Logic in AI: the case of strategic reasoning

Raul Fervari

[http://cs.famaf.unc.edu.ar/~rfervari | rfervari@unc.edu.ar]

Logics, Interaction and Intelligent Systems Group (LIIS),

FAMAF-UNC / CONICET, Argentina

GTIIT (China)

MCS Colloquium - GTIIT - 2022

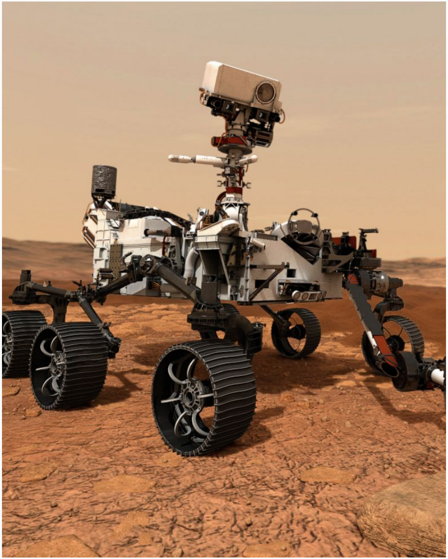
Logic in AI: from the origins

1960 – 2024

- 1962: Knowledge (Hintikka)
- 1966: Planning problem (Simon)
- 1969: Logical AI - Common Sense (McCarthy & Hayes)
- 1980: Non-monotonic reasoning (Reiter / McCarthy / McDermott & Doyle)
- 1984: Mutiple agents (Fagin, Halpern, Vardi)
- 1992: Description Logics (Woods & Schmolze)
- 1996: Logic in NLP (van Benthem & ter Meulen)
- 2015: 'Knowing How' (Y. Wang)
- 2021: Uncertainty-Based KH (Areces, Fervari, Saravia, Velázquez-Quesada)
- 2023: Logics with Abilities and Constrained Plans (Demri & Fervari)

Strategies and Knowledge

- Agents are autonomous entities, acting in a certain environment.
- Have some perception about the real world (epistemic).
- Take a certain course of action for achieving a goal (strategies/abilities).



The concept(s) of knowledge

Y. Wang (2015): *A Logic of Knowing How.* LORI 2015.

- Usually, epistemic logic is about "knowing that":
  - *John knows that it is raining in Shantou*,
  - *the robot knows that it is standing next to a wall...*
- Study other patterns of reasoning: knowing why, knowing whether, knowing who, knowing how.
  - Knowledge + Abilities.

Knowing How

- Autonomous agent: intelligent entities operating in a given environment (perception, decision making, etc).
- Knowing How is related to the abilities of the agents to achieve a certain goal.
- Inspired by AI planning.
- Interpreted as: *there exists a proper course of action (sequence of actions) that the agent can take to achieve the goal.*

Knowing How

- Autonomous agent: intelligent entities operating in a given environment (perception, decision making, etc).
- Knowing How is related to the abilities of the agents to achieve a certain goal.
- Inspired by AI planning.
- Interpreted as: *there exists a proper course of action (sequence of actions) that the agent can take to achieve the goal.*
    - What “proper” means?
    - Different courses of actions?
    - Different costs?

Models: Labelled Transition Systems (LTSs)

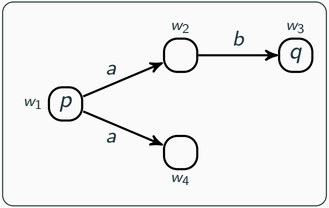
An LTS is a tuple $\mathcal{S} = \langle \mathsf{W}, \{\mathsf{R}_a\}_{a \in \mathsf{Act}}, \mathsf{V} \rangle$ where:

- W is a countable set of states
- $\mathsf{V} : \mathsf{Prop} \to 2^{\mathsf{W}}$
- $\mathsf{R}_a \subseteq \mathsf{W} \times \mathsf{W}$, for each $a \in \mathsf{Act}$.

