated: the bikes consistently agree on more and more slack to be given to the rule until a sufficient number of lootboxes become visible.

7 Related Research and Further Work

7.1 Related Research

Defeasibility, the property of a claim or rule to be falsified, changed, replaced, or ‘mutated’, acts as a pillar in dynamic agent-agent communication [3,14]. The MAS literature is not short of approaches for deciding which rules of a socio-technical multi-agent system should be changed and how. However, most approaches tackle rule change by describing processes which, despite being transparent, interpretable, and tractable, ultimately place a ‘heavy’ burden on AI agents and MAS engineers from a socio-functional requirement standpoint.

A different perspective on addressing socio-cognitive properties is taken by the agent-oriented programming languages (AOPLs) that specify how to implement agent communication languages such as FIPA and KQML. AOPLs have previously been extended to integrate the multiple layers of abstraction in the multi-agent systems literature, which are the agent layer, the environment layer, and the organisational layer.

A modular and scalable example of the approach is the JaCaMo framework [4]², which integrates the agent, environment and organisational layers under a single unified multi-agent-oriented programming ‘language’. Indeed, an important property of the rules implemented in such systems is their *defeasibility*. The processes that agents of such systems follow to interact should be able to allow the agents to change the rules when it is reasonable to do so. More recently, the JaCaMo framework has been used in the MAS community for creating distributed human loop systems that leverage argumentation and mentalisation techniques, e.g., where AI agents and human users share evidence to reach more justified conclusions about each other’s mental attitudes [25].

Despite the ‘academic’ advancements in engineering socio-cognitive MAS, fewer real-world applications of such systems have been deployed. This might be

² <https://jacamo-lang.github.io/getting-started>.

because the problem of having a speedy and cognitively efficient approach for both AI agents and experimenters regarding rule processing persists. One exception is the MAIDS framework, a JaCaMo extension, that implements intentional dialogue AI systems for human-AI teams that self-organise to optimise hospital bed allocation [5]. However, the hospital bed allocation problem does not have the same entangled complexity of the *Megabike* scenario, which involves self-selection, self-determination of social arrangements, and existential threats. *Megabike* aims to provide insights into social arrangements, whereas MAIDS is tailored to a very specific domain problem.

In more detail, there are some substantive differences between *Megabike* and the hospital bed allocation problem. For example, one aims to solve an applied problem (hospital bed allocation) and the other aims to solve an abstract problem regarding social arrangements (*Megabike*). Thus *Megabike* aims to provide insights into social arrangements, especially regarding with how social contracts between agents give rise to emergent behaviour, both within a *Megabike* and eventually between *megabikes*. Alternatively, the hospital bed allocation is a problem which does not imply social arrangements to be worked out, on the fly, by the patients and doctors, as it has well established rules and organisations that agents can follow and use to solve their problems.

Furthermore, in the *Megabike* scenario agents need to self-organise in order to maximise their own utility and ‘survivability’, which also depends on the contribution of others, whereas in the hospital bed allocation the agents want to coordinate and communicate, i.e. self-organise, in order to better cooperate such that they optimise a type of resource (bed occupancy) in multiple organisations (hospitals). *Megabike* is concerned with the agents figuring out how to run the entire organisation as a group of agents with individual interests that choose to have or to change various social arrangements to do so.

7.2 Further Work

This comparison between hospital bed allocation and *Megabike* does inspire a line of work for further research. Within the hospital bed allocation problem, there are core questions of an ethical and safety-critical nature, since it involves ‘real’ patients whose health and well-being are “at the sharp end” of an artificial intelligence-enabled decision-support system. At the same time, it is known from the Covid-19 pandemic, that doctors could face a difficult triage problem, especially with bed occupancy in intensive care units. An analogous situation could be developed within the context of the *Megabike* scenario by an extension to human-in-the-loop experiments with *Megabike* as a Serious Game, and use this to investigate questions of humanist ethics [28]. For example, from the triage perspective, it would be particularly interesting, in the context of the balloon debate dilemma, to understand how people react to and feel about being excluded by artificial agents.

Furthermore, having demonstrated that the rule representation is appropriate for use in social contracts, we aim to further develop the idea that social contracts are a flexible and satisfactory ‘proxy’ for social deliberation, and which give a

good approximation of the optimal solution. Additional experiments would be run over two (iterated) simulations: one where social deliberation is allowed and social contracts are not (to serve as a benchmark), and another with the opposite conditions. We would analyse these simulations by considering how ‘happy’ the agents are with the *Megabike* they are on, which is shown qualitatively by the (average) number of bike exclusions: both voluntary and involuntary.

We hypothesise that the use of social contracts should be able to approximate the number of exclusions occurring with social deliberation. For a ‘good’ approximation, we would find that using social contracts only overshoots the number of exclusions (under deliberation) by 10%, say, giving a confidence interval of 90%. Furthermore, through iteration, the probability of deviating by less than 10% should be within one standard deviation.

8 Summary and Conclusions

In summary, this workshop paper is set in the context of self-organising multi-agent systems, in which the agents have voluntarily joined an organisation, and now have to negotiate *ab initio*, and repeatedly, the social arrangements for their own self-governance. It has specifically addressed both the abstract problem of balancing (ideal) social deliberation vs. (practical) social contracts, and the problem of defining an expressive rule representation and efficient and effective rule processing for these social contracts. The formalism is intended to be generic,

The specific contributions of the current work are:

- to have specified the *Megabike Scenario* and to define the social arrangements designed to address the multiple inter-dependent social problems that arise in the scenario;
- to have identified a quintet of interleaved *socio-functional* requirements, namely scalability, complexity, mutability, enforceability and versatility;
- to have discussed the contrast between social deliberation (for which both consensus and majority decision-making can be problematic [9]) and social contracts, which can be equally effective in reaching a ‘correct’ decision;
- to have defined an effective representational formalism for these social arrangements, and algorithms for efficient processing; and
- to have derived some analytic results with respect to the socio-functional requirements, and some empirical results from simulation, that demonstrate the improved performance of social contracts over social deliberation.

However, the significance of this work for self-organising multi-agent systems is to highlight that to cope with the burden of self-governance, the agents need some awareness of their own limitations. This includes, firstly, realising that the cognitive and communicative overheads imposed by deliberative decision-making under constraints imposed by the environment is having a deleterious effect on their survivability; and secondly, recognising that by substituting social deliberation with social contracts they can – ideally – produce approximately as ‘correct’,

or as ‘good’, a result. It also demands some awareness of the importance of *compromise* with respect to *values* in the negotiation phase of joining a *megabike*, in deciding the original set of social arrangements, as this agreement is the essential assurance underpinning effective equivalence of outcomes of social deliberation versus social contracts.

Additionally, this work suggests that there is not necessarily an ‘optimal’ social arrangement: there may be *mutatis mutandis* more ‘preferable’ social arrangements according to values, but the important requirements seem to be (a) being able to subordinate personal preferences for benefit of the common good, (b) being able to change ‘on demand’ existing arrangements to alternative arrangements that are ‘fit for purpose’ for prevailing environmental conditions, and (c) not getting stuck in those arrangements when conditions change.

Acknowledgments. We are grateful for the constructive comments of three anonymous reviewers, which have improved this article. We also acknowledge the contribution of the Imperial College London Department of Electrical and Electronic Engineering SOMAS 23-24 Cohort, who developed the infrastructure used to run the experiments described in Sect. 6. Thanks to Ella Bettison for assistance with that experimentation.

References

1. Artikis, A., Sergot, M., Pitt, J.: Specifying norm-governed computational societies. *ACM Trans. Comput. Logic* **10**(1), 1–42 (2009)
2. Artikis, A., Sergot, M.J., Paliouras, G.: Run-time composite event recognition. In: Proceedings of the Sixth ACM International Conf. on Distributed Event-Based Systems, DEBS, pp. 69–80. ACM (2012)
3. Boella, G., Governatori, G., Hulstijn, J., Riveret, R., Rotolo, A., van der Torre, L.: Time and defeasibility in FIPA ACL semantics. In: 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology. vol. 3, pp. 634–637. IEEE (2008)
4. Boissier, O., Bordini, R.H., Hübner, J.F., Ricci, A., Santi, A.: Multi-agent oriented programming with JaCaMo. *Sci. Comput. Program.* **78**(6), 747–761 (2013)
5. Engelmann, D.C., Panisson, A.R., Vieira, R., Hübner, J.F., Mascardi, V., Bordini, R.H.: MAIDS – a framework for the development of multi-agent intentional dialogue systems. In: Proceedings of AAMAS, pp. 1209–1217 (2023)
6. Graeber, D., Wengrow, D.: *The Dawn of Everything: A New History of Humanity*. Allen Lane, London, UK (2021)
7. Jones, A., Sergot, M.: A formal characterisation of institutionalised power. *J. IGPL* **4**(3), 427–443 (1996)
8. Klein, N.: *The Shock Doctrine: The Rise of Disaster Capitalism*. Penguin, London, UK (2007)
9. Mertzani, A., Ober, J., Pitt, J.: Θ -learning: An algorithm for the self-organisation of collective self-governance. In: IEEE International Conference on Autonomic Computing and Self-Organizing Systems (ACSOS), pp. 97–106 (2023)
10. Mertzani, A., Pitt, J.: Requisite social influence in self-regulated systems. In: 16th International Conference on Agents and Artificial Intelligence (ICAART), pp. 133–140 (2024)

11. Ober, J.: *Demopolis: Democracy before liberalism in theory and practice*. Cambridge University Press, Cambridge, UK (2017)
12. Ober, J.: *Democracy and Knowledge*. Princeton University Press, Princeton, NJ (2008)
13. Ostrom, E.: *Governing the Commons: The Evolution of Institutions for Collective Action*. Cambridge University Press, Cambridge, UK (1990)
14. Panisson, A.R., Meneguzzi, F., Vieira, R., Bordini, R.H.: An approach for argumentation-based reasoning using defeasible logic in multi-agent programming languages. In: 11th International Workshop on Argumentation in Multiagent Systems, pp. 1–15 (2014)
15. Pitt, J.: *Self-Organising Multi-Agent Systems*. World Scientific, Singapore (2021)
16. Pitt, J., Busquets, D., Macbeth, A.: Distributive justice for self-organised common-pool resource management. *ACM TAAS* **9**(3), 14:1–14:39 (2014)
17. Pitt, J., Schaumeier, J., Artikis, A.: Axiomatisation of socio-economic principles for self-organising institutions: concepts, experiments and challenges. *ACM TAAS* **7**(4), 39:1–39:39 (2012)
18. Pitt, J., Ober, J.: Democracy by design: basic democracy and the self-organisation of collective governance. In: 12th IEEE International Conference on SASO, pp. 20–29 (2018)
19. Rawls, J.: *A Theory of Justice*. Harvard University Press, Harvard, MA (1971)
20. Rescher, N.: *Distributive Justice*. Bobbs-Merrill Co., Inc., Indianapolis, IN (1966)
21. Robert, S., Robert, H., Evans, W., Honemann, D., Balch, T.: *Robert's Rules of Order*, Newly Revised, 10th edn. Perseus Publishing, New York, NY (2000)
22. Rychwalska, A., Roszczyńska-Kurasinska, M., Ziembowicz, K., Pitt, J.: Fitness for purpose in online communities: Community complexity framework for diagnosis and design of socio-technical systems. *Front. Psychol.* **12**(Article 739415), 1–13 (2021)
23. Scott, M., Dubied, M., Pitt, J.: Social motives and social contracts in cooperative survival games. In: Ajmeri, N., Morris-Martin, A., Savarimuthu, T. (eds.) *COINE XV*. LNCS, vol. 13549, pp. 148–166. Springer (2022)
24. Scott, M., Pitt, J.: Interdependent self-organizing mechanisms for cooperative survival. *Artif. Life* **29**(2), 198–234 (2023)
25. da Silva, H.H., Rocha, M., Trajano, G., Morales, A.S., Sarkadi, S., Panisson, A.R.: Distributed theory of mind in multi-agent systems. In: 16th International Conference on Agents and Artificial Intelligence (ICAART), pp. 451–460 (2024)
26. Southwood, N., Eriksson, L.: Norms and conventions. *Philos. Explor.* **14**, 195–217 (2011)
27. Suber, P.: *The Paradox of Self-Amendment: A Study of Law, Logic, Omnipotence, and Change*. Peter Lang Publishing, Oxford, UK (1990)
28. Tasoulias, J.: *Artificial Intelligence, Ethics, and a Right to a Human Decision*. Harold T. Shapiro Lecture on Ethics, Science, and Technology, Princeton University (2023)

Norm Compliance and Enforcement



An Agent-Centric Perspective on Norm Enforcement and Sanctions

Elena Yan¹(✉) Luis G. Nardin¹ Jomi F. Hübner² and Olivier Boissier¹

¹ Mines Saint-Etienne, Univ Clermont Auvergne, INP Clermont Auvergne, CNRS,
UMR 6158 LIMOS, 42023 Saint-Etienne, France

{elena.yan,luisgustavo.nardin,olivier.boissier}@emse.fr

² Department of Automation and Systems Engineering, Federal University of Santa
Catarina, Florianópolis, Brazil
jomi.hubner@ufsc.br

Abstract. In increasingly autonomous and highly distributed multi-agent systems, centralized coordination becomes impractical and raises the need for governance and enforcement mechanisms from an agent-centric perspective. In our conceptual view, sanctioning norm enforcement is part of this agent-centric perspective and they aim at promoting norm compliance while preserving agents' autonomy. The few works dealing with sanctioning norm enforcement and sanctions from the agent-centric perspective present limitations regarding the representation of sanctions and the comprehensiveness of their norm enforcement process. To address these drawbacks, we propose the NPL(s), an extension of the NPL normative programming language enriched with the representation of norms and sanctions as first-class abstractions. We also propose a BDI normative agent architecture embedding an engine for processing the NPL(s) language and a set of capabilities for approaching more comprehensively the sanctioning norm enforcement process. We apply our contributions in a case study for improving the robustness of agents' decision-making in a production automation system.

Keywords: Sanctions · Normative Programming Language · Normative Agent Architecture · Norm Enforcement

1 Introduction

The concepts used for conceiving a multi-agent system (MAS) are often inspired by human societies, e.g., abstractions of agent, environment, interaction, and organization [3]. Agents are autonomous entities that make independent decisions and act to achieve their goals [32]. In Normative MAS (NMAS), the agents' actions are regulated by norms. A set of norms comprises an explicit and formal specification of the behaviors that agents ought to comply with for the system to achieve its overall objectives. Agent's autonomy, however, enables agents to

behave nonconforming to the prescribed norms [8], which may lead the system into undesirable states.

Enforcement mechanisms are introduced to nudge agents to act conforming to the norms aiming to steer the system towards its overall objectives [15]. An organization-centric perspective on norm enforcement results in mechanisms and representation external to the agents, dedicated to the monitoring of the status of the norms and enforcement of the agents w.r.t. their (un)fulfillment. Even if interesting in terms of efficacy, such a perspective may become impractical in highly distributed dynamic and large-scale systems since such mechanisms may become a bottleneck and increase the latency of the expected reactions. On the other side of the spectrum, moving to an agent-centric perspective, i.e., including the representation and the norm enforcement mechanisms in the agents themselves, may bring the ability to decentralize and adapt norm enforcement in these different agents closer to the sources of changes and adaptation. This agent-centric perspective, however, may bring extra costs about the coordination between the different agents in case of conflicts about the enforcement to apply.

A solution to overcome the drawbacks of both perspectives and profit of their advantages is to combine them to some extend for balancing the agents' autonomy and the system's regulation. In such a hybrid perspective, with a strong emphasis on the agent-centric perspective on norm enforcement, agents will self-regulate and supervise the behavior of other agents.

In our conceptual view, sanctioning is part of this agent-centric norm enforcement approach. Sanctioning is a common enforcement mechanism that uses sanctions to encourage agents to comply with norms. Sanctions are reactions to regulated behaviors, thus not all reactions can count as sanctions [14].

In the NMAS community, there has been some proposals to incorporate normative capabilities into agent architectures [25]. In these architectures, agents should represent norms and be able to (i) recognize new norms, (ii) recognize actions affected by the norms, (iii) reason about the norms, and (iv) decide whether or not to comply with them. Assuming that a sanctioning enforcement mechanism is adopted, agents should also represent sanctions and be able to (i) identify norm compliance and violation, (ii) decide for the most appropriate sanctions depending on the situation, and (iii) apply the decided sanctions.

Although norms representation and agent's normative capabilities have been well investigated in the context of NMAS, few works have proposed a comprehensive perspective on sanctioning norm enforcement from an agent-centric perspective. To reduce this disparity, we advance in norm enforcement by (i) enriching a normative programming language with the ability to represent norms and sanctions as first-class abstractions, and (ii) embedding normative capabilities into a BDI agent architecture to enable agents to deliberate on the normative state when deciding to act and enforce norms.

This agent-centric norm enforcement approach shows great potential for complex and dynamic systems, for instance, manufacturing systems. For handling fault at operation time, a centralized and predefined control system is not always practical and may limit the manufacturing system's resilience [7]. Thus, a hybrid

approach able to represent regulations and flexible in terms of managing these regulations with a strong emphasis on a decentralized norm enforcement is then recommended.

The remainder of the paper is organized as follows. In Sect. 2, we introduce the concepts of norms and sanctions, and discuss normative agent architectures and normative programming languages from the perspective of their support to norm enforcement. We then extend a normative programming language by adding sanction as a first-class abstraction (Sect. 3) and extend a BDI agent architecture with sanctioning capabilities (Sect. 4). Next, we illustrate the use of our contributions in an industrial case study (Sect. 5). Finally, we conclude with some discussions and potential future research directions (Sect. 6).

2 Related Work

Norm is an important concept in human societies that has been investigated in a wide range of disciplines. Norms prescribe how members of a group ought to behave in a given situation [31]. They are expected standards of behavior socially shared and enforced by group members [15]. In MAS, norms are typically represented as obligations, permissions, or prohibitions; they are used to enable a degree of control over the autonomous agents' decision-making processes. These processes depend and intervene through their decisions, on the state of the norms (e.g., recognized, adopted, or complied). The transitions between these states of the norm define what is called norm's life cycle.

2.1 Norm's Life Cycle

Global Overview. There are various proposals regarding the life cycle of norms in the literature. They often share a similar structure and process but differ in specific details. Hollander and Wu [15] present the norm life cycle as an overall end-to-end evolutionary process. The norm's life cycle begins with the creation of norms by system designers at design time or by autonomous normative agents at execution time as a consequence of the agents' interactions (*creation* phase). Once the norms exist, they are spread in the society from one agent to another through active or passive transmission (*transmission* phase). Next, enforcement mechanisms (e.g., sanctioning) or norm agent compliers are used to encourage other agents to adopt the transmitted norms and to comply with them (*enforcement* phase). The norms adoption leads to their use in the agents' decision-making processes (*internalization* phase). As conditions change, norms may become invalid or lose their effectiveness, which cause them to be forgotten (*forgotten* phase).

Next, we describe the *enforcement* phase in more detail since this paper focuses on norm enforcement.

Norm Enforcement. There are three traditional approaches to norm enforcement: (i) the *institutional approach*, in which there is a single authority that enforces agents' behaviors; (ii) the *social approach*, in which a subset of agents

are equipped with enforcement mechanisms (e.g., police agents) used to enforce other agents' behaviors; and (iii) the *self-enforcement approach*, in which agents enforce the norms on themselves.

Cross-cutting these three approaches, the norm enforcement can be *regimented* or *regulated*. In the regimented norm enforcement, agents are prevented from violating the norms, possibly limiting agents' autonomy. In the regulated norm enforcement, agents can act in violation to the norms, but mechanisms, such as sanctioning, can be used to inflict consequences to their misbehavior in order to encourage their future compliance with the norms.

In the regulated norm enforcement, sanctions can be used as a means to steer the agents compliance with the norms. Sanction is a negative or a positive reaction to a violation of or a compliance with a norm. A comprehensive set of dimensions to classify sanctions and an accompanying conceptual sanctioning norm enforcement process model has been proposed in [28]. The proposed typology of sanctions enables classifying the types of sanctions that are more or less effective in reducing violation or increasing compliance. This conceptual sanctioning process model combines the institutional, the social, and the self-enforcement approaches offering opportunities for agents to reason about and decide for the most appropriate sanctions to apply in each situation.

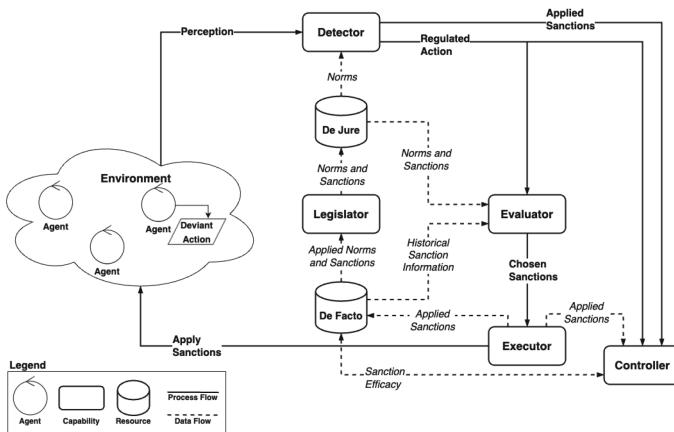


Fig. 1. Conceptual sanctioning enforcement process model (based on [28]).

This model has two normative resources (cf. Fig. 1): the *De Jure* that stores the representation of norms and sanctions and the *De Facto* that stores the historical information about sanction decisions, executions, and effectiveness. The model is enacted by five capabilities [11] (cf. Fig. 1):

- the *Detector capability* perceives the environment and detects any action regulated by the norms in the *De Jure* as well as any sanctions applied by other agents;

- the *Evaluator capability* obtains the norms and sanctions regulating the detected action from the *De Jure* to determine whether the action violates or complies with the norms. The *Evaluator capability* also combines this information with the contextual information of the agent and the historical information in the *De Facto* to decide the sanctions that should be applied, if any;
- the *Executor capability* applies the sanctions decided by the *Evaluator*;
- the *Controller capability* watches the regulated actions previously sanctioned and the sanctions applied by other agents to assess the sanctions' efficacy;
- the *Legislator capability* assesses the *De Jure* in the light of the *De Facto* information and updates the norms and sanctions in the *De Jure*.

Next, we review normative agent architectures (Sect. 2.2) and normative programming languages (Sect. 2.3) from the perspective of their support to norm enforcement.

2.2 Normative Agent Architectures

Focusing on the BDI (Belief-Desire-Intention) agent model, the literature proposes normative agent architectures using various approaches to enforce norms.

Most architectures regiment norms by preventing agents from performing actions that violate them. The BOID (Beliefs-Obligations-Intentions-Desires) [6] architecture adds the concept of *obligations* as a mental state that may conflict with the agent's intentions or desires. BOID agents use a predefined static priority function to resolve conflicts among these concepts. The priority ordering allows classifying agents in different categories, e.g., selfish (i.e., prioritize desires) or socially responsible (i.e., prioritize obligations). Although BOID does not explicitly implement a norm enforcement mechanism, social responsible agents regiment norms since they always choose actions that fulfill their obligations.

Other architectures, like NoA [20] and NBDI [30], regiment norms by filtering out actions or desires that violate norms. In the NoA architecture, agents use filtering functions to simply exclude actions that violate adopted norms. In the NBDI architecture, agents update their beliefs, desires, intentions and set of adopted norms independently. When selecting desires, NBDI agents only select those that are compliant with the adopted norms. Using yet a different approach, the Normative AgentSpeak(L) [26] architecture enables agents to change their plans at runtime to conform to new norms. The approach consists in modifying the agent's plans library, i.e., create new plans when new obligation norms are adopted and suppress plans containing actions forbidden by prohibition norms. The N-Jason [23] extends the Normative AgentSpeak(L) capabilities by scheduling intentions considering deadline and priority for runtime norm compliance.

A NoA extension [21] and the Nu-BDI [27] architectures relax the regimentation constraint allowing the agents to violate norms, but diminishing the negative impact that the violation may have. In the extended NoA architecture, agents have deliberation mechanisms to annotate actions that violate norms; these annotations can be later used by the agents to decide whether or not to

perform actions leading to the violation of norms. In the Nu-BDI architecture, agents annotate each step of the plans within the scope of a norm with constraints stemming from that norm, and rank the plans using a utility function based on these annotations. When selecting plans, Nu-BDI agents choose the plan that violates the norms less (i.e., highest ranked plan).

López y López et al. [24] implicitly incorporate regulated norm enforcement mechanisms into their agent framework. They define two levels of norms, in which the secondary norms are activated as a result of the fulfillment or violation of the primary norms. So, the secondary norms can be seen as sanctions inflicted on agents fulfilling or violating norms.

Few normative agent architectures deal explicitly with regulated norm enforcement. EMIL-A (EMergence in the Loop Architecture) [2] is a normative BDI agent architecture developed for norm emergence, innovation, and spread. EMIL-A is endowed with modules that allow (i) norm recognition (i.e., recognition and representation of norms as mental states), (ii) norm adoption (i.e., detection and dynamic updating of the norms' salience that corresponds to the perceived prominence of a norm within a relevant reference group), (iii) norm compliance (i.e., decision to comply with the adopted norms), and (iv) norm enforcement (i.e., detection of norm compliance and violation, and the application of sanctions). The n-BDI architecture [9] also incorporates a norm enforcement mechanism that always rewards or punishes the agents as a consequence to, respectively, the compliance with or violation of a norm. AORTA [18] is a normative agent module that provides normative reasoning capabilities to the agents. AORTA-powered agents are able to reason about norms, automatically detect violations as soon as they become aware of the violation state, and produce a new belief about the violation that may trigger another norm or plan.

In addition to those BDI-based normative agent architectures, Fagundes et al. [12] extend the Markov Decision Process (MDP) framework for creating self-interested agents able to reason about norms. The framework uses an enforcement mechanism based on the imperfect observation of the system state and agents, and inflicts a cost on agents if a violation is detected [13]. Agents make decisions based on the expected utility of complying with or violating a norm, which takes into account the cost of violating the norm and the probability that the norm violation will be detected.

