

Epistemic Planning

Edited by

Chitta Baral¹, Thomas Bolander², Hans van Ditmarsch³, and
Sheila McIlraith⁴

1 Arizona State University – Tempe, US, chitta@asu.edu

2 Technical University of Denmark – Lyngby, DK, tobo@dtu.dk

3 LORIA, CNRS – Univ. of Lorraine – Nancy, FR, hans.van-ditmarsch@loria.fr

4 University of Toronto, CA, sheila@cs.toronto.edu

Abstract

The seminar Epistemic Planning brought together the research communities of *Dynamic Epistemic Logic*, *Knowledge Representation and Reasoning*, and *Automated Planning* to address fundamental problems on the topic of epistemic planning. In the context of this seminar, dynamic epistemic logic investigates the formal semantics of communication and communicative actions, knowledge representation and reasoning focuses on theories of action and change, and automated planning investigates computational techniques and tools to generate plans. The original goals of the seminar were to develop benchmarks for epistemic planning, to explore the relationship between knowledge and belief in multi-agent epistemic planning, to develop models of agency and capability in epistemic planning and to explore action types and their representations (these originally separate goals were merged during the seminar), and finally to identify practical tools and resources. An additional goal explored during the workshop was the correspondence between planning problems and games.

Seminar June 5–9, 2017 – <http://www.dagstuhl.de/17231>

1998 ACM Subject Classification F.4.1 Mathematical Logic, I.2.4 Knowledge Representation Formalisms and Methods, I.2.8 Problem Solving, Control Methods, and Search, I.2.11 Distributed Artificial Intelligence

Keywords and phrases Automated Planning, Knowledge Representation and Reasoning, Reasoning About Actions, Dynamic Epistemic Logic, Multi-Agent Systems

Digital Object Identifier 10.4230/DagRep.7.6.1

1 Executive Summary

Chitta Baral

Thomas Bolander

Hans van Ditmarsch

Sheila McIlraith

License © Creative Commons BY 3.0 Unported license
© Chitta Baral, Thomas Bolander, Hans van Ditmarsch, and Sheila McIlraith

This seminar brought together three largely independent research communities: Dynamic Epistemic Logic (DEL), Knowledge Representation and Reasoning (KR&R, subsequently KR) and Automated Planning. All three communities have a tradition of investigating the interaction between dynamical systems and epistemic states, but with a different focus and by different means. In the context of this seminar, despite occasional overlap, DEL has mainly investigated the formal semantics of communication and communicative actions,



Except where otherwise noted, content of this report is licensed under a Creative Commons BY 3.0 Unported license

Epistemic Planning, *Dagstuhl Reports*, Vol. 7, Issue 06, pp. 1–47

Editors: Chitta Baral, Thomas Bolander, Sheila McIlraith, and Hans van Ditmarsch



Dagstuhl Reports

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

KR&R has mainly focussed on the theories of action and change, and AP has mainly focussed on computational techniques to automatically generate plans. This seminar aimed to encourage and nurture increasing synergies between these three strong and largely independent research communities leading to frameworks for Epistemic Planning: planning with epistemic states, actions, and goals. The seminar succeeded in strengthening and broadening the cross-disciplinary research community that emerged after a prior Dagstuhl Seminar on the subject, entitled “Planning with Epistemic Goals”, that was held in January 2014 (seminar number 14032). This follow-up seminar led to a better understanding and articulation of commonalities, synergies, and deficits between the DEL, KR and Planning communities.

The main components of the seminar were tutorial talks and group work. There were four tutorials, by Andreas Herzig, Bernhard Nebel, Tran Cao Son and Ramaswamy Ramanujam. The four tutorial talks presented epistemic planning from the perspective of four different research communities. Andreas Herzig presented epistemic planning from a knowledge representation perspective, Bernhard Nebel from a classical planning perspective, Tran Cao Son from a theories of action and change perspective, and Ramaswamy Ramanujam from a distributed systems and temporal logic perspective. Abstracts for the four tutorials are included in Section 3.

The group work was performed in five separate working groups, each addressing a distinct important objective in epistemic planning of shared interest to the DEL, KR and Planning communities. Specifically, the five groups worked on (1) *Developing Benchmarks for Epistemic Planning*, (2) *Exploring Action Types and their Representations*, (3) *Exploring the Relations Between Knowledge and Belief in Multi-Agent Epistemic Planning*, (4) *Practical Tools, Resources and Computational Techniques*, and (5) *Correspondence Between Planning Problems and Games*. These group themes are briefly described below, and the outcome of the work in each group is documented in Section 4.

1. **Developing Benchmarks for Epistemic Planning.** In the planning community benchmarks are common, but in the epistemic planning community they are not. The overall goal of the group working on this theme was to formulate a list of ten benchmark epistemic planning problems, in view of setting goals that can evolve into competitions. In particular, the focus was on targeting planning problems that are truly epistemic, meaning problems in which the epistemic dimension – knowledge and ignorance – cannot easily be disregarded.

Guiding questions for the work in this group were: What problems help define and circumscribe what we are studying? What are specific tasks that motivate this area of study, e.g. epistemic planning, protocol synthesis, automated diagnosis, verification, and communication? What problems can help drive future research in developing formalisms and implementing systems of epistemic planning? What are the features of the relevant planning problems in terms of knowledge vs. belief, single- vs. multi-agent, communicative vs. sensing vs. ontic actions, deterministic vs. non-deterministic actions, etc. How do we evaluate “hardness” of problems, e.g. in terms of level of nesting of belief/knowledge, types of actions, size of problem, scalability, and quality of solution. Do benchmarks for some of these problems already exist?

2. **Exploring Action Types and their Representations.** It is important to identify and broaden the list of action types relevant to epistemic planning. In terms of communicative actions we can for instance at least distinguish between announcements, questions, requests and instructions. How these actions are best represented is also an important issue. It should be explored whether formalizations in dynamic epistemic logic are appropriate for planning or whether a more simplified way of representing multi-agent actions is needed.

The overall goal of the group working on this theme was to identify, classify and possibly broaden the list of action types relevant to epistemic planning, as well as to explore formalisms for representing these action types.

Guiding questions for the work in this group were: What are the relevant distinctions between types of actions, e.g. epistemic vs. ontic, deterministic vs. non-deterministic vs. probabilistic, instantaneous vs. durative, sensing vs. announcements, degree of observability (public, private, semi-private), etc. What formalisms can support expressing and distinguishing between these action types (e.g. DEL, Situation Calculus, Knowledge-based Programs)? How are multi-agent actions represented, in particular how are conflicts between concurrently occurring actions specified, and how is observability of concurrent actions specified in terms of the observability of the constituting actions?

3. **Exploring the Relations Between Knowledge and Belief in Multi-Agent Epistemic Planning.** In multi-agent planning an important problem is that knowledge may turn into false belief for some agents after a partially observable action has taken place. This is a problem for several formalisms for epistemic planning, e.g. dynamic epistemic logic, since agents might not be able to recover from false beliefs. It relates to the general issues of devising appropriate formalisms for doxastic planning (treating beliefs instead of knowledge) and how to deal with belief revision in such settings. The overall goal of the work in this group was to identify the theoretical and computational challenges in planning with knowledge vs. planning with belief; when one or the other is appropriate, or both are needed.

Guiding questions for the work in this group were: How are knowledge and beliefs represented and distinguished from representations of the actual world? How do we formally handle that knowledge may turn into false belief after a partially observable action has occurred? What are the relevant formalisms for planning with knowledge and/or belief and what are their theoretical and computational properties? How do we deal with belief revision in planning? Are there specific types of interesting goals for epistemic planning? For example, in planning with beliefs, goals can be about making some agents have false beliefs. This necessitates formalizing false-belief tasks, lying and deception.

4. **Practical Tools, Resources and Computational Techniques.** The goal of the group working on this theme was to identify practical tools and resources that facilitate the development and experimental evaluation of automated techniques for epistemic planning.

Guiding questions for the work in this group were: What tools and computational techniques already exist in epistemic planning? What are the models and formulas used? What are the shortcomings and challenges of these tools and computational techniques? Are there tools or computational techniques from other communities that we are not availing ourselves of to the fullest extent (e.g., for the planning people model checking)? What are the trade-offs between different tools? What are the computational complexities of the different approaches and under different assumptions.

5. **Correspondence Between Planning Problems and Games.** This working group was introduced as an additional discussion topic during the seminar, since several participants found it very relevant and important to epistemic planning. The issue is that (epistemic) game theory studies many of the same problems as (epistemic) planning, but mainly by separate research communities using separate vocabularies. The goal of the group working on this theme was to establish formal connections between the area of automated planning and the area of game theory.

4 **17231 – Epistemic Planning**

Guiding questions for the work in this group were: What are the formalisms, tools and results from game theory relevant to automated planning? Symmetrically, what are the formalisms, tools and results from automated planning relevant to game theory? Can problems formulated in one of the settings easily be translated into the other? What is gained and lost in such translations?

2 Table of Contents

Executive Summary

Chitta Baral, Thomas Bolander, Hans van Ditmarsch, and Sheila McIlraith 1

Overview of Talks

Epistemic Planning: A Knowledge Representation Perspective

Andreas Herzig 6

Epistemic Planning Based on DEL for Researchers Working on Classical Planning

Bernhard Nebel 6

KRR and Epistemic Planning: Specification and Implementation Issues

Tran Cao Son 6

Distributed Presentations of Multi-Agent Systems

Ramaswamy Ramanujam 7

Working groups

Developing Benchmarks for Epistemic Planning

Malvin Gattinger, Fillippos Kominis, Jérôme Lang, Robert Mattmüller, Tim Miller, Ron Petrick, François Schwarzentruber, Bruno Zanuttini 7

Exploring Action Types and their Representations

Yves Lespérance, Louwe B. Kuijer, Nina Gierasimczuk, Barteld Kooi, Michael Thielscher, Ivan José Varzinczak, Thomas Bolander, Andrés Occhipinti Liberman, Tran Cao Son, Yongmei Liu, Hans van Ditmarsch 20

Exploring the Relations Between Knowledge and Belief in Multi-Agent Epistemic Planning

Chitta Baral, Hans van Ditmarsch, Jan van Eijck, Esra Erdem, Andreas Herzig, Mathias Justesen, Yongmei Liu, Richard Scherl, Tran Cao Son 23

Practical Tools, Resources and Computational Techniques

Tristan Charrier, Sophie Pinchinat, Vaishak Belle, Thorsten Engesser, Torsten Schaub, Malte Helmert, Bastien Maubert, Jens Claßen, Ioannis Kokkinis, Sunil Easaw Simon, Robert Mattmüller 28

Correspondence Between Planning Problems and Games

Guillaume Aucher, Tim French, Kai Li, Sheila McIlraith, Bernhard Nebel, Ramaswamy Ramanujam, Yanjing Wang 36

Participants 47

3 Overview of Talks

3.1 Epistemic Planning: A Knowledge Representation Perspective

Andreas Herzig (Paul Sabatier University, Toulouse, FR)

License  Creative Commons BY 3.0 Unported license
© Andreas Herzig

I start by a brief historical review of knowledge representation approaches to planning. I then argue that the integration of reasoning about knowledge requires a thorough analysis of the main concepts and models that are involved, viz. initial situation, goal, and action laws. While epistemic logic provides a satisfactory account of the former two, the representation of action laws deserves a closer look. I focus on the dynamic epistemic logic account in terms of event models and point out some conceptual and practical difficulties, including a mismatch between action tokens as modelled in the former on the one hand, and action types as needed for action laws on the other.

3.2 Epistemic Planning Based on DEL for Researchers Working on Classical Planning

Bernhard Nebel (Universität Freiburg, DE)

License  Creative Commons BY 3.0 Unported license
© Bernhard Nebel
Joint work of Bernhard Nebel, Thorsten Engesser, Robert Mattmüller

After a quick introduction of what states and actions in DEL are, we show how classical planning, fully observable non-deterministic (FOND) planning, conformant planning, and partially observable non-deterministic (POND) planning can be implemented using DEL. On top of POND planning, we introduce the notions of multi-agent planning, non-uniform observability, and implicitly coordinated plans, and demonstrate how these notions are modelled in DEL. The final part of the tutorial highlights strength and weaknesses of DEL planning by presenting three small examples: Nano-Hanabi, multi-agent path finding with destination uncertainty, and secretly travelling to Dagstuhl.

3.3 KRR and Epistemic Planning: Specification and Implementation Issues

Tran Cao Son (New Mexico State University, US)

License  Creative Commons BY 3.0 Unported license
© Tran Cao Son

In this tutorial, we discuss specification and implementation issues of epistemic planning. We start with a short motivation of the epistemic planning problem. We then discuss the issues and how they have been addressed in single agent environments. Specifically, we describe an action language formalism and present possible approaches for planning with incomplete information and sensing actions. We highlight the advantage of approximation-based approaches to epistemic planning in single agent domains. We continue with a discussion of various issues of the problem formulation in multi agent domains. We present

an action language, $m\mathcal{A}^+$, and the notion of finitary S5-theory whose combination with $m\mathcal{A}^+$ could be used for the specification of epistemic planning problems. Additionally, we show that a large class of $m\mathcal{A}^+$ theories maintains the KD45-property of the initial epistemic state, a desirable feature when one needs to distinguish between knowledge and beliefs. We conclude with a list of research issues related to the formalization and implementation of epistemic planning systems.

3.4 Distributed Presentations of Multi-Agent Systems

Ramaswamy Ramanujam (Institute of Mathematical Sciences, Chennai, IN)


License  Creative Commons BY 3.0 Unported license
© Ramaswamy Ramanujam

In Dynamic Epistemic Logics (DEL) and Epistemic Temporal Logics (ETL) the modelling of multi-agent systems tends to be ‘from above’, describing epistemic ascriptions to agents by the designer. In distributed protocols we look for ‘inside’ or local presentations of agents, where agents have partial views of system evolution and acquire knowledge about others by explicit mechanisms such as shared variables or message passing. We present a framework of systems of epistemic finite state automata that explicitly acquire and update knowledge tokens specified by a separate (pre-ordered) alphabet. These tokens can be thought of as finite representations of sets of knowledge formulas (ordered by implication). We also present an ETL built on local propositions where structural composition ensures that epistemic assertions by agents about others refer to knowledge constructed by agents based on their partial views. For every consistent formula we construct a system of automata that realizes the models of the formula (as a set of partial orders). There are refinement methodologies and theorem prover based synthesis of such behaviours as well. In general, given a global presentation of an epistemic multi-agent system, decomposing it into a system of communicating agents seems to be interesting and challenging.

4 Working groups

4.1 Developing Benchmarks for Epistemic Planning

Malvin Gattinger, Fillippos Kominis, Jérôme Lang, Robert Mattmüller, Tim Miller (coordinator), Ron Petrick, François Schwarzentruber, Bruno Zanuttini

License  Creative Commons BY 3.0 Unported license
© Malvin Gattinger, Fillippos Kominis, Jérôme Lang, Robert Mattmüller, Tim Miller, Ron Petrick, François Schwarzentruber, Bruno Zanuttini

We distinguish between *long term applications*, which can hardly be expressed by simplified examples, *examples of problems*, that can more easily be formalized and are good candidates for developing benchmarks, and *benchmarks* stricto sensu, that are parameterized families of simplified examples (such as puzzles) and that are primarily meant to test and compare the different approaches for verifying and generating plans.

4.1.1 Applications

We classify here potential long-term applications of epistemic planning.

One major application is the design of *cooperative agents*, who share the same utility function and that are generally programmed in a centralized fashion (their plans are designed off-line, but of course the online execution of these plans will be distributed and perhaps asynchronous). As an example: multirobot exploration of a nuclear plant (modelled by decentralized POMDP ([20])).

At the other extremity of this spectrum, we have *purely adversarial games*, where the utilities of the players sum to zero. This is typically the field of game playing; knowledge plays a role as soon as incomplete knowledge of the state of the game and/or communicative actions are present in the game. Two paradigmatic examples: *poker* and *bridge*.

Between these two extreme cases, we have classes of problems which are neither purely cooperative nor purely adversarial. A class of applications of crucial importance is the class of *assistive agent problems*, where an autonomous agent has to reason about the mental state of a human agent and assist them in performing various tasks, such as: robots helping disabled or elderly persons, tutorial systems and serious games, dialogue/interaction management (including interactive generation of explanations). The problems in this rich class of applications are “almost cooperative”, in the sense that the utility of the helper should generally increase with the utility of the helped agent, but there are exceptions, especially when the helped agent is not fully rational (e.g., patients with Alzheimer). Another class of applications is where the environment has agents that can be partitioned to groups and one group of agents may collaboratively try to achieve something while keeping agents in the other groups in the dark or even making them have false beliefs.