Models: Labelled Transition Systems (LTSs)

An LTS is a tuple $\mathcal{S} = \langle \mathsf{W}, \{\mathsf{R}_a\}_{a \in \mathsf{Act}}, \mathsf{V} \rangle$ where:

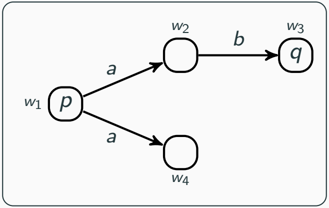
- W is a countable set of states
- $\mathsf{V} : \mathsf{Prop} \to 2^{\mathsf{W}}$
- $\mathsf{R}_a \subseteq \mathsf{W} \times \mathsf{W}$, for each $a \in \mathsf{Act}$.



Models: Labelled Transition Systems (LTSs)

An LTS is a tuple $\mathcal{S} = \langle \mathsf{W}, \{\mathsf{R}_a\}_{a\in\mathsf{Act}}, \mathsf{V}\rangle$ where:

- W is a countable set of states
- $\mathsf{V} : \mathsf{Prop} \to 2^{\mathsf{W}}$
- $\mathsf{R}_a \subseteq \mathsf{W} \times \mathsf{W}$, for each $a \in \mathsf{Act}$.

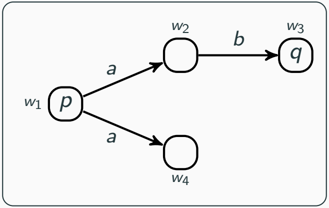


A transition a from w_1 to w_2 is read as "after executing action a at state w_1, the agent reaches state w_2".

Models: Labelled Transition Systems (LTSs)

An LTS is a tuple $\mathcal{S} = \langle \mathsf{W}, \{\mathsf{R}_a\}_{a \in \mathsf{Act}}, \mathsf{V} \rangle$ where:

- W is a countable set of states
- $\mathsf{V} : \mathsf{Prop} \to 2^{\mathsf{W}}$
- $\mathsf{R}_a \subseteq \mathsf{W} \times \mathsf{W}$, for each $a \in \mathsf{Act}$.

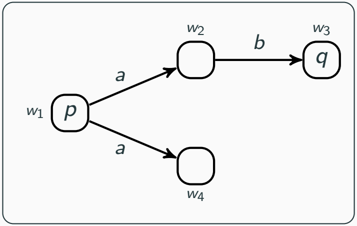


A transition a from w_1 to w_2 is read as "after executing action a at state w_1, the agent reaches state w_2".

For a set of actions Act, a plan σ is an element from Act^* (finite sequences of symbols from Act, such as a, ab and the empty plan ϵ).

Strong executability

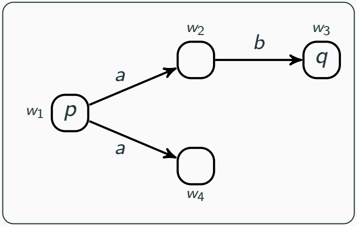
A plan must be fail-proof: each partial execution must be completed.



ab is not SE at w_1

Strong executability

A plan must be fail-proof: each partial execution must be completed.



ab is not SE at w_1

Definition:

A plan σ is strongly executable (SE) at $u \in W$ iff for all partial execution of σ from u, such an execution can be completed.

L_{Kh} over LTS

Definition (Syntax of L_{Kh})

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathsf{Kh}(\varphi, \varphi)$$

$\mathsf{Kh}(\psi, \varphi)$: *"whenever ψ holds, the agent knows how to achieve φ"*.

$\mathsf{L_{Kh}}$ over LTS

Definition (Syntax of $\mathsf{L_{Kh}}$)

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathsf{Kh}(\varphi, \varphi)$$

$\mathsf{Kh}(\psi, \varphi)$: *"whenever ψ holds, the agent knows how to achieve φ"*.