2.3 Normative Programming Languages

Normative programming languages provide abstractions that enable agents to deliberate about their actions bounded by the normative regulations. These languages complement the normative agent architectures as they provide an explicit representation of and enable the handling of normative concepts (e.g., obligation, permission, prohibition, sanction). There has been extensive work on normative programming languages with an organization-centric perspective [10, 17]; however, there has been limited interest in languages with an agent-centric perspective.

One of the first normative programming languages with an agent-centric perspective is the NoA language [19]. The language, whose semantics are implemented by the NoA architecture, contains constructs for the specification of beliefs, goals, plans, and norms. Norm specifications can regulate states (i.e., the achievement of a particular state of the world) or actions (i.e., the performance of explicit actions). In this language, normative statements are expressed as obligations, permissions, prohibitions, and sanctions. Although sanction exists as an independent construct, it is a syntactic sugar that sets an obligation for an agent to pursue certain activities that represent such sanctions.

The Normative Programming Language (NPL) [16], albeit used mainly within an organization-centric perspective [17], is a general language dedicated to the development of normative programs based on the obligation, permission, and prohibition constructs. Likewise the NoA language, the NPL also represents sanctions as obligations.

The N-2APL [1] is an extension to the agent programming language 2APL [10]. The N-2APL provides support for representing beliefs, goals, plans, norms (i.e., obligations or prohibitions), sanctions, deadlines, and durations. A sanction is a consequence to the violation of an obligation or prohibition, i.e., specify the updates to be applied on the environment due to the violation of a norm. In N-2APL, sanction is part of the norm specification, thus tightly associated to it.

2.4 Remarks

In our conceptual view, sanctions are part of social mechanisms (i.e., agent-centric perspective) for norm enforcement as they are consequences of norms violation or compliance. Although sanctioning mechanisms are built on top of norms and their state, sanctions are not simply triggered depending on norms state, but also depending on the context in which the compliance or violation takes place. Tackling these requirements call for a normative programming language that disentangles sanctions representation from, yet letting it associated to, norms, and a normative agent architecture that triggers sanctions from a normative state contingent upon the agent's contextual situation in order to fulfill the agent-centric perspective.

In the next sections, we propose an extension to the NPL normative language [16] and to the JaCaMo BDI agent [4] to tackle these requirements. We chose to extend NPL because, to the best of our knowledge, it is the only general purpose normative programming language available and the JaCaMo BDI agent because NPL has already been integrated into it in an organization-centric perspective [17].

3 NPL(s): Extending NPL with Sanctions

Our approach consists of having an enriched and separated formal description of sanctions and their triggering conditions. Here we extend the NPL [16] language

to introduce the concept of sanctions as a first-class abstraction. This language is interpreted at execution time by the NPL interpreter based on the current state of the normative system. Therefore, we also extend the NPL interpreter to compute sanctions. The syntax and (informal) semantics of the new language, named NPL(s), is presented in the sequence.

A normative program np in NPL(s) is composed of a set of: (i) facts and inference rules (based on a syntax similar to the one used in Jason [4]), (ii) norms, (iii) sanction rules, and (iv) sanctions. While the two first components come from NPL, the last two are extensions introduced by NPL(s). Next, we briefly introduce the representation of norms in NPL. For additional details in the NPL's formal semantics, please refer to [16].

An NPL norm has the general form:

$$\text{norm } id : \varphi \rightarrow \psi .$$

where id is a unique *identifier* of the norm; φ is a formula that determines the *activation condition* for the norm; and ψ is the *consequence* of the activation of the norm. Here we focus on obligations, albeit other possibilities are permissions, prohibitions, and failures.

$\psi = \text{obligation}(a, m, g, d)$: represents the case where an obligation for some agent a is created. Argument m is the maintenance condition for the obligation; g is the formula that represents the obligation itself (a state of the world that the agent must bring about, i.e., a goal the agent has to achieve or an action the agent must perform); and d is the deadline condition to fulfill the obligation.

The semantics defines that whenever φ is true, an obligation for a is created and its initial state is **active**. If m becomes false, the obligation state changes to **inactive**. If g becomes true before d is true, the obligation state changes to **fulfilled**. If d becomes true before g is true, the obligation state changes to **unfulfilled**. The final state of an obligation may thus be **unfulfilled**, **fulfilled**, or **inactive**.

In NPL(s), a norm can be followed by sanction rules to be evaluated according to the final state of an obligation. For instance, the sanction rules for the fulfillment, unfulfilment, and inactivity of norm id has the general form (see *tsr* in Table 1):

```
norm id :  $\varphi \rightarrow \psi$ 
  if fulfilled:  $sr_1(args), sr_2(args), \dots, sr_n(args)$ 
  if unfulfilled:  $sr_{n+1}(args), sr_{n+2}(args), \dots, sr_m(args)$ 
  if inactive:  $sr_{m+1}(args), sr_{m+2}(args), \dots, sr_z(args)$  .
```

A sanction rule has the following general form (see *srule* in Table 1):

```
sanction-rule  $sr_i(args) : \rho \rightarrow \text{sanction}(a, s)$  .
```

Table 1. EBNF of the Normative Programming Language for Sanctions — NPL(s). Non-terminals *atom*, *id*, *var*, and *number* correspond, respectively, to predicates, identifiers, variables, and numbers as used in Prolog.

<i>np</i>	$::= \text{"np"} \text{ atom } \{\} (\text{ rule } \text{ norm } \text{ srule })^* \{\}$
<i>rule</i>	$::= \text{ atom } [\text{ :- } \text{ formula }] \text{ . }$
<i>norm</i>	$::= \text{ "norm"} \text{ id } \text{ ":" formula } \text{ "-> } (\text{ fail } \text{ obl } \text{ per } \text{ pro }) (\text{ tsr })^* \text{ . }$
<i>fail</i>	$::= \text{ "fail"}(\text{ atom })$
<i>obl</i>	$::= \text{ "obligation"} \text{ dargs}$
<i>per</i>	$::= \text{ "permission"} \text{ dargs}$
<i>pro</i>	$::= \text{ "prohibition"} \text{ dargs}$
<i>dargs</i>	$::= (\text{ (var } \text{ id }) \text{ , formula } \text{ , formula } \text{ , (time } \text{ formula) } \text{) } \text{) }$
<i>tsr</i>	$::= \text{ "if"} (\text{ fulfilled } \text{ unfulfilled } \text{ inactive }) \text{ ":" (atom) }^*$
<i>srule</i>	$::= \text{ "sanction-rule"} \text{ atom } \text{ ":" formula } \text{ "-> } \text{ "sanction"}(\text{ (var } \text{ id }) \text{ , atom })$
<i>formula</i>	$::= \text{ atom } \text{ "not"} \text{ formula } \text{ atom } (\text{ "&" } \text{ " " }) \text{ formula }$
<i>time</i>	$::= \text{ "(" number } (\text{ "second"} \text{ "minute"} \dots) \text{ ")" }$

where sr_i is a unique *identifier* of the sanction rule; *args* are terms passed as parameters (optional); ρ is a formula that determines the *activation condition* for the sanction (optional, assumed true if not specified); a is the *target agent* of the sanction; and s is the *sanction* content. Sanction rules can be read as “In the case that a norm has triggered the sanction rule sr_i because it is (un)fulfilled or inactive, if ρ is true, then the sanction s is created for agent a .” We opted to define a sanction rule with its own activation condition to create a sanction, rather than defining it in the *if* statements of the norm. This choice allows us (i) to define multiple sanction rules per normative event (i.e., fulfilled, unfulfilled or inactive), each triggering the sanction whose activation condition is fulfilled; (ii) to reuse and associate the same sanction rule (consequently sanction) to multiple norms.

A simple example to illustrate the language is given below. Two norms are defined, each associated with a sanction-rule. Norm **n1** specifies that **alice** is obligated to achieve **b(0)** within 3 s. Unfulfillment to do so activates sanction-rule **sr1**, resulting in the imposition of a **fine** to **alice**. If the sanction is created and the condition **extra(C)** is met, norm **n2** is activated. This delegates **bob** to apply the fine to **alice** in 2 s. In the event of unfulfillment, a sanction is issued to remove **bob** from the system.

```
// * NORMS *
// alice has 3 seconds to achieve b(0), or else evaluate sr1
norm n1: vl(X) & X > 5
-> obligation(alice,true, b(0), '3 seconds')
    if unfulfilled: sr1(alice,X) .
```

```

// bob is obliged to apply fines in 2 seconds
norm n2: sanction(A,fine(X)) & extra(C)
-> obligation(bob,true, apply_fine(A,X*C), '2 seconds')
  if unfulfilled: sr2 .

// * SANCTION RULES *
// if A is not in an emergency, create fine sanction for it
sanction-rule sr1(A,V) : not emergency(A) -> sanction(A,fine(V)) .

sanction-rule sr2 -> sanction(bob,remove_from_systems) .

```

Based on this NPL(s) example, we can have the following example story line:

1. A fact `v1(20)` is produced and norm `n1` is triggered.
2. `alice` is thus obliged to achieve `b(0)` in 3s.
3. After 3s, `alice` does not fulfill the obligation, triggering the sanction rule `sr1(alice,20)`.
4. Since `alice` has no fact like `emergency(alice)`, the sanction rule `sr1` produces the `sanction(alice,fine(20))` fact.
5. This new sanction fact together with the fact `extra(10)` triggers the norm `n2` that obliges agent `bob` to `apply_fine(alice,200)` in 2s.
6. Supposing that `bob` fulfills its obligation, the story ends here.
7. However, if `bob` does not fulfill the obligation, the sanction rule `sr2` is triggered and the `sanction(bob,remove_from_systems)` fact is produced.

Here, we assume the environment is capable of handling and applying the sanction.

From this story, we can notice that the NPL(s) interpreter is also able to compute sanction facts. In our conceptual view, we distinguish the sanction as a fact (e.g., the content *fine*) and the sanction as a procedure to follow (e.g., the action *to pay the fine*). In NPL(s) sanctions are computed as facts.

For the above example, the sanction as the procedure to follow is specified as an obligation norm for the agent that is the target of the sanction fact to pay the fine. See below.

```

// A is obliged to pay the fine in 2 seconds, or else evaluate sr_extra
norm n3: fulfilled(obligation(_,_, apply_fine(A,V), _))
-> obligation(A, true, pay_fine(V), '2 seconds')
  if unfulfilled: sr_extra(A, V) .

// create an extra fine sanction for A
sanction-rule sr_extra(A,V) -> sanction(A, fine(V + 20)) .

```

The previous story line continues from point 6:

6. Supposing that `bob` fulfills its obligation to apply the fine, norm `n3` is then triggered.
7. `alice` is thus obliged to `pay_fine(200)` in 2 s.
8. After 2 s, if `alice` does not fulfill the obligation, the sanction rule `sr_extra` will be triggered and the `sanction(alice, fine(220))` fact is produced.

The choice of considering the sanction as a fact allows us, firstly, to distinguish between an obligation norm and a sanction, and secondly, to provide more flexibility in defining sanctions. As in the example, we can create a chain of sanctions (i.e., unfulfilment to pay the fine will increase the fine) through obligation norms to implement the sanctions.

The procedural consequences of sanction facts on the running system depend on a mechanism that reads these facts and implements them. First, the NPL(s) interpreter itself can read sanction facts, as in the example of the activation of norm `n2`. However, the management of norms and sanctions could be handled along an organization-centric, environment-centric or agent-centric perspective. For instance, in an organization-centric perspective on management, sanctions could be applied directly by an external mechanism or combined with an agent-centric perspective, i.e., rely on a specialized agent who supervises other agents. It could also be managed from an environment-centric perspective that the environment is capable to implement and trigger the action corresponding to the sanction fact. An example could be fixed radar cameras placed in roads. In case a vehicle is detected exceeding the speed limit, an official fine (i.e., sanction fact) is automatically produced. Finally, in an agent-centric perspective, it could be an agent itself that reads the sanction fact and decides to run a procedure to implement the sanction. In this case, a sanctioner agent would apply the sanction to a target sanctioned agent. Then, the sanctioned agent receives the sanction and may decide to enact it, i.e., execute the enforced behavior. Although relevant, we do not discuss here about the capacity and power of the sanctioner in applying the sanction to the sanctioned agent.

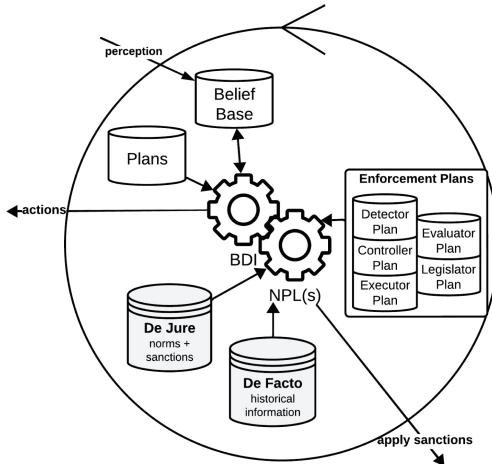
In this paper, we focus on the agent-centric perspective to manage the norm enforcement and sanctions.

4 Normative Agent Architecture

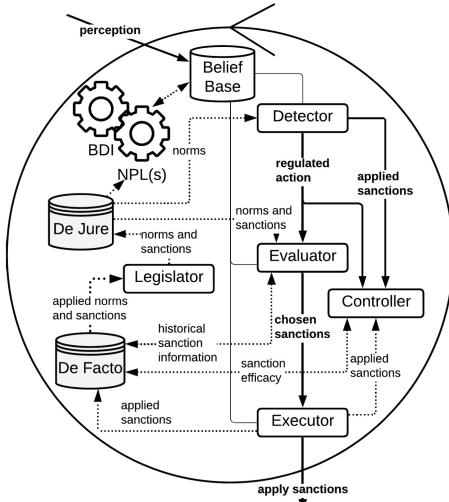
Here we describe an extension of the JaCaMo BDI agent architecture to handle normative states¹.

The normative architecture draws inspiration from the sanctioning norm enforcement process model proposed in [28] (Sect. 2.1) to define the specialized resources and enforcement capabilities needed to implement a normative agent. As illustrated in Fig. 2, we extend the JaCaMo BDI agent architecture by integrating:

¹ The source code of the normative agent architecture is available at <https://github.com/moise-lang/npl/blob/master/src/main/java/npl/NormativeAg.java>.



(a) Overview of the normative BDI agent architecture extended to handle normative states.



(b) Detail of the normative BDI agent architecture showing, in particular, the interaction among the enforcement plans.

Fig. 2. Normative BDI agent architecture (Note that this architecture depicts only norm enforcement capabilities. Other capabilities were omitted due to our interest in norm enforcement in this paper.)

1. a *NPL(s)* Engine to enable agents to interpret and compute the current state of norms and sanctions specified in the *NPL(s)* language (Sect. 2.3);
2. a *De Jure* resource, which stores norms, sanction rules, and sanctions, specified in *NPL(s)* language;

3. a *De Facto* resource, which stores the actual applied norms and sanctions and their historical information; and
4. a set of *Enforcement plans* for managing the enforcement capabilities.

The NPL(s) Engine reads the normative specification from the De Jure and based on the information in the agent's belief base, adds normative facts to the belief base. Contrary to the De Jure, the De Facto is part of the belief base of the agent that contains particular types of beliefs, i.e., beliefs related to the decision and application of sanctions and their efficacy.

The Enforcement Plans implement the norm enforcement capabilities (see Sect. 2.1). They are carried out by the BDI and/or the NPL(s) engines. While the detector capability (i.e., *Detector Plan*) is generic and performed by the NPL(s) engine, the other domain specific plans are performed by the BDI engine using the normative states updated by the NPL(s) Engine.

5 Case Study: Laboratory Plant MyJoghurt

Here we present a case study illustrating the practical potential of the extended normative programming language NPL(s) integrated into the proposed normative agent architecture. The case study is focused on improving the robustness of agents' decision-making in a production automation system. We took as reference the industrial case study of the laboratory plant *myJoghurt* introduced in [22].

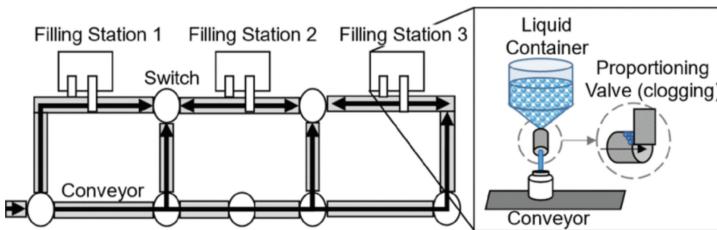


Fig. 3. Process Plant with several filling stations (left) and clogging of proportioning valve of exemplary filling station (right) [22] (Reprinted with permission from 2023 IEEE 21st International Conference on Industrial Informatics, © 2023 IEEE).

The *myJoghurt* is composed of filling stations responsible for filling recipe-specific liquids (e.g., yogurt, milk) into bottles that are transported by the logistics system. Orders are received and the filling tasks are distributed among the process plant. The filling process uses a proportioning valve that opens for a specific time window to achieve the requested fill level in the bottle (Fig. 3). Over time, residues can adhere to the valve, reducing the flow rate and causing the valve to clog. The extent and timing of this clogging are unpredictable and vary with the liquid type (e.g., yogurt clogs the valve faster than milk or water

because of its higher viscosity). The orders and filling process are controlled by a multi-agent system (illustrated in Fig. 4), whose agents control the process plant into filling the correct amount of liquid in the bottles according to the order received.

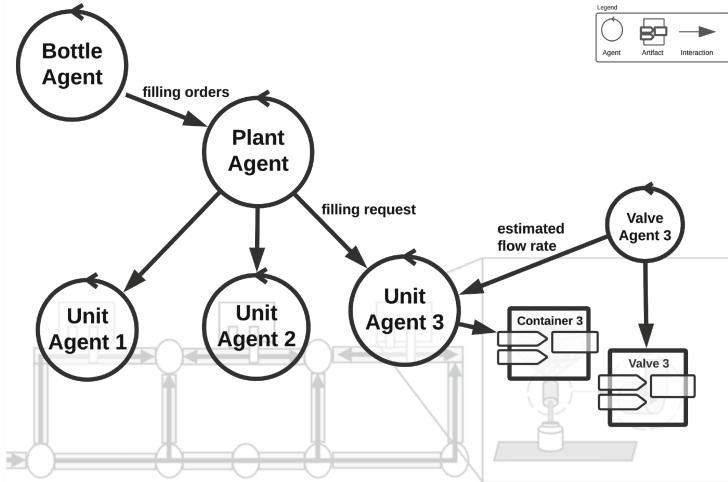


Fig. 4. *myJoghurt* multi-agent architecture (based on [22]).

The *myJoghurt* control system is composed of five types of components: bottle, plant, unit, container, and valve. They are controlled by four types of agents that interact with each other and two types of artifacts. The *bottle agent* receives and sends filling orders from customers to the *plant agent* with the type and the amount of liquid to fill the bottles. The *plant agent* negotiates with the *unit agent* when orders are received , and selects and assigns the orders to *unit agents* based on their availability and image. The *unit agent*, which controls the filling station, interacts with the filling station's container artifact and *valve agent* for the filling process. The *container artifact* manages the level of liquids in the container. The *valve agent* estimates the flow rate considering the type of liquid (e.g., a liquid with a higher viscosity leads to a faster flow) and whether the valve is clogged (e.g., residues of liquid could clog the valve over time).

Each *plant agent* has the goal of correctly filling the bottles according to the filling request. Each *unit agent* has the goal to maximize the number of filled bottles over time. Upon receiving a filling request from the *plant agent*, the *unit agent* checks the liquid type in the container and the estimated flow rate of the *valve agent* to calculate the number of bottles that it can fill . If the *unit agent* accepts the order, it coordinates with the *valve agent* that controls the opening window of the valve, through the *valve artifact*, according to the estimated flow rate to fill the bottles. Agents regulate the correct fulfillment of order requests.

The norms defined in this case study guide the agents to filling each bottle with the correct liquid and volume within a specified tolerance range.

Enforcement mechanisms and sanctions are used to steer the decisions of agents to improve the accuracy of the flow rate estimation in various types of liquid and in the presence of non-deterministic situations. The enforcement mechanisms take into account the deviation and learning factors to determine whether to sanction and which sanctions to apply. The deviation factor comprises polarity, determined by whether an action complied with (positive polarity) or violated (negative polarity) the norm, and the magnitude which measures the degree of the deviation from the expected fill level. The learning factor considers the historical behavior of the *valve agents*, using the image of the agent, the frequency of violations, and the efficacy of sanctions applied. We consider the image of the agent instead of the reputation to refer to the agent's own assessment based on direct experience with the sanctioned agent [29].

Table 2. Sanctions defined for the case study. The sanctioner agent applies the sanction to the sanctioned agent if the activation condition holds.

Id	Sanctioner	Sanctioned	Sanction	Activation condition
S1	unit agent	valve agent	Adjust the estimated flow rate	The image is below a threshold
S2	unit agent	valve agent	Activate the self-cleaning procedure	The violation occurs three consecutive times
S3	plant agent	unit agent	Adjust the unit agent's image	The image is below a threshold
S4	plant agent	unit agent	Disregard the unit agent as an option for subsequent filling orders	The violation occurs five consecutive times
S5	plant agent	unit agent	Require manual intervention to repair the filling station	The violation occurs five consecutive times

Table 2 lists the sanctions defined for the case study. The activation conditions of sanctions are evaluated based on the compliance with or violation of the norm, and sanctions are applied accordingly. In this case study, all defined sanctions have a negative polarity [28], intended to punish the *valve agent* or the *unit agent* for their misbehavior. Sanction S1 aims to guide the *valve agent* by adjusting its estimated flow rate. Sanctions S2 and S5 are used to punish the agent for their misbehavior by temporarily preventing the agent from receiving new orders, thus reducing their utility. Similarly, sanction S3 aims to reduce the

number of orders given to the agent, while sanction S4 completely excludes the agent from the subsequent filling orders.

We used JaCaMo [3] with the extended NPL(s) normative language to implement the case study².

5.1 Representation of Norms and Sanctions

Norms and sanctions are stored in the De Jure repository and represented using the NPL(s) normative language (see Sect. 3). The listing below illustrates the De Jure repository with the representation of the *unit agent*'s norms, sanction rules, and sanctions³.

```

norm n1: fill_bottle(LQ, X, MN, MX) & .my_name(U)
-> obligation(U, n1, fill(LQ,X,MN,MX), level(X,L) & (L<MN | L>MX)).

norm n2: level(V,X,L) & .my_name(U)
-> obligation(U, n2, update_factors(V,X,L),
  deviation_factor(V,X,"negative",_) &
  learning_factor(V,X,I,_,_,_) & threshold(T,_) & I<T)
if unfulfilled: s1(V,X), s2(V,X).

sanction-rule s1(V,X): learning_factor(V,X,_,_,_,C) & threshold( _, T) &
C<T
-> sanction(V, adjust_flow_rate(X)).

sanction-rule s2(V,X): learning_factor(V,X,_,_,_,C) & threshold( _, T) &
C>=T
-> sanction(V, self_cleaning(X)).
```

The working principle of the agent-centric sanctioning mechanism including the NPL(s) engine operates as follows. The NPL(s) activates the norm **n1** whenever the *unit agent* accepts a filling request `fill_bottle(LQ, X, MN, MX)` from the *plant agent*. This norm produces an obligation `fill(LQ, X, MN, MX)` for the *unit agent* to fill the current bottle *X* with the liquid type *LQ* within the filling level range *MN-MX*. Once the bottle is filled, the norm **n2** is activated and produces an obligation for the *unit agent* updating the deviation and learning factors based on the bottle filled. If the polarity is negative and the image of the *valve agent* (*I*) drops below a threshold *T*, the norm becomes `unfulfilled` and one of two sanction rules are triggered, i.e., *s1* and *s2*. If the number of consecutive violations (*C*) of the *valve agent* (*V*) is less than a threshold *T*, then the sanction rule *s1* is triggered ordering the *valve agent* (*V*) to adjust its flow rate; otherwise, the sanction rule *s2* is triggered ordering the *valve agent* (*V*) to activate the self cleaning procedure in the *valve artifact*.

Next the implementation of this sanctioning enforcement mechanism is shown in the context of the *myJoghurt* MAS architecture.

² The source code is available at <https://github.com/yan-elena/myjoghurt-case-study>.

³ The *plant agent*'s norms, sanction rules, and sanctions are available at https://github.com/yan-elena/myjoghurt-case-study/blob/main/src/npl/plant_de_jure.npl.

5.2 *myJoghurt* MAS Architecture

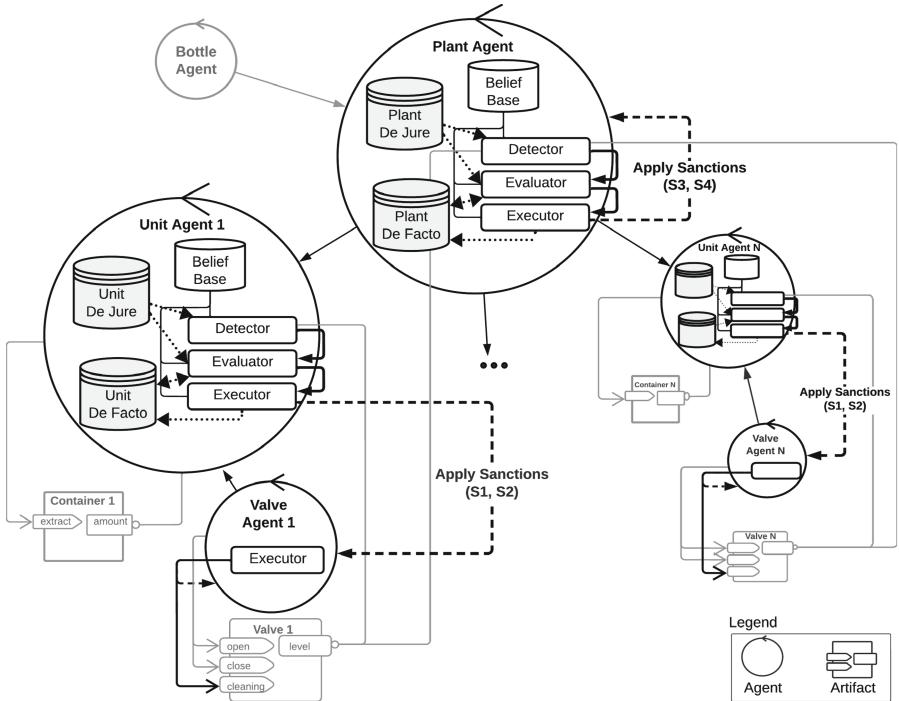


Fig. 5. Overview of integration of enforcement mechanisms into the MAS Architecture.