Another important class of applications of epistemic planning is *security games*, where an autonomous agent or a collection of autonomous agents have to reason about the mental states of an adversary and to prevent it from performing malevolent actions, such as organization of airport security, phishing detection, reasoning about a malware that could detect a sandbox (e.g. WannaCry), etc.

The difference with games in the ‘game playing’ category is that there is a safety/maintenance goal, agents are not symmetric, the airport’s actions are only checking properties, etc.

4.1.2 Problems as Benchmarks

In this section, we provide a list of examples that could be potentially transformed into benchmarks. Indeed, the applications presented in the previous section are too informal to lead to benchmarks. However we keep the same classification.

Purely cooperative problems

Let us start with a list of purely cooperative problems, that is, in which there is no adversary.

- Tiger [49]
- Hanabi (recently proven to be NP-hard [6])
- Active Muddy Children (one active child, asking questions with the goal of knowing its own state) [57, p. 152]
- Multirobot exploration of a nuclear plant (modeled by decentralized POMDP [20])
- 100 Prisoners and a Lightbulb [108]
- Word Rooms (see [57])
- Corridor (see [57])
- Gossip (see below and [46])
- Grapevine (combination of Corridor and Gossip, see [74])
- Dining Cryptographers [30]

Security games

In the following examples, several agents cooperate for an adversary not to perform bad actions and not to obtain certain knowledge.

- Russian cards
- Scotland Yard
- Sandbox and detection thereof (e.g. WannaCry)
- Organization of airport security

There are key differences with general games. First, the goal is essentially a safety/maintenance condition. Second, agents are not symmetric in the sense that they do not have the same role at all. For instance, in the organization of an airport security, the airport's actions are checking properties only while other agents have physical actions.

Adversary game playing

The following list provides examples where agents are programmed to react against other agents in a non-cooperative imperfect information game.

- Clue (Cluedo)
- Mobile battleships
- Military strategy
- Magic tricks [86]

Multi-agent deception

This scenario can be illustrated with respect to a simple example¹ taken from the storybook, *Winnie the Pooh*. At one point in the story, Rabbit, Pooh and Piglet are forming a joint plan to distract Kanga's attention so that they can substitute Piglet for Roo so that Kanga will not notice that Roo is gone so that Rabbit can run off with Roo.

In the above example, the goal of Rabbit, Pooh and Piglet is not only to substitute Piglet for Roo but also to keep Kanga in the dark whereby Kanga has false beliefs.

Features to evaluate

Each problem is characterized by the following features:

- *Knowledge vs. belief*: Is the problem one of true knowledge, of (possibly incorrect) belief, or both? Typically, knowledge involves indistinguishability between possible situations modeled as equivalence relations over possible worlds, one for each agent. On the contrary, (possibly incorrect) belief is modeled by means of so-called KD45 relation (see [31]). Note also that belief may require complex mechanisms such as belief revision [42].
- *Single vs. multi-agent*: Is the problem a multi- or single-agent problem?
- *Communication vs. sensing vs. ontic*: Are actions ontic (effect only the environment), sensing (only sense the environment and have no effect), or complex form of communication (effect the mental states of other, etc. as for action models of dynamic epistemic logic [9]) actions, or a mix of these?
- *Determinism/reliability of communication*: Are actions deterministic or non-deterministic, in which the latter includes concepts such as unreliable sending and communication?
- *Epistemic goals*: Are goals epistemic, ontic, or both?

¹ A version of this example was presented by Ernie Davis in his invited talk at the ACS 2017 conference.

- *Cooperative/competitive*: Is the problem cooperative, adversarial/competitive, or both; or potentially neither (other agents as ‘noise’ in the environment)?
- *Benchmarks*: Are there already existing benchmarks that can be accessed?

4.1.3 Application-Inspired Benchmarks

4.1.3.1 Gossip

Description: The Gossip problem is a communication problem in which agents must share secrets with each other; thus goals are epistemic. The actions available to each agent are to share a secret with another agent, as in a telephone call. In the simplest version, the goal is that all agents know each others’ secrets. However, more sophisticated goals are possible:

- Everybody knows all secrets
- All secrets are common knowledge
- Deception (of some subset of agents): some agents believe the opposite of some secrets
- Privacy-preserving: Some agents know all secrets, while some can only know part of the secrets. For example, a single source agent knows a set of secrets and wants a destination agent to know all of these, but can communicate only via other agents, to whom the secret should not be revealed. Thus, it must communicate some secrets via some paths and other secrets via other paths.

Similarly, the *nature* of the secrets can be more expressive than simple propositions, such as:

- s_i (just the secret)
- $K_j s_i$ (secret + epistemic info)
- histories (secret + when I got it + from whom)
- how much does the agent say per turn? (one secret, all secrets, some secrets, everything that agent knows)
- unilateral or not?
- negotiation of secrets? values of secrets?

Different variations of the gossip protocols have been investigated [109, 3, 106].

Motivation: The Gossip problem has the nice properties that one would like for a running example in a paper, but can be easily made more challenging by increasing the number of agents, number of secrets, nature of the goals, nature of the secrets, and the structure of the network in which the secrets are shared. Thus, it forms an excellent benchmark problem.

Further, variants of the problem, parameters, and complexity have been well studied [34, 50, 107].



■ **Figure 1** Gossip problem (Taken from Faustine Maffre’s thesis.) [67]

Features:

- *Knowledge vs. belief*: Either.
- *Single vs. multi-agent*: Multi-agent.
- *Communication vs. sensing vs. ontic*: Communication.
- *Determinism/reliability of communication*: Reliable.
- *Epistemic Goals*: Yes.
- *Cooperative/competitive*: Cooperative, but adversarial versions possible.
- *(De)Centralized*: Centralized.
- *Benchmarks*: Different implementations available, such as at Toulouse/IRIT² and Melbourne/Toronto³.
- *Evaluation of hardness*: Cooper et al. [34] for optimality, levels of nesting of belief and knowledge, size of problems, and scalability.

4.1.3.2 Security and patrolling games

Description: In security games, the aim is to find a strategy for a defender to deter attackers from reaching their targets. Usually, the defender’s strategy is a mixed strategy which is a best response to what the defender believes to be the attacker’s strategy. Patrolling games is a specific subfamily of security games where actions consist in performing some security checks at some points at each time step. Depending on the type of game, the nature of the attackers and the cost involved, the techniques used and the rationality assumptions about the attacker largely differ. Some examples of high-stake security games are protecting airports against terrorist attacks [95] ; some middle-stake games include “green security games”, such as protecting the environment against smuggling and poaching; low-stake domains include deterring fare evasion in public transport.

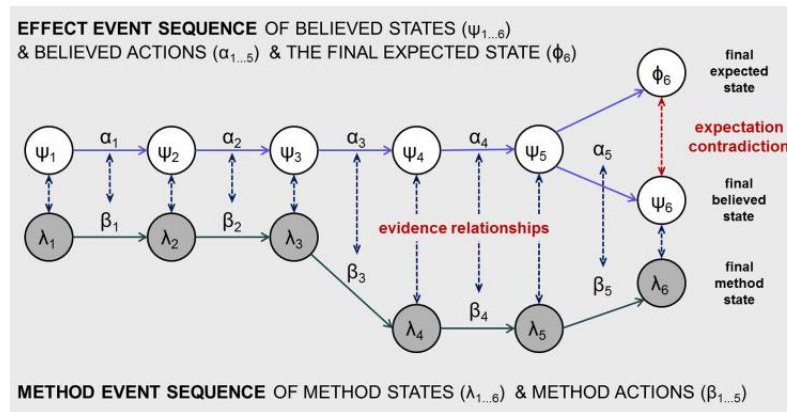
Motivation: Programming a defender’s strategy can be seen as an adversarial epistemic planning problem. The defender has to reason about the opponent’s intended actions and about his higher-order beliefs (his beliefs about the defender’s strategy and about the defender’s beliefs about his own strategy). The base version consists in casting the problem as a Stackelberg game: a defender must defend a set of targets using a limited amount of resources whereas the attacker can observe and learn the (mixed) strategy of the defender. In this base model, the payoffs of the two agents depend only on the target attacked and whether or not it was protected by the defender. More sophisticated versions have been developed, including versions where the attacker can only partially observe the attacker’s actions and/or where his rationality is bounded.

Features:

- *Knowledge vs. belief*: Belief (with deception).
- *Single vs. multi-agent*: Multi-agent.
- *Communication vs. sensing vs. ontic*: Sensing and ontic.
- *Determinism/reliability of communication*: Either.
- *Epistemic Goals*: Usually not, but possible (maintenance goal to always know the position of the attacker, for instance).
- *(De)Centralized*: Centralized.

² See <https://github.com/FaustineMaffre/GossipProblem-PDDL-generator>.

³ See <https://bitbucket.org/haz/pdkb-planning>.



■ **Figure 2** Parallel processes involved in conjuring/magic.

- *Cooperative/competitive*: Competitive (adversarial).
- *Benchmarks*: Existing problem description from Tambe [95]; see also follow-up work of *Sim.Patrol: Establishing a Testbed for Multi-agent Patrolling*.

4.1.3.3 Conjuring Magic Tricks

Description: The aim of these problems is to generate an element of ‘surprise’ in a group observers. A magic trick consists of a set of actions, in which the state of the world is partially observable to some or all of the observers. The aim of the magician is to transition the real state of the world into some state that is different from that expected by some of the observers. Figure 2 presents an abstraction of this, in which two parallel processes occur: the effect sequence, which is what the observer believes has occurred, and the method sequence, which is what the magician knows has occurred. At the end, there are three states: the *expected* state of the observer, the *believed* state of the observer, and the *method* state, which is the real state of the affairs only known by the magician.

As a first step, we could aim to discover the correct sequences for existing magic tricks, by specifying action models describing these, and using an epistemic planner to try to discover the correct set of moves. As a second step, epistemic planning could be used to find new tricks using existing or modified actions models.

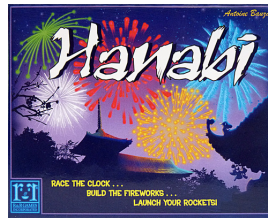
Smith et al. [86] define a formalism of conjuring using propositional dynamic logic (PDL), as well as examples of some tricks using this formalism.

Motivation: Being able to derive magic tricks, even in a limited setting, is a somewhat creative task, and something that explicitly requires a Theory of Mind of the observers. In addition, the goals are *temporal epistemic goals* – the true state of the world must remain hidden until such time that the observer’s perspective is incorrect.

Higher-order beliefs also play a part in some tricks; for example, in tricks involving more than two people, in which the magician has a helper for the trick, and thus needs to track not only the perspectives that the helper and ‘victim’ have of the world, but also of each other and the magician.

Further to this, generating surprise is key to many tactical scenarios, such as game playing, military planning, and story telling.

These are interesting and fun problems to solve, so this may be a good way to attract masters and internship students to work on epistemic planning.



■ **Figure 3** Cover of Hanabi, © Asmodée éditions.

Features:

- *Knowledge vs. belief*: Belief (with deception).
- *Single vs. multi-agent*: Multi-agent.
- *Communication vs. sensing vs. ontic*: All three.
- *Determinism*: Non-deterministic, or probabilistic (uncertain whether participants will infer certain beliefs).
- *(De)Centralized*: Decentralized.
- *Epistemic Goals*: Temporal epistemic goals.
- *Collaborative/competitive*: Both (collaborative helper, competitive).
- *Benchmarks*: Existing model of conjuring from Smith et al. [86].

4.1.3.4 Hanabi

Description: Hanabi is a fully collaborative card game⁴. The game consists of cards each with a given colour and number. Each player has a hand drawn from the deck, which they cannot see, while they can see the others' hand. The cooperative goal is to stack up cards of each colour in sequential order. For this, on their turn, players can either (i) play a card from their hand (and draw a new one), (ii) discard a card and draw a new one, or (iii) give a clue to another player about what cards they have. A clue is the indication of all cards of a given colour, or of all cards of a given number (e.g., “the blue cards in your hand are the first and fourth ones”).

When a card is played, it must be either the first of its colour played by any player, or the one numbered immediately after the last one played. Otherwise, the move is a mistake; if a predefined number of mistakes is reached, the players lose the game. Moreover, a maximal number of clues to be given by anyone is predefined. Giving a clue uses one, and discarding a card gives one back. When the deck is empty (and players have not lost the game before), the score is computed as the sum of the highest number in each colour.

Motivation: Hanabi requires the player to maintain knowledge about their own cards (e.g., “I know that my second card is a 2”, “I know that I don’t have any red card”, etc.) and to reason about what information is needed by their mates, so as to inform them while using as few tokens as possible. When playing it for real, players most often agree on a strategy about which information they will give to each other (e.g., “Always give information about what next card to play”), which can be seen as centralized planning (with decentralized execution of the strategy agreed upon). Of course, it also makes sense to plan in a decentralized or distributed fashion, which corresponds in real life to players starting a game without knowing each other.

⁴ Created by Antoine Bauza and published in 2010.

An interesting feature of Hanabi is that it has a formal and simple description, easily amenable to description in an input language for epistemic planners. Moreover, by adjusting the numbers of players, colors, and numbers in each, it can be used for analyzing the scalability of algorithms. Other features can also be controlled; for instance, loosely speaking, the number of clue tokens is related to the number of communication actions which can be taken in a row, and the number of mistakes allowed is related to whether we need “cautious” plans.

Finally, the goal can be controlled by framing it as an optimization problem (maximize the score while not losing) or one of reachability (empty the deck before losing). Orthogonally, the goal can be defined taking into account the probabilities involved (maximize the expected score, maximize the expected time-to-defeat, etc.) or looking for a winning strategy in the adversarial sense.

Features:

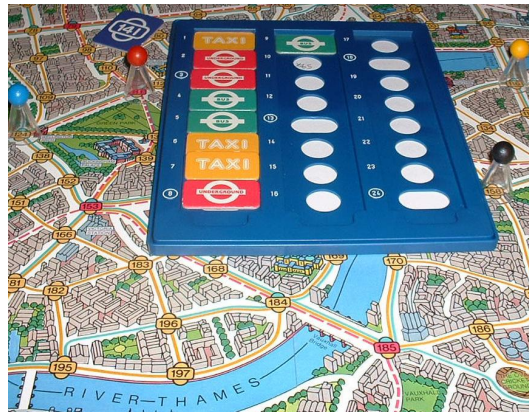
- *Knowledge vs. belief*: Knowledge in the first place (“what am I/is my teammate sure of?”), but probabilistic knowledge (“I know that there is a 33 % chance that my second card is blue”) or beliefs (“since all but one red card have been played, let me bet that I have none”) make sense.
- *Single vs. multi-agent*: Multi-agent.⁵
- *Communication vs. sensing vs. ontic*: Communication (mostly) and ontic (but these are mostly dictated by the information which the player has).
- *Determinism*: Stochastic because the order of the deck is unknown (and random), otherwise deterministic.
- *(De)Centralized*: Decentralized for the execution of the plan; naturally centralized for computation of a strategy but may be solved in decentralized or distributed fashion.
- *Epistemic Goals*: Mo, but reasoning with epistemic subgoals is clearly needed for playing optimally (first-order knowledge should be enough).
- *Collaborative/competitive*: Collaborative.
- *Benchmarks*: Game rules are already formal, and easy to encode; can be made small, tiny, nano, or huge by adjusting the number of colours, cards and players. AI implementations exist, see for instance <https://github.com/Quuxplusone/Hanabi> and <https://github.com/WuTheFWasThat/hanabi.rs>

4.1.3.5 Clue (Cluedo)

Description: It is an adversarial board game in which the goal of a player is to guess rightly the value of each of three variables: who committed the crime; where; and how. Players iteratively ask other players to privately tell them if they possess one card of a specified triple of values (one for each variable). The value of variable ‘where’ must correspond to the place where the player is located at the request time; players can move from one place to another subject to some constraints.

Motivation: Clue is perhaps the most well-known game with epistemic goals, and it possesses almost all the possible features of an epistemic planning problem. In order to rightly guess the three variables, a player exploits information from the cards that other agents show them,

⁵ Interestingly, deciding whether there is a winning sequence of plays even with full information *and* a single player is already NP-complete [6].