Definition ($\mathsf{L_{Kh}}$ over LTS)

$\mathcal{S}, w \models p \;$ iff $\; w \in \mathsf{V}(p)$

L_{Kh} over LTS

Definition (Syntax of L_{Kh})

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathsf{Kh}(\varphi, \varphi)$$

$\mathsf{Kh}(\psi, \varphi)$: *"whenever ψ holds, the agent knows how to achieve φ"*.

Definition (L_{Kh} over LTS)

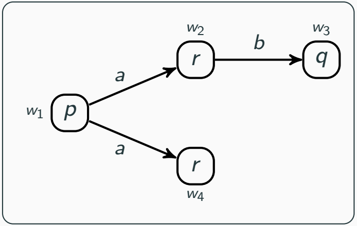
$\mathcal{S}, w \models p$ iff $w \in \mathsf{V}(p)$

$\mathcal{S}, w \models \mathsf{Kh}(\psi, \varphi)$ iff there exists a plan $\sigma \in \mathsf{Act}^*$ such that:

(1) σ is SE at every state satisfying ψ; and,

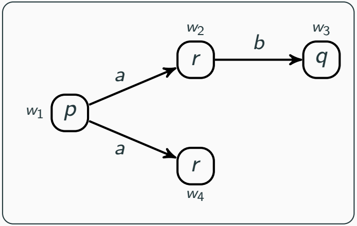
(2) from every ψ-state, executing σ always ends at φ-states.

Example



$\mathcal{S}, w_1 \models \mathsf{Kh}(p, r)$

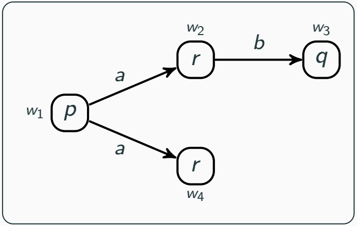
Example



$\mathcal{S}, w_1 \models \mathsf{Kh}(p, r)$

the plan a is SE w_1 (the only p-state), and takes the agent from p only to r-states.

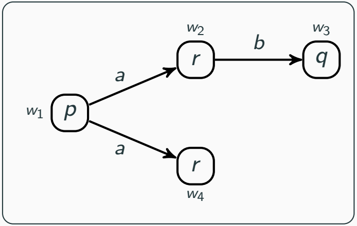
Example



$\mathcal{S}, w_1 \models \mathsf{Kh}(p, r)$
the plan a is SE w_1 (the only p-state), and takes the agent from p only to r-states.

$\mathcal{S}, w_1 \not\models \mathsf{Kh}(p, q)$

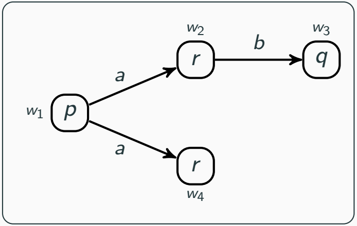
Example



$\mathcal{S}, w_1 \models \mathsf{Kh}(p, r)$
the plan a is SE w_1 (the only p-state), and takes the agent from p only to r-states.

$\mathcal{S}, w_1 \not\models \mathsf{Kh}(p, q)$
- ϵ and a: are SE at w_1 (p-state), but do not lead to q;

Example



$\mathcal{S}, w_1 \models \mathsf{Kh}(p, r)$
the plan a is SE w_1 (the only p-state), and takes the agent from p only to r-states.

$\mathcal{S}, w_1 \not\models \mathsf{Kh}(p, q)$
- ϵ and a: are SE at w_1 (p-state), but do not lead to q;
- ab is **not** SE at w_1.

Conceptual analysis

- The logic reached some consensus in the community.
- Simple language and semantics, and features nice properties (e.g. decidability, axiomatizability).
- But look at these properties:
    - $\mathsf{Kh}(\psi, \chi) \land \mathsf{Kh}(\chi, \varphi)$ implies $\mathsf{Kh}(\psi, \varphi)$

      Should the agent know how to compose plans?