The normative agent architecture equipped with the NPL(s) engine proposed in Sect. 4 is used to design agents in the *myJoghurt* MAS architecture (Fig. 5). The *plant agent* and *unit agents* set the norms and sanctions in their De Jure repository. Both types of agents are endowed with the enforcement mechanisms. Because the applied agent architecture is modular, we can adapt each agent to specific requirements of the *myJoghurt* architecture. So, the *plant agent* and the *unit agents* implement the *Detector*, *Evaluator*, and *Executor* plans while the *valve agent* implements only the *Executor* plan.

The *Detector* plan of the *unit agents* and *plant agent* perceives the filling level information of each completed bottle and detects any violation or compliance with the norms.

Once a violation is detected, both the *plant agent* and *unit agents* calculate and update the factors to determine whether and which sanctions to apply using their *Evaluator* plan. The *Evaluator* considers the norms in the De Jure and the historical information in the De Facto to make these decisions. If the activation conditions of sanctions listed in Table 2 are satisfied, the *unit agent* may apply sanctions S1 and S2 to the *valve agent* responsible for the violation. S2 and S3

are applied by the *plant agent* to itself in order to indirectly update the *unit agents*' image or even disregard it of future orders. S5 is not represented in the diagram since it is a sanction to be sent to an external actor in the system (e.g., a human operator) responsible for repairing the filling station.

The Executor plan of *plant agent*, *unit agents*, and *valve agent* handle the actual enforcement of the sanctions. We assume Executor plans are obedient, i.e., they enforce the sanctions decided on the Evaluator plan. However, there may be cases in which the Executor plan may be implemented differently and it may decide whether or not to enforce the decided sanctions. The actual sanction enforcement is specified in the agent code as a plan. The following listing shows the *unit agent* internal plan for reacting to the sanction fact of sanction S1 linked to adjusting the *valve agent* flow rate estimation.

```
+sanction(V, adjust_flow_rate(X))
<- //...
    .send(V, signal, update_estimation(M)).
```

For the sake of simplicity in this paper, we have not considered the Controller and Legislator capabilities.

6 Conclusion and Future Work

In this paper, we have presented an agent-centric perspective on norm enforcement and sanctions by proposing a twofold contribution. We have extended the normative programming language NPL to NPL(s) with an explicit representation of sanctions as first-class abstraction in defining regulations (i.e., norms and sanctions). Therefore, regulation specifications gain flexibility and expressivity by (i) enabling the association of sanctions with different norms and (ii) opening the possibility for agents to locally decide based on normative states and how to enforce them.

Based on this enriched normative programming language, we proposed a normative agent architecture able to reason explicitly on norms and sanctions. The use of these new features has been illustrated in a multi-agent based control production system in the context of the industrial case study of the laboratory plant *myJoghurt*. We have shown how this normative agent-centric architecture opens a comprehensive, modular, and flexible perspective on norm enforcement and sanctions in a multi-agent system.

These advancements push towards self-regulation by enabling self-adaptation of regulations and sanctions. As future work, we plan to consider the distinctions between different types of obligations (i.e., an obligation as an action to be performed or to achieve a certain state) and how agents adapt their sanctioning strategy effectively, especially in complex social scenarios. In addition, we plan to investigate self-adaptation approaches by enabling the Legislator capability to revise and adapt norms and sanctions. For instance, the Legislator could evaluate the effectiveness of the applied sanctions by analyzing the De Facto repository and revise (i.e., modify or discard less effective) sanctions in the De

Jure repository. Although the architecture allows reasoning and evaluation of sanctions, further investigation is required when integrating with the NPL(s) Engine.

In an orthogonal direction, the agent-centric perspective on managing norms and sanctions can be carried out from different management perspectives across the multi-agent oriented dimensions, e.g., by connecting the sanctioning process directly to the environment [5] or the organization [17]. From a theoretical point of view, our proposal opens the path to explore transparency and explainability of the normative functioning of multi-agent systems profiting from the agent-centric reasoning on norms and sanctions to extend the approach proposed in [33]. Finally, we have not yet tested the case study in the real production setting of the *myJoghurt* plant to assess the effectiveness of the approach. It would be valuable to assess how the manufacturing system can be improved in terms of flexibility and runtime adaptation by using an agent-centric perspective to regulation enforcement. If successful, this approach could enhance the system's robustness and contribute to advancements in the Industry of the Future.

Acknowledgments. Partially funded by ANR-FAPESP NAIMAN project (ANR-22-CE23-0018-01), GFA through RAMP-UP II project, and CAPES, The Brazilian Agency for Higher Education, under the project PrInt CAPES-UFSC “Automation 4.0”.

References

1. Alechina, N., Dastani, M., Logan, B.: Programming norm-aware agents. In: Conitzer, V., Winikoff, M., Padgham, L., van der Hoek, W. (eds.) Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012). vol. 2, pp. 1057–1064. IFAAMAS, Valencia (2012)
2. Andrichtetto, G., Conte, R., Turrini, P., Paolucci, M.: Emergence in the loop: simulating the two way dynamics of norm innovation. In: Dagstuhl Seminar Proceedings. Schloss Dagstuhl-Leibniz-Zentrum für Informatik (2007)
3. Boissier, O., Bordini, R.H., Hubner, J., Ricci, A.: Multi-agent oriented programming: programming multi-agent systems using JaCaMo. Mit Press (2020)
4. Bordini, R.H., Hübner, J.F., Wooldridge, M.: Programming Multi-Agent Systems in AgentSpeak using Jason. John Wiley & Sons (2007)
5. de Brito, M., Hübner, J.F., Boissier, O.: Coupling the normative regulation with the constitutive state management in situated artificial institutions. Knowl. Eng. Rev. **34** (2019)
6. Broersen, J., Dastani, M., Hulstijn, J., Huang, Z., van der Torre, L.: The boid architecture: conflicts between beliefs, obligations, intentions and desires. In: Proceedings of the Fifth International Conference on Autonomous Agents, pp. 9–16. ACM, New York, NY (2001)
7. Brtis, J.S., McEvilley, M.A.: Systems engineering for resilience. Tech. Rep. MP1909495, The MITRE Corporation (2019)
8. Conte, R.: Emergent (info)institutions. Cogn. Syst. Res. **2**(2), 97–110 (2001)
9. Criado, N., Argente, E., Noriega, P., Botti, V.: Reasoning about norms under uncertainty in dynamic environments. Int. J. Approximate Reasoning **55**(9), 2049–2070 (2014)

10. Dastani, M., Tinnemeier, N.A.M., Meyer, J.J.C.: A programming language for normative multi-agent systems. In: Dignum, V. (ed.) *Handbook of Research on Multi-Agent Systems: Semantics and Dynamics of Organizational Models*, pp. 397–417. IG Global, Hershey, PA (2009)
11. de Lima, I.C.A., Nardin, L.G., Sichman, J.S.: Gavel: a sanctioning enforcement framework. In: Weyns, D., Mascardi, V., Ricci, A. (eds.) *Engineering Multi-Agent Systems*, Lecture Notes in Computer Science, vol. 11375, pp. 225–241. Springer, Cham (2019)
12. Fagundes, M.S., Billhardt, H., Ossowski, S.: Normative reasoning with an adaptive self-interested agent model based on markov decision processes. In: Kuri-Morales, A., Simari, G.R. (eds.) *Advances in Artificial Intelligence – IBERAMIA 2010*, pp. 274–283. Springer (2010)
13. Fagundes, M.S., Ossowski, S., Meneguzzi, F.: Imperfect norm enforcement in stochastic environments: an analysis of efficiency and cost tradeoffs. In: Bazzan, A.L., Pichara, K. (eds.) *Advances in Artificial Intelligence - IBERAMIA 2014*, pp. 523–535. Springer International Publishing, Cham (2014)
14. Gibbs, J.P.: Sanctions. *Soc. Probl.* **14**(2), 147–159 (1966)
15. Hollander, C.D., Wu, A.S.: The current state of normative agent-based systems. *J. Artif. Soc. Soc. Simul.* **14**(2), 6 (2011)
16. Hübner, J.F., Boissier, O., Bordini, R.H.: A normative programming language for multi-agent organisations. *Ann. Math. Artif. Intell.* **62**(1–2), 27–53 (2011)
17. Hübner, J.F., Boissier, O., Kitio, R., Ricci, A.: Instrumenting multi-agent organisations with organisational artifacts and agents: “giving the organisational power back to the agents”. *J. Auton. Agent. Multi-Agent Syst.* **20**(3), 369–400 (2010)
18. Jensen, A.S.: The AORTA Reasoning Framework – Adding Organizational Reasoning to Agents. Ph.D. thesis, Danmarks Tekniske Universitet (DTU) (2015)
19. Kollingbaum, M.J., Norman, T.J.: Supervised interaction: creating a web of trust for contracting agents in electronic environments. In: Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems: Part 1, pp. 272–279. AAMAS '02, ACM (2002)
20. Kollingbaum, M.J., Norman, T.J.: NoA - a normative agent architecture. In: Gottlob, G., Walsh, T. (eds.) *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, pp. 1465–1466 (2003)
21. Kollingbaum, M.J., Norman, T.J.: Informed deliberation during norm-governed practical reasoning. In: Boissier, O., et al. (eds.) *Coordination, Organizations, Institutions, and Norms in Multi-Agent Systems*, pp. 183–197. Springer, Berlin Heidelberg, Berlin, Heidelberg (2006)
22. Land, K., Nardin, L., Vogel-Heuser, B.: Increasing robustness of agents' decision-making in production automation using sanctioning. In: 2023 IEEE 21st International Conference on Industrial Informatics (INDIN), pp. 1–6 (2023)
23. Lee, J., Padget, J., Logan, B., Dybalova, D., Alechina, N.: N-Jason: Run-time norm compliance in AgentSpeak(L). In: Dalpiaz, F., Dix, J., van Riemsdijk, M.B. (eds.) *Engineering Multi-Agent Systems*. EMAS 2014. Lecture Notes in Computer Science, vol. 8758, pp. 367–387. Springer, Cham (2014)
24. López y López, F., Luck, M., d'Inverno, M.: A normative framework for agent-based systems. *Comput. Math. Organ. Theory* **12**, 227–250 (2006)
25. Luck, M., et al.: Normative agents. In: Ossowski, S. (ed.) *Agreement Technologies*, pp. 209–220. Springer (2013)
26. Meneguzzi, F., Luck, M.: Norm-based behaviour modification in BDI agents. In: Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS. vol. 1, pp. 177–184 (2009)

27. Meneguzzi, F., Vasconcelos, W., Oren, N., Luck, M.: Nu-BDI: norm-aware BDI agents. In: Proceedings of the 10th European Workshop on Multi-Agent Systems, Dublin, Ireland (2012)
28. Nardin, L.G., Balke-Visser, T., Ajmeri, N., Kalia, A.K., Sichman, J.S., Singh, M.P.: Classifying sanctions and designing a conceptual sanctioning process model for socio-technical systems. *Knowl. Eng. Rev.* **31**(2), 142–166 (2016)
29. Sabater-Mir, J., Paolucci, M., Conte, R.: REPage: reputation and image among limited autonomous partners. *J. Artif. Soc. Soc. Simul.* **9**, 1–18 (2006)
30. dos Santos Neto, B.F., da Silva, V.T., de Lucena, C.J.: Developing goal-oriented normative agents: The NBDI architecture. In: Agents and Artificial Intelligence: Third International Conference, ICAART 2011, Rome, Italy, January, 28–30, 2011. Revised Selected Papers 3, pp. 176–191. Springer (2013)
31. Ullmann-Margalit, E.: The emergence of norms. Oxford University Press (1977)
32. Wooldridge, M.: An Introduction to MultiAgent Systems. Wiley Publishing, 2nd edn. (2009)
33. Yan, E., Burattini, S., Hübner, J.F., Ricci, A.: Towards a multi-level explainability framework for engineering and understanding BDI agent systems. In: Falcone, R., Castelfranchi, C., Sapienza, A., Cantucci, F. (eds.) Proceedings of the 24th Workshop “From Objects to Agents” (WOA 2023). CEUR Workshop Proceedings, vol. 3579. CEUR-WS.org, Rome (2023)



Knowledge Level Support for Programming Agents to Interact in Regulated Online Forums

Nesrine Hafiene^(✉), Luis G. Nardin, and Olivier Boissier

Mines Saint-Etienne, Univ Clermont Auvergne, INP Clermont Auvergne, CNRS,
UMR 6158 LIMOS, Saint-Etienne 42023, France
`{nesrine.hafiene,luisgustavo.nardin,Olivier.Boissier}@emse.fr`

Abstract. The Web is populated by autonomous agents that act on behalf of users to perform tasks on the Web. These agents are required to interact with online systems to achieve their goals. However, they are usually designed to interact with specific online systems, preventing them from interacting with those whose structure, interaction methods, and regulations are *a priori* unknown. We propose a means to overcome this limitation by representing the online systems' resources and regulations that agents can use to make decisions and act. Our approach can be split into two parts. First, we introduce a knowledge layer structured along with the Multi-Agent Oriented Programming approach to describe Web resources and their regulations using ontology. Second, we introduce an agent that uses this knowledge layer for reasoning and acting in compliance with the regulations towards achieving its goals. We use the online forum as a running example to highlight the applicability and usefulness of our approach.

Keywords: Multi-agent systems · Knowledge representation · Regulation · Online forum · Hypermedia

1 Introduction

The World Wide Web (or simply the Web) is recognized as a universal and world wide platform of resources that are accessible to humans. Nowadays, the Web has also become the playground of an increasing number of autonomous agents [7] (e.g., crawlers [3], curators [13], recommenders [12], and moderators [11]). These agents, however, usually have limited to no ability to adapt to correctly interacting with other systems rather than the ones they have been designed to. This is due to, at least in part, the agents lack of knowledge about these other systems' interaction mechanisms and regulations. For example, an autonomous agent designed to operate (e.g., post or read messages) in the Reddit online forum is usually unable to also operate in the Quora online forum without changes.

To address this challenge we have to ensure that agents can effectively interact with online systems (i.e., Web resources) without having prior knowledge

of the latter's structure, interaction methods, and regulations. We claim that a knowledge layer representing the online systems' Web resources can support overcoming this challenge by enabling agents (i) to discover the systems' functionalities, (ii) to understand how to interoperate with them, (iii) to discover their regulations, and (iv) to reason about and behave according to the regulations. Additionally, agents have to be endowed with capabilities to discover, reason about, and use the Web resources' representations to be capable of performing tasks in *a priori* unknown online systems.

In this paper we tackle this challenge in two parts. First, we introduce a knowledge layer structured along with the Multi-Agent Oriented Programming (MAOP) approach [1] to describe Web resources and their regulations using ontology. Second, we illustrate the use of this knowledge layer by an agent that can reason and act in compliance with the Web resources' regulations towards achieving its goals. We show the applicability and usefulness of our approach with the practical use case of agents interacting with the Reddit online forum supported by the JaCaMo framework¹. In this use case, we show that agents can dynamically discover and use knowledge about the Reddit online forum's interaction methods and regulations to correctly interact with it.

The remaining of the paper is structured as follows. First, we provide an overview of the tools and ontology used to demonstrate our approach (Sect. 2). Then, we introduce the two parts of our approach in the context of an online forum: the knowledge layer representation (Sect. 3) and the use of the knowledge layer by an agent (Sect. 4). Finally, we give some concluding remarks (Sect. 5).

2 Background

The Web is a comprehensive media through which agents can access the external world. Ciortea et al. [7] proposes a deeper integration of agents into the Web's hypermedia structure for transforming the Web into a suitable place for agents to act and interact in tight cooperation with human agents. The Hypermedia Multi-Agent Systems (Hypermedia MAS) [6, 16] approach has been proposed to support the creation of dynamic and open MAS and to enable seamless interaction with resources on the Web. In Hypermedia MAS, agents perceive and act on resources represented and deployed in the hypermedia environment, such as websites, APIs, and semantic resources. For agents to discover, access, and use hypermedia resources across the Web [5], these resources must be represented in a way that agents are capable of handling and reasoning about.

Semantic Web technologies provide a natural and explicit accessible means to describe Web resources and their relationships. The use of these technologies enables agents to automate real-world tasks as long as they are able to handle and reason about these representations. Ontologies, in particular, provide the required features to represent Web resources. They may act as shared vocabularies with well-defined concepts and relationships enabling agents to understand information, reason about their environment, and interact with other agents.

¹ <https://jacamo-lang.github.io/>.

By using ontologies, agents can make informed decisions and take appropriate actions on the Web [9]. Additionally, ontologies enable agents to communicate and exchange information effectively, facilitating collaboration and knowledge sharing [4].

The structure of the knowledge of Web resources plays a central role in enabling agents to automate their tasks. In the context of MAS, the MAOP approach offers four dimensions that are useful for structuring the knowledge of Web resources: Agent [2], Environment [14], Interaction [17], and Organization [10].

The knowledge related to the Agent dimension consists of the description of the agents' capabilities to perceive and act on the environment. The knowledge related to the Environment dimension denotes the necessary information for agents to access shared, dynamic, and evolving spaces where (1) they are situated and (2) where they can interact with other agents and artifacts. Artifacts are shared non-autonomous resources that can be observed and acted on by agents. The knowledge related to the Interaction dimension provides the necessary descriptions for agents to interact with other agents and artifacts [15]. Finally, the knowledge related to the Organization dimension defines the necessary information for representing coordination strategies and regulations in a MAS. The Organization dimension consists of social structures (roles played by agents in groups), coordinated activities (missions and collective goals undertaken by agents) and norms (obligations, permissions, and prohibitions aimed at agents while playing roles in groups with respect to missions and goals) governing the behavior of agents within the system.

The representation of resources and regulations in the hypermedia environment can be accomplished using different ontologies. WoT Thing Description (TD)² is an ontology for describing the capabilities of IoT devices and services. These representations enable agents to discover services and learn how to interact with them. The Hypermedia MAS Ontology (hMAS ontology)³ is an ontology for describing MAS' structure and regulations. We chose these ontologies because the WoT TD ontology is a W3C standard and the hMAS is a comprehensive and modular ontology that allows the representation of MAS developed using the MAOP approach.

3 Knowledge Level Support for Agent Programming

In this section, we provide an overview of the various components of our approach. First, we present a motivating scenario (Sect. 3.1) that will be used as a running example to introduce our approach. We then present the knowledge layer that describes the resources and regulations of an online system (Sect. 3.2). Next, we present what agents can use to decide how to act to achieve their goals (Sect. 3.3). Finally, we describe what agents can use to act in the environment (Sect. 3.4).

² <https://www.w3.org/TR/wot-thing-description11/>.

³ <https://ci.mines-stetienne.fr/hmas>.

3.1 Online Forum Motivating Scenario

We have chosen an online forum to illustrate our proposal (Fig. 1). In this online forum, agents can submit and read posts. Playing the **contributor** role, agents have the permission to submit posts (**Post**); while playing the **subscriber** role, agents are obliged to retrieve⁴ (**Read**) posts.

Alice and *Maria* are two users in the **Online Forum**, each with a personal autonomous agent acting on their behalf. *Alice*'s agent has adopted a **subscriber** role in the context of the **Forum** thread. By playing this role, *Alice*'s agent has the obligation to achieve the 'Message Read' goal. *Maria* is a new user with the intention of posting a message in the **Forum**, i.e., achieving the 'Message Posted' goal. Therefore, *Maria*'s agent has to choose a suitable role under the regulation of the **Online Forum** that allows her to achieve her goal. Her agent reasons using the knowledge layer describing the online forum resources and regulations, and decides to adopt the **contributor** role in the **Forum** thread to achieve the 'Message Posted' goal.

In the next sections, we use this motivating scenario to introduce our approach.

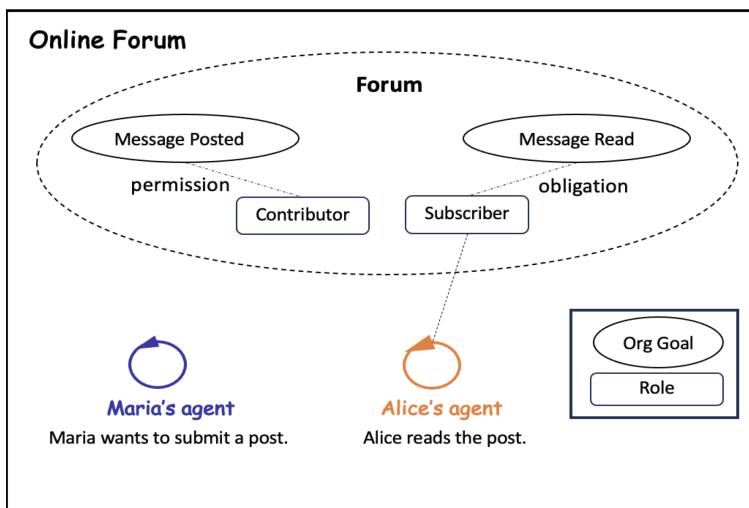


Fig. 1. Online forum motivating scenario

3.2 Knowledge Level Support for Discovering

Description of Action Possibilities. To describe the action possibilities in the environment that the agents can discover, we represent the hypermedia

⁴ Agents playing the **subscriber** role are obliged to retrieve posts because they are personal assistants that should read all posts in a thread to provide their users with summaries and key insights about everything posted.

environment using the hMAS ontology. Action possibilities are represented as *signifiers* in the hMAS ontology. Signifiers describe behaviors that agents can perform to exploit an affordance. Signifier is a perceivable sign or cue that can be interpreted meaningfully by an agent to reveal information about a behavior possibility. An affordance is a behavior possibility that is a relationship between an ability of an agent and a situation that includes agents and features of the environment⁵. Signifiers are not simple informational resources, but rather constructs available in hypermedia environments that enable situated agents to discover and interpret affordances [15].

In the context of the motivating scenario, we use two signifiers: `ex:PostSubmission` and `ex:PostRetrieval`. Each signifier describes a behavior that agents can perform to act in the environment (see Fig. 2). More specifically, the `ex:PostSubmission` signifier `hmas:signifies` the specification of the behavior `ex:PostSubmissionAction`. This action is specified through a *form* defined by the property `td:hasForm` that describes an *HTTP POST* request and an *input* defined by the property `td:hasInputSchema`. We have a similar description for `ex:PostRetrieval` signifier.

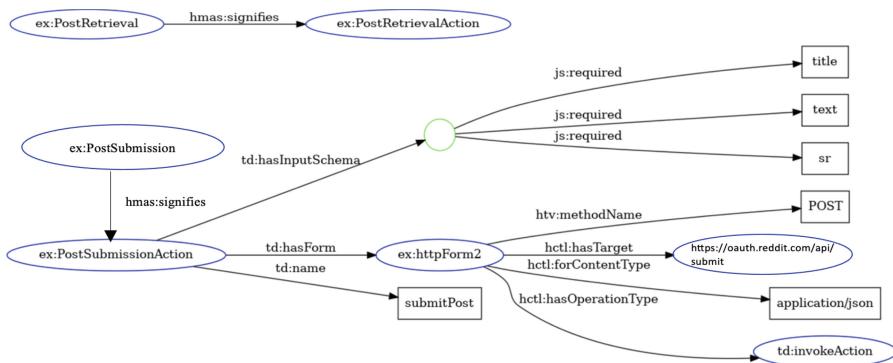


Fig. 2. Representation of signifiers and affordances using the hMAS ontology

Description of Regulations. Going a step further in the building of the knowledge layer to make accessible to the agents a description of the online forum's structure, we represent its regulations using the organization abstractions in the hMAS ontology. The organization of the Reddit online forum, depicted in Fig. 3, revolves around key entities that create a structured environment for regulating agent interactions. Within this organization, agents can discover organizational goals (e.g., `submitted_RedditPost`), missions (e.g., `submit_RedditPost`), roles (e.g., `reddit_contributor`), and groups (e.g., `reddit_group`). Organizational goals are the specific objectives that agents are expected to accomplish within

⁵ Please see more about these definitions at <https://ci.mines-stetienne.fr/hmas/core>.

the organization. Missions define the set of organizational goals that an agent committed to the mission has to accomplish. Roles define positions of agents in the organization. Groups define a set of links, roles and compatibility relations used to define a shared context for agents playing roles in it. Membership (e.g., `reddit_contributor_Membership`) indicates the role played by an agent in a group of an organization. An Activity Scheme (e.g., `submitting_RedditPost`) is a plan for an activity intended to achieve the `submitted_RedditPost` organizational goal.

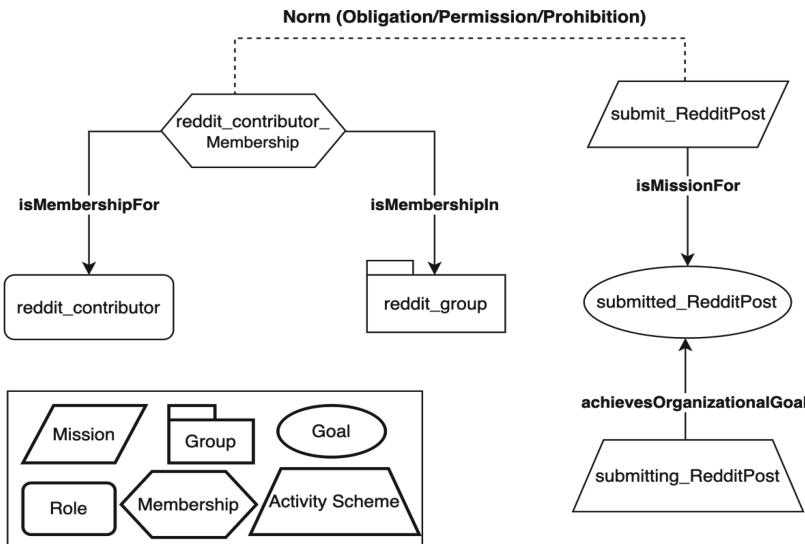


Fig. 3. Structure of the Online Forum organization

At the core is the concept of a `reddit_group`, representing a subreddit which is a specific category or topic within the Reddit online forum. A subreddit is a dedicated section within Reddit where users can share and discuss content related to a particular theme, interest, or subject. Within this online forum, participants can adopt specific roles such as `subscriber` and `contributor`. `Subscribers` follow discussions in a thread to retrieve posts, while `contributors` actively contribute to discussions by submitting new posts to the subreddit.