■ **Figure 4** Scotland Yard.

from the fact that an agent denies having any one of a triple of cards, or from the fact that an agent shows a card to another agent, but also from what other agents ask. A good player should maintain probabilistic beliefs and choose questions maximizing expected information gain, but should also try to reason about the inferences that the other agents will draw from his questions.

Features:

- *Knowledge vs. belief*: Knowledge (acquired from non-noisy observations) but the players may perform some probabilistic reasoning.
- *Single vs. multi-agent*: Multi-agent.
- *Communication vs. sensing vs. ontic*: Communication and ontic.
- *Determinism/reliability of communication*: Movement resources and card drawing probabilistic, communication reliable.
- *Epistemic Goals*: Yes, and only that (guess the right triple).
- *Collaborative/competitive*: Competitive.
- *Benchmarks*: Do probably not exist, but have been discussed several times in the epistemic planning community. Parameters that can vary is the number of players, the number of variables, and the number of values per variable.

This problem and its relationship to epistemic reasoning has been discussed by van Ditmarsch [105].

4.1.3.6 Scotland Yard

This board game is played on a map of London. A group of detectives is hunting a criminal using public transport. The position of the criminal is only revealed occasionally and detectives have to reason about possible paths the criminal could have taken. Additionally, resources for movement (bus tickets etc.) are limited for all players.

References and existing Implementations:

- Wikipedia: [https://en.wikipedia.org/wiki/Scotland_Yard_\(board_game\)](https://en.wikipedia.org/wiki/Scotland_Yard_(board_game))
- Spiel des Jahres 1983: <http://spieldesjahres.com/de/scotland-yard>.

- Scotland Yard is PSPACE: <http://www.ilc.uva.nl/Research/Publications/Reports/PP-2006-18.text.pdf>.
- Monte-Carlo Tree Search for Scotland Yard: [76].
- Cooperative planning: <https://eldorado.tu-dortmund.de/bitstream/2003/2653/1/86.pdf>.

Features:

- *Knowledge vs. belief*: Both, there is hard (e.g. criminal is at certain position) and probabilistic (e.g. bus ticket was used) information.
- *Single vs. multi-agent*: Multi-agent, though this depends on model and implementation. If detectives use centralized planning then it is essentially a two-player game. For decentralized planning each detective should work on their own and communication between them might be limited, in particular to public announcements.
- *Communication vs. sensing vs. ontic*: Seen as a two-player game there are only ontic actions, if detectives are multiple agents then a range of communication can be allowed. The only sensing actions are when one can sense whether a criminal is in their sensing agent's current location.
- *Determinism and reliability of communication*: Probabilistic uncertainty is in the initial state only, all communications are reliable successful public announcements.
- *Epistemic Goals*: The goal is not explicitly epistemic but just to be at the same position. But the implied and intermediate goal is to know where the criminal is.
- *Collaborative/competitive*: both: 1 vs n players.
- *Benchmarks*: Different planning algorithms and their implementations could play against each other. Alternatively, a tool could assist the detectives.

4.1.3.7 Bitstring

Description: *Bitstring* is a contingent problem of communication where we have a string of bits, two agents that have the same uncertainty concerning the actual truth values of the bits in the string and each agent can sense only a subset of the bits, disjoint between them.

The simple version of the problem is the one where the goal is for both agents to know the actual truth values of all the bits in the string.

Formally, we have two agents $A = \{A_1, A_2\}$, we have a set B of bits, $B = \{b_1, b_2, \dots, b_n\}$, which denotes a set S of possible states, where $|S| = 2^n$ and a hidden, true state s^* . It is common knowledge among the two agents that only a set S' is actually possible, where $S' \subset S$. Allowed actions are communication actions of the knowledge of each agent concerning the truth value of each bit ($K_i b_j / \neg b_j$, for $i \in A$ and $j = 1..n$). The goal then is $G = \bigwedge_{j=1..n} (K_i b_j \vee K_i \neg b_j)$, for $i \in A$.

The simple version of the problem allows for simple plans where, for example, we can have an agent deriving that he does not need to communicate the truth value of a certain bit since he knows that based on the knowledge the other agent has acquired through his observations, he already knows the truth value of the specific bit.

A more interesting view on the *Bitstring* is its extended version: Assume a third agent A_3 who is not aware of S' but only of S – at the same time he is able to listen to all communication between the two agents A_1 and A_2 and he knows what A_1 and A_2 have sensed. Now, the question becomes: is there a plan which achieves the goal $G_s = \bigwedge_{j=1..n} (K_i b_j \vee K_i \neg b_j) \wedge \bigwedge_{j=1..n} (\neg K_{A_3} b_j \wedge \neg K_{A_3} \neg b_j)$. In other words, we are checking whether there is a plan where the first two agents know the truth values of all bits, while the third one learns nothing.

This problems allows for a number of approaches:

- Checking for existence of such a plan.
- Treat the second part of the goal as a soft goal: the first two agents must learn the truth value of all bits and, at the same time, try to minimize the knowledge the third agent acquires.
- Increase the number of agents and/or restrict the set of agents that each one can communicate with.
- Is there a way to derive a set of states S' such that (i) $S' \subset S$, and (ii) there exists a plan that satisfies G_s for some $s^* \in S'$.

Motivation: It allows for different planning approaches. We can focus on existence of a plan, minimizing leaked information, maximizing size of S' and more. It easily allows decrease/increase in difficulty either by size of state space, or number of agents or restriction of communication. It is related to other examples in dynamic epistemic logic (russian cards, gossip protocols etc).

Features:

- *Knowledge vs. belief:* Knowledge.
- *Single vs. multi-agent:* Multi-agent.
- *Communication vs. sensing vs. ontic:* Communication with, possibly, sensing.
- *Determinism/reliability of communication:* Deterministic.
- *(De)Centralized:* Possibly both.
- *Epistemic Goals:* Yes.
- *Collaborative/competitive:* Collaborative, since the third agent does not act.
- *Benchmarks:* None known.

See also work on communication complexity about related problems (A very short introduction to this and references can be found in Capelli's thesis [27]).

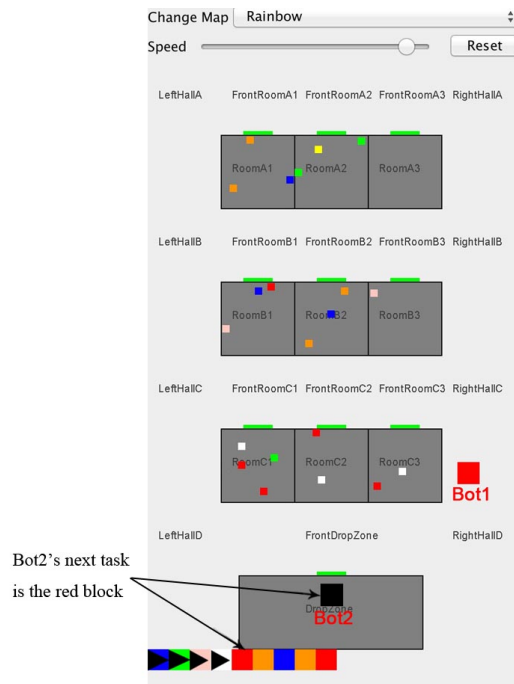
4.1.3.8 BW4T – BlocksWorld For Teams

Description: Blocks World for Teams (BW4T) [54] is a multi-agent simulation platform that is an abstraction of several different applications and scenarios. In this problem, we have a number of rooms and a drop zone. Each room contains a number of colored blocks (or it is empty), and, initially, none of the agents know where the blocks are. Each agent can carry at most one block at a time, and agents can communicate with each other concerning their observations.

The goal is for the agents to deliver a number of colored blocks, in a particular order, to the drop zone. Figure 5 shows an example scenario, taken from the BW4T simulation environment.

Motivation: This problem is an abstraction of several important application domains, such as disaster management and search & rescue. For example, blocks represent survivors of a disaster, the drop zone represents a medical tent, agents represent autonomous vehicles and humans with differing capabilities, such as searchers, medical officers, or aid delivery vehicles.

From an epistemic point of view, we can define a BW4T with private actions, restrict communication between subgroups of agents, and, if in a decentralized approach, reason about the other agents' intentions and plan accordingly. In any case, agents need to reason in terms of *who* to ask about *what*.



■ **Figure 5** Blocks World for Teams simulation interface.

In addition, the BW4T simulator is designed as a testbed for human-agent interaction experiments, making it an interesting framework to study human interaction in epistemic terms: bounded reasoning about high-order beliefs, belief change, interaction between belief, intention, and action.

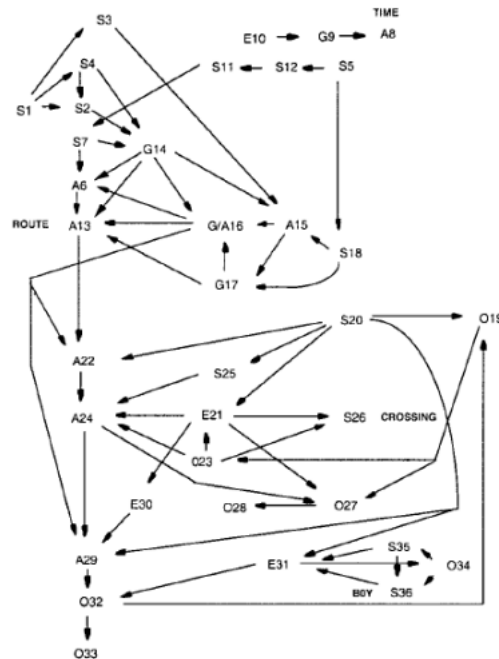
The problem is easy to scale by modifying the number of rooms, blocks, agents, depth of epistemic goals, and area to be searched.

While the original problem is not particularly rich with respect to epistemic reasoning problems, there are many extensions that one can make that are seen in domains that BW4T abstracts:

1. Joint activities: Certain coloured blocks (e.g. blue) are ‘heavy’ blocks, and need two agents to carry them simultaneously. This requires agents to participate in and coordinate this joint activity; something that can be done only via communication or by reasoning about the other agents’ perspectives and intentions. In the scenario of no communication, it explicitly requires reasoning about higher-order beliefs and intentions.
2. Heterogenous agents: Different agents may have different capabilities; for example, some agents can just search and find blocks, while others must retrieve them, and certain rooms may only be accessible to certain agents.
3. Knowledge goals & Communication: Another scenario is one in which a ‘commander’ of the task must be told of the location of all blocks once found. Only some agents can communicate back to the commander, so the problem is deciding what and when to communicate block locations.

Features:

- *Knowledge vs. belief*: Possibly both.
- *Single vs. multi-agent*: Multi-agent.
- *Communication vs. sensing vs. ontic*: All three.



■ **Figure 6** Example causal chain.

- *Determinism*: Yes, but non-determinism is possible.
- *(De)Centralized*: Both.
- *Epistemic Goals*: Yes.
- *Collaborative/competitive*: Both, if two teams try to get the same blocks while their number is not sufficient for both teams.
- *Benchmarks*: A simulation tool for BW4T: <http://eishub.github.io/BW4T/>

4.1.3.9 Social Explanation Dialogue in AI

Description: Consider an agent that makes a decision, and a human observer asks why that decision was made. The aim of a social explanation can be described as follows. Given a causal chain of ‘events’, such as the one in Figure 6, and a request to provide a socially-acceptable explanation for a specific event in that chain, taking into account what the receiver already knows, to extract an explanation that allows the person who posed the query to understand why the event occurred.

The explainer may have a model of the explainee’s (partial) understanding of the causal chain. The explainee can communicate values of the variables in the chain, or relations between variables. Halpern and Pearl offer a formal model for the problem using structural equations [45].

This problem can be extended to a dialogue. At each step, the explainee provides only one part of the explanation, selecting/prioritising the ‘best’ explanation, measured by e.g. simplicity, generality, conformance with prior beliefs of explainee.

Motivation: The ability to explain a decision to a human observer is key to providing transparency and trust to human operators, and is currently one of the most challenging and important problems in artificial intelligence.

Further, such social explanations require a Theory of Mind to avoid explaining redundant information. In richer settings, explanations can be given to multiple observers at once. The challenge in this setting is to provide the explanation with the most explanatory power to the most observers.

Features:

- *Knowledge vs. belief*: Both.
- *Single vs. multi-agent*: Multi-agent.
- *Communication vs. sensing vs. ontic*: Communication.
- *Determinism/reliability of communication*: Deterministic.
- *(De)Centralized*: Decentralized.
- *Epistemic Goals*: Yes (perhaps only first-order).
- *Collaborative/competitive*: Collaborative (but deceptive versions could exist).
- *Benchmarks*: None known.

4.1.4 Modelling

Currently there is no standardized formal language which can express all or even just a substantial subset of these epistemic planning problems and benchmarks. For the future it would be good to fix a notation for DEL and generalize (or pick one of the existing generalizations of) PDDL.

Further, it would be fruitful to build a centralized website that collects the benchmarks (as <http://www.satcompetition.org/> for SAT and <http://planning.domains/> for planning).

4.2 Exploring Action Types and their Representations

Yves Lespérance (coordinator), Louwe B. Kuijer (coordinator), Nina Gierasimczuk, Barteld Kooi, Michael Thielscher, Ivan José Varzinczak, Thomas Bolander, Andrés Occhipinti Liberman, Tran Cao Son, Yongmei Liu, Hans van Ditmarsch

License © Creative Commons BY 3.0 Unported license

© Yves Lespérance, Louwe B. Kuijer, Nina Gierasimczuk, Barteld Kooi, Michael Thielscher, Ivan José Varzinczak, Thomas Bolander, Andrés Occhipinti Liberman, Tran Cao Son, Yongmei Liu, Hans van Ditmarsch

4.2.1 The Need of Pure Action Types in DEL

One of the topics discussed was the distinction between action tokens and action types. In particular, we focused on the question how the action models in DEL fit this distinction. Can DEL action models be seen as action types? It turns out that some aspects of actions which could be seen as properties of tokens are by design included in action models.

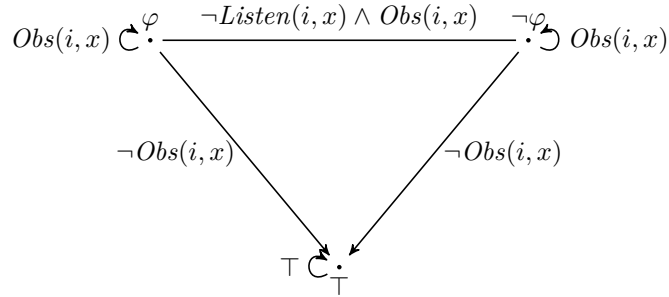
As an example, consider an action model corresponding to a private announcement. Such an action model contains information about which agents learn the content of the information, which agents learn that there has been an announcement but do not learn the actual content, and which agents learn nothing. The problem here is that which agents are “paying attention” should be independent of the action of announcing, but rather follow from the situation in which the announcement is made.

In other words, who does or does not learn the content of an announcement should be determined not by the action model, but by the state model to which the action model is applied. What is needed, then, is an alternative formulation of such an event where the

“observability” information is encoded in the state model. In the context of high level action languages such an approach was suggested in [13] and elaborated in [14, 15, 16].

One can also do this in DEL itself, but there are also other (equivalent) formalisms where this comes a bit more natural, such as Generalized Arrow Update Logic [58] and Edge-Conditioned Event Models [22]. The (epistemic) Game Description Language [97] provides an alternative action representation formalism for epistemic domains that also comes with a clear separation between an action itself and information about which agents are observing it; see below.

One can introduce *observation atoms*, combined with *agentive actions* and *observability change*. Agent i announcing φ can then be the action $i : \varphi$, represented by the following event model.



Here $Listen(i, x)$ means that agent x is listening to agent i , and therefore x will learn the content of i 's announcement. The atom $Obs(i, x)$, meanwhile, means that x observes i making the announcement, but does not necessarily listen to the content of the message.

Importantly, in addition to the preconditions on the worlds of the event model, there are also preconditions on the arrows in the event model. In the notation used above, there would be an arrow for agent x from event e to event f if $\psi(x)$ is true at e , where ψ is the label on the arrow from e to f .⁶

As a consequence of these additional preconditions on arrows, one does not need to put any information about who observes the announcement in the event model. Agent i takes an action of the type “announcing”, whether this is a public or a private announcement depends on the state model.

In this framework, if i definitely wants to make a *public* announcement of φ , this means that she has to perform two actions. First, she has to take an action which has the result that $Listen(i, x)$ becomes common knowledge for all x , then she takes the action of announcing φ .