Conceptual analysis

- The logic reached some consensus in the community.
- Simple language and semantics, and features nice properties (e.g. decidability, axiomatizability).
- But look at these properties:
    - $\mathsf{Kh}(\psi, \chi) \land \mathsf{Kh}(\chi, \varphi)$ implies $\mathsf{Kh}(\psi, \varphi)$

      Should the agent know how to compose plans?
    - If $\psi \to \varphi$ holds **everywhere** then $\mathsf{Kh}(\psi, \varphi)$ holds.

      Universal validities should imply knowing how?

Conceptual analysis

- The logic reached some consensus in the community.
- Simple language and semantics, and features nice properties (e.g. decidability, axiomatizability).
- But look at these properties:
  - $\mathsf{Kh}(\psi, \chi) \wedge \mathsf{Kh}(\chi, \varphi)$ implies $\mathsf{Kh}(\psi, \varphi)$

    Should the agent know how to compose plans?
  - If $\psi \to \varphi$ holds **everywhere** then $\mathsf{Kh}(\psi, \varphi)$ holds.

    Universal validities should imply knowing how?
- Moreover, in this setting, abilities = knowledge.

Conceptual analysis

- The logic reached some consensus in the community.
- Simple language and semantics, and features nice properties (e.g. decidability, axiomatizability).
- But look at these properties:
  - $\mathsf{Kh}(\psi,\chi) \land \mathsf{Kh}(\chi,\varphi)$ implies $\mathsf{Kh}(\psi,\varphi)$

    Should the agent know how to compose plans?
  - If $\psi \rightarrow \varphi$ holds **everywhere** then $\mathsf{Kh}(\psi,\varphi)$ holds.

    Universal validities should imply knowing how?
- Moreover, in this setting, abilities = knowledge.

Arguably, this is a logic of knowing how, but not an **epistemic** logic of knowing how.

Towards an epistemic logic of knowing how

There are many reasons to **not** knowing how. What if...

- The agent is not aware of the existence of certain plans?

Towards an epistemic logic of knowing how

There are many reasons to **not** knowing how. What if...

- The agent is not aware of the existence of certain plans?
- The agent is not able to distinguish certain plan from another?

Towards an epistemic logic of knowing how

There are many reasons to **not** knowing how. What if...

- The agent is not aware of the existence of certain plans?
- The agent is not able to distinguish certain plan from another?
- The agent does not care about the difference among certain plans?

Towards an epistemic logic of knowing how

There are many reasons to **not** knowing how. What if...

- The agent is not aware of the existence of certain plans?
- The agent is not able to distinguish certain plan from another?
- The agent does not care about the difference among certain plans?

We introduced the notion of epistemic indistinguishability at the level of plans, to fix these issues.

Knowing How + Uncertainty

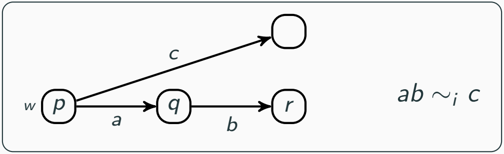
C. Areces, R. Fervari, A. Saravia, F. Velázquez-Quesada. *Uncertainty-Based Semantics for Multi-Agent Knowing How Logics. (TARK 2021).*

https://cs.famaf.unc.edu.ar/~rfervari/files/papers/2021-tark.pdf

Uncertainty-based LTS (LTS^{U})

An LTS^{U} is a tuple $\mathcal{M} = \langle \mathrm{W}, \{\mathrm{R}_a\}_{a \in \mathrm{Act}}, \{\sim_i\}_{i \in \mathrm{Agt}}, \mathrm{V} \rangle$ where:

- $\langle \mathrm{W}, \{\mathrm{R}_a\}_{a \in \mathrm{Act}}, \mathrm{V} \rangle$ is an LTS,
- $\sim_i$ is an equivalence relation over a non-empty set of plans, for each $i \in \mathrm{Agt}$ (a set of agent symbols).



(Multi-Agent) semantics over LTS^U

Definition ($\mathsf{L}_{\mathsf{Kh}_i}$ over LTS^U).