3.3 Knowledge Level Support for Agent Reasoning

Suppose an agent has the goal of submitting a post in an Online Forum, but it does not know yet how to achieve this goal.

Reasoning on the description of the missions and their organizational goals, and on the relation between missions and roles defined in the norms, the agent

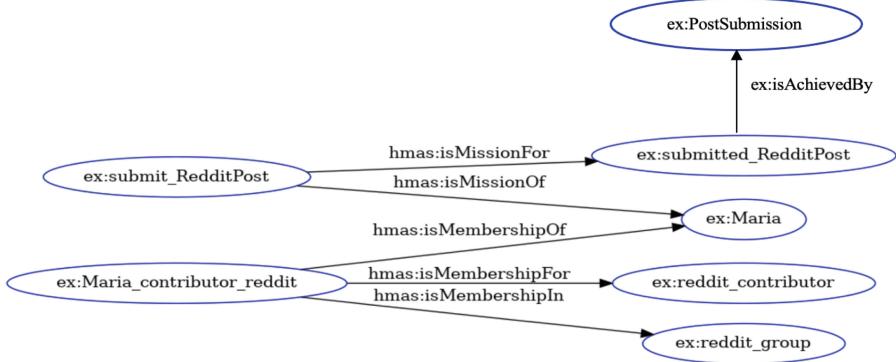


Fig. 4. Result of a role adoption process and the commitment to a mission

is able to select which role to adopt to achieve its goal. Thus, the agent starts by retrieving the organizational goal that matches with its own goal. Then, it uses missions associated organizational goals to select a role taking into account norms. Next, it adopts this role that obligates the agent to commit to the mission for achieving the goal (i.e., the organizational goal and its own).

Figure 4 shows the new state of the organization after the agent (e.g., *Maria*'s agent) adopts the `reddit_contributor` role in the `reddit_group` group of the Reddit organization. Once the agent adopts the `reddit_contributor` role, it becomes obliged to commit to the `submit_RedditPost` mission and to achieve the corresponding `submitted_RedditPost` organizational goal. The agent being obedient commits to this mission and to achieve the organizational goal.

3.4 Knowledge Level Support for Acting

In the previous section, we describe how agents process information to decide how to act to achieve their goals. In this section, we extend this by providing the framework and information for agents to act in the environment. For instance, the online forum TD in Listing 1 is used to facilitate interactions via agent actions.

The `retrievePost` action affordance enables users to retrieve posts. It offers a GET method, targeting a retrieve endpoint, and expects data in `application/json` format. This affordance supports reading operations.

The `submitPost` action affordance allows users to submit new posts. It uses a POST method, targeting the submit endpoint, and requires input data in `application/json` format. Users must provide certain required inputs that vary from one online forum to another. For instance, in the following example, `title`, `text`, and `sr` information are necessary when invoking this action. In the given context, `sr` refers to the subreddit.

This online forum TD instance serves as an abstract representation within the Reddit online forum environment, enabling interactions through the `retrievePost` and `submitPost` affordances.

```
<#thing> a td:Thing ;
  td:title "Online Forum Thing" ;
  td:hasSecurityConfiguration [ a wotsec:NoSecurityScheme ] ;
  td:hasPropertyAffordance [
    a td:PropertyAffordance , js:ObjectSchema ;
    td:name "retrievePost" ;
    td:title "retrieve post" ;
    td:hasForm [
      htv:methodName "GET" ;
      hct1:hasTarget <retrieve> ;
      hct1:forContentType "application/json" ;
      hct1:hasOperationType td:readProperty ;
    ];
  ];
  td:hasActionAffordance [ a td:ActionAffordance ;
    td:name "submitPost";
    td:title "submit post";
    td:hasForm [
      htv:methodName "POST";
      hct1:hasTarget <submit>;
      hct1:forContentType "application/json" ;
      hct1:hasOperationType td:invokeAction
    ];
    td:hasInputSchema [ a js:ObjectSchema ;
      js:properties [ a js:StringSchema;
        js:propertyName "title"
      ];
      js:properties [ a js:StringSchema;
        js:propertyName "text"
      ];
      js:properties [ a js:StringSchema;
        js:propertyName "sr"
      ];
      js:properties [ a js:StringSchema;
        js:propertyName "kind"
      ];
      js:properties [ a js:StringSchema;
        js:propertyName "url"
      ];
      js:required "title", "text", "sr"
    ];
  ];

```

Listing 1. Online forum Thing Description

4 Knowledge Level Support in Action

In the previous section we presented the knowledge level support that agents can use to act in the hypermedia environment. In this section, we illustrate how the knowledge level support can be used to regulate agents' interactions within our illustrative Reddit online forum use case. Reddit provides a dynamic and collaborative environment where users, or agents acting on their behalf, can engage in discussions, share content, and contribute to specific topics (i.e., **subreddits**).

We have used JaCaMo [1] and Hypermedea [5] to implement this use case. Hypermedea brings to the JaCaMo platform a set of dedicated artifacts to handle the access to Web resources and knowledge layer. The Hypermedea used artifacts are the *Linked Data* and the *Thing* artifacts. The Linked Data artifact is used to crawl Linked Data from the Web, to store it in a local database, and to query it. It is responsible for retrieving RDF representations of resources. The *Thing* artifact is used to create a proxy for a WoT device systems, which allows agents to interact with devices and systems in a standardized way. For JaCaMo, the organizational artifacts enable organization and environment integration.

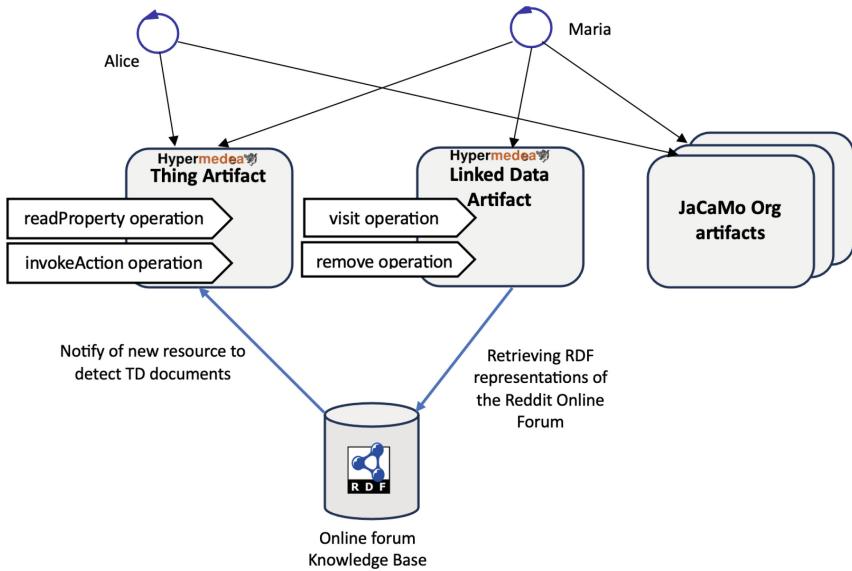


Fig. 5. Reddit use case Multi-Agent Architecture

Figure 5 illustrates the architecture of our multi-agent architecture with the artifacts used to access the knowledge layer and Web resources (i.e., **Thing Artifact** and **Linked Data Artifact**), and the artifacts used to monitor and regulate the activities of agents under the governance of normative organizations (i.e., **JaCaMo Org artifacts**). In this use case, we focus on two agents: *Alice* and *Maria*.

Both agents are playing roles in the **reddit_group** group within the Reddit organization's structure that defines two distinct roles: the **reddit_subscriber** and the **reddit_contributor**. Each role comes with specific obligations and permissions expressed by norms. Agents playing the **reddit_subscriber** role are obliged to read posts, while those playing the **reddit_contributor** role are permitted to submit posts to Reddit. This structure creates agents that can violate norms (read posts obligation) and trigger alarms within the JaCaMo organization if obligations or prohibitions are not fulfilled. In our implementation, *Alice*'s agent is a dedicated hard-coded obedient agent that we have developed to retrieve posts from the Reddit online forum (see details in [8]). This agent plays the **reddit_subscriber** role. Let's suppose that, in this organization, *Maria*'s agent has the goal to submit a post to the Reddit online forum, **submitted_redditPost**. To achieve that goal, the agent needs to identify the most suitable mission. *Maria*'s agent initiates the decision-making process by retrieving the goal from her beliefs. She crawls the organization ontology to extract the relevant missions to that goal. Based on the norms and the extracted mission, she infers that she has to adopt the **reddit_contributor** role within the **reddit_group**. When done, the **reddit_contributor_Membership** is created as a new state of the organization. Thus, *Maria*'s agent is allowed to commit to the **submit_RedditPost** mission. She commits to that mission and follows the plan for **submitting_RedditPost** activity scheme to achieve the **submitted_RedditPost** goal. The reasoning steps that *Maria*'s agent follows to act on the Reddit online forum are illustrated in Fig. 6.

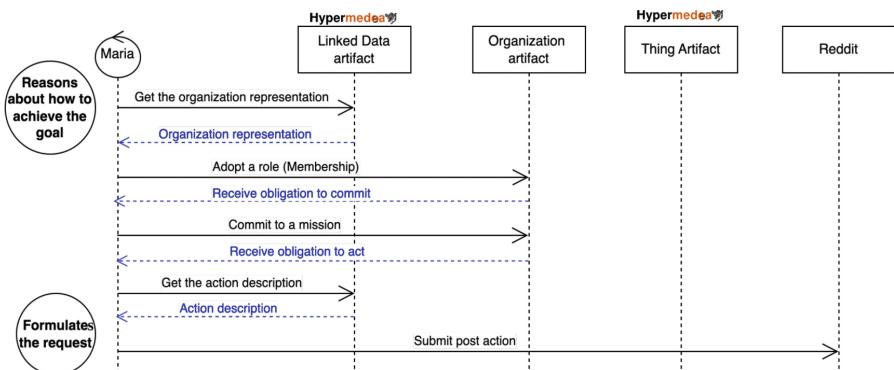


Fig. 6. Maria's agent achieves the goal of submitting a post

1. *Maria*'s agent starts the reasoning process by retrieving the `submitted_redditPost` goal from the organization set of organizational goals.
2. Using the knowledge graph describing the Reddit organization, it retrieves the missions of the organization including the organizational goal.
3. Then, considering the norms of the organization in relation to the retrieved missions, it retrieves the set of roles that are involved in those norms and make decisions about the appropriate roles to adopt.
4. Once a role is adopted, the agent gets the obligation to commit to the mission in relation to the norm activated by the adoption of the role and achieve her goal.
5. *Maria*'s agent then extracts all the signifiers related to the `submitted_redditPost` goal and decides about the actions to take within the Reddit online forum. *Maria*'s agent submits successfully a post in Reddit.

The agent's successful submission of posts to Reddit demonstrates its ability to use a knowledge layer for discovering, reasoning, and acting on any online forum.

In Algorithm 1, we present the reasoning process that outlines the various steps that *Maria*'s agent follows to reason about the knowledge layer and make decisions, such as submitting a post to Reddit.

Algorithm 1. Reasoning process

```

1: Forms ← FindAll(Form, rdf(Action, "https://www.w3.org/2019/wot/td#hasForm",
  Form))
2: Names ← FindAll(Name, rdf(Action, "https://www.w3.org/2019/wot/td#name",
  Name))
3: for each Na ∈ Names do
4:   for each Fr ∈ Forms do
5:     Operations ← FindAll(Operation, rdf(Fr, "https://www.w3.org/2019/wot/
    hypermedia#hasOperationType", Operation))
6:     AA ← []
7:     for each U ∈ Operations do
8:       AA ← Append(AA, ConstructOperationWithParameter(U, Na, Parameters))
9:     end for
10:   end for
11: end for
12: return AA

```

Maria's agent initiates the reasoning process by finding all actions associated with a relationship defined by the property `td:hasForm` from the W3C TD ontology, that describes how an operation can be performed. The algorithm systematically identifies these actions and their corresponding names. For each

action and its related form, the agent explores further to identify the semantic intention of performing the operations described by the form linked through the property `hypermedia:hasOperationType` in compliance with the W3C WoT ontology. Thus, the agent has the name of the action which is `Na=submitPost` and the name of the operation which is `U=invokeAction`. Using the retrieved information, the agent constructs the action *ConstructOperationWithParameter* with the type of the action to take *U*, name of the action *Na*, and a list of required parameters: the title of the post, the text, the subreddit, and the type of the post (i.e., link, self, image, video, videogif). Thus, the action affordance **AA** is constructed based on the extracted features, enabling *Maria*'s agent to reason about the available knowledge and formulate decisions based on the identified actions and the associated operations.

5 Conclusions

In this paper, we illustrate how a knowledge layer can be used to discover and guide agent interactions in online forums on the Web without requiring their prior knowledge of the online forum's structure. Based on a MAOP framework, the use of the Hypermedia MAS Ontology (hMAS) and on the use of the Thing Description (TD) to act on the Reddit online forum, we show the feasibility of accessing the Reddit online forum resources and making agents able to discover and get a structured view of the online forum to submit and retrieve posts on this platform. This includes understanding the relationships between different entities like agents, roles, groups, missions and signifiers. Thus, we assume our approach effective in guiding agents to discover, reason and act on online forums.

As future work, we will investigate how to generalize this approach, making it easy for agents to achieve their goals without changing their core code. We will also investigate how this approach ensures flexibility and maintainability since agents can interact with various online forums seamlessly, using a shared knowledge to understand their structure and regulations. By sharing this knowledge, agents can work together across multiple forums efficiently without needing to be adapted for each one.

Acknowledgment. This work was partially funded by the ANR-SNF HyperAgents project (ANR-19-CE23-0030).

References

1. Boissier, O., Bordini, R.H., Hübner, J.F., Ricci, A.: Multi-Agent Oriented Programming: Programming Multi-Agent Systems Using JaCaMo. MIT Press (2020)
2. Bordini, R.H., Hübner, J.F., Wooldridge, M.: Programming Multi-Agent Systems in AgentSpeak Using Jason. Wiley (2007)
3. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.* **30**(1), 107–117 (1998). [https://doi.org/10.1016/S0169-7552\(98\)00110-X](https://doi.org/10.1016/S0169-7552(98)00110-X)

4. Canito, A., Santos, G., Corchado, J.M., Marreiros, G., Vale, Z.: Semantic web services for multi-agent systems interoperability. In: Moura Oliveira, P., Novais, P., Reis, L.P. (eds.) EPIA 2019. LNCS (LNAI), vol. 11805, pp. 606–616. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-30244-3_50
5. Charpenay, V., Zimmermann, A., Lefrançois, M., Boissier, O.: Hypermedea: a framework for web (of things) agents. In: Companion Proceedings of the Web Conference 2022, pp. 176–179. Association for Computing Machinery, New York, NY (2022). <https://doi.org/10.1145/3487553.3524243>
6. Ciortea, A., Boissier, O., Ricci, A.: Engineering world-wide multi-agent systems with hypermedia. In: Weyns, D., Mascardi, V., Ricci, A. (eds.) Engineering Multi-Agent Systems. EMAS 2018, LNCS, vol. 11375, pp. 285–301. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-25693-7_15
7. Ciortea, A., Mayer, S., Gandon, F., Boissier, O., Ricci, A., Zimmermann, A.: A decade in hindsight: the missing bridge between multi-agent systems and the world wide web. In: Proceedings of the International Conference on Autonomous Agents and Multiagent Systems, pp. 1659–1663 (2019)
8. Hafiene, N., Nardin, L.G., Zimmermann, A., Boissier, O.: Governing online forum interactions with multi-agent system: a Reddit use case with the JaCaMo platform. In: Mathieu, P., Dignum, F., Novais, P., De la Prieta, F. (eds.) Advances in Practical Applications of Agents, Multi-Agent Systems, and Cognitive Mimetics. The PAAMS Collection, LNCS, vol. 13955, pp. 113–124. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-37616-0_10
9. Hendler, J.: Agents and the semantic web. *IEEE Intell. Syst.* **16**(2), 30–37 (2001)
10. Hubner, J.F., Sichman, J.S., Boissier, O.: Developing organised multiagent systems using the MOISE+ model: programming issues at the system and agent levels. *Int. J. Agent-Oriented Softw. Eng.* **1**(3–4), 370–395 (2007)
11. Kolla, M., Salunkhe, S., Chandrasekharan, E., Saha, K.: LLM-Mod: can large language models assist content moderation? In: Extended Abstracts of the 2024 CHI Conference on Human Factors in Computing Systems. Association for Computing Machinery, New York, NY (2024). <https://doi.org/10.1145/3613905.3650828>
12. Montaner, M., López, B., de la Rosa, J.L.: A taxonomy of recommender agents on the internet. *Artif. Intell. Rev.* **19**, 285–330 (2003). <https://doi.org/10.1023/A:1022850703159>
13. Niederer, S., van Dijck, J.: Wisdom of the crowd or technicity of content? Wikipedia as a sociotechnical system. *New Media Soc.* **12**(8), 1368–1387 (2010). <https://doi.org/10.1177/1461444810365297>
14. Ricci, A., Piunti, M., Viroli, M., Omicini, A.: Environment programming in CArAgO. In: Multi-Agent Programming: Languages, Tools and Applications, pp. 259–288 (2009)
15. Vachtsevanou, D., Ciortea, A., Mayer, S., Lemée, J.: Signifiers as a first-class abstraction in hypermedia multi-agent systems. In: Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems, pp. 1200–1208. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC (2023)
16. Vachtsevanou, D., Junker, P., Ciortea, A., Mizutani, I., Mayer, S.: Long-lived agents on the web: continuous acquisition of behaviors in hypermedia environments. In: Companion Proceedings of the Web Conference 2020, pp. 185–189 (2020)
17. Zatelli, M.R., Hübner, J.F.: The interaction as an integration component for the JaCaMo platform. In: Dalpiaz, F., Dix, J., van Riemsdijk, M.B. (eds.) Engineering Multi-Agent Systems, pp. 431–450. Springer, Cham (2014)

Norms and Large Language Models



Extracting Norms from Contracts Via ChatGPT Opportunities and Challenges

Amanul Haque^(✉) and Munindar P. Singh

North Carolina State University, Raleigh, USA
`{ahaque2,mpsingh}@ncsu.edu`

Abstract. We investigate the effectiveness of ChatGPT in extracting norms from contracts. Norms provide a natural way to engineer multiagent systems by capturing how to govern the interactions between two or more autonomous parties. We extract norms of commitment, prohibition, authorization, and power, along with associated norm elements (the parties involved, antecedents, and consequents) from contracts. Our investigation reveals ChatGPT’s effectiveness and limitations in norm extraction from contracts. ChatGPT demonstrates promising performance in norm extraction without requiring training or fine-tuning, thus obviating the need for annotated data, which is not generally available in this domain. However, we found some limitations of ChatGPT in extracting these norms that lead to incorrect norm extractions. The limitations include oversight of crucial details, hallucination, incorrect parsing of conjunctions, and inferring incorrect norm types and elements. Enhanced norm extraction from contracts can foster the development of more transparent and trustworthy formal agent interaction specifications, thereby contributing to improving multiagent systems.

1 Introduction

Normative relationships—norms, for short—typically refer to a social construct that governs the behavior of individuals in a given social context. The behaviors could range from etiquette and social expectations to moral and ethical standards, depending on the context. These norms help capture the essential aspects of interactions between individuals independent of low-level implementation details. Recent approaches have shown how we might engineer a multiagent system by specifying norms between member agents [10].

Contracts are a rich source of norms. Although contracts are expressed in natural language, it is well recognized that they can be understood as capturing the norms between the contracting parties [15]. Norms in contracts define the terms of agreement between the contracting parties and their expected behaviors. This paper shows how we might begin from existing annotated contracts to identify relevant norms. Even if we do not fully formalize the norms in this way, progress made in identifying the norms can help in producing valid norm specifications for further implementation.

We use ChatGPT, a pretrained Large Language Model (LLM), to extract norms from contracts. LLMs are sophisticated AI systems capable of understanding language and have pushed the state-of-the-art on many standard language benchmark tasks [14, 32, 35, 52]. We use a model of norms based on Custard [8] and its variants, e.g., [43], and extract norms such as commitment, prohibition, authorization, and power. We conduct a qualitative analysis of the extracted norms manually. We use contract clauses extracted from CUAD (Contract Understanding Atticus Dataset) [19] for our analysis. We focus on contract clauses related to the termination of a contract, such as Termination for Convenience, Right of First Refusal/Offer/Negotiation (Rofr/Rofo/Rofn), and Post-Termination Services, and clauses related to the Change of Control.

Whereas ChatGPT demonstrates notable effectiveness in norm extraction, our analysis uncovers recurring challenges. Some common problems include overlooking crucial details, hallucinations, incorrect parsing of conjunctions, and inferring incorrect norm types and elements. Additionally, our investigation emphasizes the scarcity of high-quality annotated datasets crucial for furthering norm extraction from contracts and, hence, advancement toward automated contract comprehension. Existing datasets are marred by limitations, including a small number of manually annotated contracts, a lack of diversity in types of contracts and clauses, and poor manual annotations.

Norms offer a principled basis for engineering flexible, autonomous multiagent systems for business and other cross-organizational settings. Automating norm extraction from unstructured text documents can help us design more robust and efficient decentralized autonomous multiagent systems. A clear path to engineering a multiagent system based on norms derived from contracts can improve the comprehensibility of the resulting specifications and clearer accountability and, thus, trust. Our efforts are a step forward in that direction. The insights from this study can guide future research aimed at overcoming barriers to effective automated norm extraction from natural text.

The rest of this paper is organized as follows. Section 2 discusses related work. Section 3 describes the methodology (including definitions, dataset, and the ChatGPT prompt used). Section 4 discusses the findings of our qualitative (manual) analysis, followed by a conclusion in Sect. 5.

2 Related Work

Automating comprehension of legal documents has long been a topic of investigation. Many prior works have automated the extraction of useful information from unstructured legal documents. This includes extracting software requirements [39], unfair or risky clauses [29, 31], legal norms [18], conflicting norms [1], and business events and their temporal constraints [16]. However, despite considerable progress, the goal of achieving comprehensive automation in contract understanding is far from complete.

Automating contract execution from unstructured text documents involves two fundamental steps. *First*, identifying the contract clauses and then inferring

the relationship between the identified clauses. This involves identifying relevant text spans in the document (i.e., context) and extracting relationships between different entities from the context. *Second*, translating the identified relationships into formal specifications suitable for computational processing, such as Hercule [10] and Symboleo [41]. In this work, we focus on the first task, more precisely, extracting norms (normative relationships) from unstructured legal contracts.

A major roadblock in automating contract understanding is the lack of manually annotated gold-standard datasets. The manual annotation of contracts has long posed a challenge essentially due to the complexity and intricacy of language in contract documents that require experts for annotations. Additionally, contract documents are extensive in length (a contract may be a few pages to hundreds and sometimes even thousands of pages long). Although some previous works have created manually annotated contract datasets, they are often small (i.e., contain few manually annotated contracts), lack coverage (i.e., identify a limited number of different types of clauses and contracts), and exhibit poor quality human annotations.

Leivaditi et al. [30] annotate 179 lease agreements for named entities and red flags. Although their dataset is useful for automating contract comprehension, it focuses exclusively on one type of contract (leases) and a small number of types of clauses. Koreeda and Manning [28] create ContractNLI, a dataset for Natural Language Inference (NLI) on contracts. Given a contract and a set of hypotheses, they annotate each hypothesis as entailed by, contradicting, or not mentioned by (neutral to) the contract. They also identify evidence for the annotation as text spans in the contract. Although ContractNLI offers more manually annotated contracts (≈ 600) than most datasets containing annotated contracts, its annotations are less granular. For instance, ContractNLI does not identify different contract clauses, essential for automated contract understanding.

For our analysis, we chose CUAD (Contract Understanding Atticus Dataset) [19], a Contract Clause Extraction (CCE) dataset containing more than 500 contracts manually annotated by experts (lawyers and law students). CUAD proposes the task of contract comprehension as a question-answering task and contains text spans of the contract clauses manually identified as belonging to one of 41 categories. CUAD is more diverse than previous datasets on contracts (i.e., contains contracts from different domains), has more annotated contracts, and is annotated by experts. Further, the ontology is developed for the given dataset (as opposed to adapting an ontology from another work that may or may not be suitable for the current dataset).

Prior research has proposed several methodologies for extracting essential clauses from unstructured legal documents, including Case-Based Reasoning (CBR) [13], grammars, word lists, and heuristics [27], rule-based NLP [45], hand-crafted rules and machine learning [12], neural models such as hierarchical BiLSTM with attention and CNN (Convolution Neural Networks) [6, 7], and fine-tuned transformer-based models, such as Legal-BERT [34]. Although some of the prior approaches are effective, they usually require a lot of manual effort, either in hand-crafting rules or annotating large amounts of data for training.

Recent advances in NLI and NLU (Natural Language Understanding) present promising avenues for alleviating the burden of manual effort in automated contract understanding. Vaswani et al. [47] proposed Transformers, a deep learning model architecture based on multihead self-attention, which pushed the state-of-the-art performance for multiple NLI and NLU benchmarks, including the General Language Understanding Evaluation (GLUE) [49], Stanford Question Answering (SQuAD) [37, 38], and Multigenre NLI (MNLI) [50]. We use ChatGPT (model GPT-3.5-turbo), an instance of the Generative Pretrained Transformer (GPT), to extract norms from contracts. ChatGPT has pushed the state of the art on multiple benchmarking language tasks. ChatGPT showcases capabilities that closely mirror human-level performance on language understanding and inference and occasionally exceed human proficiency on specific tasks [5]. ChatGPT has been used in several domains including healthcare [3, 20, 26, 40], business and finance [2, 17], creative content generation [11, 46], writing code [21, 24], and education and training [4, 25].

Although transformer-based models show promising results on a variety of NLU and NLI tasks, evidence suggests that most LLMs perform poorly in contract understanding [51]. Further, there is a lack of qualitative analysis of how well the LLMs' perform in contract understanding. Such analysis is essential to devise reliable automated contract understanding systems that could streamline the process of contract analysis and review, saving valuable time and resources for legal professionals and organizations. To facilitate such efforts, we qualitatively evaluate the effectiveness of ChatGPT in understanding contracts via norm extraction.