There is still one significant restriction on this framework: the agents that observe but do not listen to the announcement $i : \varphi$ will still learn that it was the truth or falsity of φ that was announced. There does not seem to be an elegant way around this in a DEL-like system like the one presented; an agent who observes an announcement, but has no idea what the announcement is about would, in theory, need to be represented by an exponential (or even infinite) model with one copy for each possible announcement.

It seems the single “announcement” action type can model public, semi-private, and private announcements. It is conjectured that similar representations can be found for every action type that can be represented in DEL.

⁶ See [22] for details. This formalization is very similar, but not identical, to that of Generalized Arrow Update Logic [58].

Unfortunately, one cannot give this conjecture a technical formulation, because it is unclear what, on a technical level, the objective is. For every type of action, which is represented by multiple action models in DEL, one would like to have a single event model in the language that that was just described. However, it is not clear what, on a technical level, it means for different action models to represent actions of the same type.

As a final note, it should be noted that lunch is usually not free. In the above formalism, information about which agents observe which parts of an action need not be encoded in the event models. In some sense, this makes them simpler. However, one pays for this by having to include information about observability in the state model. So complexity is not removed, but moved to a different part of the problem.

It is believed that in some cases it is useful to move this complexity from event models to state models. The first advantage of doing this is the motivation for this system, namely that it allows a clearer representation of action types. The second advantage is that it is often considered harder to intuitively understand event models than state models, so keeping event models as simple as possible may aid comprehensibility. But one should keep in mind that in some cases the complexity shift may be harmful, and that in those cases one is probably better off using DEL.

Action Types in the Epistemic Game Description Language

The (epistemic) game description language (GDL-III, [97]) may provide an alternative to DEL for representing actions and knowledge especially when the aim is to avoid complex event models. A special feature of this language is that only what agents can see and do needs to be specified. It is then implicit in the semantics how actions and observations affect the knowledge of agents. The scenario from above, for example, would be axiomatized thus (using a first-order variation of the actual syntax):

$$\begin{aligned} \mathbf{Sees}(x, i_a) &\Leftrightarrow \mathbf{Does}(i, \text{announce}(z)) \wedge \text{Obs}(i, x) \\ \mathbf{Sees}(y, \varphi) &\Leftrightarrow \mathbf{Does}(i, \text{announce}(\varphi)) \wedge \text{Obs}(i, y) \wedge \text{Listen}(i, y) \wedge \varphi \end{aligned}$$

Both **Sees** and **Does** are keywords that loosely correspond to modalities. The two axioms can be understood as follows: If agent i makes some announcement z then all agents x observing i will receive the “information token” i_a . Any agent y that observes and also listens to i will get to see the announcement φ itself. From the semantics of GDL-III it follows that agents who only see i_a will know that i has made an announcement but without learning the content. Agents who see φ , however, will know that φ must be true. Moreover, if for example an agent x only observes agent i but at the same time knows that another agent y listens to i , then the semantics entails that x will know that y knows the content of the announcement afterwards.

In the Game Description Language, actions such as $\text{announce}(z)$ can be seen as action types. Their effects, especially on the knowledge of other agents, depends on the state in which they are executed. Our example action can thus model public, semi-public, and private announcements. From this perspective GDL-III might be a viable alternative to DEL in cases where the complexity shift from the event model to the state representation is desirable. It has been shown that many of the standard DEL-examples can be expressed in GDL-III [97]; however, a precise characterization of the expressivity of the two languages in comparison has yet to be given.

4.2.2 Concurrent Actions

The issue of finding a suitable representation for *concurrent actions* that can be used in multiagent epistemic planning was also discussed. Amongst other things, concurrency has attracted attention in multiagent planning because parallelizing actions can reduce overall plan length and lead to a significant speed-up in plan execution time.

Concurrent actions can have interaction effects; their parallel execution sometimes alters the effects that the actions would have if executed individually. The following example from [24] illustrates this. Suppose agents x and y need to move a large set of blocks from *room 1* to *room 2*. While x and y could each pick up and transport single blocks repeatedly, they can do better by using an existing table as follows. First, the agents put all the blocks on the table. Then, they each lift one side of the table. Lifting the table simultaneously from both sides allows the agents to jointly carry all blocks to *room 2* in one trip. However, if only one side of the table is lifted, all the blocks will fall to the floor. That is, the *table lifting* action has different effects when applied individually or jointly, and its joint execution allows the agents to achieve their goal faster.

In the reasoning about actions community several formalisms and approaches, such as in [63, 17], have been proposed to represent and reason about concurrent actions. More recently, Boutilier [24] and Knoblock [56] extended the STRIPS representation of actions to allow for concurrent actions. Reiter proposed axioms for concurrent planning in the situation calculus framework [81]. There has also been some work to model concurrent actions in the DEL framework [110], as well as in the GDL framework [96]. It would be useful to provide a comparison of these different approaches to concurrency as well as their applicability in epistemic planning.

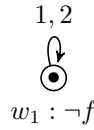
4.3 Exploring the Relations Between Knowledge and Belief in Multi-Agent Epistemic Planning

Chitta Baral, Hans van Ditmarsch, Jan van Eijck, Esra Erdem, Andreas Herzig, Mathias Justesen (coordinator), Yongmei Liu, Richard Scherl (coordinator), Tran Cao Son

License © Creative Commons BY 3.0 Unported license
 © Chitta Baral, Hans van Ditmarsch, Jan van Eijck, Esra Erdem, Andreas Herzig, Mathias Justesen, Yongmei Liu, Richard Scherl, Tran Cao Son

In multi-agent domains actions can happen without all agents being aware of their occurrence. This leads to situations where some of the agents' beliefs may not match with reality. On the other hand agents' may have incomplete information about the world and there may be actions such as sensing actions that provide additional knowledge about the world to the agent that does the sensing. Agents may have knowledge goals that are about knowing part of the world and agents may have goals that are about deceiving other agents in terms of mismatch between the beliefs of the other agents and reality. Thus the notions of knowledge and belief play crucial roles in a multi-agent epistemic planning domain and several important questions about them need to be addressed. Some of those questions are: How do we identify the theoretical and computational challenges in planning with knowledge vs. planning with belief? When is one or the other appropriate and when are both needed?

Every approach has to have a way of distinguishing the representation of the knowledge and beliefs of different agents from the representation of the actual world. Even if knowledge is being used, the knowledge may turn into false belief after a partially observable action has occurred. This necessitates addressing questions such as:



■ **Figure 7** Initial state s_0 .

- What are the different formalisms for planning in knowledge and/or belief and what are their theoretical and computational properties?
- How is revision dealt with in planning?
- Are there specific types of interesting goals for epistemic planning? (Note that, in planning with beliefs, goals can be false beliefs with relevance to formalizing false-belief tasks such as lying and deception.)

4.3.1 Knowledge vs. Belief

In planning we sometimes want to achieve something with certainty and therefore knowledge is then needed. However, the certainty required for knowledge is not always achievable and then we should work with belief. From an external perspective the planner has a goal involving knowledge, but from an internal perspective each agent executing its part of the plan only needs belief. One approach is to define knowledge in terms of belief as true belief. There are also a variety of ways of defining belief in terms of knowledge [4].

In the planning based on dynamic epistemic logic (DEL) work in [10], states are modeled using Kripke structures and depending on the accessibility relations among the worlds, different axioms of knowledge and belief are obtained. If the accessibility relations satisfy that all relations are reflexive, it corresponds to the axiom of truthfulness, T: $K_a\varphi \rightarrow \varphi$. When modelling the planning problems that satisfy axiom T (e.g. S5), the agents cannot have false beliefs. However, this is often not a suitable model of the problem, so for illustrating the difference between knowledge and belief, KD45 is used. (In this context, in the work [88] it is shown that the $KD45_n$ property is maintained for a large class of actions.)

The rest of this subsection is organized as follows. First, we motivate the use of planning with beliefs with an example based on DEL. Continuing with the example, we proceed to show that with DEL, agents cannot recover from a false belief. Using various formalisms, we then explore different options for incorporating belief revision in epistemic planning.

4.3.2 Planning with beliefs

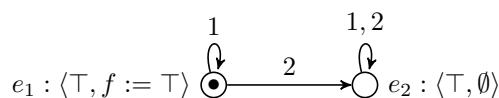
Consider the following scenario. There are two agents. Initially, there is common knowledge of $\neg f$. The goal is to have agent 1 correctly believe f and agent 2 to incorrectly believe $\neg f$. So agent 1 has the plan of changing f to true while agent 2 is not looking.

- Initial State: $C_{\{1,2\}}\neg f$
- Goal: $K_1f \wedge K_1K_2\neg f$
- Plan: agent 1 makes f true while agent 2 is not looking

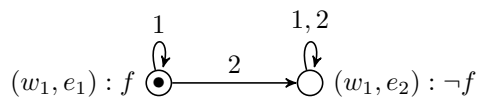
Consider the example in DEL. We first have the initial state, where there is common knowledge of $\neg f$ depicted in Figure 7.

The plan can be modelled by an action a depicted in Figure 8, i.e., agent 1 makes f true while agent 2 is not looking.

The result of executing action a is $s_1 = s_0 \otimes a$ as depicted in Figure 9, where \otimes is the usual product-update operator. Observe that s_1 satisfies the goal, $s_1 \models K_1f \wedge K_1K_2\neg f$.



■ **Figure 8** Event model for action a .



■ **Figure 9** Resulting state $s_1 = s_0 \otimes a$.



■ **Figure 10** Event model for action a_2 .

Now let us imagine that there is a public announcement of f . The action model for this action a_2 is given in Figure 10.

The result in computing $s_2 = s_1 \otimes a_2$ is that agent 2 now believes everything (including contradictions), as shown in Figure 11, since there are no accessibility relations for agent 2. This appears counter-intuitive. We would expect agent 2 to revise their beliefs, and now believe f . Hence, we need some sort of belief revision to handle such scenarios where an agent has a false belief.

4.3.3 Belief revision

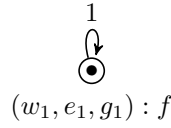
The topic of belief revision in general, and with regard to DEL in particular [38, 113, 12, 102], has been studied thoroughly in the literature. We look at several approaches of incorporating belief revision:

- Plausibility models [11]
- Syntactic belief revision [53]
- Recovery [112]
- Modifying the product update of DEL (a recent work by Chitta Baral and Tran Cao Son).

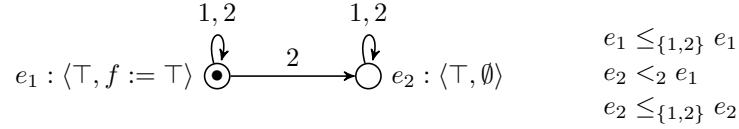
Plausibility models

Here there is a plausibility ordering on the worlds considered possible by a given agent. The belief of the agent is captured by the most plausible worlds. If worlds are lost due to application of an action, then previously less plausible worlds become the most plausible. It is a requirement of plausibility orderings that they are reflexive, hence, we get the a action in Figure 12, and the resulting state s_1 (Figure 13) now also has a reflexive edge for agent 2 in the designated world. Still, in s_1 agent 2 believes $\neg f$ more strongly than f as indicated by the plausibility orderings (\leq_i) on the worlds, where the minimum element is considered most plausible.

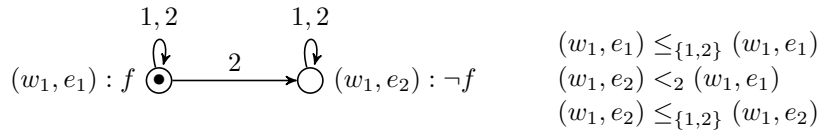
Now, we apply the public announcement action of f and the result is given in Figure 14. Note that we no longer lose the indistinguishability relations of agent 2, as their belief was revised by rejecting the world that was previously believed to be most plausible.



■ **Figure 11** State s_2 after the public announcement of f .



■ **Figure 12** Plausibility event model for action a .



■ **Figure 13** Resulting state $s_1 = s_0 \otimes a$.

Syntactic revision

In [53] formulas are kept in a normal form called Alternating Cover Disjunctive Formula (ACDF). There is an algorithm for syntactically manipulating the form for update and revision so that the result is still in ACDF. The running example can be done with an initial state of

$$\neg f \wedge K_1 \neg f \wedge K_2 \neg f.$$

The action a with effect $f \wedge K_1 f$ is used with the update algorithm to yield

$$f \wedge K_1 f \wedge K_2 \neg f.$$

Next the action a_2 with effect $K_1 f \wedge K_2 f$ is used along with the revision algorithm to yield

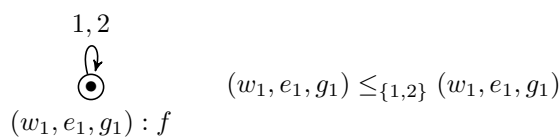
$$f \wedge K_1 f \wedge K_2 f.$$

Recovery

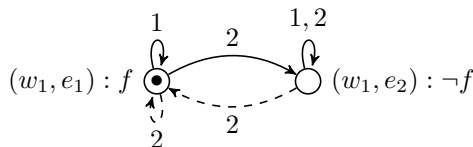
Another approach is to do a recovering action before the public announcement [112]. The recovering action is defined as $R \cup (R; R^u) \cup R^u$, where R is the original set of accessibility relations for an agent, R^u is the inverse relation, and $R; R^u$ is the composition of the relation and its inverse. Recovery can be seen as a form of epiphany, where agents suddenly expand their beliefs and consider worlds that they previously did not. Recovery is applied to the state of Figure 9 to yield Figure 15. Then when the public announcement is applied, the desired result is obtained.

Modifying the DEL product operator

An alternative approach, suggested and being worked on by Baral and Son, is based on modifying the DEL product-update operator. In the above discussion of the product update,



■ **Figure 14** State after public announcement of f is $s_2 = s_1 \otimes a_2$.



■ **Figure 15** State after recovery. Dashed edges are the edges added by recovery.

we saw that when executing an action a in the epistemic state s , there is an edge with label l between nodes n_1 and n_2 in the resulting epistemic state, if there is a similar edge in both a and s . In the modified operator, this requirement is relaxed.

More precisely, if node n_1 is a composition of node ns_1 (of s) and na_1 (of a) and node n_2 is a composition of node ns_2 (of s) and na_2 (of a) then the traditional DEL operator will add an edge with label l from n_1 and n_2 if there is an edge both from ns_1 to ns_2 and from na_1 to na_2 with label l . Baral and Son suggest that, when one can conclude that the action a requires revising s , then a *relaxed* product operator can be used where a gets preference over s . For example, with respect to a simple relaxed operator, there will be an edge with label l from n_1 and n_2 , if there is an edge with label l from na_1 to na_2 . This addresses the example in question. Finding the right relaxed product operator, and conditions when they should be used is a topic of ongoing work.

Update and revision for proper epistemic knowledge bases

Miller and Muise [70] present a belief update and revision approach for proper epistemic knowledge bases, which underlies the approach for compiling epistemic planning to classical planning discussed in Section 4.4.1. The syntactical restrictions on proper epistemic knowledge bases allow entailment, update, revision, and erasure operations that have quadratic time and space complexity, and thus enable efficient search over epistemic states.

4.3.4 Story understanding

The distinction between knowledge and belief is important in many scenarios, e.g., story understanding. As a story progresses, the characters can obtain false beliefs. As is common in TV serials, the truth is revealed at the end and all characters need to revise their beliefs.

This is related to the Sally-Anne test. A short skit is enacted. Sally takes a marble and hides it in her basket. She then leaves the room and Anne takes the marble out of Sally’s basket and puts it in her own basket. The question of where Sally will look for her marble is asked. Autistic children tend to reply that the marble is in Anne’s basket. The test shows the inability of people on the autism spectrum to distinguish between the actual world and a person’s belief about the world.

Story understanding relates more directly to plan verification than to plan synthesis: Answering questions about the beliefs of agents at the end of a story corresponds to checking whether certain epistemic or doxastic formulas (“goal formula”) hold after the execution of an action sequence (where the action sequence represents the story).

4.4 Practical Tools, Resources and Computational Techniques

Tristan Charrier (coordinator), Sophie Pinchinat, Vaishak Belle, Thorsten Engesser, Torsten Schaub, Malte Helmert, Bastien Maubert, Jens Claßen, Ioannis Kokkinis, Sunil Easaw Simon, Robert Mattmüller

License © Creative Commons BY 3.0 Unported license
 © Tristan Charrier, Sophie Pinchinat, Vaishak Belle, Thorsten Engesser, Torsten Schaub, Malte Helmert, Bastien Maubert, Jens Claßen, Ioannis Kokkinis, Sunil Easaw Simon, Robert Mattmüller

We survey existing techniques to deal with epistemic planning. For each of these techniques, we explain what are the models and formulas considered, and what are the advantages and the drawbacks of such techniques. We do not restrict the survey to already implemented epistemic planners, but also detail theoretical techniques to deal with epistemic planning.