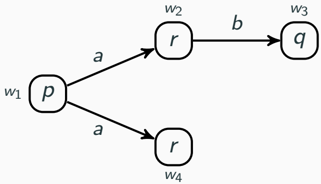
Let S_i be the set of equivalence classes (over plans) by $\sim_i$.

$\mathcal{M}, w \models \mathsf{Kh}_i(\psi, \varphi)$ iff there exists a set of plans $\pi \in \mathsf{S}_i$ such that:

(1) each plan in π is SE at every ψ-state; and

(2) from ψ-states, each plan in π always ends at φ-states.

Example

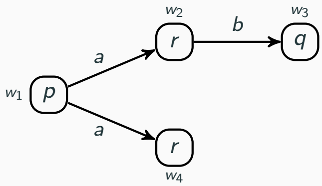


$a \sim_i ab \; (S_i = \{\{a, ab\}\})$

$a \sim_j a, \;\; ab \sim_j ab \; (S_j = \{\{a\}, \{ab\}\})$

$\mathcal{M}, w_1 \models \neg \mathsf{Kh}_i(p, r)$

Example



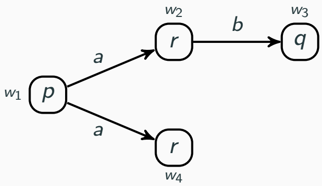
$a \sim_i ab$ ($S_i = \{\{a, ab\}\}$)

$a \sim_j a,\; ab \sim_j ab$ ($S_j = \{\{a\}, \{ab\}\}$)

$\mathcal{M}, w_1 \models \neg \mathsf{Kh}_i(p, r)$

take $\pi = \{a, ab\}$:

Example



$a \sim_i ab$ ($S_i = \{\{a, ab\}\}$)

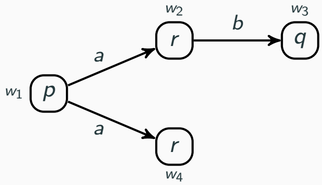
$a \sim_j a$, $ab \sim_j ab$ ($S_j = \{\{a\}, \{ab\}\}$)

$\mathcal{M}, w_1 \models \neg \mathsf{Kh}_i(p, r)$

take $\pi = \{a, ab\}$:

- a is SE at w_1 (p-state), and takes from p-states to r-states.

Example



$a \sim_i ab \; (S_i = \{\{a, ab\}\})$

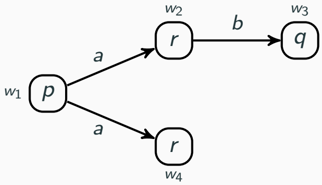
$a \sim_j a, \; ab \sim_j ab \; (S_j = \{\{a\}, \{ab\}\})$

$\mathcal{M}, w_1 \models \neg \mathsf{Kh}_i(p, r)$

take $\pi = \{a, ab\}$:

- a is SE at w_1 (p-state), and takes from p-states to r-states.
- ab is not SE at w_1 (p-state).

Example



$a \sim_i ab$ ($S_i = \{\{a, ab\}\}$)

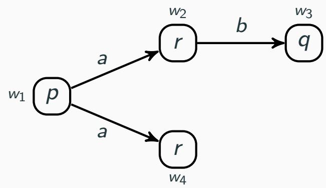
$a \sim_j a, \; ab \sim_j ab$ ($S_j = \{\{a\}, \{ab\}\}$)

$\mathcal{M}, w_1 \models \neg \mathsf{Kh}_i(p, r)$

take $\pi = \{a, ab\}$:

- a is SE at w_1 (p-state), and takes from p-states to r-states.
- ab is not SE at w_1 (p-state).
- thus, $\pi = \{a, ab\}$ is not SE at w_1.