3 Methodology

We extract norms (normative relationships) from contracts (unstructured text documents) to facilitate the engineering of agents and multiagent systems that respect stakeholder needs as captured in contracts.

In abstract terms, a norm creates a legitimate expectation of one autonomous party over another. The sense of *norm* used here is the one established in the legal literature [48] and is formal and prescriptive. A norm can be viewed as an element of a contract. Or, conversely, a contract is (primarily) a set of norms. Norms in contracts serve multiple purposes and fall into a variety of types of clauses [19, 44].

Norms provide a natural way to engineer multiagent systems because they provide a natural basis (i.e., one that is close to stakeholders' conception) for governance [23, 42] and accountability [9] in settings involving two or more autonomous entities. The benefits of using norms include improvements in the trustworthiness and resilience of sociotechnical systems [22, 33].

We adopt Gao and Singh's [2014] approach to define normative relationships to extract from a contract. A norm is directed from a subject to an object and is constructed as a conditional relationship involving an antecedent (which brings the norm into force, i.e., the condition on which the action of the subject depends)

and a consequent (which brings the norm to satisfaction, i.e., the outcomes of the action).

A norm can be one of the following four types.

- A *commitment*** means that its subject commits to its object to ensure the consequent if the antecedent holds.
- A *prohibition*** means that its subject is forbidden by its object from bringing about the consequent if the antecedent holds.
- An *authorization*** means that its object authorizes its subject to bring about the consequent if the antecedent holds.
- A *power*** means that its subject is empowered by its object to bring about the consequent if the antecedent holds.

The norm types of authorization and power involve similar language; however, some differences exist [15]. An authorization norm is when the subject is allowed by an object to bring about the consequent such that the consequent doesn't change an existing normative relationship. In contrast, a power norm is when the subject brings about the consequent as a way to change an existing normative relationship. For instance, consider a manufacturing contract. The manufacturer may use some equipment the purchaser provides under the stated conditions. No norms change as a result of such usage. Therefore, the use of equipment maps most naturally to an authorization norm. The purchaser may cancel an order with prior notice, i.e., it can terminate a commitment at will, thereby changing the normative relationship between itself and the manufacturer. Therefore, termination in this case (and usually) is mapped to power.

3.1 Dataset

There is a lack of good-quality gold standard datasets for contract understanding. This is partly because of the complexity of the task itself; annotating contracts often requires expert judgment. Further, there is no limit to the variety of clauses that may be present in a contract depending on its purpose. For instance, a manufacturing contract may have clauses that are not found in a lease agreement, and vice versa. Similarly, for other types of contracts. Not surprisingly, prior works focus on a few clauses to annotate depending on the task.

For our analysis, we chose CUAD (Contract Understanding Atticus Dataset), which contains 510 manually annotated contracts. To ensure diversity in the dataset, CUAD includes 25 types of contracts, such as manufacturing, service, and leasing. Contracts in CUAD have varying lengths, ranging from a few pages to over one hundred pages. Each contract in CUAD is annotated manually by legal experts by identifying essential clauses (i.e., text spans in the contracts) that facilitate contract comprehension. CUAD defines 41 types of clauses (categories), including *temporal clauses* such as Effective Date, Renewal Date, and Expiration Date, *service-related clauses* such as Governing Laws, Revenue-Profit Sharing, and Price Restrictions, and *termination-related clauses* such as Termination for Convenience and Post Termination Services. CUAD contains 13 000

of such clauses identified manually by experts. In addition to identifying the relevant spans, CUAD contains a short answer (derived from the relevant text spans) to a question for each clause, such as *on what date will the contract's initial term expire?* (Expiration Date), or *Is one party required to share revenue or profit with the counterparty for any technology, goods, or services?* (Revenue/Profit Sharing).

Although CUAD annotations are claimed to be reliable and double-checked by experts, we found instances where the annotations were lacking. We found some annotations leave out contexts (text spans from the contract) that may help answer the clause-related questions more accurately. We manually verified some contracts and found termination-related clauses, such as Post-Termination Services and Termination For Convenience, often overlooked relevant text spans casting doubt on the quality of these annotations. This shows the inherent difficulties of the task that is challenging even for experts. This may be due to long and complex sentence structures common in contracts or information relevant to one clause often being spread across the document.

Some clause types in CUAD are easy to identify. For instance, temporal clauses such as *agreement date* and *expiration dates* have a distinctive syntax. Likewise, the *parties* involved in a contract are easy to identify as they are explicitly mentioned at the beginning of the contract and repeated multiple times across the document. For simplicity, we focus our analysis on clause types related to the termination of a contract, such as *Termination For Convenience*, *Right of first refusal/offer/negotiation* (Rofr/Rofo/Rofn), and *Post-Termination Services*, and clause types related to the change of control of a party to the contract, such as *Change of Control*. These clause types are a good venue to look for norms. Further, these are some of the toughest clause types to identify, as evidenced by the fact that three of these four clause types were in the top six worst-performing clause types benchmarked using a Transformer model (DeBERTa) [19]. We pick clauses from the CUAD dataset of the above-mentioned types and analyze the norms identified by ChatGPT. We randomly sample 50 clauses to evaluate and identify an additional 50 that potentially contain more challenging logical forms to extract, such as conditional actions and multiple conjunctions (OR, AND). We use keywords such as *and*, *or*, *upon*, *unless*, and so on to identify such challenging clauses. We use ChatGPT to extract norms from these 100 clauses and manually evaluate the extracted norms.

3.2 Extracting Norms via ChatGPT

ChatGPT is a state-of-the-art LLM based on the GPT (Generative Pretrained Transformer) architecture. ChatGPT pushed the state-of-the-art for multiple NLP tasks such as language translation, text summarization, sentiment analysis, and question-answering [36]. A major advantage of using ChatGPT is that it requires no training data. ChatGPT can be instructed via prompts to perform the norm extraction.

We craft prompts to extract the norms. We incrementally refine the prompt by manually inspecting the extracted norms against the expected norms for a few

sample clauses. ChatGPT shows low sensitivity to small changes in the prompt. Small changes here refer to rephrasing the prompts and substituting words with synonyms. Such minor changes to the prompt yield similar results as long as the meaning of the prompt remains the same. However, ChatGPT performs better when given more detailed prompts, i.e., when they contain more information about the concepts to be extracted, such as definitions and elaborations of the concepts to extract. For instance, we initially did not specify the direction of the relationship between the subject and object for different norm types in the prompt. The norms extracted via such prompts often involve the correct parties, i.e., objects and subjects, but in incorrect roles. This may be because the roles (directionality) differ depending on the norm type. For example, in the case of commitments, the subject commits to the object, whereas in the case of prohibitions, the object prohibits the subject. Including the directions explicitly in the definitions for the subject and object in the prompt improved the results by reducing such mismatches.

As input, we give ChatGPT the prompt followed by the clause from the contract (from which the norm needs to be extracted). Figure 1 shows the prompt we use to extract the norms given a contract clause.

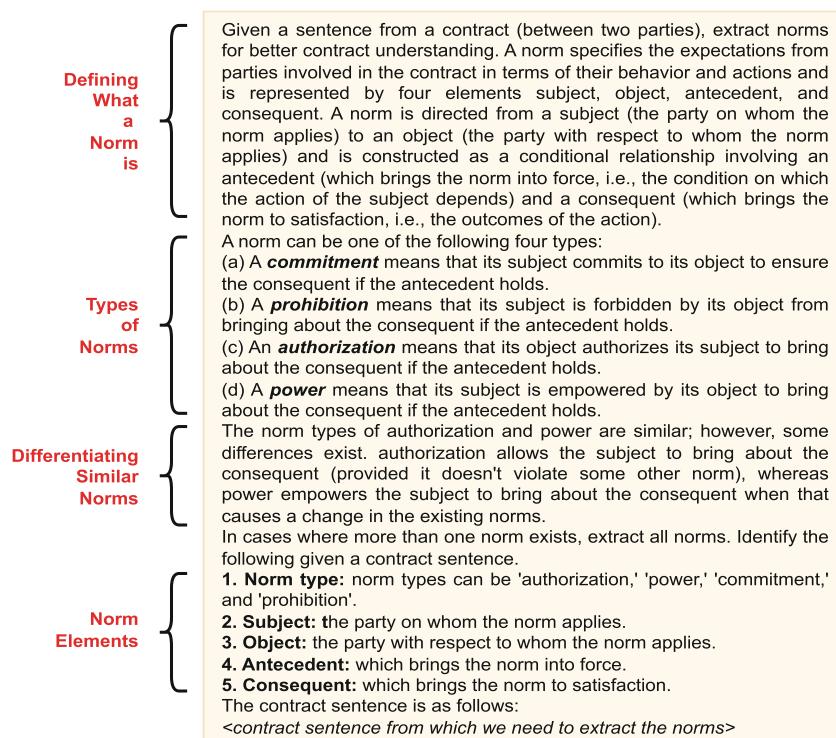


Fig. 1. ChatGPT prompt for norm extraction.

We run our experiments on the GPT-3.5-Turbo version of ChatGPT. Since the ChatGPT model is nondeterministic by default, we set the temperature equal to 0.0 for the reproducibility of our results.

4 Findings

ChatGPT did a good job in extracting norms from contract clauses in many cases. However, the extractions are not without limitations. We evaluate 100 clauses sampled from CUAD. Based on the quality of extraction by ChatGPT, we label each extraction into four categories. The four categories are *correct extractions* (norm extraction without mistakes), *incorrect norm types* (extracted norms that contain an incorrect norm type), *incorrect norm elements* (extracted norms that contain an incorrect norm element), and *overlooking crucial details* (extracted norms that omit crucial details that should be included). An extraction can belong to multiple categories (except for correct extractions).

Table 1. Distribution of norm extractions across various categories.

Category	Count
Correct Extractions	31
Incorrect Norm Types	43
Incorrect Norm Elements	30
Overlooking Crucial Details	16

The figure shows a user interface for ChatGPT. On the left, there is a prompt icon followed by the text "Prompt +". Below it is a red box containing a norm clause from an agreement:

Notwithstanding any other provision of this Agreement,
Rogers may terminate this Agreement, at any time, upon
sixty (60) days prior written notice to Licensor.

Underneath the clause, parts are labeled with terms: "subject" under "Rogers", "consequent" under "terminate", "antecedent" under "prior written notice", and "object" under "Licensor".

To the right, there is a green box titled "ChatGPT Extracted Norms" containing the extracted norm information:

Norm 1: Authorization
 Subject: Rogers
 Object: Licensor
 Antecedent: Rogers providing sixty (60) days' prior written notice
 Consequent: Termination of the Agreement

Fig. 2. A norm extracted via ChatGPT

Table 1 shows the distribution across the four categories. Of the 100 extractions that we analyze, about one-third (31) of the extractions are correct and

contain no mistakes. However, 43 extractions contain an incorrect norm type (for example, incorrectly inferring an authorization as commitment), 30 extractions contain an incorrect norm element (for example, incorrect subject, object, antecedent, or consequent), and 16 extractions omit crucial details (details that are important to understand the norm completely and correctly). We discuss each category in more detail with examples.

Figure 2 shows an example of successful norm extraction from a contract clause using ChatGPT with the given prompt (Fig. 1). ChatGPT extracts an *authorization* norm and correctly infers all norm elements.

We notice contract clauses containing redactions lead to more incorrect extracted norms. This implies that contract clauses with redacted information are more challenging for ChatGPT than clauses without redactions. This could be critical in situations where some of the information in the contract is redacted for confidentiality reasons, such as contracts involving national security and intellectual property. We leave further analysis of norm extraction from clauses containing redactions for future work and only discuss norm extraction from clauses that do not contain any redactions.

Below, we discuss some recurring issues in ChatGPT’s norm extractions with examples.

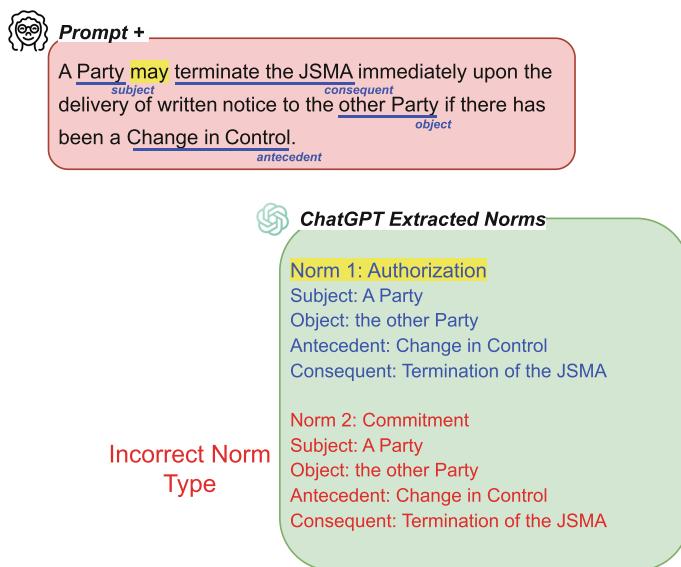


Fig. 3. A norm extraction containing incorrect norm types

Incorrect Norm Types. In some cases, ChatGPT extracts incorrect norm types. For instance, ChatGPT often extracts a commitment norm (in addition to

the authorization norm) from clauses that contain only an authorization clause. Figure 3 shows one such example. ChatGPT extracts two norms, both have the same norm elements. Whereas norm 1 (authorization) is correct, norm 2 (commitment) is incorrect. This is because the original clause uses *may* to describe the potential action, making the action optional (has the authority to) but not mandatory (i.e., commitment). Hence, the extracted commitment norm is incorrect. We notice that ChatGPT often confuses what *must* be done with what *can* be done, leading to incorrect norm extraction.

Incorrect Norm Elements. In some of the extracted norms, the norm elements are incorrect. Figure 4 shows one such example where multiple extracted norm elements are incorrect. The consequent and object are incorrect, and the antecedent is only partially captured.

The figure shows a user interface for ChatGPT. On the left, there is a 'Prompt +' button with a person icon. Below it is a text input field containing a legal clause about rights assignment. On the right, there is a green box titled 'ChatGPT Extracted Norms' containing the extracted norm elements. To the left of this green box, the text 'Incorrect Norm Elements' is written in red.

Prompt +

Notwithstanding the foregoing, each Party may assign the rights and obligations under this Agreement in whole, without consent of the other Party, to a Third Party or Affiliate in connection with the transfer or sale of all or substantially all of its business or in the event of a merger, consolidation or change in control provided that the assignee assumes in writing and becomes directly obligated to the other Party to perform all of the obligations of assignor under this Agreement.

ChatGPT Extracted Norms

Norm 1: Authorization
Subject: Each party
Object: Third Party or Affiliate
Antecedent: Assignment of rights and obligations under the Agreement in connection with the transfer or sale of business, merger, consolidation, or change in control
Consequent: The assignee assumes in writing and becomes directly obligated to perform all obligations of the assignor under the Agreement

Incorrect Norm Elements

Fig. 4. A norm extraction containing incorrect norm elements.

Overlooking Crucial Details. ChatGPT sometimes overlooks crucial details while extracting norms. For instance, in Fig. 5, the detail that the agreement will be in effect until March 18th is not captured by any norm (the date is overlooked). Further, the detail that 30 days written notice is needed for termination is also lacking from all extracted norms. Moreover, the extraction contains incorrect norms.

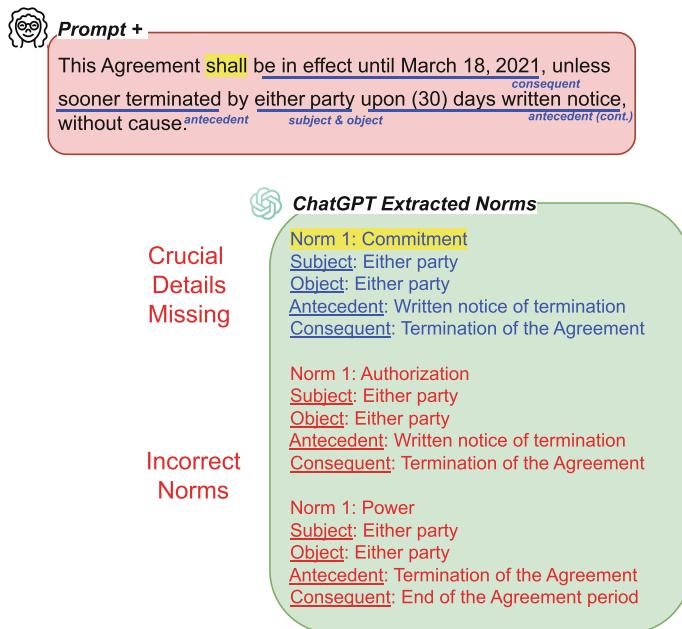


Fig. 5. A norm extraction that overlooks crucial details.

Apart from these four common errors in norm extraction, we notice a few more errors worth mentioning. ChatGPT often hallucinates information, finds it challenging to parse clauses containing multiple conjunctions, and leaves norm elements empty sometimes.

Hallucination. *Hallucination* refers to a phenomenon where a model generates outputs not grounded in the input data. Hallucination can be highly problematic for norm extraction from contracts that rely heavily on the accuracy of the extracted information, and even minor hallucinations could change the meaning and applicability of clauses. Hallucination could lead to incorrect inferences based on the extracted norms.

Figure 6 shows a norm extraction that contains hallucination. The subject is inferred as VS (Vendor Services). Although the acronym (VS) is correct (i.e., the entity mentioned in the clause), its expansion *Vendor Services* is incorrect (the correct expansion is *VirtualScopics*). Interestingly, the input clause does not contain the expansion of the acronym, but ChatGPT hallucinated it (the word *Service* occurs in the input clause but not *Vendor*). Additionally, the extracted norm identifies two norm types for the same extracted norm, which is incorrect. The commitment norm type is correct, but the prohibition norm is incorrect.

Incorrect Parsing of Conjunctions. Clauses with multiple conjunctions, such as AND and OR, are sometimes not parsed correctly, leading to incorrect

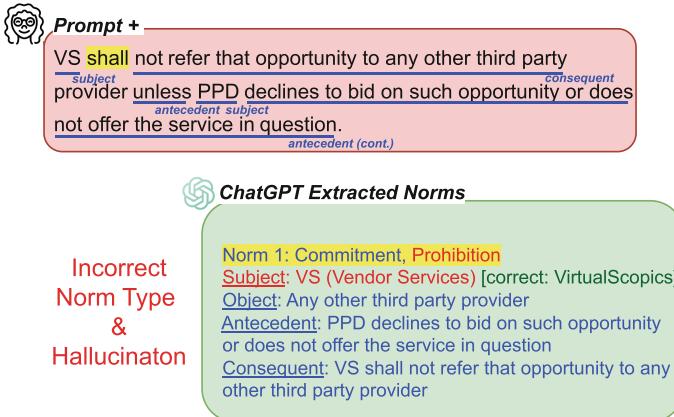


Fig. 6. A norm extraction containing hallucination.

norm extractions. Norm extractions from clauses containing multiple conjunctions lead to long, complex, and incorrect antecedents and consequents. For instance, Fig. 4 shows norm extraction from a sentence containing multiple conjunctions leading to incorrect norm elements, including consequent, object, and antecedent. A potential solution could be to have a separate prompt that first breaks down the logical form based on the conjunctions before extracting the norms. We leave this to future work as the focus of this study is to identify the challenges and opportunities in norm extraction.

Empty Norm Elements. For some extracted norms, ChatGPT leaves some norm elements empty (i.e., using values such as ‘’, N/A, NULL, or Not Specified). This is more common for subjects and objects, followed by antecedents. Consequent is rarely left empty. Although an empty element in an extracted norm may indicate poor extraction in some cases, it may be the correct response in others. For instance, when no precondition (or trigger) exists for a consequent, it implies an empty antecedent. Similarly, an empty subject (or object) could mean either *no* party or *all* parties subject to the contract. However, in many cases, empty norm elements imply incorrect norm extraction.

5 Conclusion

This study delves into using LLMs such as ChatGPT to extract norms from contracts, shedding light on the opportunities and challenges inherent in automating contract comprehension. We uncover critical insights through a qualitative analysis of the norms extracted by ChatGPT.

We find that the extraction is influenced by the prompts used. A well-crafted prompt that includes definitions of the formal terms used in the prompt

yields much better results than less detailed prompts. Although a comprehensive prompt consistently yields superior outcomes, the results do not seem too sensitive to minor changes in the prompt (such as rephrasing and substitution with synonyms).

Despite ChatGPT’s overall efficacy in norm extraction, our scrutiny unveils recurrent issues. Overlooking crucial information, hallucination, incorrect processing of conjunctions (AND and OR), incorrect norm types, and incorrect norm elements in the extracted norms emerge as persistent challenges. Moreover, ChatGPT occasionally identifies norms with empty norm elements, long consequents (that include the subject, object, and antecedents), and infers incorrect objects and subjects.

Furthermore, our exploration highlights the dearth of high-quality annotated datasets essential for advancing automated contract understanding. Existing datasets suffer from limitations such as small size, limited diversity, and subpar annotations, hindering progress in this domain.

Despite these challenges, our findings underscore the promise of using ChatGPT for norm extraction from unstructured text documents. Particularly compelling is the fact that no training or fine-tuning is necessitated, positioning ChatGPT (and similar LLMs) as a preferred model for tasks like contract understanding that lack high-quality annotated datasets.

In conclusion, this research not only provides valuable insights into the efficacy and challenges of employing LLMs for norm extraction from contracts but also underscores their potential as powerful tools in the realm of automated contract understanding.

Acknowledgments. Thanks to the US NSF (grant IIS-1908374) for support for this research.

References

1. Aires, J.P., Meneguzzi, F.: Norm conflict identification using a convolutional neural network. In: International Workshops on Coordination, Organizations, Institutions, Norms, and Ethics (COINE) for Governance of Multi-Agent Systems XIII, LNCS, vol. 12298, pp. 3–19, Springer (2020). https://doi.org/10.1007/978-3-030-72376-7_1
2. AlAfnan, M.A., Dishari, S., Jovic, M., Lomidze, K.: ChatGPT as an educational tool: opportunities, challenges, and recommendations for communication, business writing, and composition courses. *J. Artif. Intell. Technol.* **3**(2), 60–68 (2023). <https://doi.org/10.37965/jait.2023.0184>
3. Ali, S.R., Dobbs, T.D., Hutchings, H.A., Whitaker, I.S.: Using ChatGPT to write patient clinic letters. *Lancet Digital Health* **5**(4), e179–e181 (2023). [https://doi.org/10.1016/S2589-7500\(23\)00048-1](https://doi.org/10.1016/S2589-7500(23)00048-1)
4. Baidoo-Anu, D., Owusu Ansah, L.: Education in the era of generative artificial intelligence (AI): understanding the potential benefits of ChatGPT in promoting teaching and learning. *J. AI* **7**(1), 52–62 (2023). <https://doi.org/10.61969/jai.1337500>

5. Bubeck, S., et al.: Sparks of artificial general intelligence: early experiments with GPT-4. arXiv preprint [arXiv:2303.12712](https://arxiv.org/abs/2303.12712) (2023)
6. Chalkidis, I., Androutsopoulos, I., Michos, A.: Obligation and prohibition extraction using hierarchical RNNs. In: Gurevych, Iryna and Miyao, Yusuke (ed.) Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 254–259, Association for Computational Linguistics, Melbourne, Australia (2018). <https://doi.org/10.18653/v1/P18-2041>
7. Chalkidis, I., Fergadiotis, M., Malakasiotis, P., Androutsopoulos, I.: Neural contract element extraction revisited. In: Workshop on Document Intelligence at NeurIPS 2019 (2019). <https://openreview.net/forum?id=B1x6fa95UH>
8. Chopra, A.K., Singh, M.P.: Custard: computing norm states over information stores. In: Proceedings of the 15th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS), pp. 1096–1105, IFAAMAS, Singapore (2016). <https://doi.org/10.5555/2936924.2937085>
9. Chopra, A.K., Singh, M.P.: Accountability as a foundation for requirements in sociotechnical systems. IEEE Internet Comput. (IC) **25**(6), 33–41 (2021). <https://doi.org/10.1109/MIC.2021.3106835>
10. Christie V, S.H., Chopra, A.K., Singh, M.P.: Hercule: representing and reasoning about norms as a foundation for declarative contracts over blockchain. IEEE Internet Comput. (IC) **25**(4), 67–75 (2021). <https://doi.org/10.1109/MIC.2021.3080982>
11. Cox, C., Tzoc, E.: ChatGPT: implications for academic libraries. Coll. Res. Libr. News **84**(3), 99 (2023). <https://doi.org/10.5860/crln.84.3.99>
12. Curtotti, M., McCreath, E.: Corpus based classification of text in Australian contracts. In: Indurkhy, N., Zwarts, S. (eds.) Proceedings of the Australasian Language Technology Association Workshop 2010, pp. 18–26, Melbourne, Australia (2010). URL <https://aclanthology.org/U10-1005>
13. D'Agostini Bueno, T.C., Von Wangenheim, C.G., Da Silva Mattos, E., Hoeschl, H.C., Barcia, R.M.: Jurisconsulto: retrieval in jurisprudencial text bases using juridical terminology. In: Proceedings of the 7th International Conference on Artificial Intelligence and Law, pp. 147–155. ICAIL, Association for Computing Machinery, New York, NY, USA (1999). <https://doi.org/10.1145/323706.323789>
14. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 17th Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, vol. 1, pp. 4171–4186, Association for Computational Linguistics, Minneapolis, Minnesota (2019). <https://doi.org/10.1162/v1/N19-1423>
15. Gao, X., Singh, M.P.: Extracting normative relationships from business contracts. In: International Conference on Autonomous Agents and Multi-Agent Systems, AAMAS, Paris, France, May 5-9, 2014, pp. 101–108, IFAAMAS/ACM (2014). <http://dl.acm.org/citation.cfm?id=2615751>
16. Gao, X., Singh, M.P.: Mining contracts for business events and temporal constraints in service engagements. IEEE Trans. Serv. Comput. **7**(3), 427–439 (2014). <https://doi.org/10.1109/TSC.2013.21>
17. George, A.S., George, A.S.H.: A review of ChatGPT AI's impact on several business sectors. Partners Univers. Int. Innovation J. **1**(1), 9–23 (2023). <https://doi.org/10.5281/zenodo.7644359>
18. Hashmi, M.: A methodology for extracting legal norms from regulatory documents. In: 2015 IEEE 19th International Enterprise Distributed Object Computing Workshop, pp. 41–50 (2015). <https://doi.org/10.1109/EDOCW.2015.29>