4.4.1 Compilation to Classical Planning: Syntactic Approaches

An approach to epistemic planning that recently gained attention is the idea of resorting to a suitably restricted fragment of DEL and use a STRIPS-style encoding of action operators over epistemic formulas instead of event models and product updates. This allows to compile the given problem into an instance of classical planning, and subsequently leverage the performance of state-of-the-art PDDL planners for solving it efficiently. The proposed methods differ in

1. the restrictions that are applied,
2. whether a centralised or first-person view is adopted,
3. the type of knowledge that is assumed.

Muise *et al.* [74] present a planner for multi-agent systems where agents are assumed to have so-called *proper epistemic knowledge bases*, which effectively rule out disjunctions. With the additional restriction to a finite depth of nesting of modalities, the compilation is exponential in the depth of the maximum length of modal strings. The expressiveness of the action model is independent of the approach, but depends only on that of the underlying planner. Both $KD45_n$ and $S5_n$ are supported. Planning is from the view of a single agent, including a “perspective shift” where one agent reasons about the belief of another from that agent’s point of view.

Kominis and Geffner [57] compile a multi-agent epistemic planning problem into a classical one using techniques from single-agent conformant planning. The compilation is similarly exponential in the maximal depth of nesting of modal operators. They assume $S5_n$ type knowledge and adopt a centralised perspective with command-and-control first-person planning. Preconditions and effects of actions are expressed by Boolean combinations of propositional literals with modality prefixes, including disjunctions.

Cooper *et al.* [33] use an encoding based on “seeing” variables for atomic propositions, which represent the agent “knowing whether” the corresponding variable is true. Knowing a variable p can then be defined as knowing whether p holds and p being actually true at the same time. The approach supports both $S5_n$ and K_n knowledge types and plans from a centralized perspective.

A related, earlier approach is Petrick and Bacchus’ PKS (“planning with knowledge and sensing”) system [78] for single-agent epistemic planning. While they do not compile into classical planning, they also adopt a syntactic representation for a planning domain where an agent’s knowledge is assumed to be expressed by three different types of formulas: the knowledge base is partitioned into one part for “knowing that”, one for “knowing whether”,

and one for “knowing what”. Their planner then generates conditional plans over the space of knowledge states expressible by means of such knowledge bases, relying on an efficient, yet incomplete, inference algorithm.

4.4.2 Forward Search in Space of Epistemic Models

Forward search in the space of epistemic models is possibly one of the most obvious approaches to solving epistemic planning tasks. The first definitions of (centralized) epistemic plans over epistemic states and DEL action models were given by Bolander and Andersen [23] and independently by Löwe *et al.* [66]. The plans are sequences of DEL action models applicable in the initial epistemic state that guarantee to reach an epistemic state satisfying the goal formula. Those were later generalized to branching plans [2] where branching is essential, since epistemic action models allow non-deterministic action outcomes, possibly necessitating different follow-up actions.

While the above-mentioned definitions of plans may not, strictly speaking, imply a computation based on forward search in the space of epistemic states, they at least suggest such a search as an obvious first approach to explore. To our knowledge, so far only simple uninformed explicit-state forward search algorithms such as breadth-first search (BFS) have been implemented.

Whereas the first definitions assumed a centralized perspective, the definitions were later adapted to a more decentralized view. Engesser *et al.* [40] introduced the notion of implicitly-coordinated (linear and branching) plans. While these plans are still computed from a single-agent perspective, they consider explicitly the perspectives of the owner of each subsequent action (*i.e.*, the agent that will execute it). The intention is to facilitate cooperative behavior by ensuring that agents can come up with those actions when planning on their own. The approach heavily relies on the framework’s capacity to arbitrarily switch perspective between the agents. Since the necessary depth of knowledge nesting is dependent on the plan length and therefore not known beforehand, compilation into classical planning is not possible.

4.4.3 Efficient progression for epistemic states

One way to handle epistemic planning is to adapt a standard forward search (progression) or backward search (regression) algorithm to the epistemic setting. Compared to a classical (non-epistemic) setting, a major question in this approach is how to represent the search states. In the classical setting, search states are either propositional models (in the case of planning with complete information) or representations of sets of propositional models, such as propositional logic formulas (in the case of planning with incomplete information, for example conformant planning). This latter representation can be generalized to the epistemic setting by using epistemic formulas instead of classical ones. An algorithmic challenge, then, is how to efficiently perform the basic operations on search states that are required by standard search algorithms like A^* or greedy best-first search, such as testing whether a goal state has been reached, testing whether a given action is applicable, and computing the (epistemic) successor state for a given (epistemic) search state and (epistemic) action. The paper by Bienvenu *et al.* [21] addresses these issues in the context of the (single-agent) epistemic logic $S5$. They present a fragment of $S5$ formulas, called $S5DNF_{DNF,CNF}$, which efficiently supports the progression operation: given an epistemic formula describing the possible (epistemic) states of the world and a description of an epistemic or classical action in a certain formula representation, compute a formula representing the possible (epistemic)

successor states after applying the action. The operation is polynomial-time in the description size of the inputs (epistemic state and action), but presumably can be exponential in the sequence length if sequences of actions are applied, as each application can increase the size of the resulting formula. Miller and Muise [70] present a similar approach for proper epistemic knowledge bases, which are epistemic knowledge bases that forbid disjunction. Progression of such knowledge bases is quadratic in the number and depth of epistemic formulae. Presumably, the same techniques can be used for regression instead of progression, *i.e.*, for computing possible predecessor states given formulas representing current states and actions. More generally, the paper is framed in the context of knowledge compilation pioneered by Darwiche and Marquis [36]. It emphasizes the strengths and weaknesses of different kinds of representations of knowledge (usually in terms of propositional models, here extended to an epistemic setting) by focusing on their closure properties and the set of operations (such as logical connectives, logical implication, existential and universal abstraction etc.) that can or cannot be performed efficiently for a given representation. Beyond the immediate contribution of the paper by Bienvenu *et al.*, the knowledge compilation perspective might be a useful angle for coming up with search state representations for epistemic planning.

4.4.4 Symbolic model checking of strategy logics with knowledge (AETL)

Logics for strategic reasoning have been developed in the last two decades, first with Alternating-time Temporal Logic (ATL) and its many variants, and more recently with Strategy Logic (SL). ATL, introduced in [1], extends the branching-time logic Computation Tree Logic (CTL) [7] by replacing CTL temporal operators with strategic cooperation modalities expressing what state of affairs a coalition of agents can bring about in a system, irrespective of the actions of the other agents. It can thus naturally express the existence of a strategy, or plan, to reach a given objective. So can SL, as it is strictly more expressive than ATL [71].

MCMAS [65] is a model checker for logics for strategic reasoning that handles several variants of ATL and SL, including AETL, an extension of ATL with knowledge operators [104] that can talk about epistemic objectives. Models are concurrent game structures, which are essentially finite state automata where states represent positions of the game, or states of the system, and transitions are labelled by tuples of actions, one for each agent. In addition, equivalence relations on the set of states represent agents' observation of the system. The specification of the system is completed by providing MCMAS with a set of initial states, an optional set of fairness conditions, and the set of formulas to be verified.

Given an epistemic formula φ , the AETL formula $\langle A \rangle F\varphi$ means that “there exists a strategy for coalition A such that every outcome of this strategy hits a state where goal φ holds, irrespective of the strategies of the other agents”. The model checker MCMAS can therefore be used to solve a form of epistemic planning problem, even in an adversarial setting. Since MCMAS, as other standard model checkers, can also generate witnesses of strategies when they exist, it is possible to synthesize an (adversarial) plan when the answer to the epistemic planning problem is positive. Notice that AETL can express more than mere reachability. This is an advantage if one wants to solve variants of the epistemic planning, *e.g.* with extra constraints on the searched plan. For example, one may seek a plan that reaches the goal without ever taking the system into some predefined set of bad/risky states.

However, there are some strong limitations when using the MCMAS model checker. First, it does not allow reasoning about general beliefs, as epistemic relations are $S5$. Second, it is not possible to reason about systems with perfect recall: an agent does not distinguish between two executions of the system as long as their last states are equivalent to her (*i.e.*

memoryless knowledge). Third, unlike other frameworks such as DEL, the entire system needs to be described as a finite state automaton (or concurrent game structure). Therefore, only regular systems can be considered, as opposed to those described by arbitrary DEL presentations.

We also mention MCK, a model checker for logics of knowledge and time (that supports both symbolic and explicit model checking); DEMO, an explicit model checker for DEL; and SMCDEL, a symbolic model checker for DEL [103]. A comparison of MCMAS, MCK and DEMO can be found in [111].

4.4.5 Model checking via automata

Automata theory provides very powerful techniques to study a wide range of logics of programs, such as the linear-time temporal logic LTL, the branching-time temporal logic CTL or the mu-calculus (see, e.g. [99] and [114]). Concerning the interplay between time and knowledge, little work has been done using automata. The reason may be that adding the “transversal” dimension of knowledge in trees representing the temporal evolution transform the usual tree models into graphs, for which no good model of automata is known.

A few years ago, Bozzelli *et al.* [25] showed that if the transversal, or epistemic, relation is representable by a two-tape automaton (*i.e.* it is a rational relation), then we can model-check branching-time temporal logic with epistemic operators. This result was then applied to solve the propositional DEL-based epistemic planning problem: the automata constructions presented in [5] can, given a propositional DEL presentable domain of the epistemic planning problem, produce a finite word automaton that accepts the set of successful plans. Indeed, in this restricted class of domains, the epistemic relations of agents with perfect recall fit in the framework considered in [5]: they are rational relations, and in fact quite simple ones as they can be accepted by transducers with only one state.

The first main advantage of this approach is that it generates a finite representation of the whole set of solutions to the planning problem, which may be useful for instance if the addition of some new constraint reduces the set of solutions: it may be more efficient to cut some transitions in the automaton to obtain the new set of solutions, instead of computing from scratch new solutions to the over-constrained problem.

A second strength of this approach is that it can actually solve, without additional effort, a much more general problem, which is called Epistemic Protocol Synthesis in [5]. In this problem, the input is again a DEL-based epistemic planning problem, but instead of an epistemic goal formula φ that has to be achieved, the problem takes an arbitrary formula from CTL^*K_n , the full branching-time temporal logic with knowledge, and produces, if possible, one possible infinite tree of events that satisfies it. The tree may represent a non-deterministic plan, a set of plans, or a strategy for instance. In CTL^*K_n one can express, for instance, the existence of a path that eventually satisfies φ , which corresponds to the classic epistemic planning problem. But one can also for example require instead that in the solution tree there is a path where φ_1 holds infinitely often, another path in which after some point φ_2 always holds, and in addition that in all paths of the solution tree, some fairness property holds.

The main drawbacks of this approach are that, because of the perfect recall semantics resulting from the DEL setting, it only works for propositional events, *i.e.*, events with propositional preconditions and effects/postconditions. However the problem does not come from the automata approach: the DEL-based epistemic planning problem is known to be undecidable for alternation depth one in the pre/post-conditions. However it is still a seemingly difficult question whether it is decidable for alternation depth 0 and even modal depth 1.

Finally, this approach gives an algorithm in $(k + 1)$ -EXPTIME, where k is the modal depth of the goal formula. Note that it is the same complexity for the epistemic planning problem and its generalization, epistemic protocol synthesis. Also, for DEL-based epistemic planning it is the best known upper bound, but matching lower bounds are an open question.

4.4.6 Reduction of Epistemic Reasoning to Theorem Proving

One technique proposed for dealing with epistemic formulas in a Situation-Calculus context is a reduction to theorem proving (*e.g.* see [59] for a modal variant of the epistemic Situation Calculus). In the single agent case (see [18] for an extension to the multi-agent case), the idea is that the agent’s belief is represented by a knowledge base in the form of a collection of (non-epistemic) first-order formulas about the current state of the world, and it is assumed that this knowledge base is all that is known by the agent. An epistemic subformula such as $K(\text{Open}(\text{window}))$ can then be evaluated by checking whether $\text{Open}(\text{window})$ is entailed by the knowledge base, and subsequently substituting it by either true or false, depending on the outcome of the entailment check. Furthermore, open formulas like $K(\text{OnTable}(x))$ will be substituted by a representation of the known instances with respect to the knowledge base in terms of equalities, *e.g.* $(x = \text{mug} \vee x = \text{laptop})$, and nested K’s can be dealt with by means of recursion. Together with regression (a syntactical method for rewriting a formula about the situation after the execution of a sequence of actions to a formula talking only about the initial state of the world), this corresponds to a reduction of the projection problem to standard first-order theorem proving as follows: In order to decide whether formula $[\text{Action}_1] \dots [\text{Action}_k] K(\varphi)$ holds, use regression to eliminate the action modalities $[\text{Action}_i]$, turning the input into an equivalent formula (with respect to axioms describing action effects and sensing outcomes in the knowledge base), use reduction (which involves theorem proving in the form of entailment checks) to eliminate K modalities, and finally check the resulting formula against the knowledge base (another entailment check). Projection is of course only a subtask needed for planning. To create a planner, one has to perform a search over possible action sequences, where in each case projection is used to decide whether it constitutes a plan achieving the given goal formula. One option to constrain the search space and thus speed up planning is by incorporating procedural domain knowledge in the form of knowledge-based agent programs (*e.g.* [83, 69, 82, 61, 32]). Apart from imperative constructs such as while-loops and if-conditionals, such programs may also contain nondeterministic choices. A program thus can be understood as a plan sketch, where certain parts are left open and have to be resolved by the system. As the general Situation Calculus allows for full first-order expressivity, in practice one needs to integrate further restrictions to obtain decidability or tractability. Apart from the usual syntactical restrictions on the language, another interesting approach is resorting to a limited (sound, but incomplete) form of reasoning [60].

4.4.7 Language $m\mathcal{A}$

The language $m\mathcal{A}$ is described in [16]. Some of its earlier versions appeared in [14, 15]. It could be seen as an extension of the high-level action languages to reasoning about actions and changes in single agent environment [43] to multi-agent environment.

In $m\mathcal{A}$, a multi-agent environment is characterized by a set of agents \mathcal{AG} , a set of fluents \mathcal{F} , and a set of actions \mathcal{A} .

A *belief formula* over \mathcal{AG} and \mathcal{F} is of the following form

$$\varphi \mid \neg\varphi \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \varphi \rightarrow \psi \mid B_i\varphi \mid C\varphi \mid E_\alpha\varphi$$

where φ and ψ can be a propositional formulas over \mathcal{F} or belief formulas, $i \in \mathcal{AG}$, and $\alpha \subseteq \mathcal{AG}$.

Syntax

A *dynamic domain* over \mathcal{AG} , \mathcal{F} , and \mathcal{A} consists of a set D of statements describing the effects of actions in \mathcal{A} , and a set O of statements describing the observability of agents when actions are executed. Specifically,

- Each statement in D is of a following form

$$\mathbf{executable} \ a \ \mathbf{if} \ \psi \tag{1}$$

$$a \ \mathbf{causes} \ \ell \ \mathbf{if} \ \psi \tag{2}$$

$$a \ \mathbf{determines} \ f \tag{3}$$

$$a \ \mathbf{announces} \ \ell \tag{4}$$

where $a \in \mathcal{A}$, f is a fluent \mathcal{F} , ℓ is a fluent $f \in \mathcal{F}$ or its negation $\neg f$, and ψ is a belief formula. (1) encodes the executability condition of action a , *i.e.*, the execution of a is possible only if ψ is true. (2) represents the effects of action a . Intuitively, if the real state of the world and of the beliefs match the condition described by ψ , then the real state of the world is affected by the change that makes the literal ℓ true after the execution of a . (3) encodes a sensing action a which enables the agent(s) to learn the value of the fluent f . (4) states that action a announces the truth value of ℓ .

- Each statement in O is of a following form

$$\alpha \ \mathbf{observes} \ a \ \mathbf{if} \ \varphi \tag{5}$$

$$\alpha \ \mathbf{aware_of} \ a \ \mathbf{if} \ \varphi \tag{6}$$

where $\alpha \subseteq \mathcal{AG}$ is a set of agents, $a \in \mathcal{A}$, and φ is a fluent formula.