Example



$a \sim_i ab$ ($S_i = \{\{a, ab\}\}$)

$a \sim_j a, \; ab \sim_j ab$ ($S_j = \{\{a\}, \{ab\}\}$)

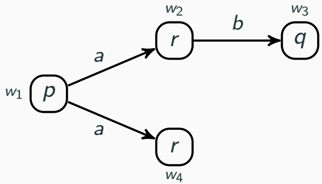
$\mathcal{M}, w_1 \models \neg \mathsf{Kh}_i(p, r)$

take $\pi = \{a, ab\}$:

- a is SE at w_1 (p-state), and takes from p-states to r-states.
- ab is not SE at w_1 (p-state).
- thus, $\pi = \{a, ab\}$ is not SE at w_1.

$\mathcal{M}, w_1 \models \mathsf{Kh}_j(p, r)$

Example



$a \sim_i ab$ ($S_i = \{\{a, ab\}\}$)

$a \sim_j a,\ ab \sim_j ab$ ($S_j = \{\{a\}, \{ab\}\}$)

$\mathcal{M}, w_1 \models \neg \mathsf{Kh}_i(p, r)$

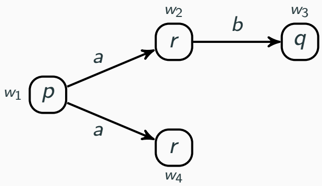
take $\pi = \{a, ab\}$:

- a is SE at w_1 (p-state), and takes from p-states to r-states.
- ab is not SE at w_1 (p-state).
- thus, $\pi = \{a, ab\}$ is not SE at w_1.

$\mathcal{M}, w_1 \models \mathsf{Kh}_j(p, r)$

take $\pi' = \{a\}$:

Example



$a \sim_i ab$ ($S_i = \{\{a, ab\}\}$)

$a \sim_j a$, $ab \sim_j ab$ ($S_j = \{\{a\}, \{ab\}\}$)

$\mathcal{M}, w_1 \models \neg\mathsf{Kh}_i(p, r)$

take $\pi = \{a, ab\}$:

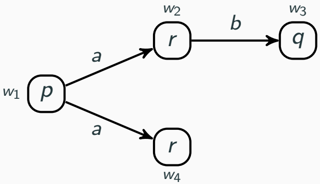
- a is SE at w_1 (p-state), and takes from p-states to r-states.
- ab is not SE at w_1 (p-state).
- thus, $\pi = \{a, ab\}$ is not SE at w_1.

$\mathcal{M}, w_1 \models \mathsf{Kh}_j(p, r)$

take $\pi' = \{a\}$:

- a is SE at w_1 (p-state), and takes from p-states to r-states.

Example



$a \sim_i ab$ ($S_i = \{\{a, ab\}\}$)

$a \sim_j a, \; ab \sim_j ab$ ($S_j = \{\{a\}, \{ab\}\}$)

$\mathcal{M}, w_1 \models \neg \mathsf{Kh}_i(p, r)$

take $\pi = \{a, ab\}$:

- a is SE at w_1 (p-state), and takes from p-states to r-states.
- ab is not SE at w_1 (p-state).
- thus, $\pi = \{a, ab\}$ is not SE at w_1.

$\mathcal{M}, w_1 \models \mathsf{Kh}_j(p, r)$

take $\pi' = \{a\}$:

- a is SE at w_1 (p-state), and takes from p-states to r-states.
- thus, $\pi' = \{a\}$ works as a witness.

LTS vs. $\mathsf{LTS}^{\mathsf{U}}$ approaches

- Over LTS, there is **no distinction** between abilities and knowledge.

LTS vs. $\mathsf{LTS^U}$ approaches

- Over LTS, there is **no distinction** between abilities and knowledge.
- Over $\mathsf{LTS^U}$:

LTS vs. LTS$^{\mathsf{U}}$ approaches

- Over LTS, there is **no distinction** between abilities and knowledge.
- Over LTS$^{\mathsf{U}}$:
    - **abilities** are given by the LTS (common to all agents);

LTS vs. $\mathsf{LTS}^{\mathsf{U}}$ approaches

- Over LTS, there is **no distinction** between abilities and knowledge.
- Over $\mathsf{LTS}^{\mathsf{U}}$:
  - **abilities** are given by the LTS (common to all agents);
  - **knowledge** is determined by $\sim_i$ (individual, own perception).