19. Hendrycks, D., Burns, C., Chen, A., Ball, S.: CUAD: an expert-annotated NLP dataset for legal contract review. In: Vanschoren, J., Yeung, S. (eds.) Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks, December 2021 (2021). <https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/6ea9ab1baa0efb9e19094440c317e21b-Abstract-round1.html>
20. Howard, A., Hope, W., Gerada, A.: ChatGPT and antimicrobial advice: the end of the consulting infection doctor? *Lancet. Infect. Dis.* **23**(4), 405–406 (2023)
21. Jalil, S., Rafi, S., LaToza, T.D., Moran, K., Lam, W.: ChatGPT and software testing education: promises & perils. In: IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW), pp. 4130–4137 (2023). <https://doi.org/10.1109/ICSTW58534.2023.00078>
22. Kafali, Ö., Ajmeri, N., Singh, M.P.: DESEN: specification of sociotechnical systems via patterns of regulation and control. *ACM Trans. Softw. Eng. Methodol. (TOSEM)* **29**(1), 7:1–7:50 (2020). <https://doi.org/10.1145/3365664>
23. Kampik, T., et al.: Governance of autonomous agents on the web: challenges and opportunities. *ACM Trans. Internet Technol. (TOIT)* **22**(4), 104:1–104:31 (2022). <https://doi.org/10.1145/3507910>
24. Kashefi, A., Mukerji, T.: ChatGPT for programming numerical methods. *J. Mach. Learn. Model. Comput.* **4**(2), 1–74 (2023)
25. Kasneci, E., et al.: ChatGPT for good? On opportunities and challenges of large language models for education. *Learn. Individ. Differ.* **103**, 102274 (2023). <https://doi.org/10.1016/j.lindif.2023.102274>
26. Khan, R.A., Jawaid, M., Khan, A.R., Sajjad, M.: ChatGPT - reshaping medical education and clinical management. *Pak. J. Med. Sci.* **39**(2), 605–607 (2023). <https://doi.org/10.12669/pjms.39.2.7653>
27. Kiyavitskaya, N., et al.: Automating the extraction of rights and obligations for regulatory compliance. In: Li, Q., Spaccapietra, S., Yu, E., Olivé, A. (eds.) ER 2008. LNCS, vol. 5231, pp. 154–168. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-87877-3_13
28. Koreeda, Y., Manning, C.: ContractNLI: a dataset for document-level natural language inference for contracts. In: Findings of the Association for Computational Linguistics: EMNLP 2021, pp. 1907–1919. Association for Computational Linguistics, Punta Cana, Dominican Republic (2021). <https://doi.org/10.18653/v1/2021.findings-emnlp.164>
29. Lee, J., Yi, J.S., Son, J.: Development of automatic-extraction model of poisonous clauses in international construction contracts using rule-based NLP. *J. Comput. Civ. Eng.* **33**(3), 04019003 (2019). [https://doi.org/10.1061/\(ASCE\)CP.1943-5487.0000807](https://doi.org/10.1061/(ASCE)CP.1943-5487.0000807)
30. Leivaditi, S., Rossi, J., Kanoulas, E.: A benchmark for lease contract review. arXiv preprint [arXiv:2010.10386](https://arxiv.org/abs/2010.10386) (2020)
31. Lippi, M., et al.: Automated detection of unfair clauses in online consumer contracts. In: Legal Knowledge and Information Systems - JURIX 2017: The 13th Annual Conference, Luxembourg, 13-15, Frontiers in Artificial Intelligence and Applications, vol. 302, pp. 145–154. IOS Press (2017). <https://doi.org/10.3233/978-1-61499-838-9-145>
32. Liu, Y., et al.: RoBERTa: a robustly optimized BERT pretraining approach. arXiv preprint [arXiv:1907.11692](https://arxiv.org/abs/1907.11692) (2019)
33. Mahala, G., Kafali, Ö., Dam, H.K., Ghose, A., Singh, M.P.: A normative approach for resilient multiagent systems. *JAAMAS* **37**(2), 46:1–46:40 (2023). <https://doi.org/10.1007/s10458-023-09627-4>

34. Mamakas, D., Tsotsi, P., Androutsopoulos, I., Chalkidis, I.: Processing long legal documents with pre-trained transformers: modding LegalBERT and longformer. In: Proceedings of the Natural Legal Language Processing Workshop 2022, pp. 130–142. Association for Computational Linguistics, Abu Dhabi, United Arab Emirates (Hybrid) (2022). <https://doi.org/10.18653/v1/2022.nllp-1.11>
35. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al.: Improving language understanding by generative pre-training. OpenAI (2018). <https://www.cs.ubc.ca/~amuhamed/LING530/papers/radford2018improving.pdf>
36. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al.: Language models are unsupervised multitask learners, Technical report, Open AI (2019). <https://s3.amazonaws.com/insightcivic.s3.us-east-1.amazonaws.com/language-models.pdf>
37. Rajpurkar, P., Jia, R., Liang, P.: Know what you don't know: unanswerable questions for SQuAD. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 784–789. Association for Computational Linguistics, Melbourne, Australia (2018). <https://doi.org/10.18653/v1/P18-2124>
38. Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P.: SQuAD: 100,000+ questions for machine comprehension of text. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 2383–2392. Association for Computational Linguistics, Austin, Texas (2016). <https://doi.org/10.18653/v1/D16-1264>
39. Sainani, A., Anish, P.R., Joshi, V., Ghaisas, S.: Extracting and classifying requirements from software engineering contracts. In: IEEE 28th International Requirements Engineering Conference (RE), pp. 147–157 (2020). <https://doi.org/10.1109/RE48521.2020.00026>
40. Sallam, M.: ChatGPT utility in healthcare education, research, and practice: systematic review on the promising perspectives and valid concerns. Healthcare **11**(6) (2023). <https://doi.org/10.3390/healthcare11060887>
41. Sharifi, S., Parvizmosaed, A., Amyot, D., Logrippo, L., Mylopoulos, J.: Symboleo: towards a specification language for legal contracts. In: 2020 IEEE 28th International Requirements Engineering Conference (RE), pp. 364–369 (2020). <https://doi.org/10.1109/RE48521.2020.00049>
42. Singh, M.P.: Norms as a basis for governing sociotechnical systems. ACM Trans. Intell. Syst. Technol. (TIST) **5**(1), 21:1–21:23 (2013). <https://doi.org/10.1145/2542182.2542203>
43. Singh, M.P., Chopra, A.K.: Computational governance and violable contracts for blockchain applications. IEEE Comput. **53**(1), 53–62 (2020). <https://doi.org/10.1109/MC.2019.2947372>
44. Singh, M.P., Gao, X.: A gray box conceptual model for accountability and ethics in business contracts. IEEE Internet Comput. (IC) **25**(4), 13–19 (2021). <https://doi.org/10.1109/MIC.2021.3083295>
45. Sleimi, A., Sannier, N., Sabetzadeh, M., Briand, L.C., Ceci, M., Dann, J.: An automated framework for the extraction of semantic legal metadata from legal texts. Empir. Softw. Eng. **26**(3), 43 (2021). <https://doi.org/10.1007/S10664-020-09933-5>
46. Taecharungroj, V.: What can ChatGPT do? Analyzing early reactions to the innovative AI chatbot on twitter. Big Data Cogn. Comput. **7**(1) (2023). <https://doi.org/10.3390/bdcc7010035>
47. Vaswani, A., et al.: Attention is all you need. In: Guyon, I., et al. (eds.) Advances in Neural Information Processing Systems, vol. 30. Curran Associates, Inc., Long Beach, CA (2017). <https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf>

48. Von Wright, G.H.: Norm and action: a logical enquiry. In: International Library of Philosophy and Scientific Method. Humanities Press, New York (1963)
49. Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., Bowman, S.: GLUE: a multi-task benchmark and analysis platform for natural language understanding. In: Proceedings of the EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, pp. 353–355. Association for Computational Linguistics, Brussels, Belgium (2018). <https://doi.org/10.18653/v1/W18-5446>
50. Williams, A., Nangia, N., Bowman, S.: A broad-coverage challenge corpus for sentence understanding through inference. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pp. 1112–1122. Association for Computational Linguistics, New Orleans, Louisiana (2018). <https://doi.org/10.18653/v1/N18-1101>
51. Xu, W., Deng, Y., Lei, W., Zhao, W., Chua, T.S., Lam, W.: ConReader: exploring implicit relations in contracts for contract clause extraction. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 2581–2594. Association for Computational Linguistics, Abu Dhabi, United Arab Emirates (2022). <https://doi.org/10.18653/v1/2022.emnlp-main.166>
52. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R.R., Le, Q.V.: XLNet: generalized autoregressive pretraining for language understanding. In: Wallach, H., Larochelle, H., Beygelzimer, A., d’Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 32. Curran Associates, Inc., Vancouver (2019). <https://proceedings.neurips.cc/paper/2019/file/dc6a7e655d7e5840e66733e9ee67cc69-Paper.pdf>



Harnessing the Power of LLMs for Normative Reasoning in MASs

Bastin Tony Roy Savarimuthu¹(✉) ID, Surangika Ranathunga²(✉) ID,
and Stephen Cranefield¹(✉) ID

¹ University of Otago, Dunedin, New Zealand

{tony.savarimuthu, stephen.cranefield}@otago.ac.nz

² Massey University, Auckland, New Zealand

S.Ranathunga@massey.ac.nz

Abstract. Software agents, both human and computational, do not exist in isolation and often need to collaborate or coordinate with others to achieve their goals. In human society, social mechanisms such as norms ensure efficient functioning, and these techniques have been adopted by researchers in multi-agent systems (MAS) to create socially aware agents. However, traditional techniques have limitations, such as operating in limited environments often using brittle symbolic reasoning. The advent of Large Language Models (LLMs) offers a promising solution, providing a rich and expressive vocabulary for norms and enabling *norm-capable* agents that can perform a range of tasks such as norm discovery, normative reasoning and decision-making. This paper examines the potential of LLM-based agents to acquire normative capabilities, drawing on recent Natural Language Processing (NLP) and LLM research. We present our vision for creating *normative LLM agents*. In particular, we discuss how the recently proposed “LLM agent” approaches can be extended to implement such normative LLM agents. We also highlight challenges in this emerging field. This paper thus aims to foster collaboration between MAS, NLP and LLM researchers in order to advance the field of normative agents.

Keywords: norms · agents · normative reasoning · Large Language Models (LLMs)

1 Introduction

In the dynamic field of multi-agent systems (MAS), the integration of Large Language Models (LLMs) with software agents is emerging as a promising area of research [47, 51]. In traditional MAS, agents are often limited by brittle symbolic reasoning and confined to specific environments. In contrast, agents powered by LLMs (*LLM agents*) have the potential to be equipped with implicit world knowledge. This evolution is crucial for their effective participation in sociotechnical systems, ensuring more efficient coexistence and collaboration between humans

and agents in societies. However, given the novelty of this emerging research area, current LLM-based agent research has not discussed the use of LLMs to assist agents with social knowledge and understanding, including normative aspects. This paper explores the potential of equipping LLM-based agents with advanced reasoning skills, particularly in the context of normative decision-making.

Norms prescribe behaviours that are expected in societies. In normative multi-agent systems, researchers have investigated how human norms can be operationalised in computational societies where an agent's behaviour is constrained by norms [25]. However, existing MAS research has primarily focused on agents making normative decisions based on symbolic expressions, which are specific to the formalisms used [8].

Recently, LLM-based reasoning about social norms has gained the attention of Natural Language Processing (NLP) researchers. There is emerging research in using LLMs for norm discovery and moral judgement over textual data such as stories and chat communications [18, 56]. However, so far this strand of research has not been harnessed to implement normative multi-agent systems.

Such NLP research suggests that the advent of LLMs introduces the possibility of employing a rich, expressive vocabulary for norms, enabling agents to gain a range of normative capabilities. Norm discovery, reasoning and conformance are some examples of such normative capabilities. These capabilities are required to seek appropriate knowledge about norms, based on the phase of the norm life-cycle of a particular norm [40]. In particular, LLMs will enable agents to reason with norms using natural language that includes asking questions such as *what norms apply in a given context?*, *what are the punishments for norm violations?*, and *what action must an agent pursue to be norm compliant?* This development opens avenues for more human-like cognitive abilities in agents, which are essential for collaborative tasks in various domains such as human-robot interaction.

The goal of this paper is to present a vision for creating **normative LLM agents** that have capabilities such as norm discovery, reasoning and compliance. Our proposal has the potential to instill superior norm capabilities at both the individual agent level and the societal level, overcoming limitations of traditional techniques. This paper also emphasizes the need for further collaboration among MAS, NLP, and LLM researchers to fully realize the potential of normative LLM agents. Such interdisciplinary effort will foster a new era of normatively adept computational agents.

2 The Promise of LLMs for MAS

In recent years, the rapidly increasing capabilities of LLMs have astounded both AI practitioners and users, especially since the release of OpenAI's ChatGPT. LLMs are nothing more than complex probabilistic models that calculate the most likely word (or, more accurately, token) given a sequence of previous tokens. However, their massive amount of textual training data and self-supervised training regime means that they exhibit a sophisticated understanding of syntax and

semantics of natural language. Furthermore, fine-tuning these models with carefully constructed prompts, as well as techniques such as reinforcement learning, can condition the model to produce answers that appear highly relevant to the prompt, giving the impression of human-like understanding.

An emerging and rapidly advancing thread of research on LLMs is addressing the question of how intelligent software agents can leverage the abilities of LLMs to help them make decisions (e.g., choose their actions). A recent survey paper by Wang et al. [47] assimilates developments in the literature to propose a standard set of modules required for an LLM-based autonomous agent to effectively leverage an LLM’s capabilities: 1) a *profile* module to store information about agent’s roles, personalities, relationships, etc., 2) a *memory* module to record the agent’s experiences in its environment, 3) a *planning* module to break complex tasks into simpler ones, and 4) an *action* module to translate decisions into actions in the environment. The LLM can then provide the agent with the ability to operate in unconstrained settings unlike the traditional approaches which are usually restricted to specific problem domains. In addition, Xi et al. [51] note that “LLMs can adapt to various languages, cultures, and domains, making them versatile and reducing the need for complex training processes and data collection”.

Furthermore, Andreas et al. [3] argues that “Language Models are models of intentional communication” in the sense that providing information about agents’ beliefs, desires and intention within prompts causes LLMs to generate outputs that appear to relate to these mental attitudes in predictable ways.

As part of a survey of LLM-based agents, Xi et al. [51] discuss research on LLM-based agents interacting in a society. Work in this area has included studies on the interactions between agents with individual personalities and emotions, cooperation in teams and spontaneous social interactions. For example, Park et al. [32] observe that LLMs “encode a wide range of human behavior from their training data” and describe a multi-agent simulation where *generative agents* use LLMs to reflect on their memories to produce high level insights on the activities and motivations of the agents. In combination with memory and planning modules, this approach generates “believable individual and emergent social behaviors”.

Notably, none of the papers mentioned above discuss the use of LLMs to assist software agents in real time with social knowledge and understanding, including normative aspects. Just as Andreas concluded that LLMs are able to model intentional communication, we posit that LLMs are also capable of revealing rich social understanding of narratives (see Sect. 3). We believe they hold great promise for assisting social agents with normative reasoning, especially in sociotechnical-technical systems where human participants naturally conceive and discuss norms in natural language rather than using symbolic representations.

3 NLP Research on Social Norms

In the last five years, there has been a surge in interest amongst NLP researchers to study social norms. In particular, a nascent NLP task of norm-related reasoning based on text (chatbot conversations, stories, situation descriptions, etc.) has recently emerged. This research can be broadly categorised under norm discovery (or norm generation), normative reasoning and norm conformance. This new-found interest of NLP researchers is not surprising. As AI systems gradually evolve to assist and collaborate with humans, they are expected to be aware of social norms that govern human cultural and social spaces. As described below, with the power of recent advancements in LLMs, there have been promising advances in norm-related reasoning over textual data.

NLP researchers have treated *norm discovery* as the task of identifying applicable norms given the textual description of a situation, and addressed it as a text generation problem. Early approaches relied on human annotated data to fine-tune LLMs in order to train them to generate new norms from unseen situations. Here, the situation has been referred by a single sentence [11] or a conversational setup [56]. Kiehne et al. [20] argue that LLMs fine-tuned with human-annotated norms may have biases. As a solution, they investigate how to remove or introduce contrasting norms during model training. Fung et al. [13] is the first to omit the need for human annotated data altogether. Instead, they prompt the LLM for norm discovery over dialogue situations. Li et al. [24] bootstrap from an existing set of social norms and prompt ChatGPT to generate new norms.

Normative Reasoning has been referred as predicting ‘moral judgement’ [18, 55]. The common approach to training a moral judgement prediction system is to fine-tune an LLM with *situation_description*, *moral_judgement* pairs. In its simplest form, normative reasoning has been treated as a text classification problem, where a textual description of an action gets classified as normative or not [31]. More recent work has explored the possibility of more fine-grained reasoning. For example, Ziems et al. [55] categorize a given situation as *ok*, *expected* or *unexpected*. The publicly available Delphi system¹ [18] can be prompted with a situation (expressed in a single sentence), and it responds with one of the following judgements: *rude*, *ok*, *bad*, *wrong*, *expected*, *shouldn’t*. However, in Delphi, normative reasoning is done in isolation with no reference to surrounding context. ClarifyDelphi [33] is a system capable of learning to ask questions to elicit salient context, which overcomes this limitation.

Norm Conformance (or ‘value alignment’ as termed by NLP researchers) refers to creating agents whose behaviours conform to expected moral and social norms for a given context and a group of people [1]. Emelin et al. [9] present LLM-based classification and generative models to determine the most suitable action given a situation and a norm. Using the Delphi system [18], Ammanabrolu et al. [1] make an agent learn to constrain its action space such that its actions align with socially beneficial human values.

¹ <https://delphi.allenai.org/>.

In order to train LLMs for the above tasks, many human-annotated text datasets have been created. Some such datasets are SOCIAL-CHEM-101 [11], Moral Stories [9], SCRUPLES [26], the MORAL INTEGRITY CORPUS [56], NORM-BANK [55] and NORMDIAL [24].

This LLM-based normative reasoning research presents an exciting opportunity to enhance an agent’s social reasoning potential. However, none of these work have been harnessed by MAS researchers as a part of an agent’s social decision-making engine.

4 Implementing Normative LLM Agents

A normative agent in a traditional MAS is typically created by incorporating domain knowledge expressed in symbolic logical representations [4, 27], which is then used for normative reasoning. In many cases, agents using this approach know about the norm *a priori*. However, such *a priori* knowledge is unrealistic for agents participating in open systems where norms change. Also, the complexity and nuances associated with norms can make it difficult to model them using formal approaches.

In order to overcome these limitations of traditional normative agents, we propose the MAS community should actively engage in developing normative LLM agents. In particular, we propose to extend the emerging proposals on creating generic LLM agent architectures [47] such that these LLM agents are enriched with norm capabilities.

The key question in developing normative LLM agents is how to train an LLM to reliably demonstrate normative capability (e.g., norm discovery, reasoning and conformance), and integrate them to an agent-based system. We relate these capabilities to NLP-related research on norms discussed in Sect. 3. Below, we characterise the nature of these normative capabilities and suggest how these can be realised in the context of building a normative LLM agent.

Norm Discovery - First and foremost, the LLM should be equipped with norm-related knowledge. As done by Fung et al. [13] (see Sect. 3), one could prompt a general-purpose LLM to discover norms implicitly captured in its world knowledge. However, generic norms most likely would not work for domain- or culture-specific agent systems. In such cases, the LLM can be further fine-tuned with human-annotated norm data. Note that, as discussed in Sect. 3, this is a common approach to teaching the notion of norms to LLMs.

A particular limitation of fine-tuning is that it results in a static set of norms (i.e., these don’t change). However, norms may not be static and some tend to emerge from complex interactions in an agent society. In order to identify emerging norms, we propose that the *planning module* of Wang et al. [47]’s LLM-agent architecture can be extended with normative reasoning capabilities. For example, an agent should be able to identify sanctions based on what it perceives from the environment (e.g. being told off), and then learn norms based on that. Approaches such as Theory of Mind (ToM) for LLM agents [23] can be used for this kind of norm identification.

Another issue that needs to be addressed is to avoid the need for fine-tuning of an LLM every time a new norm is discovered. We suggest extending the memory architectures (such as vector databases) proposed by Wang et al. [47] to store such dynamically discovered norms.

Norm Reasoning - Norm reasoning involves capabilities such as norm identification², violation detection, sanction identification, norm spreading to other agents and norm enforcement of non-compliant agents. As mentioned above, approaches such as ToM for LLM agents can be explored for violation detection and sanction identification. Norm identification (i.e., which norm currently applies in the given situation) also involves the *planning module* of the LLM agent. While the normative reasoning research in the NLP field (see Sect. 3) would be a good starting point, we envisage more sophisticated modes of LLM usage by a normative agent for explicitly reasoning about a norm (e.g., *what actions are prohibited?*). Some possibilities for norm identification are as follows.

- Prompt the LLM with the situation information, without explicitly stating a norm. For example, a robot working with humans can provide the contextual information to the LLM and ask for the norm that applies for its current situation.
- Prompt the LLM with the situation information and applicable norm(s), and ask whether the specified norm is applicable to that situation (in order to check its norm understanding).
- Prompt the LLM in an iterative manner in order to obtain further normative information, say by asking a series of questions (e.g., using techniques such as Chain of Thought [49])

Norm Conformance - Once the agent knows which norm applies in a given situation, it should be capable of selecting a norm-adhering action (i.e., to choose to conform to the norm). We propose to extend the *action module* of the LLM agent architecture [47] in order to achieve this. Note that selecting a norm adhering action is very similar to the task of norm conformance (value alignment) discussed in Sect. 3. Instead of generating a textual output as the action as done in that NLP research, here, we expect the agent architecture to generate an action to be executed in the environment that the agent is situated in. One approach to achieve this is by providing specific a priori guidance to the LLMs (i.e., prompting the LLM to choose from a set of norm-aligned actions) using techniques such as Retrieval Augmented Generation (RAG) [22]. The challenges in achieving this are discussed in Sect. 5.

While the discussion above only focused on normative LLM agent capabilities related to prior work in NLP (i.e., norm discovery, reasoning and conformance), there is an opportunity to explore the use of LLMs in achieving richer and nuanced normative capabilities for agents.

² This is somewhat different from norm discovery discussed in the previous sub-section, which aims to discover a set of all possible norms that may apply in a society, while identification refers to reasoning about specific norms that apply in a given context.

First, capabilities such as the ability to spread norms through communication and enforce a norm through punishment can be explored in depth. For example, a norm spreading agent should be able convey the norm in a convincing way (e.g., by providing evidence that the norm is in force, and the penalty for non-adherence), and the evidence can come from the LLM. More generally, this rich set of capabilities can be provided to an agent by enabling the LLM-agent to recognise the current phase of the norm in the norm life-cycle model (e.g., whether a norm is being created, identified, spread and enforced [12, 30, 39]).

Second, certain assumptions made by prior work can be relaxed. Prior work assumes that a sanction is observed in the form of a signal (a symbol [41]), and the question of where sanctions come from, their different types, and why agents sanction have remained unanswered. With the advent of LLMs, the task of reasoning about sanctions (e.g., seeking explanation) can be delegated to the LLM (after training with appropriate data). Thus, existing gaps in reasoning can be plugged using LLMs. *Third*, norm-relevant explanation can be offered by an LLM agent - e.g., *why does an agent perform action X and not Y in a given normative context?*. Norm-related explanation generation is an area which has been underexplored, and an LLM agent is well-poised to explain the reasons behind action selection and avoidance, corresponding to obligation and prohibition norms respectively. *Fourth*, there is limited prior work that has investigated the dual (slow and fast) nature of normative reasoning, which is routinely employed by humans [19]. The core idea is that some norms are applied fast, based on how often an agent employs it, such as offering a greeting, while some require norm identification from scratch. With the use of LLMs, agents will have the option to do fast reasoning based on its local memory, and consult an LLM for slow reasoning. This presents the opportunity to experiment with different application modes (i.e., when to cache well-known norms and when to consult the LLM), considering the performance goals of the system.

5 An Application of a Normative LLM Agent

We consider a childcare scenario that has been already considered for investigating normative reasoning [6, 54], where Rosie the robot cares for a child in the child's home. Figure 1 shows the interactions between the child and Rosie (an embodied normative LLM agent). In this scenario, the child who is less than three years old asks the robot to fetch a large pair of scissors so that he can cut paper and then colour each piece. The robot knows generic norms, as shown inside the thought bubble in the figure, but not all possible instantiations of these generic norms. It consults with the specific LLM model that has been trained on childcare scenarios. The four reasoning steps the agent pursues are shown on the right of Fig. 1. The robot consults its LLM module to first check whether there might be norm violations if it were to proceed with the request. Having found that two norms would be violated, it asks for an alternative solution that does not violate the norm, yet satisfies the request from the child. The solution from the LLM is provided by the agent as a response to the child. Once the response

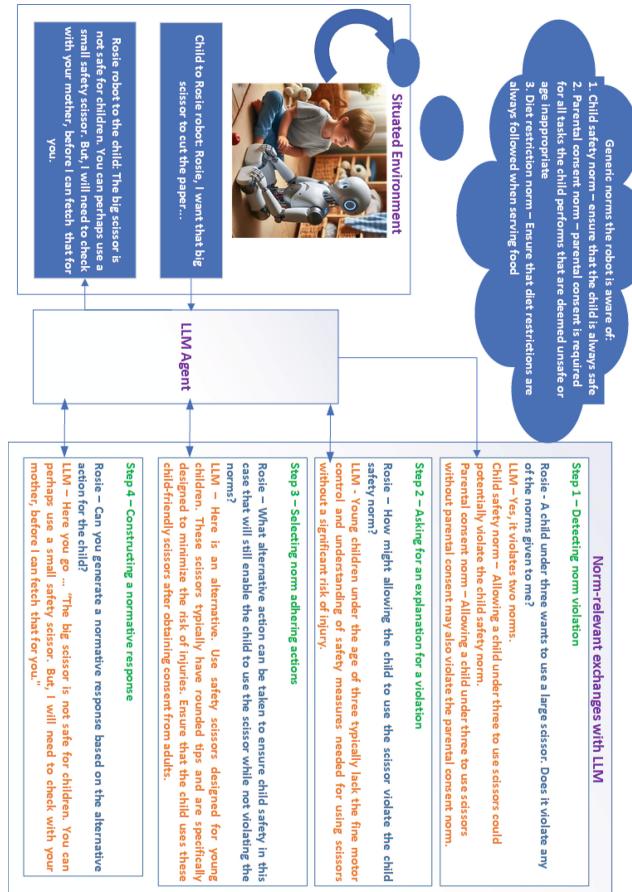


Fig. 1. Normative reasoning of a normative LLM agent in a child care scenario

is provided, the agent must invoke its planning module to perform the actions that correspond to what has been communicated to the child (i.e., seek consent from the child's mother to use safety scissors and then fetch the scissors and give them to the child). Having learnt the new specific instantiations of the generic norms, it stores them in its memory for future use. If these specific norms are violated in the future, the agent does need to consult the LLM.