(5) (resp. (6)) states that if φ is true in the real state of the world then agents in α are fully (resp. partially) aware of the occurrence of the action a .

4.4.7.1 Semantics

Let \mathcal{S} be the set of pointed Kripke structures over \mathcal{AG} and \mathcal{F} . The semantics of a dynamic domain (D, O) over \mathcal{AG} , \mathcal{F} , and \mathcal{A} is defined by a function $\Phi_{(D,O)}$ that maps pairs of actions and pointed Kripke structures in \mathcal{S} to subsets of \mathcal{S} , *i.e.*, given an action a and a pointed Kripke structure M , $\Phi_{(D,O)}(a, M)$ is a set of pointed Kripke structures that results from the execution of a in M . It does so by

- constructing the update model of the action occurrence a given M , denoted by $\mathcal{U}(a, M)$;
- defining $\Phi_{(D,O)}(a, M)$ as the update of M by $\mathcal{U}(a, M)$, $M \otimes \mathcal{U}(a, M)$, following *e.g.* [8].

4.4.7.2 Relationship to Epistemic Planning

The language $m\mathcal{A}$ provides a practical tool as well as the foundation for generalizing contemporary planning techniques in single agent domains to epistemic planning in presence of multiple agents. The features of $m\mathcal{A}$ that are suitable for the development of epistemic planning systems are:

- *The separation between the description of effects of actions and the observability of action occurrences of the agents:* In $m\mathcal{A}$, action types and action tokens are clearly separated. Statements of the forms (1) – (4) specify action types. Action types are specialized with respect to a pointed Kripke structure and statements of the forms (1) – (4) to create action tokens. In this sense, $m\mathcal{A}$ addresses the issues raised in the discussion for the need of pure action types in DEL (Sub-section 4.2.1). Given the similarity between action languages for single-agent domains and PDDL and the fact that statements of the forms (1) – (3) have been a part of PDDL, the development of $m\mathcal{A}$ shows that PDDL can be easily extended to specifying multi-agent planning domains. To do so, PDDL needs to be extended with (*i*) the specification of agents; (*ii*) consideration of belief formulas and an additional type of actions (announcement actions); and (*iii*) the specification of observability.
- *Definition for the transition function:* The function $\Phi_{(D,O)}$ can easily be realized and utilized by heuristic search algorithms. Indeed, the function $\Phi_{(D,O)}$ can be implemented in heuristic search planner similar to the implementation of the `result` function of PDDL domains. This paves way for the studying and development of heuristics in epistemic planning.
- *Conditions for the finiteness of the set of pointed Kripke structures satisfying a belief formula and algorithms for computing this set:* The initial state of an epistemic planning problem can be represented by a belief formula. Planners employing pointed Kripke structures as states need to create the initial state by computing the set of pointed Kripke structures satisfying a belief formula. This set, however, might be infinite. As such, conditions for the finiteness of the set of initial pointed Kripke structures and algorithms for computing this set are important for the development of epistemic planners. Such conditions are provided in [87].

4.4.8 Multi-agent planning in general (with epistemic side-effects)

In multi-agent planning, one of the settings usually studied is cooperative planning involving agents with heterogeneous capabilities. An even more specific setting that gained attention recently is privacy-preserving multi-agent planning [77]. While the problem is not epistemic per se (*i.e.*, there are no knowledge goals and preconditions), it certainly possesses some epistemic characteristics. First, some dimensions of the state space can be completely unknown to all but some agents. Private variables can then be changed by private actions that are only known to their owners. The intention is for the agents to come up with a joint plan to achieve their goal without having to give away their private information (variables, actions) in the process.

There already exists a variety of search algorithms [101, 26] as well as different kind of heuristics [93, 68, 91, 92]. An overview of state-of-the-art planners can be found in [94].

4.4.9 Classical planning techniques

For approaches based on compilation to classical planning, or perhaps also to draw inspiration from classical planning algorithms, it is useful to consider the state of the art in classical planning. The three prevalent approaches in the classical planning literature at the moment are heuristic forward search, planning as satisfiability, and planning as symbolic model checking. In planning competitions, for more than a decade almost all successful planning systems in the sequential planning tracks have been based on one of these paradigms or are portfolios of planning systems based on these paradigms. For an overview of planning tech-

niques used in state-of-the-art planning systems, the planner descriptions in the competition booklets of the International Planning Competitions (IPC) are a good starting point (see the “Deterministic Track” entries on <http://icaps-conference.org/index.php/Main/Competitions>). Most current planning systems based on heuristic forward search are built on top of the Fast Downward planner [47], and some others are based on FF [51]. It should be pointed out that the paper describing the Fast Downward planner is more than a decade old, and the current version of Fast Downward contains many features (in particular more heuristics) not described in the paper. A good place for discussion and practical information about planning as heuristic forward search is the Fast Downward mailing list, linked from the planner homepage (<http://www.fast-downward.org>). Generally speaking, there is a wide difference in scalability between planning systems that guarantee optimality of solutions (optimal planners) and ones that do not (satisficing planners). Most optimal heuristic search planners use A^* with an admissible heuristic, in some cases enhanced with pruning techniques based on symmetries and partial order reduction. The paper by Helmert and Domshlak [48] is a good starting point to get an overview of heuristics for classical planning, although it has become a bit dated. For satisficing search algorithms, the design space and hence the list of available techniques is larger. In recent work, much emphasis has been placed on guaranteeing sufficient exploration, *i.e.*, making sure to visit diverse parts of the search space, as opposed to exploitation guided by a heuristic. Exploration can be based on randomness or based on systematically favouring states with high novelty. The latter approach has been very successful, and many relevant papers have been published recently by Lipovetzky and Geffner [64]. Planners based on compilations to satisfiability have also been very successful over the last ten years, thanks mainly to the work by Jussi Rintanen [84, 85]. They are a very strong alternative to heuristic search planners in the case of satisficing planning; for optimal planning, no competitive SAT-based planners exist at the moment. SAT-based planners and planners based on heuristic search tend to have different strengths and weaknesses, so both approaches are very worth trying out in the context of epistemic planning. Similar remarks apply to planning approaches based on symbolic search, *i.e.*, BDD-based exploration. Symbolic search planners have been very successful for optimal planning, where they are as strong as the best heuristic planners (again, as with SAT-based planners, the different approaches excel in different domains). A very good reference for BDD-based search techniques is Torralba’s PhD thesis [100].

4.4.10 Generalized planning

Generalized planning [19, 52, 62, 89, 90] is an extension of classical planning where the goal is, instead of solving a single planning task, to solve a general class of planning tasks (e.g., all blocks world planning tasks) by essentially synthesizing a program with loops. Semantically, one posits a set of possible worlds, where each world is one particular plan instance. A generalized plan, then, is one which is correct for all world states: that is, in each world state, the plan enables the goal and then terminates (by reaching a distinguished stop state).

Typically, as input it accepts a standard planning problem with one exception: a particular parameter for the initial state is left open. For example, in a blocks world setting, the number of blocks on the table may be left open, in which the possible worlds are all possible instantiations of this parameter. The output can be a sequential plan, a conditional plan or a loopy plan. The obvious advantage of plans created by generalized planners is that they have a broad domain coverage. However in order use a generalized planner for a planning problem instance, an applicability test and then (if the test is successful) a plan instantiation are necessary. These procedures, if they have high complexity, may add computational effort to the planner.

4.5 Correspondence Between Planning Problems and Games

Guillaume Aucher (coordinator), Tim French, Kai Li, Sheila McIlraith, Bernhard Nebel, Ramaswamy Ramanujam (coordinator), Yanjing Wang

License © Creative Commons BY 3.0 Unported license
 © Guillaume Aucher, Tim French, Kai Li, Sheila McIlraith, Bernhard Nebel,
 Ramaswamy Ramanujam, Yanjing Wang

The research communities dealing with games and planning problems both deal with similar issues: a number of agents can perform actions in a given context and aim at achieving an objective. However, these two research communities use different vocabularies. For example, a *goal* in a planning problem corresponds to an *objective* in a game; a *plan* or *policy* in a planning problem corresponds to a *strategy* in a game and a *planning domain* corresponds to an *arena* in a game.

Setting rigorous and formal connections between both parties has a number of benefits for both of them. For example, meta-theorems of game theory are of interest to the planning community and heuristic for planning synthesis could be used to synthesize strategies more efficiently. As it turns out, some connections have already been set [37].

The rest of this subsection is organized as follows. First, in Section 4.5.1, we recall the formal setting of games based on graphs. Then, in Section 4.5.2, we recall the *Fully Observable and Non-Deterministic* planning problem (FOND) as it is usually defined in the planning community and we show how it can be reformulated in the setting of games based on graphs. In Section 4.5.3, we discuss what is gained and lost in this translation. Finally, in Section 4.5.4, we discuss which novel perspectives and questions arise in this confrontation between planning problems and games.

4.5.1 Games Based on Graphs

Games are natural models for modeling interaction between multiple agents. In this report, we consider infinite games played on finite graphs [28, 39]. These games are supposed to model reactive systems like an operating system evolving in an arbitrary or hostile environment. The first player represents a program (or “control”, here identified with Player 1), and the second player represents the environment (or “disturbance”, here identified with Player 2). More precisely, we consider zero-sum two-player turn-based games of infinite duration played on finite graphs. This means that when a player wins a game, the other player loses it (“zero-sum”), that players play in turn (“turn-based”), that the game lasts forever (“infinite duration”), and that plays are determined by a finite graph. The graph describes the possible interactions of the system. The game is of infinite duration, because reactive systems are usually not expected to terminate. As recalled by [29], zero-sum graph games have been widely used in the synthesis (or control) of reactive systems, as well as for defining and checking the realizability of specifications, the compatibility of interfaces, simulation relations between transition systems, and for generating test cases.

We introduce the definitions and notations that we will use in the sequel. They are (almost) identical to the definitions and notations of [28, p. 290-292].

Game Graphs

A *game arena* (of imperfect information) is a tuple $G = (L, l_0, \Sigma, \Delta, \mathcal{O}, \gamma)$ where L is the finite set of states, $l_0 \in L$ is the initial state, $\Delta \subseteq L \times \Sigma \times L$ is a set of labeled transitions,

\mathcal{O} is a finite set of observations and $\gamma : \mathcal{O} \rightarrow 2^L - \{\emptyset\}$ maps each observation to the set of states that it represents. We require the following two properties on G :

- *Total*: for all $l \in L$ and all $\sigma \in \Sigma$, there exists $l' \in L$ such that $(l, \sigma, l') \in \Delta$;
- *Partition*: the set $\{\gamma(o) : o \in \mathcal{O}\}$ partitions L .

When the partition is composed of singletons only, the game is a *game of perfect information*. The non-determinism of the transition function is very important since it allows players to play in turn.

Plays and Strategies

In a game arena, at each turn, Player 1 chooses a letter in Σ , and Player 2 resolves nondeterminism by choosing the successor state. A *play* in G is an infinite sequence $\pi = l_0\sigma_0l_1\dots l_n\sigma_nl_{n+1}\dots$ such that for all $i \geq 0$, $(l_i, \sigma_i, l_{i+1}) \in \Delta$. The *prefix up to* l_n of the play π is denoted $\pi(0, n)$; its length is $|\pi(0, n)| = n + 1$; and its last element is $\text{Last}(\pi(0, n)) = l_n$. The *observation sequence* of π is the unique infinite sequence $\gamma^-(\pi) = o_0o_1\dots o_{n-1}o_n\dots$ such that for all $i \geq 0$, we have $l_i \in \gamma(o_i)$. Similarly, the *observation sequence* $\pi(0, n)$ is the prefix up to o_n of $\gamma^-(\pi)$. The set of infinite plays in G is denoted $\text{Plays}(G)$, and the set of corresponding finite prefixes is denoted $\text{Prefs}(G)$.

A *deterministic strategy* in G for Player 1 is a function $\alpha : \text{Prefs}(G) \rightarrow \Sigma$. A strategy α for Player 1 is *observation-based* if for all prefixes $\rho, \rho' \in \text{Prefs}(G)$, if $\gamma^-(\rho) = \gamma^-(\rho')$, then $\alpha(\rho) = \alpha(\rho')$. A strategy for Player 1 is *memory-less* if for all prefixes $\rho, \rho' \in \text{Prefs}(G)$ such that $\text{Last}(\rho) = \text{Last}(\rho')$, $\alpha(\rho) = \alpha(\rho')$. A *deterministic strategy* for Player 2 is a function $\beta : \text{Prefs}(G) \times \Sigma \rightarrow L$ such that for all $\rho \in \text{Prefs}(G)$ and all $\sigma \in \Sigma$, we have $(\text{Last}(\rho), \sigma, \beta(\rho, \sigma)) \in \Delta$. We denote by \mathcal{A}_G the set of all strategies for Player 1.

The *outcome* of two deterministic strategies α (for Player 1) and β (for Player 2) in G is the play $\pi = l_0\sigma_0l_1\dots\sigma_{n-1}l_n\sigma_n\dots \in \text{Plays}(G)$ such that for all $i \geq 0$, we have $\sigma_i = \alpha(\pi(0, i))$ and $l_{i+1} = \beta(\pi(0, i), \sigma_i)$. This play is denoted $\text{outcome}(G, \alpha, \beta)$. The *outcome set* of the deterministic strategy α for Player 1 in G is the set $\text{Outcome}_1(G, \alpha)^\omega$ of plays π such that there exists a deterministic strategy β for Player 2 with $\pi = \text{outcome}(G, \alpha, \beta)$. The set $\text{Outcome}_1(G, \alpha)^*$ denotes the set of finite prefixes of $\text{Outcome}_1(G, \alpha)^\omega$.

Objectives

An *objective* for a game of imperfect information is a set $\mathcal{K} \subseteq (\mathcal{O} \times \Sigma)^\omega$ which is assumed to be Borel measurable: a Borel objective is a Borel set in the Cantor topology on $(\mathcal{O} \times E)^\omega$ [55]. Reachability, safety, Büchi, coBüchi, and parity objectives are all Borel measurable. The parity objectives are a canonical form to express all ω -regular objectives [98]. An objective is induced by a set of target observations $\mathcal{T} \subseteq \mathcal{O}$. Here are some examples of objectives:

- $\text{Reach}_\mathcal{T}^\omega := \{l_0\sigma_0l_1\sigma_1\dots \in \text{Plays}(G) : \text{there is } k \geq 0, \text{ there is } o \in \mathcal{T} \text{ such that } k \in \gamma(o)\}$. The *reachability objective* $\text{Reach}_\mathcal{T}^\omega$ requires that an observation in \mathcal{T} be visited at least once;
- $\text{Safe}_\mathcal{T}^\omega := \{l_0\sigma_0l_1\sigma_1\dots \in \text{Plays}(G) : \gamma^-(l_k) \in \mathcal{T} \text{ for all } k \geq 0\}$. The *safety objective* $\text{Safe}_\mathcal{T}^\omega$ requires that only observations in \mathcal{T} be visited;
- $\text{Buchi}_\mathcal{T}^\omega := \{\pi : \text{Inf}(\pi) \cap \mathcal{T} \neq \emptyset\}$. The *Büchi objective* $\text{Buchi}_\mathcal{T}^\omega$ requires that an observation in \mathcal{T} be visited infinitely often;
- $\text{Parity}(p)^\omega := \{\pi : \min\{p(o) : o \in \text{Inf}(\pi)\} \text{ is even}\}$, where $p : \mathcal{O} \rightarrow \{0, 1, \dots, d\}$ is a *priority function* that maps each observation to a nonnegative integer priority (where $d \in \mathbb{N}$). The *parity objective* $\text{Parity}(p)^\omega$ requires that the minimum priority that appears infinitely often be even.

In this report, we consider only the *reachability objective* $\text{Reach}_{\mathcal{T}}^{\omega}$. A *game* $\mathcal{G} = \{G, \mathcal{T}\}$ is a pair consisting of a game arena and a set of target observations \mathcal{T} . Then, we say that a strategy $\alpha \in \mathcal{A}_G$ is *winning* for $\text{Reach}_{\mathcal{T}}^{\omega}$ when $\text{Outcome}_1(G, \alpha)^{\omega} \subseteq \text{Reach}_{\mathcal{T}}^{\omega}$.

4.5.2 Translation from Planning Problems to Games

Planning problems in the most general sense are specified over some logic (often propositional logic), where the initial situation and the goals are given as formulas and the dynamics of the world is described by action models. These action models induce a transition system that is similar to a game arena. Depending on the form of the initial situation description and the action description language, you end up with different planning problems. In this report, we only consider Fully Observable Non-Deterministic (FOND) planning problems, but we conjecture that the result that we obtain can be extended to other forms of planning problems such as *Partially Observable Non-Deterministic* planning problems (POND) and classical planning problems.

FOND Planning

Following [44], a *Fully Observable Non-Deterministic* (FOND) planning problem consists of a tuple $\langle \mathcal{F}, \mathcal{I}, \mathcal{G}, \mathcal{A} \rangle$, where \mathcal{F} is a set of propositions that we call *fluents*, $\mathcal{I} \subseteq \mathcal{F}$ characterizes what holds in the initial state; $\mathcal{G} \subseteq \mathcal{F}$ characterizes what must hold for the goal to be achieved. Finally \mathcal{A} is the set of actions. The set of literals of \mathcal{F} is $\text{Lits}(\mathcal{F}) = \mathcal{F} \cup \{\neg f \mid f \in \mathcal{F}\}$.

Each action $a \in \mathcal{A}$ is associated with $\langle \text{Pre}_a, \text{Eff}_a \rangle$, where $\text{Pre}_a \subseteq \text{Lits}(\mathcal{F})$ is the precondition

and Eff_a is a set of outcomes of a .

Each outcome $e \in \text{Eff}_a$ is a set of conditional effects, each of the form $(C \rightarrow \ell)$, where $C \subseteq \text{Lits}(\mathcal{F})$ and $\ell \in \text{Lits}(\mathcal{F})$. Given a planning state $s \subseteq \mathcal{F}$ and a fluent $f \in \mathcal{F}$, we say that s satisfies f , denoted $s \models f$ iff $f \in s$. In addition $s \models \neg f$ if $f \notin s$, and $s \models L$ for a set of literals L , if $s \models \ell$ for every $\ell \in L$.

Action a is *applicable* in state s if $s \models \text{Pre}_a$. We say s' is a *result of applying a in s* iff, for one outcome

e in Eff_a , s' is equal to $s \setminus \{p \mid (C \rightarrow \neg p) \in e, s \models C\} \cup \{p \mid (C \rightarrow p) \in e, s \models C\}$.

Unlike classical planning, where a solution is a sequence of actions, a solution to a FOND planning task is a policy that maps a state to an appropriate action (so that the agent eventually reaches the goal). A state s is said to be *reachable* by a policy if the policy can lead the agent to s , and a policy is *closed* if it returns an action for every non-goal state that a policy reaches. When the agent executes a non-deterministic action the effect is randomly chosen, so a closed policy must handle every possible outcome of an action it returns. There are three primary variations of plans for a FOND planning problem [35]: *weak*, *strong*, and *strong cyclic*.

► **Definition 1** (Weak Plan [35]). A *weak plan* is a policy that achieves the goal with at least one possible set of outcomes for the non-deterministic actions.

A weak plan may be as simple as a sequence of actions that achieves the goal with a particular setting of non-deterministic action outcomes. The policy for a weak plan need not be closed.

► **Definition 2** (Strong Plan [35]). A *strong plan* is a closed policy that achieves the goal and never visits the same state twice.

A strong plan provides a guarantee on the maximum number of steps to achieve the goal but is often too restrictive. For example, a strong plan cannot contain an action that can fail with no effect on the state.

► **Definition 3** (Strong Cyclic Plan [35]). A *strong cyclic plan* is a closed policy where the goal is reachable from every state reachable using the policy.

A strong cyclic plan guarantees that the agent eventually reaches the goal, but does not guarantee that the agent can do so in a fixed number of steps. Researchers in the planning community are primarily interested in computing strong cyclic plans, as they are guaranteed to achieve the goal under an assumption of fairness – every outcome of a non-deterministic action will occur infinitely often if the action is executed infinitely often.

► **Remark.** The text in this section is taken near-verbatim from the following two sources: [72, 73].

Translation Theorem

► **Theorem 4.** Given a FOND planning problem $\langle \mathcal{F}, \mathcal{I}, \mathcal{G}, \mathcal{A} \rangle$, one can define an associated game graph $G = (L, l_0, \Sigma, \Delta, \mathcal{O}, \gamma)$ of perfect information and a target observation \mathcal{T} such that:

- $\Sigma = \mathcal{A}$;
- there exists a bijective mapping f between sets of fluents and states of L such that $f(\mathcal{F}) = l_0$ and $f(\mathcal{G}) = \bigcup_{t \in \mathcal{T}} \gamma^{-}(t)$;
- every policy π can be mapped to a (memory-less) strategy in the associated game G , denoted $g(\pi)$, such that for all reachable states $s \subseteq \mathcal{F}$, $f(\pi(s)) = g(\pi)(f(s))$;
- there exists a strong plan π in the original FOND planning problem if, and only if, Player 1 has a winning strategy $g(\pi)$ in the associated game of perfect information G with reachability objective $\text{Reach}_{\mathcal{T}}^{\omega}$.

We conjecture that this theorem can be extended to other forms of planning, such as planning with *partial observability* and even *implicitly coordinated* planning [41].

Muise et al. [75] have informally identified the same relationship between multi-agent games and FOND planning. They modified an existing FOND planner to find strong and weak solutions for first-person multi-agent planning problems. They treated the outcome of other players moves as non-deterministic outcomes of the actions from the first-person perspective, but used standard planning heuristics to prioritize which states to explore given a fixed time budget. Calculating a strong solution for this is equivalent to calculating a winning strategy for the game.

4.5.3 Lost and Gained in Translation

Translating planning problems into games has a number of benefits, but also some shortcomings. On the one hand, the succinctness of the description of a planning problem and in particular of actions is lost when it is translated into a game problem. Moreover, the planning approach allows to provide a more fine-grained and rich description of actions than in the game-theoretical reformulation of the problem, where actions are simply represented as (labeled) transitions between states. On the other hand, one inherits from the game-theoretical formulation some meta-theorems such as determinacy and formal methods to synthesize the plans/winning strategies of the planning problem/game. Moreover, we can also consider other objectives than mere reachability objectives that deal with the overall desired behavior of the system when a plan/strategy is executed.

We should also point out some important differences between planning problems and games. Games are often non-cooperative while in planning the agents (most often) cooperate to achieve a common objective. Moreover, the game arena is usually assumed to be common knowledge among agents, while in planning transition systems are often not explicitly given.

4.5.4 New Perspectives

While the idea of mapping between multi-player co-ordination games and multi-agent planning is to make use of techniques and results from one domain in the other, it is also instructive to examine ideas from one to widen perspectives in the other, in the process generating new interesting questions for study in that domain. Below we list some instances of such possibilities.

Distributed Games

In the context of coordination games against an environment, one interesting question that arises is the existence of not merely a winning strategy for the coalition, but a *distributed* one: these are strategies played out by players locally, based on their partial views of game position. These are imperfect information games and strategy existence is in general undecidable [80, 79] for synchronous executions. The winning conditions in these games are naturally definable in epistemic temporal logics and correspond to distributed plans for multi-agent systems.

In synchronous play, the strategy question is decidable for broadcast environments and hierarchical systems of agents (where if $i \leq j$ and $s \sim_i s' \rightarrow s \sim_j s'$; i is j 's boss, so whatever information is available to j is also available to i). It is also decidable when the goal formulas have local structure. Mapping the decidable subclasses is an interesting exercise.

When players move asynchronously, the strategy question has been shown to be decidable for several subclasses of communication architectures, but the general question is open.

All these questions have an obvious reformulation in the world of multi-agent epistemic planning, and can be answered by mapping planning into games. In the process, we may have new subclasses inspired by plans enriching game theory as well.

Large Games

Games with a large number of players can be called large games. In such games, payoffs are associated not with strategy profiles but with choice distributions. Repeated normal forms of this kind can lead to herd behavior etc. Studying planning in the context of multi-agent systems with a large number of agents might be interesting as well.

Goals, Preferences

In planning, it is customary to distinguish between hard goals (that need to be met definitely) and soft ones (where we have a preference for achieving them over otherwise, but non-achievement is OK). Notions of rationality may be relevant in such goal setting. In game theory, an important question is where utilities arise from and the structure of outcomes. This again suggests a close relationship between plans and games.

Another interesting question in the multi-agent case is coordination by separating 'social' goals from individual agent goals: as long as the social goals are achieved, agents might behave competitively. Such behavior has been studied in game theory and it may be relevant for some planning problems.

References

- 1 Rajeev Alur, Thomas A Henzinger, and Orna Kupferman. Alternating-time temporal logic. *Journal of the ACM (JACM)*, 49(5):672–713, 2002.
- 2 Mikkel Andersen, Thomas Bolander, and Martin Jensen. Conditional epistemic planning. *Logics in Artificial Intelligence*, pages 94–106, 2012.
- 3 Maduka Attamah, Hans van Ditmarsch, Davide Grossi, and Wiebe van der Hoek. Knowledge and gossip. In *ECAI 2014 - 21st European Conference on Artificial Intelligence, 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems (PAIS 2014)*, pages 21–26, 2014.
- 4 Guillaume Aucher. Intricate Axioms as Interaction Axioms. *Studia Logica*, pages 1035–1062, 2015.
- 5 Guillaume Aucher, Bastien Maubert, and Sophie Pinchinat. Automata techniques for epistemic protocol synthesis. *arXiv preprint arXiv:1404.0844*, 2014.
- 6 Jean-François Baffier, Man-Kwun Chiu, Yago Diez, Matias Korman, Valia Mitsou, André van Renssen, Marcel Roeloffzen, and Yushi Uno. Hanabi is np-hard, even for cheaters who look at their cards. *Theor. Comput. Sci.*, 675:43–55, 2017.
- 7 Christel Baier, Joost-Pieter Katoen, and Kim Guldstrand Larsen. *Principles of model checking*. MIT press, 2008.
- 8 A. Baltag and L. Moss. Logics for epistemic programs. *Synthese*, 2004.
- 9 Alexandru Baltag, Lawrence S. Moss, and Slawomir Solecki. The logic of public announcements and common knowledge and private suspicions. In *Proceedings of the 7th Conference on Theoretical Aspects of Rationality and Knowledge (TARK-98), Evanston, IL, USA, July 22-24, 1998*, pages 43–56, 1998.
- 10 Alexandru Baltag and Bryan Renne. Dynamic epistemic logic. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, winter 2016 edition, 2016.
- 11 Alexandru Baltag and Sonja Smets. Probabilistic dynamic belief revision. *Synthese*, 165(2):179, 2008.
- 12 Alexandru Baltag and Sonja Smets. *A Qualitative Theory of Dynamic Interactive Belief Revision*, pages 813–858. Springer International Publishing, Cham, 2016.
- 13 Chitta Baral and Gregory Gelfond. On representing actions in multi-agent domains. In M. Balduccini and T. Son, editors, *Logic programming, knowledge representation, and non-monotonic reasoning*, pages 213–232. Springer, 2011.
- 14 Chitta Baral, Gregory Gelfond, Enrico Pontelli, and Tran Cao Son. An action language for reasoning about beliefs in multi-agent domains. In *Proceedings of NMR*, 2012.
- 15 Chitta Baral, Gregory Gelfond, Enrico Pontelli, and Tran Cao Son. Reasoning about the beliefs of agents in multi-agent domains in the presence of state constraints: The action language mal. In J. Leite, T. C. Son, P. Torroni, L. van der Torre, and S. Woltran, editors, *Proceedings of the 14th International Workshop, Computational Logic in Multi-Agent Systems, CLIMA VIX, Coruna, Spain, September 16-18, 2013*, volume 8143 of *Lecture Notes in Computer Science*, pages 290–306. Springer, 2013.
- 16 Chitta Baral, Gregory Gelfond, Enrico Pontelli, and Tran Cao Son. An Action Language for Multi-Agent Domains: Foundations. *arXiv.org*, page <http://arxiv.org/abs/1511.01960>, 2015.
- 17 Chitta Baral and Michael Gelfond. Representing concurrent actions in extended logic programming. In *IJCAI*, pages 866–873, 1993.
- 18 Vaishak Belle and Gerhard Lakemeyer. Multiagent only knowing in dynamic systems. *Journal of Artificial Intelligence Research*, 49:363–402, 2014.
- 19 Vaishak Belle and Hector J Levesque. Foundations for generalized planning in unbounded stochastic domains. In *KR*, pages 380–389, 2016.

- 20 Daniel S. Bernstein, Robert Givan, Neil Immerman, and Shlomo Zilberstein. The complexity of decentralized control of markov decision processes. *Math. Oper. Res.*, 27(4):819–840, 2002.
- 21 Meghyn Bienvenu, Hélène Fargier, and Pierre Marquis. Knowledge compilation in the modal logic s_5 . In *AAAI*, 2010.
- 22 Thomas Bolander. Seeing is believing: Formalising false-belief tasks in dynamic epistemic logic. In Andreas Herzig and Emiliano Lorini, editors, *Proceedings of the European Conference on Social Intelligence (ECSI-2014)*, volume 1283 of *CEUR Workshop Proceedings*, pages 87–107. CEUR-WS.org, 2014.
- 23 Thomas Bolander and Mikkel Birkegaard Andersen. Epistemic planning for single and multi-agent systems. *Journal of Applied Non-Classical Logics*, 2011.
- 24 Craig Boutilier, Ronen I Brafman, et al. Planning with concurrent interacting actions. In *AAAI/IAAI*, pages 720–726, 1997.
- 25 Laura Bozzelli, Bastien Maubert, and Sophie Pinchinat. Uniform strategies, rational relations and jumping automata. *Information and Computation*, 242:80–107, 2015.
- 26 Ronen I Brafman and Israel Be’er Sheva. A privacy preserving algorithm for multi-agent planning and search. In *IJCAI*, pages 1530–1536, 2015.
- 27 Florent Capelli. *Structural restriction of CNF-formulas: Application to model counting and knowledge compilation*. PhD thesis, Université Paris Diderot, 2016.
- 28 Krishnendu Chatterjee, Laurent Doyen, Thomas A. Henzinger, and Jean-François Raskin. Algorithms for omega-regular games with imperfect information. In *CSL*, pages 287–302, 2006.
- 29 Krishnendu Chatterjee, Thomas A. Henzinger, and Nir Piterman. Strategy logic. *Inf. Comput.*, 208(6):677–693, 2010.
- 30 David Chaum. The dining cryptographers problem: Unconditional sender and recipient untraceability. *J. Cryptology*, 1(1):65–75, 1988.
- 31 Brian F Chellas. *Modal logic: an introduction*. Cambridge university press, 1980.
- 32 Jens Claßen and Gerhard Lakemeyer. Foundations for knowledge-based programs using \mathcal{ES} . In *KR*, pages 318–328, 2006.
- 33 Martin C Cooper, Andreas Herzig, Faustine Maffre, Frédéric Maris, and Pierre Régnier. A simple account of multi-agent epistemic planning. In *ECAI*, pages 193–201, 2016.
- 34 Martin C. Cooper, Andreas Herzig, Faustine Maffre, Frédéric Maris, and Pierre Régnier. Simple epistemic planning: generalised gossiping. *CoRR*, abs/1606.03244, 2016.
- 35 Marco Daniele, Paolo Traverso, and Moshe Y. Vardi. Strong cyclic planning revisited. In *5th European Conference on Planning*, pages 35–48. Springer, 2000.
- 36 Adnan Darwiche and Pierre Marquis. A knowledge compilation map. *Journal of Artificial Intelligence Research*, 17(1):229–264, 2002.
- 37 Giuseppe De Giacomo, Aniello Murano, Sasha Rubin, and Antonio Di Stasio. Imperfect-information games and generalized planning. In Subbarao Kambhampati, editor, *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 1037–1043. IJCAI/AAAI Press, 2016.
- 38 Hans van Ditmarsch. Prolegomena to dynamic logic for belief revision. *Synthese*, 147(2):229–275, 2005.
- 39 Laurent Doyen and Jean-François Raskin. Games with imperfect information: Theory and algorithms. *Lecture Notes in Game Theory for Computer Scientists*, pages 185–212, 2011.
- 40 Thorsten Engesser, Thomas Bolander, Robert Mattmüller, and Bernhard Nebel. Cooperative epistemic multi-agent planning for implicit coordination. *arXiv preprint arXiv:1703.02196*, 2017.
- 41 Thorsten Engesser, Thomas Bolander, Robert Mattmüller, and Bernhard Nebel. Cooperative epistemic multi-agent planning for implicit coordination. In Sujata Ghosh and