As can be observed, this scenario involves several normative capabilities enabled by the LLM: norm identification through violation detection, seek norm explanation, recommendation of non-violating action and normative response generation. This scenario can be further extended. For example, if the robot was also responsible for entertaining and educating the child, e.g. by playing games, reading books together, then there would be culture specific norms that govern choices of books and games.

Application domains where a normative LLM-agent could prove useful include checking business contract compliance [14], policy design (e.g., based on the results of simulating norm emergence [36]), and designing virtual organisations where self-regulating agents take preventative or remedial actions when they detect norm conflicts [38]. Other application areas that could be considered include those discussed by the work of Singh et al. [43].

6 Summary and Challenges

This paper proposed to improve normative capabilities of LLMs and to use them in building normative LLM agents, which will enable richer natural language based normative competencies in agent societies. However, we identify five main challenges that need to be addressed to realize their full potential.

1. Retrofitting agent architectures: Integrating LLMs as a part of the decision-making cycle of an agent may require re-engineering of the existing agent frameworks such as the BDI architecture [35]. For example, LLMs produce natural language responses that need to be converted back to actions performed by an agent. Thus, in an agent's planning module, responses need some mapping function that link natural language text to actions (classes, functions etc.). A similar effort is needed to map an agent's perception and beliefs into natural language prompts (e.g. during normative reasoning).

There have been some prior work on converting textual observations into logic [7, 10, 53] and vice versa [5], which can be harnessed for this purpose. For example, the work of Chaudhury et al. [7] has proposed a neuro-symbolic approach where an agent learns to perform a specific action (e.g., whether to enter a room or not) based on symbolic abstractions from textual observations (i.e., description of the state of the room). However, these works have not considered the use of LLMs for this purpose.

2. Inherent capability limitations of current LLMs: As probabilistic models trained to predict the next token, LLMs are prone to hallucination and may produce inaccurate facts [17]. Many researchers have argued (and empirically shown) that LLMs are still far behind human reasoning capabilities [28, 52]. This suggests that a substantial effort needs to be employed to teach human-level normative reasoning capabilities to LLMs. Moreover, capability to obtain proper responses from an LLM requires a carefully generated prompt, which requires human expertise.

3. Resource constraints: LLMs are prohibitively expensive to train: they need large GPU clusters and training expands over days to months. Thus it is not practical for an average researcher to build custom LLMs from scratch. Even fine-tuning LLMs requires a computer with a GPU that is well-equipped with memory. Prompting proprietary LLMs such as ChatGPT incurs non-trivial amounts of time and money [32].

4. Data constraints: To fine-tune LLMs for specific agent scenarios, human-annotated domain data and also culture-specific data may be needed. The domain experts needed to curate domain-specific data may be costly. Obtaining

culture-specific data is even harder. Also, there is a significant linguistic disparity in the field of NLP [34], and the performance of LLMs for low-resource languages is far behind that of their high-resource counterparts [21].

5. Ethical issues: Using LLMs require careful consideration as a range of ethical issues need to be addressed including privacy, transparency, reproducibility, explainability, and potential data biases. In particular, using AI systems for moral reasoning has been criticized by some researchers [44]. Thus, integrating frameworks that enable systematic mitigation of ethical issues is required [48].

It is worth noting that NLP and LLM researchers are advancing in an unprecedented speed to address the LLM-related limitations. For example, solutions such as RAG [22] have been employed to overcome the problem of hallucination, and Low-rank Adaptation (LoRA) [16] has been very successful in reducing the memory requirements during LLM fine-tuning. Thus, we expect that most of the LLM-related challenges to be resolved to a greater extent in the near future. We invite MAS researchers to tackle challenges on the MAS side (e.g., creating integrated architectures) in order to harness the potential of LLMs to create socially-situated, expressive and norm-competent agents. Our proposal here complements the proposal of Wells and Snaith [50] on using LLMs and dialogue based systems to deliberate about norms.

In the short term, we suggest that researchers focus on specific agendas such as a) investigating how an LLM-agent can employ sanctions [46] and reputation mechanisms [37] to facilitate norm emergence, and b) exploring how norm entrepreneurship [2] can be facilitated by an LLM-agent, and how these norms may go on to become established norms, and then (potentially) superseded by other norms based on changing circumstances. The recent work of Ren et al. [36] demonstrates how many phases of the norm-life cycle model can be simulated using LLM-based normative agents, and their framework can be extended to explore the emergence of complex and conditional norms. These studies can demonstrate the potential for the LLM-agent to demonstrate real-life norm emergence processes.

Beyond creating normative LLM-agents which has been the focus of this paper (and there have been efforts in this direction [14, 15, 36]), LLMs can also be useful in enabling smoother functioning of a normative system where it can play the role of an overseer or a governor agent that monitors normative activities of an agent society. For example, LLMs can be useful for detecting the specific life-cycle phase of a norm (whether a norm is in the emergent phase, established phase or declining phase [42]) based on observing all interactions in an agent society. Additionally, LLMs can be used to synthesise and propose new norms that are more efficient than current norms [29], detect norm conflicts at the societal level and offer conflict resolution mechanisms [45]. We anticipate such developments will be beneficial for the creation and embedding of norm-capable agents (e.g., robots) in real world settings.

References

- Ammanabrolu, P., Jiang, L., Sap, M., Hajishirzi, H., Choi, Y.: Aligning to social norms and values in interactive narratives. In: Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 5994–6017 (2022)
- Anavankot, A.M., Cranefield, S., Savarimuthu, B.T.R.: NEMAS: norm entrepreneurship in multi-agent systems. *Systems* **12**(6), 187 (2024)
- Andreas, J.: Language models as agent models. In: Goldberg, Y., Kozareva, Z., Zhang, Y. (eds.) Findings of the Association for Computational Linguistics. EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7–11, 2022, pp. 5769–5779. Association for Computational Linguistics (2022). <https://doi.org/10.18653/v1/2022.findings-emnlp.423>
- Broersen, J., van der Torre, L.: Ten problems of deontic logic and normative reasoning in computer science. Lectures on Logic and Computation. ESSLLI 2010 Copenhagen, Denmark, August 2010, ESSLLI 2011, Ljubljana, Slovenia, August 2011, Selected Lecture Notes, pp. 55–88 (2012)
- Calò, E., van der Werf, E., Gatt, A., van Deemter, K.: Enhancing and evaluating the grammatical framework approach to logic-to-text generation. In: Proceedings of the 2nd Workshop on Natural Language Generation, Evaluation, and Metrics (GEM), pp. 148–171 (2022)
- Canavotto, I., Horty, J.: Piecemeal knowledge acquisition for computational normative reasoning. In: Proceedings of the 2022 AAAI/ACM Conference on AI, Ethics, and Society, pp. 171–180 (2022)
- Chaudhury, S., Sen, P., Ono, M., Kimura, D., Tatsumori, M., Munawar, A.: Neuro-symbolic approaches for text-based policy learning. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pp. 3073–3078 (2021)
- Choi, Y.: The curious case of commonsense intelligence. *Daedalus* **151**(2), 139–155 (2022)
- Emelin, D., Le Bras, R., Hwang, J.D., Forbes, M., Choi, Y.: Moral stories: situated reasoning about norms, intents, actions, and their consequences. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pp. 698–718 (2021)
- Ferraro, G., et al.: Automatic extraction of legal norms: evaluation of natural language processing tools. In: New Frontiers in Artificial Intelligence: JSAl-isAI International Workshops, JURISIN, AI-Biz, LENLS, Kansei-AI, Yokohama, Japan, November 10–12, 2019, Revised Selected Papers 10, pp. 64–81. Springer (2020)
- Forbes, M., Hwang, J.D., Shwartz, V., Sap, M., Choi, Y.: SOCIAL CHEMISTRY 101: learning to reason about social and moral norms. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 653–670 (2020)
- Frantz, C., Pigozzi, G.: Modeling norm dynamics in multiagent systems. *J. Appl. Log.* **5**(2), 491–564 (2018)
- Fung, Y., Chakrabarty, T., Guo, H., Rambow, O., Muresan, S., Ji, H.: NORM-SAGE: multi-lingual multi-cultural norm discovery from conversations on-the-fly. In: Bouamor, H., Pino, J., Bali, K. (eds.) Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pp. 15217–15230. Association for Computational Linguistics, Singapore (2023). <https://doi.org/10.18653/v1/2023.emnlp-main.941>

14. Haque, A., Singh, M.P.: Extracting norms from contracts via ChatGPT: opportunities and challenges. arXiv preprint [arXiv:2404.02269](https://arxiv.org/abs/2404.02269) (2024)
15. He, S., Ranathunga, S., Cranefield, S., Savarimuthu, B.T.R.: Norm violation detection in multi-agent systems using large language models: a pilot study. arXiv preprint [arXiv:2403.16517](https://arxiv.org/abs/2403.16517) (2024)
16. Hu, E.J., et al.: LoRA: low-rank adaptation of large language models. In: International Conference on Learning Representations (2022). <https://openreview.net/forum?id=nZeVKeeFYf9>
17. Huang, L., et al.: A survey on hallucination in large language models: principles, taxonomy, challenges, and open questions. arXiv preprint [arXiv:2311.05232](https://arxiv.org/abs/2311.05232) (2023)
18. Jiang, L., et al.: Can machines learn morality? the Delphi experiment. arXiv preprint [arXiv:2110.07574](https://arxiv.org/abs/2110.07574) (2021)
19. Kahneman, D.: Thinking, Fast and Slow: Farrar, Straus and Giroux (2017)
20. Kiehne, N., Kroll, H., Balke, W.T.: Contextualizing language models for norms diverging from social majority. In: Findings of the Association for Computational Linguistics: EMNLP 2022, pp. 4620–4633 (2022)
21. Lee, E.S., et al.: Pre-trained multilingual sequence-to-sequence models: a hope for low-resource language translation? In: Findings of the Association for Computational Linguistics: ACL 2022, pp. 58–67 (2022)
22. Lewis, P., et al.: Retrieval-augmented generation for knowledge-intensive NLP tasks. *Adv. Neural. Inf. Process. Syst.* **33**, 9459–9474 (2020)
23. Li, H., et al.: Theory of mind for multi-agent collaboration via large language models. In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pp. 180–192 (2023)
24. Li, O., Subramanian, M., Saakyan, A., CH-Wang, S., Muresan, S.: NORMDIAL: a comparable bilingual synthetic dialog dataset for modeling social norm adherence and violation. In: Bouamor, H., Pino, J., Bali, K. (eds.) Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 15732–15744. Association for Computational Linguistics, Singapore (2023). <https://doi.org/10.18653/v1/2023.emnlp-main.974>
25. López y López, F., Luck, M., d'Inverno, M.: Constraining autonomy through norms. In: Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems: Part 2, pp. 674–681 (2002)
26. Lourie, N., Le Bras, R., Choi, Y.: SCRUPLES: a corpus of community ethical judgments on 32,000 real-life anecdotes. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 13470–13479 (2021)
27. Meyer, J.J.C., Wieringa, R.J.: Deontic Logic in Computer Science: Normative System Specification. Wiley (1994)
28. Mialon, G., Fourrier, C., Wolf, T., LeCun, Y., Scialom, T.: GAIA: a benchmark for general AI assistants. In: The Twelfth International Conference on Learning Representations (2024). <https://openreview.net/forum?id=fibxvalvhs3>
29. Morales, J., López-Sánchez, M., Rodríguez-Aguilar, J.A., Wooldridge, M.J., Vasconcelos, W.W.: Minimality and simplicity in the on-line automated synthesis of normative systems. In: Bazzan, A.L.C., Huhns, M.N., Lomuscio, A., Scerri, P. (eds.) International Conference on Autonomous Agents and Multi-Agent Systems, AAMAS '14, Paris, France, May 5-9, 2014, pp. 109–116. IFAAMAS/ACM (2014). <http://dl.acm.org/citation.cfm?id=2615752>
30. Morris-Martin, A., De Vos, M., Padget, J.: Norm emergence in multiagent systems: a viewpoint paper. *Auton. Agent. Multi-Agent Syst.* **33**(6), 706–749 (2019). <https://doi.org/10.1007/s10458-019-09422-0>

31. Nahian, M.S.A., Frazier, S., Riedl, M., Harrison, B.: Learning norms from stories: a prior for value aligned agents. In: Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society, pp. 124–130 (2020)
32. Park, J.S., O'Brien, J.C., Cai, C.J., Morris, M.R., Liang, P., Bernstein, M.S.: Generative agents: interactive simulacra of human behavior. In: 36th Annual ACM Symposium on User Interface Software and Technology (UIST '23). ACM (2023)
33. Pyatkin, V., et al.: ClarifyDelphi: reinforced clarification questions with defeasibility rewards for social and moral situations. In: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 11253–11271 (2023)
34. Ranathunga, S., de Silva, N.: Some languages are more equal than others: probing deeper into the linguistic disparity in the NLP world. In: Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing, pp. 823–848 (2022)
35. Rao, A.S., Georgeff, M.P.: Modeling rational agents within a BDI-architecture. In: Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning, pp. 473–484. Morgan Kaufmann Publishers Inc. (1991)
36. Ren, S., Cui, Z., Song, R., Wang, Z., Hu, S.: Emergence of social norms in large language model-based agent societies. arXiv preprint [arXiv:2403.08251](https://arxiv.org/abs/2403.08251) (2024)
37. Santos, F.P., Santos, F.C., Pacheco, J.M.: Social norm complexity and past reputations in the evolution of cooperation. *Nature* **555**(7695), 242–245 (2018)
38. Santos, J.S., Zahn, J.O., Silvestre, E.A., Silva, V.T., Vasconcelos, W.W.: Detection and resolution of normative conflicts in multi-agent systems: a literature survey. *Auton. Agent. Multi-Agent Syst.* **31**(6), 1236–1282 (2017). <https://doi.org/10.1007/s10458-017-9362-z>
39. Savarimuthu, B.T.R., Cranefield, S.: Norm creation, spreading and emergence: a survey of simulation models of norms in multi-agent systems. *Multiagent Grid Syst.* **7**(1), 21–54 (2011)
40. Savarimuthu, B.T.R., Cranefield, S., Purvis, M.A., Purvis, M.K.: Obligation norm identification in agent societies. *J. Artif. Soc. Soc. Simul.* **13**(4), 3 (2010)
41. Savarimuthu, B.T.R., Cranefield, S., Purvis, M.A., Purvis, M.K.: Identifying prohibition norms in agent societies. *Artif. Intell. Law* **21**, 1–46 (2013)
42. Savarimuthu, B.T.R., Padget, J., Purvis, M.A.: Social norm recommendation for virtual agent societies. In: Boella, G., Elkind, E., Savarimuthu, B.T.R., Dignum, F., Purvis, M.K. (eds.) PRIMA 2013. LNCS (LNAI), vol. 8291, pp. 308–323. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-44927-7_21
43. Singh, M.P., et al.: The uses of norms. In: Andriagetto, G., Governatori, G., Noriega, P., van der Torre, L.W.N. (eds.) Normative Multi-Agent Systems, Dagstuhl Follow-Ups, vol. 4, pp. 191–229. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany (2013)
44. Talat, Z., Blix, H., Valvoda, J., Ganesh, M.I., Cotterell, R., Williams, A.: On the machine learning of ethical judgments from natural language. In: Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 769–779 (2022)
45. Vasconcelos, W.W., Kollingbaum, M.J., Norman, T.J.: Normative conflict resolution in multi-agent systems. *Auton. Agent. Multi-Agent Syst.* **19**, 124–152 (2009)
46. Villatoro, D., Sen, S., Sabater-Mir, J.: Of social norms and sanctioning: a game theoretical overview. *Int. J. Agent Technol. Syst. (IJATS)* **2**(1), 1–15 (2010)

47. Wang, L., et al.: A survey on large language model based autonomous agents. *Front. Comp. Sci.* **18**(6), 1–26 (2024)
48. Watkins, R.: Guidance for researchers and peer-reviewers on the ethical use of large language models (LLMs) in scientific research workflows. *AI Ethics*, 1–6 (2023)
49. Wei, J., et al.: Chain-of-thought prompting elicits reasoning in large language models. *Adv. Neural. Inf. Process. Syst.* **35**, 24824–24837 (2022)
50. Wells, S., Snaith, M.: On the role of dialogue models in the age of large language models - extended abstract. In: Grasso, F., Green, N.L., Schneider, J., Wells, S. (eds.) *Proceedings of the 23nd Workshop on Computational Models of Natural Argument (CMNA 2023)*, Virtual Event, December 3rd, 2023. CEUR Workshop Proceedings, vol. 3614, pp. 49–51. CEUR-WS.org (2023)
51. Xi, Z., et al.: The rise and potential of large language model based agents: a survey. arXiv preprint [arXiv:2309.07864](https://arxiv.org/abs/2309.07864) (2023)
52. Yiu, E., Kosoy, E., Gopnik, A.: Transmission versus truth, imitation versus innovation: what children can do that large language and language-and-vision models cannot (yet). *Perspect. Psychol. Sci.*, p. 17456916231201401 (2023)
53. Zhan, N., Sarkadi, S., Such, J.: Privacy-enhanced personal assistants based on dialogues and case similarity. In: European Conference on Artificial Intelligence. IOS Press (2023)
54. Zhang, Z.: Research on child care robot and the influence on children. In: 2022 5th International Conference on Humanities Education and Social Sciences (ICHESS 2022), pp. 225–233. Atlantis Press (2022)
55. Ziems, C., Dwivedi-Yu, J., Wang, Y.C., Halevy, A., Yang, D.: NORMBANK: a knowledge bank of situational social norms. In: Rogers, A., Boyd-Graber, J., Okazaki, N. (eds.) *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7756–7776. Association for Computational Linguistics, Toronto, Canada (2023). <https://doi.org/10.18653/v1/2023.acl-long.429>
56. Ziems, C., Yu, J., Wang, Y.C., Halevy, A., Yang, D.: The MORAL INTEGRITY CORPUS: a benchmark for ethical dialogue systems. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3755–3773 (2022)



Norm Violation Detection in Multi-Agent Systems Using Large Language Models

A Pilot Study

Shawn He¹, Surangika Ranathunga¹⁽⁾, Stephen Cranefield²⁽⁾,
and Bastin Tony Roy Savarimuthu²⁽⁾

¹ Massey University, Auckland, New Zealand

Shawn.He.1@uni.massey.ac.nz, S.Ranathunga@massey.ac.nz

² University of Otago, Dunedin, New Zealand

{stephen.cranefield,tony.savarimuthu}@otago.ac.nz

Abstract. Norms are an important component of the social fabric of society by prescribing expected behaviour. In Multi-Agent Systems (MAS), agents interacting within a society are equipped to possess social capabilities such as reasoning about norms and trust. Norms have long been of interest within the Normative Multi-Agent Systems community with researchers studying topics such as norm emergence, norm violation detection and sanctioning. However, these studies have some limitations: they are often limited to simple domains, norms have been represented using a variety of representations with no standard approach emerging, and the symbolic reasoning mechanisms generally used may suffer from a lack of extensibility and robustness. In contrast, Large Language Models (LLMs) offer opportunities to discover and reason about norms across a large range of social situations. This paper evaluates the capability of LLMs to detecting norm violations. Based on simulated data from 80 stories in a household context, with varying complexities, we investigated whether 10 norms are violated. For our evaluations we first obtained the ground truth from three human evaluators for each story. Then, the majority result was compared against the results from three well-known LLM models (Llama 2 7B, Mixtral 7B and ChatGPT-4). Our results show the promise of ChatGPT-4 for detecting norm violations, with Mixtral some distance behind. Also, we identify areas where these models perform poorly and discuss implications for future work.

Keywords: Norms · Norm violation detection · Large language models · ChatGPT

1 Introduction

Norms prescribe behaviours that are expected by agents in a society. Norms are pivotal to establishing social order within a society as they facilitate coordination and cooperation between participating agents [2]. A software agent that is *norm capable* should be able to identify existing norms, detect situations where norms

are violated and plan to either comply with norms or choose to violate them with an understanding of the likely consequences. These capabilities (and more) are the subject of the research field of normative multi-agent systems.

This paper focuses on norm violation detection, also known as norm monitoring. The importance of this ability in human society is evidenced by our innate ability to detect norm violations. Brain studies in humans using EEG recordings show a negative deflection of electrocortical potential that occurs at around 400ms, which shows they have recognised the violation of expectations, including those that arise from norms [18]. Without a robust mechanism to detect norm violations, effective norm enforcement (e.g., by sanctioning) is not possible, undermining the spreading and emergence of norms and reducing coordination between agents.

Prior normative MAS research has a number of limitations. Norms are encoded and formalised in a variety of representations with no standard approach emerging. We believe this slows progress in the field as advances made by one researcher are not easily exploited by others. Also, norm representations and reasoning mechanisms are often developed for and evaluated in simple static problem domains. Furthermore they are predominantly based on the symbolic AI paradigm, and may therefore lack robustness when deployed in domains outside their initial design constraints [12]. Thus, current techniques may have limited applicability to the types of domain-independent sociotechnical system that are enabled today by smartphones, social media networks and personal assistants (which users are rapidly becoming accustomed to since the advent of ChatGPT).

Large Language Models (LLMs) offer promise in addressing these issues since LLMs can work with norms expressed in natural language rather than committing to one of the many norm representations in the literature. Recently, Natural Language Processing (NLP) researchers have started to investigate LLM capabilities for normative reasoning. For example, pre-trained language models already have been shown to possess some notion of norms, and they can be further trained with publicly available norm-related datasets (discussed in Sect. 2.2 below). However, in that research, normative reasoning has been done on text such as chatbot conversations, narratives or situation descriptions. In contrast, in an MAS, an agent must deliberate on norm violation based on a sequence of events it perceives from its environment. How an LLM would be capable of determining norm violations from such an event sequence has not been investigated before. This work aims to bridge this gap.

In this work, we evaluated the norm identification capability of commonly used public and proprietary LLMs. We prompted them with a sequence of events that occurred in a simulated environment related to a household setup, along with a set of norms defined in that environment. The LLMs were asked to determine whether each norm was violated or not, in a given scenario. Comparing these results with human-generated answers we found that the capabilities of different LLMs for norm violation detection varied, but the best model, ChatGPT-4, identified norm violations with an accuracy of 86%. However, the performance depended on the type of norm.

The rest of the paper is organised as follows. Section 2 discusses prior work related to norm violation detection in MAS and NLP research. Section 3 discusses our experimental setup and Sect. 4 provides the results. We provide a detailed discussion of our results in Sect. 5. Finally, Sect. 6 concludes the paper.

2 Prior Work

2.1 MAS Research on Norm Violation Detection

In the MAS literature, norms are expressed using a variety of notations such as conditional rules, temporal logic constraints and action description languages (such as the event calculus). A short survey is given by Cranefield et al. [6, Sec. 2]. Various abstract models of norms have also been proposed that specify separate components of a norm such as the type (obligation, etc.), triggering condition, the normative constraint, expiration condition and sanction. There are also a host of ad hoc domain-specific notations for norms.

Given this diversity, it is not surprising that many mechanisms for norm violation detection have been proposed. These include those that arise naturally from the underlying formalisms such as forward-chaining rule engines, temporal projection in the event calculus and temporal logic model checking as well as the design of separate entities to monitor norms (see Dastani et al. for a survey of the latter approach [7]).

2.2 NLP Research for Norm Violation Detection

In NLP, instead of norm violation detection, researchers focus on predicting ‘moral judgement’ in a given situation [10, 25]. The common approach to training such a prediction system is to fine-tune a pre-trained language model with *situation_description*, *moral_judgement* pairs. Here, the *moral_judgement* refers to whether an action was normative or not. This kind of model has been commonly implemented as a text classification system, where a textual description of an action gets classified as normative or not [13].

More recent work has produced models that are capable of more fine-grained reasoning. For example, NORMBANK [25] categorises a given situation as *ok*, *expected* or *unexpected*. The publicly available Delphi system¹ [10] responds with one of the following judgements: *rude*, *ok*, *bad*, *wrong*, *expected*, *shouldn’t*, when prompted with a situation (expressed in a single sentence). While Delphi carries out normative reasoning in isolation with no reference to surrounding context, ClarifyDelphi [16] is capable of asking questions to elicit salient context. While all this research made use of pre-trained language models, none has investigated the capabilities of the recently released LLMs, which we are using in this research. Moreover, this NLP research focused on norm violation present in textual descriptions such as narratives or chatbot output. In contrast, we are focusing on an event sequence received by an agent.

¹ <https://delphi.allenai.org/>.

2.3 LLM Agents

The field of LLM agents is a rapidly advancing thread of research that considers how intelligent software agents can leverage the abilities of LLMs to help them make decisions. Recent survey papers by Wang et al. [20] and Xi et al. [21] give an overview of this field. A key benefit is that an LLM can provide an agent with the ability to operate in a wide range of environments unlike traditional approaches to agent development that are usually restricted to specific problem domains.

There has been prior research on how LLM-based agents can interact in a society [21], including mechanisms that enable agents to exhibit “believable individual and emergent social behaviors” [15]. However, prior work has not considered how LLMs could assist software agents with social reasoning, including norm awareness and monitoring.