- R. Ramanujam, editors, *Proceedings of the Ninth Workshop on Methods for Modalities, M4M@ICLA 2017, Indian Institute of Technology, Kanpur, India, 8th to 10th January 2017.*, volume 243 of *EPTCS*, pages 75–90, 2017.
- 42 Peter Gärdenfors and Hans Rott. Belief revision. In Dov M. Gabbay, C. J. Hogger, and J. A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming (Vol. 4)*, pages 35–132. Oxford University Press, Oxford, UK, 1995.
 - 43 M. Gelfond and V. Lifschitz. Representing actions and change by logic programs. *Journal of Logic Programming*, 17(2,3,4):301–323, 1993.
 - 44 Malik Ghallab, Dana Nau, and Paolo Traverso. *Automated planning: theory & practice*. Elsevier, 2004.
 - 45 Joseph Y Halpern and Judea Pearl. Causes and explanations: A structural-model approach. part ii: Explanations. *The British journal for the philosophy of science*, 56(4):889–911, 2005.
 - 46 Sandra Mitchell Hedetniemi, Stephen T. Hedetniemi, and Arthur L. Liestman. A survey of gossiping and broadcasting in communication networks. *Networks*, 18(4):319–349, 1988.
 - 47 Malte Helmert. The fast downward planning system. *Journal of Artificial Intelligence Research*, 26:191–246, 2006.
 - 48 Malte Helmert and Carmel Domshlak. Landmarks, critical paths and abstractions: What’s the difference anyway? In *Proceedings of the 19th International Conference on Automated Planning and Scheduling (ICAPS)*, 2009.
 - 49 Andreas Herzig, Jérôme Lang, Dominique Longin, and Thomas Polacsek. A logic for planning under partial observability. In *AAAI/IAAI*, pages 768–773, 2000.
 - 50 Andreas Herzig and Faustine Maffre. How to share knowledge by gossiping. *AI Communications*, 30(1):1–17, 2017.
 - 51 Jörg Hoffmann and Bernhard Nebel. The ff planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research*, 14:253–302, 2001.
 - 52 Yuxiao Hu and Giuseppe De Giacomo. A generic technique for synthesizing bounded finite-state controllers. In *ICAPS*, 2013.
 - 53 Xiao Hung, Biqing Gang, Hai Wan, and Yongmei Liu. A general multi-agent epistemic planner based on higher-order belief change. In *IJCAI 2017, Proceedings of the 26th International Joint Conference on Artificial Intelligence, Melbourne, Australia, Sydney, 2017*, 2017.
 - 54 Matthew Johnson, Catholijn M Jonker, M Birna van Riemsdijk, Paul J Feltovich, and Jeffrey M Bradshaw. Joint activity testbed: Blocks world for teams (bw4t). In *ESAW*, volume 9, pages 254–256. Springer, 2009.
 - 55 Alexander S Kechris. *Classical descriptive set theory*, volume 156. Springer-Verlag, New York, 1995.
 - 56 Craig A Knoblock. Generating parallel execution plans with a partial order planner. In *AIPS*, volume 94, pages 98–103, 1994.
 - 57 Filippos Kominis and Hector Geffner. Beliefs in multiagent planning: From one agent to many. In *Proceedings of the Twenty-Fifth International Conference on Automated Planning and Scheduling, ICAPS 2015, Jerusalem, Israel, June 7-11, 2015.*, pages 147–155, 2015.
 - 58 Barteld Kooi and Bryan Renne. Generalized arrow update logic. In *Proceedings of the 13th Conference on Theoretical Aspects of Rationality and Knowledge*, pages 205–211. ACM, 2011.
 - 59 Gerhard Lakemeyer and Hector J Levesque. Situations, si! situation terms, no! In *KR*, pages 516–526, 2004.
 - 60 Gerhard Lakemeyer and Hector J Levesque. Decidable reasoning in a fragment of the epistemic situation calculus. In *KR*, 2014.

- 61 Noël Laverny and Jérôme Lang. From knowledge-based programs to graded belief-based programs, part I: on-line reasoning*. *Synthese*, 147(2):277–321, 2005.
- 62 Hector Levesque. What is planning in the presence of sensing? In *Proceedings of the 13th National Conference on Artificial Intelligence (AAAI)*, pages 1139–1146, Portland, Oregon, 1996.
- 63 Fangzhen Lin and Yoav Shoham. Concurrent actions in the situation calculus. In *AAAI*, volume 92, pages 590–595, 1992.
- 64 Nir Lipovetzky and Hector Geffner. Best-first width search: Exploration and exploitation in classical planning. In *AAAI*, pages 3590–3596, 2017.
- 65 Alessio Lomuscio, Hongyang Qu, and Franco Raimondi. Mcmas: An open-source model checker for the verification of multi-agent systems. *International Journal on Software Tools for Technology Transfer*, 19(1):9–30, 2017.
- 66 Benedikt Löwe, Eric Pacuit, Andreas Witzel, et al. Del planning and some tractable cases. *LORI*, 6953:179–192, 2011.
- 67 Faustine Maffre. *Ignorance is bliss : observability-based dynamic epistemic logics and their applications*. PhD thesis, Université Paul Sabatier - Toulouse III, September 2016.
- 68 Shlomi Maliah, Guy Shani, and Roni Stern. Privacy preserving landmark detection. In *Proceedings of the Twenty-first European Conference on Artificial Intelligence*, pages 597–602. IOS Press, 2014.
- 69 Sheila A. McIlraith and Richard B. Scherl. What sensing tells us: Towards a formal theory of testing for dynamical systems. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI)*, pages 483–490, 2000.
- 70 Tim Miller and Christian J. Muise. Belief update for proper epistemic knowledge bases. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 1209–1215, 2016.
- 71 Fabio Mogavero, Aniello Murano, Giuseppe Perelli, and Moshe Y Vardi. Reasoning about strategies: On the model-checking problem. *ACM Transactions on Computational Logic (TOCL)*, 15(4):34, 2014.
- 72 Christian Muise. *Exploiting Relevance to Improve Robustness and Flexibility in Plan Generation and Execution*. PhD thesis, University of Toronto, 2014.
- 73 Christian Muise, Sheila A. McIlraith, and J. Christopher Beck. Improved Non-deterministic Planning by Exploiting State Relevance. In *The 22nd International Conference on Automated Planning and Scheduling*, The 22nd International Conference on Automated Planning and Scheduling, 2012.
- 74 Christian J Muise, Vaishak Belle, Paolo Felli, Sheila A McIlraith, Tim Miller, Adrian R Pearce, and Liz Sonenberg. Planning over multi-agent epistemic states: A classical planning approach. In *AAAI*, pages 3327–3334, 2015.
- 75 Christian J. Muise, Paolo Felli, Tim Miller, Adrian R. Pearce, and Liz Sonenberg. Planning for a single agent in a multi-agent environment using FOND. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 3206–3212, 2016.
- 76 J. A. M. Nijssen and M. H. M. Winands. Monte-carlo tree search for the game of scotland yard. In *2011 IEEE Conference on Computational Intelligence and Games (CIG'11)*, pages 158–165, Aug 2011.
- 77 Raz Nissim and Ronen I Brafman. Distributed heuristic forward search for multi-agent planning. *J. Artif. Intell. Res.(JAIR)*, 51:293–332, 2014.
- 78 Ronald PA Petrick and Fahiem Bacchus. Extending the knowledge-based approach to planning with incomplete information and sensing. In *ICAPS*, pages 2–11, 2004.
- 79 Amir Pnueli and Roni Rosner. On the synthesis of a reactive module. In *POPL*, pages 179–190, 1989.

- 80 Amir Pnueli and Roni Rosner. On the synthesis of an asynchronous reactive module. In *Automata, Languages and Programming*, pages 652–671. Springer, 1989.
- 81 Raymond Reiter. Natural actions, concurrency and continuous time in the situation calculus. *KR*, 96:2–13, 1996.
- 82 Raymond Reiter. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. MIT Press, Cambridge, MA, 2001.
- 83 Raymond Reiter. On knowledge-based programming with sensing in the situation calculus. *ACM Trans. Comput. Log.*, 2(4):433–457, 2001.
- 84 Jussi Rintanen. Planning as satisfiability: Heuristics. *Artificial Intelligence*, 193:45–86, 2012.
- 85 Jussi Rintanen, Keijo Heljanko, and Ilkka Niemelä. Planning as satisfiability: parallel plans and algorithms for plan search. *Artificial Intelligence*, 170(12-13):1031–1080, 2006.
- 86 Wally Smith, Frank Dignum, and Liz Sonenberg. The construction of impossibility: a logic-based analysis of conjuring tricks. *Frontiers in Psychology*, 7:748, 2016.
- 87 Tran Cao Son, Enrico Pontelli, Chitta Baral, and Gregory Gelfond. Finitary s5-theories. In *European Workshop on Logics in Artificial Intelligence*, pages 239–252. Springer, 2014.
- 88 Tran Cao Son, Enrico Pontelli, Chitta Baral, and Gregory Gelfond. Exploring the kd45 property of a kripke model after the execution of an action sequence. In *AAAI*, pages 1604–1610, 2015.
- 89 Siddharth Srivastava. Foundations and applications of generalized planning. *AI Communications*, 24(4):349–351, 2011.
- 90 Siddharth Srivastava, Neil Immerman, and Shlomo Zilberstein. Computing applicability conditions for plans with loops. In *ICAPS*, pages 161–168, 2010.
- 91 Michal Štolba, Daniel Fišer, and Antonín Komenda. Admissible landmark heuristic for multi-agent planning. In *International Conference on Automated Planning and Scheduling (ICAPS)*, 2015.
- 92 Michal Štolba, Daniel Fišer, and Antonín Komenda. Potential heuristics for multi-agent planning. In *Proceedings of the Twenty-Sixth International Conference on International Conference on Automated Planning and Scheduling*, pages 308–316. AAAI Press, 2016.
- 93 Michal Štolba and Antonín Komenda. Relaxation heuristics for multiagent planning. In *ICAPS*, 2014.
- 94 Michal Štolba, Antonín Komenda, and Daniel L Kovacs. Competition of distributed and multiagent planners (codmap). *The International Planning Competition (WIPC-15)*, 24, 2015.
- 95 Milind Tambe. *Security and game theory: algorithms, deployed systems, lessons learned*. Cambridge University Press, 2011.
- 96 Michael Thielscher. A general game description language for incomplete information games. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*, pages 994–999, 2010.
- 97 Michael Thielscher. Gdl-iii: A description language for epistemic general game playing. In *Proceedings of the International Joint Conference on Artificial Intelligence*, Melbourne, August 2017. AAAI Press.
- 98 Wolfgang Thomas. *Handbook of formal languages, vol. 3: beyond words*, chapter Languages, automata, and logic. Springer-Verlag, New York, 1997.
- 99 Wolfgang Thomas et al. *Automata, logics, and infinite games: a guide to current research*, volume 2500. Springer Science & Business Media, 2002.
- 100 Alvaro Torralba. Symbolic search and abstraction heuristics for cost-optimal planning. *Universidad Carlos III de Madrid*, 2015.
- 101 Alejandro Torreno, Eva Onaindia, and Oscar Sapena. Fmap: Distributed cooperative multi-agent planning. *Applied Intelligence*, 41(2):606–626, 2014.

- 102 Johan van Benthem. Dynamic logic for belief revision. *Journal of Applied Non-Classical Logics*, 17(2):129–155, 2007.
- 103 Johan Van Benthem, Jan Van Eijck, Malvin Gattinger, and Kaile Su. Symbolic model checking for dynamic epistemic logic. In *International Workshop on Logic, Rationality and Interaction*, pages 366–378. Springer, 2015.
- 104 Wiebe Van der Hoek and Michael Wooldridge. Cooperation, knowledge, and time: Alternating-time temporal epistemic logic and its applications. *Studia Logica*, 75(1):125–157, 2003.
- 105 Hans van Ditmarsch. *Knowledge games*. PhD thesis, Groningen University, 2000.
- 106 Hans van Ditmarsch. Epistemic gossip protocols. In *Proceedings of the 1st Chinese Conference on Logic and Argumentation (CLAR 2016), Hangzhou, China, April 2-3, 2016.*, page 1, 2016.
- 107 Hans van Ditmarsch, Davide Grossi, Andreas Herzig, Wiebe van der Hoek, and Louwe B Kuijter. Parameters for epistemic gossip problems. *Proc. LOFT 2016*, 2016.
- 108 Hans van Ditmarsch and Barteld Kooi. *One Hundred Prisoners and a Light Bulb*. Springer, 2015.
- 109 Hans van Ditmarsch, Jan van Eijck, Pere Pardo, Rahim Ramezani, and François Schwarzentruber. Epistemic protocols for dynamic gossip. *J. Applied Logic*, 20:1–31, 2017.
- 110 Hans P van Ditmarsch, Wiebe van der Hoek, and Barteld P Kooi. Concurrent dynamic epistemic logic for mas. In *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, pages 201–208. ACM, 2003.
- 111 Hans P van Ditmarsch, Wiebe Van Der Hoek, Ron Van Der Meyden, and Ji Ruan. Model checking russian cards. *Electronic Notes in Theoretical Computer Science*, 149(2):105–123, 2006.
- 112 J. van Eijck and K. Li. Conditional belief, knowledge and probability. In J. Lang, editor, *Proc. of 16th TARK*, 2017. to appear.
- 113 Jan van Eijck and Yanjing Wang. Propositional dynamic logic as a logic of belief revision. In *Logic, Language, Information and Computation, 15th International Workshop, WoLLIC 2008, Edinburgh, UK, July 1-4, 2008, Proceedings*, pages 136–148, 2008.
- 114 Moshe Y Vardi. Alternating automata and program verification. In *Computer Science Today*, pages 471–485. Springer, 1995.

Participants

- Guillaume Aucher
IRISA – Rennes, FR
- Chitta Baral
Arizona State University –
Tempe, US
- Vaishak Belle
University of Edinburgh, GB
- Thomas Bolander
Technical University of Denmark
– Lyngby, DK
- Tran Cao Son
New Mexico State University, US
- Tristan Charrier
INRIA – Rennes, FR
- Jens Claßen
RWTH Aachen, DE
- Thorsten Engesser
Universität Freiburg, DE
- Esra Erdem
Sabanci University –
Istanbul, TR
- Tim French
The Univ. of Western
Australia, AU
- Malvin Gattinger
University of Amsterdam, NL
- Nina Gierasimczuk
Technical University of Denmark
– Lyngby, DK
- Malte Helmert
Universität Basel, CH
- Andreas Herzig
Paul Sabatier University –
Toulouse, FR
- Mathias Justesen
Technical University of Denmark
– Lyngby, DK
- Ioannis Kokkinis
LORIA – Nancy, FR
- Filippos Kominis
UPF – Barcelona, ES
- Barteld Kooi
University of Groningen, NL
- Louwe B. Kuijter
University of Liverpool, GB
- Jérôme Lang
University Paris-Dauphine, FR
- Yves Lesperance
York University – Toronto, CA
- Kai Li
Peking University, CN
- Yongmei Liu
Sun Yat-sen University –
Guangzhou, CN
- Robert Mattmüller
Universität Freiburg, DE
- Bastien Maubert
University of Napoli, IT
- Sheila McIlraith
University of Toronto, CA
- Timothy Miller
The University of Melbourne, AU
- Bernhard Nebel
Universität Freiburg, DE
- Andrés Occhipinti Liberman
Technical University of Denmark
– Lyngby, DK
- Ron Petrick
Heriot-Watt University –
Edinburgh, GB
- Sophie Pinchinat
IRISA – Rennes, FR
- Ramaswamy Ramanujam
Institute of Mathematical
Sciences – Chennai, IN
- Torsten Schaub
Universität Potsdam, DE
- Richard Scherl
Monmouth Univ. – West Long
Branch, US
- Francois Schwarzenruber
IRISA – ENS Rennes, FR
- Maayan Shvo
Utrecht University, NL
- Sunil Easaw Simon
Indian Institute of Technology
Kanpur, IN
- Michael Thielscher
UNSW – Sydney, AU
- Hans Van Ditmarsch
LORIA – Nancy, FR
- Jan van Eijck
CWI – Amsterdam, NL
- Ivan José Varzinczak
CRIL, University Artois &
CNRS, FR
- Yanjing Wang
Peking University – Beijing, CN
- Bruno Zanuttini
Caen University, FR