A companion paper [19] discusses the above points in more detail and proposes a number of potential paths to extending LLM agents to become *normative* LLM agents.

3 Methodology

3.1 Simulating an Agent Environment

We used the grammar of Nematzadeh et al. [14] to create stories that describe a sequence of actions performed by a set of agents in a pre-defined environment. It has been used to test Theory of Mind (ToM) capabilities of LLMs [1, 14]. This grammar is defined in terms of a set of *entities* (i.e., objects that may have *properties*, and agents) and a set of *predicates*. A predicate takes an entity as its subject or object. A predicate can be an action that an agent carries out (e.g. move, enter). Each story is generated in such a manner that it constitutes of a set of tasks. In each task, several agents (usually two) carry out different actions such as entering or exiting rooms or moving things around. In addition to the agent actions, the stories may contain random events that represent noise.

We introduced the following modifications to the implementation of Nematzadeh et al. [14] to generate more complex scenarios:

1. In their work, tasks in a story are arranged one after the other, meaning that the agent would perceive actions of one agent before perceiving the other’s. However, this simple scenario is not what would happen in an agent environment—an agent can perceive actions of different agents in an interleaved fashion. In order to achieve this realistic scenario, we interchanged actions of different agents using a topological sort algorithm [11].
2. Their stories have only one random event (*phone rang*). We added two more events (*kettle whistled*, and *cat meowed*), and randomly added them as noise to the generated stories.
3. We generate stories by varying the number of tasks from one to four. This results in stories that have the number of events ranging from 6 to 30.

We used the same set of agents and objects that were defined in the original implementation. Altogether, we generated 80 stories, where every 20 stories had a different number of tasks (i.e., 20 stories each for one, two, three and four tasks). Figure 1 shows a sample story (with four tasks) generated from the modified version of the implementation of Nematzadeh et al. [14]. Figure 2 shows the number of occurrences of each event across the 80 stories. This figure confirms that most stories are unique. To be specific, out of a total of 1317 events across the 80 stories, 367 are unique events and only 219 are duplicated events.

```

Alexander entered the sunroom.
Alexander moved the clock to the black suitcase.
Peter entered the sunroom.
Peter exited the sunroom.
Peter entered the study.
Peter exited the study.
Emily entered the sunroom.
Emily moved the fork to the blue bucket.
Emily exited the sunroom.
Phone rang.
Alexander exited the sunroom.
Emily entered the study.
Emily moved the tomato to the white crate.
Emily exited the study.
Peter entered the basement.
The hat is in the black bottle.
The fork is in the black suitcase.
Phone rang.
The clock is in the blue bucket.
Peter moved the hat to the red carpet.
Peter exited the basement.
Emily entered the sunroom.
Emily exited the sunroom.
Emily entered the basement.
The tomato is in the green refrigerator.
Emily exited the basement.

```

Fig. 1. An example story used in our experiment

3.2 Defining Norms

The list of objects and agents in the simulation suggest that Nematzadeh et al. [14] have considered events taking place in a home. Therefore we defined 10 norms that can be realistically applied in a household setup. We defined a home with four agents: mother (Emily) and father (Peter) with two young children (Alexander, who is a 5-year-old boy and Ann, who is a 7-year-old girl). The set of norms we used is shown in Fig. 3. We defined two types of norms: generic

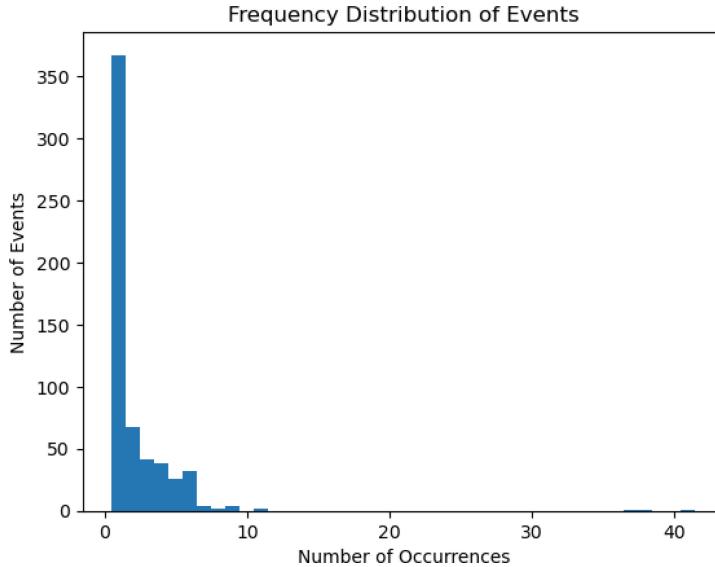


Fig. 2. Frequency distribution of events across the 80 stories

norms (6 norms: 1 to 6) and role-based norms (4 norms: 7 to 10). They are further divided into Obligation Norms (ON) and Prohibition Norms (PN) as indicated at the end of each norm in Fig. 3.

3.3 Prompting LLMs and Human Evaluation

We used ChatGPT-4 as our proprietary LLM and Llama 2 (Llama-2-7b-chat-hf) and Mixtral (Mixtral-8 × 7B-Instruct-v0.1) as the open source LLMs. The two open source models were selected considering their promising performance despite the smaller model size. After a preliminary set of experiments and discussions had within the research team over several rounds, the prompt shown in Fig. 4 was used in the final experiments. The figure shows the ChatGPT-4 version. Prompts for other two models are more or less the same, except for the model-specific tags. Each of the 80 stories we generated were provided as prompts to each of the LLMs along with the ten norms investigated for each story.

We also carried out a human evaluation, in order to obtain the ground truth. The same 80 stories were provided to external human evaluators, along with the scenario description and the set of norms. For each story and each of the norms, they were asked to decide among the following three possibilities: Norm violated, norm not violated, or not enough information to give a judgement. Each story was evaluated by three human annotators. Altogether we used nine annotators for the evaluation. Six were Computer Science undergraduates and three were recent Computer Science graduates. Two were females and the rest were males.

Generic norms:

- 1 - You should not enter an occupied bathroom. (PN)
- 2 - Vegetables and fruits (except bananas) should be kept in the refrigerator. (ON)
- 3 - Before using the staircase, you should wait until it is free. (ON)
- 4 - If sharp objects are present in the household, they should be kept out of reach of children. (ON)
- 5 - If the phone rings, the person closest to the phone must pick it up. (ON)
- 6 - There should be no fruits and vegetables containing citric acid inside the house. (PN)

Role-based norms:

- 7 - Kids should not enter crawl spaces. (PN)
- 8 - You should not enter a couple's bedroom without knocking. (PN)
- 9 - Kids should be supervised inside the workshop all the time. (ON)
- 10 - Father should not be disturbed while he is in his study room. (PN)

Fig. 3. The norms used in our experiment

Inter-annotator agreement for the human evaluation was conducted. The Fleiss Kappa value was 0.6826, which indicates a substantial agreement.

4 Results

From the human annotated data, we used majority voting to select the decision corresponding to a norm with respect to a given story. We evaluated model performance using human annotation as the ground truth. However, despite being given the option to decide between three options, yes/no/cannot be determined, humans have always selected one of the two options: yes or no. Therefore, if a model has the output ‘cannot be determined’, we map that output to ‘no’.

Table 1 shows the performance (accuracy) of each LLM against human judgement. The result indicates that ChatGPT-4 has the overall best performance, with 86%. Mixtral, the second best technique, falls behind by a slight margin. The result also indicates that the model performance is not consistent across norms - i.e., for some norms it performs much better (e.g., for Mixtral, 97.5% for norm 9 when compared to 28.75% for norm 2). Therefore we conducted further analysis based on norm category (whether the norm is generic or role-based) and also based on the type of the norm considered (whether the norm is a prohibition norm or an obligation norm). Result in Table 2 shows that all the models struggle to detect generic norms than role-based norms. Result in Table 3 shows that all the three models were able to detect prohibition norms better than obligation norms (82% vs. 68%).

We also calculated the disagreement between ground truth and each of the models in determining norm violation. The result for the worst-performing model (Llama 2) is shown in Fig. 5 in the form of a confusion matrix. This model seems

There is a family that adheres to certain norms, listed below from Norm 1 to 10. The family consists of Emily (wife), Peter (husband), and their children, Alexander (5-year-old boy) and Ann (7-year-old girl).

As a sophisticated assistant, you are tasked with analyzing a story related to this family to identify any violations of the given norms. Your goal is to systematically break down the analysis for each norm, ensuring a thorough evaluation based on the events listed in the story.

The family's story unfolds with the whole family initially being in the living room, alongside a phone.

There is no other person in the house.

At the beginning of the story, no norms are violated.

If an event is not mentioned in the story, then it did not occur.

Your job is to evaluate each event in the story against the below given norms to identify any violations.

Norms to Evaluate:
 {norms}

Below is the story about the family's events.

Story:

{story}

Your evaluation should consider each norm individually. For each norm, provide a clear response in the following format:

- Norm 1
- Violation: [Choose one of three options given below]
- No: The norm was not violated or its irrelevance was given the story.
- Cannot be determined: There isn't clear evidence to determine whether the norm was violated or not.
- Yes: The norm was violated.
- Reasoning: [Explain your rationale here, referencing specific events in the story as evidence.]

Please proceed with your analysis, ensuring a thoughtful and comprehensive examination of each norm as it relates to the story's events.

Fig. 4. Our prompt template for ChatGPT-4

to struggle in identifying norm violations, more than determining that there is no violation. In other words, out of the instances where a human identified a norm violation, 81% of the time the model has not been able to identify this violation. On the contrary, when the human determines a non-violation, the model's decision differs only by 31.8%. Figure 6 shows the confusion matrix for the best performing model ChatGPT-4. For this model, out of the instances where a human identified a norm violation, only 51% of the time the model has not been able to identify this violation. When the human determines a

Table 1. Overall accuracy (in percentages) for the three models for the 10 norms (N1 to N10)

Model	N1	N2	N3	N4	N5	N6	N7	N8	N9	N10	Average
Llama 2	43.75	21.25	92.50	37.50	70.00	58.75	95.00	17.50	93.75	51.25	58.0
Mixtral	97.50	28.75	90.00	85.00	68.75	93.75	96.25	96.25	97.50	51.25	81.0
ChatGPT-4	100.00	62.50	96.25	87.50	70.00	92.50	93.75	96.25	93.75	71.25	86.0

Table 2. Accuracy results (in percentages) based on norm category

Norm category	ChatGPT-4	Mixtral	Llama 2	Average
Role-based norm	94.06	95.94	66.25	85.42
Generic norm	81.25	70.21	52.71	68.05

Table 3. Accuracy results (in percentages) based on norm type

Norm category	ChatGPT-4	Mixtral	Llama 2	Average
Prohibition Norm (PN)	90.25	87.25	68.50	82.00
Obligation Norm (ON)	82.50	73.75	47.75	68.00

non-violation, the model decision differs only by 4%. This further highlights the superior results of ChatGPT-4.

5 Discussion

We make several observations based on the results. First, ChatGPT-4 performed better than the other models with an accuracy of 86%. This result is not surprising given that GPT Turbo, the model used in ChatGPT-4 topped the leaderboard for general reasoning at the time of writing [8]. However, we note that the result obtained is far from being perfect. Software agents performing actions based on the result even from this best performing LLM might still be performing norm-violating actions based on the advice received, and hence may be subject to sanctions. However, note that in this work we simply prompted the LLMs in a zero-shot manner (i.e. without showing any examples to the LLM, either via in-context learning or fine-tuning). Therefore, we suggest the creation of fine-tuned or RAG (Retrieval Augmented Generation)-based models to further improve norm detection ability of LLMs. Use of advanced reasoning techniques such as Tree-of-Thought prompting [22] may further improve outcomes. More broadly, norm competence of LLMs [19], of which norm violation detection is only one aspect, should be improved.

Second, we observed that for a few norms all the LLMs performed poorly (Llama 2 7B and Mixtral 7B, more so than ChatGPT-4). For example, violations of norm 2 about keeping vegetables and fruits in the refrigerator have been

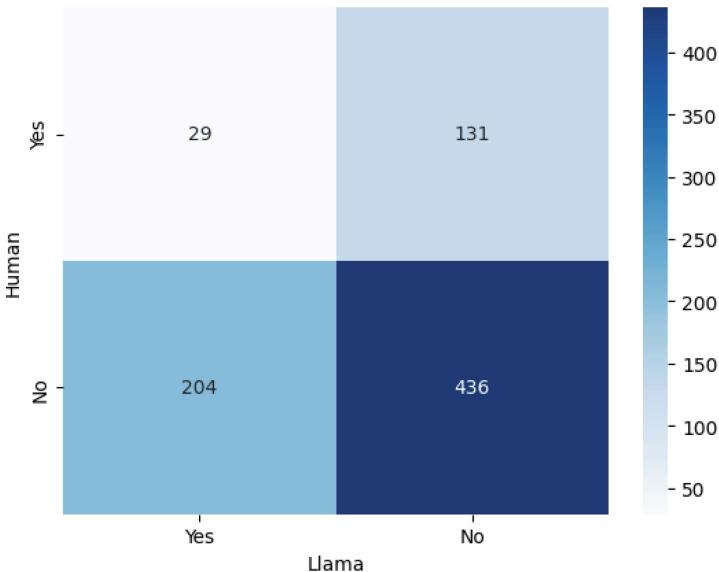


Fig. 5. Confusion Matrix for Llama 2 result against ground truth

identified poorly. Perhaps, it could be that this norm may have been in conflict with common knowledge reasoning where only green vegetables are kept in the refrigerator. In some cases, the relatively poor result of Llama 2 may have been because of the non-availability of data. For example, norm 6 requires the ability to determine whether a vegetable or fruit contains citric acid (e.g., beetroot in one story). However, it may be difficult for the LLM to identify such information. There are limited sources on the Internet that may have this information so it may not be in the LLM's training data. Also, as we did not provide a floor plan of the house to the LLM, its performance may have been impacted. For example, we noticed that in certain instances, Llama 2 made incorrect inferences related to various rooms of the house (e.g. the kitchen) having crawlspaces.

Third, we found that prohibition norm violations were more successfully identified by LLMs than obligation norm violations. This may be because there may be more instances of prohibition norms than obligation norms in the dataset to pre-train the LLM, since prohibitions are more commonly stated than obligations.

Fourth, our work shows that certain LLMs were better at identifying non-violation of norms in a story than identifying violations (e.g., for Llama 2 this difference is about 50%). This may be because the dataset in which these LLMs are pre-trained are not balanced (i.e., imbalance between violations and non-violations). This suggests these models need to be trained with more data with violations.

Fifth, while most of the human-coded data contained only yes and no classifications, indicating violations and non-violations respectively, two of the models

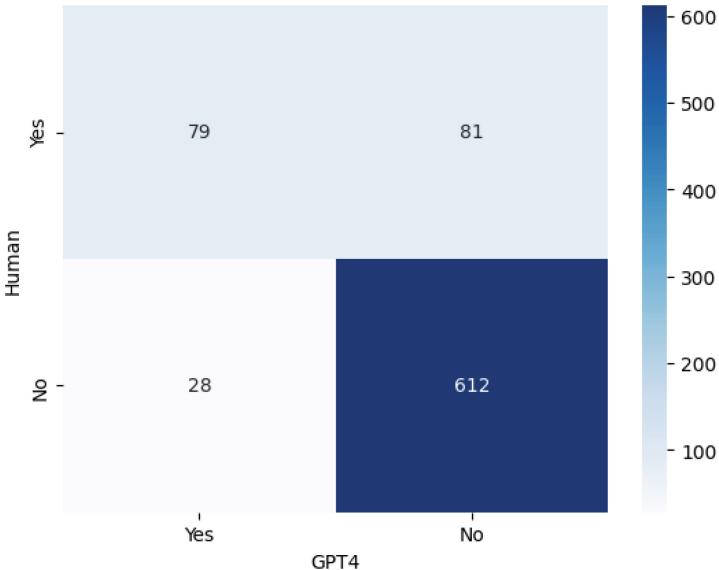


Fig. 6. Confusion Matrix for ChatGPT-4 result against ground truth

(Mixtral and ChatGPT-4) contained results that were more nuanced. In many cases ChatGPT-4 returned ‘cannot be determined’—an option that was available to the humans also, which implies there is not enough evidence to determine whether a norm was violated. Also Mixtral returned ‘not applicable’ (which could mean a conditional norm was not triggered and therefore not violated). For example, for the norm that prohibits the second person entering the staircase when someone is already on it, if the story does not involve a staircase, it returned the result ‘not applicable’ since the staircase was not mentioned. This was a surprising result since we had provided only *yes*, *no* and *cannot be determined* as the three options for the language model. In future work, we can consider providing four answer options: *yes*, *no*, *not applicable* and *cannot be determined*.

Sixth, we observed that the norm identification capability of an LLM depends on the prompt. In fact, prompt brittleness has been highlighted as a limitation of LLMs [23]. As mentioned earlier, we had to experiment with multiple prompts before reaching the most promising prompt. For example, during our initial experiments, we identified that the LLM is not capable of determining the gender of a character from their name. Therefore, we explicitly had to mention who is the husband, wife and children etc. are. Also, the LLMs seemed to get confused with the location of characters. Therefore we had to explicitly mention that all the characters are in the living room at the beginning of the story. Some LLMs tended to assume that the story may contain other characters that have not been mentioned in the story. Therefore we explicitly had to mention that there are no other characters in the house. In order to reduce LLM hallucination, we

had to explicitly state that if an event is not mentioned in a story, that event did not occur. Additionally, explicitly stating the expected format of the LLM output was also needed.

Seventh, while this work considered both generic and role-specific norms, these are relatively simple norms. In the future, we intend to experiment with more complex norms where there are several conditions involving multiple entities.

Eighth, this work does not propose how norm violation detection information is used by agents for decision-making (e.g., to punish other agents), which can form the focus of future work.

Finally, this work is a pilot study on investigating normative reasoning focusing on one aspect, norm violation detection. We encourage other researchers to pursue other aspects of normative reasoning as suggested in our companion discussion paper [19]. Additionally, this paper provides a comparison of norm violation detection ability of LLMs against human evaluations. In the future, we will investigate how well LLMs perform when compared to logic-based approaches. Having said that there has been prior work that have considered neuro-symbolic approaches where textual descriptions of norms are converted to logic [5, 9, 24] and vice-versa [4] which can be employed here. Also, LLMs themselves can be used for this purpose (i.e., convert text to logic [3]). For our current experiments, we used a simple simulation. However, there have already been studies to deploy autonomous agents in more complex simulations [17]. Therefore in the future, we intend to experiment with more complex simulations.

6 Conclusion

This work aimed at investigating the effectiveness of LLMs in detecting norm violations based on 80 stories. Of the three LLMs considered in the work (Llama 2 7B, Mixtral 7B and ChatGPT-4), we observed that ChatGPT-4 offered the most promise, with 86% accuracy. We also observed the LLMs detected violations of prohibition norms with higher accuracy than those of obligation norms. Additionally, violations of role-based norms were more accurately identified than those of generic norms. Also, LLMs were better at identifying when norms weren't violated than when they were violated. These results demonstrate that, while LLMs show promise in norm violation detection, there is scope for creating better models for detecting norm violations. We believe such enhanced models will be beneficial for recommending actions to be taken by an agent within a multi-agent system, in order to create norm-conforming agent societies where potential norm violations are effectively identified and punished. In this study, we focused only on norm violation detection, which is an example of the normative reasoning capability of an agent. However, we believe that LLMs can be leveraged to achieve other norm competencies in agents, such as norm discovery and norm conformance.

Acknowledgements. We would like to thank the following individuals who volunteered for the human evaluation: Dilan Nayanajith, Paul Janson, Sanjeepan Sivapiran,

Kumuthu Athukorala, Yasith Heshan, Rumal Costa, Thevin Senath, Harshani Bandara and Nimesh Ariyarathne.

This research was funded by a research grant from the College of Sciences, Massey University, New Zealand.

References

1. Alechina, N., Halpern, J.Y., Kash, I.A., Logan, B.: Incentive-compatible mechanisms for norm monitoring in open multi-agent systems. *J. Artif. Intell. Res.* **62**, 433–458 (2018)
2. Bicchieri, C., Muldoon, R., Sontuoso, A., et al.: Social norms. In: *The Stanford Encyclopedia of Philosophy* (2014)
3. Borazjanizadeh, N., Piantadosi, S.T.: Reliable reasoning beyond natural language. arXiv preprint [arXiv:2407.11373](https://arxiv.org/abs/2407.11373) (2024)
4. Calò, E., van der Werf, E., Gatt, A., van Deemter, K.: Enhancing and evaluating the grammatical framework approach to logic-to-text generation. In: *Proceedings of the 2nd Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*, pp. 148–171 (2022)
5. Chaudhury, S., Sen, P., Ono, M., Kimura, D., Tatsumori, M., Munawar, A.: Neuro-symbolic approaches for text-based policy learning. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3073–3078 (2021)
6. Cranefield, S., Winikoff, M., Vasconcelos, W.W.: Modelling and monitoring interdependent expectations. In: Cranefield, S., van Riemsdijk, M.B., Vázquez-Salceda, J., Noriega, P. (eds.) *Coordination, Organizations, Institutions, and Norms in Agent Systems VII*, No. 7254, pp. 149–166. LNCS, Springer (2011). https://doi.org/10.1007/978-3-642-35545-5_9
7. Dastani, M., Torroni, P., Yorke-Smith, N.: Monitoring norms: a multidisciplinary perspective. *Knowl. Eng. Rev.* **33**, e25 (2018). <https://doi.org/10.1017/S0269888918000267>
8. Fan, L., Hua, W., Li, L., Ling, H., Zhang, Y., Hemphill, L.: NPHardEval: dynamic benchmark on reasoning ability of large language models via complexity classes. arXiv preprint [arXiv:2312.14890](https://arxiv.org/abs/2312.14890) (2023)
9. Ferraro, G., et al.: Automatic extraction of legal norms: evaluation of natural language processing tools. In: *New Frontiers in Artificial Intelligence: JSAI-isAI International Workshops, JURISIN, AI-Biz, LENLS, Kansei-AI, Yokohama, Japan, November 10–12, 2019, Revised Selected Papers 10*, pp. 64–81. Springer (2020)
10. Jiang, L., et al.: Can machines learn morality? the Delphi experiment. arXiv preprint [arXiv:2110.07574](https://arxiv.org/abs/2110.07574) (2021)
11. Kahn, A.: Topological sorting of large networks. *Commun. ACM* **5** (1962)
12. Minsky, M.L.: Logical versus analogical or symbolic versus connectionist or neat versus scruffy. *AI Mag.* **12**(2), 34–51 (1991). <https://doi.org/10.1609/aimag.v12i2.894>
13. Nahian, M.S.A., Frazier, S., Riedl, M., Harrison, B.: Learning norms from stories: a prior for value aligned agents. In: *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pp. 124–130 (2020)
14. Nematzadeh, A., Burns, K., Grant, E., Gopnik, A., Griffiths, T.: Evaluating theory of mind in question answering. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2392–2400 (2018)

15. Park, J.S., O'Brien, J.C., Cai, C.J., Morris, M.R., Liang, P., Bernstein, M.S.: Generative agents: interactive simulacra of human behavior. In: 36th Annual ACM Symposium on User Interface Software and Technology (UIST '23). ACM (2023)
16. Pyatkin, V., et al.: ClarifyDelphi: reinforced clarification questions with defeasibility rewards for social and moral situations. In: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 11253–11271 (2023)
17. Ranathunga, S., Cranefield, S., Purvis, M.: Interfacing a cognitive agent platform with a virtual world: a case study using second life. In: The 10th International Conference on Autonomous Agents and Multiagent Systems, vol. 3, pp. 1181–1182 (2011)
18. Salvador, C.E., Mu, Y., Gelfand, M.J., Kitayama, S.: When norm violations are spontaneously detected: an electrocortical investigation. *Soc. Cogn. Affect. Neurosci.* **15**(3), 319–327 (2020)
19. Savarimuthu, B.T.R., Ranathunga, S., Cranefield, S.: Harnessing the power of LLMs for normative reasoning in mass. arXiv preprint [arXiv:2403.16524](https://arxiv.org/abs/2403.16524) (2024)
20. Wang, L., et al.: A survey on large language model based autonomous agents. *Front. Comp. Sci.* **18**(6), 186345 (2024)
21. Xi, Z., et al.: The rise and potential of large language model based agents: a survey. arXiv preprint [arXiv:2309.07864](https://arxiv.org/abs/2309.07864) (2023)
22. Yao, S., et al.: Tree of thoughts: deliberate problem solving with large language models. *Adv. Neural Inf. Process. Syst.* **36** (2024)
23. Zamfirescu-Pereira, J., Wong, R.Y., Hartmann, B., Yang, Q.: Why Johnny can't prompt: how non-AI experts try (and fail) to design LLM prompts. In: Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems, pp. 1–21 (2023)
24. Zhan, N., Sarkadi, S., Such, J.: Privacy-enhanced personal assistants based on dialogues and case similarity. In: European Conference on Artificial Intelligence. IOS Press (2023)
25. Ziems, C., Dwivedi-Yu, J., Wang, Y.C., Halevy, A., Yang, D.: NORMBANK: a knowledge bank of situational social norms. In: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 7756–7776 (2023)

Author Index

A

Alon, Nitay 41

B

Boissier, Olivier 79, 100

C

Cranefield, Stephen 132, 146

D

Dennis, Louise A. 3

F

Fisher, Michael 3

H

Hafiene, Nesrine 100

Haque, Amanul 115

He, Shawn 146

Hübner, Jomi F. 79

M

Mertzani, Asimina 57

Mirsky, Reuth 41

N

Nakar, Yonatan 41

Nardin, Luis G. 79, 100

O

Osman, Nardine 32

P

Pitt, Jeremy 57

R

Rachum, Ram 41

Ranathunga, Surangika 132, 146

S

Sarkadi, Stefan 57

Savarimuthu, Bastin Tony Roy 132, 146

Scott, Matthew 57

Singh, Munindar P. 115

Smit, Ciske 57

Stachura-Zurek, Dorota 15

T

Tomlinson, Bill 41

W

Wyner, Adam 15

Y

Yan, Elena 79

Z

Zurek, Tomasz 15