LTS vs. $\mathsf{LTS^U}$ approaches

- Over LTS, there is **no distinction** between abilities and knowledge.
- Over $\mathsf{LTS^U}$:
  - **abilities** are given by the LTS (common to all agents);
  - **knowledge** is determined by $\sim_i$ (individual, own perception).
- $\mathsf{Kh}(\psi, \chi) \wedge \mathsf{Kh}(\chi, \varphi) \rightarrow \mathsf{Kh}(\psi, \varphi)$ no longer valid;
- $\psi \rightarrow \varphi$ true **everywhere** does not entail $\mathsf{Kh}(\psi, \varphi)$ anymore.

LTS vs. LTS^{U} approaches

- Over LTS, there is **no distinction** between abilities and knowledge.
- Over LTS^{U}:
  - **abilities** are given by the LTS (common to all agents);
  - **knowledge** is determined by $\sim_i$ (individual, own perception).
- $\text{Kh}(\psi, \chi) \land \text{Kh}(\chi, \varphi) \rightarrow \text{Kh}(\psi, \varphi)$ no longer valid;
- $\psi \rightarrow \varphi$ true **everywhere** does not entail $\text{Kh}(\psi, \varphi)$ anymore.
- Over LTS^{U}, we preserve good properties, even improve some features (good computational complexity).
  - Checking satisfiability of a formula is NP-complete.
  - Checking whether $\mathcal{M}, w \models \varphi$? can be done in polynomial time.

On the agenda

- Dynamic Operations: how to update each agent's "knowing how"?

  [Dalí 2022 - Areces, Fervari, Saravia, Velázquez-Quesada]

On the agenda

- Dynamic Operations: how to update each agent's "knowing how"?

  [Dalí 2022 - Areces, Fervari, Saravia, Velázquez-Quesada]

- Knowing how + knowing that: interaction between types of knowledge.

  [IJCAI 2017 - Fervari, Herzig, Li, Wang]

  [Work in progress - Saravia's thesis]

On the agenda

- Dynamic Operations: how to update each agent's "knowing how"?

  [Dalí 2022 - Areces, Fervari, Saravia, Velázquez-Quesada]

- Knowing how + knowing that: interaction between types of knowledge.

  [IJCAI 2017 - Fervari, Herzig, Li, Wang]

  [Work in progress - Saravia's thesis]

- Deontic interpretations: reasoning about normative systems.

  [Paper submitted - Areces, Cassano, Castro, Fervari, Saravia]

On the agenda

- Dynamic Operations: how to update each agent's "knowing how"?

  [Dalí 2022 - Areces, Fervari, Saravia, Velázquez-Quesada]

- Knowing how + knowing that: interaction between types of knowledge.

  [IJCAI 2017 - Fervari, Herzig, Li, Wang]

  [Work in progress - Saravia's thesis]

- Deontic interpretations: reasoning about normative systems.

  [Paper submitted - Areces, Cassano, Castro, Fervari, Saravia]

- Constrained plans: revisit the definition of a "proper" plan.
  - (Infinite) equivalence classes as finite state automata.
  - Budgets (actions with costs).

  [AAAI-23 - Demri & Fervari]

On the agenda

- Dynamic Operations: how to update each agent's "knowing how"?

  [Dalí 2022 - Areces, Fervari, Saravia, Velázquez-Quesada]

- Knowing how + knowing that: interaction between types of knowledge.

  [IJCAI 2017 - Fervari, Herzig, Li, Wang]

  [Work in progress - Saravia's thesis]

- Deontic interpretations: reasoning about normative systems.

  [Paper submitted - Areces, Cassano, Castro, Fervari, Saravia]

- Constrained plans: revisit the definition of a "proper" plan.
  - (Infinite) equivalence classes as finite state automata.
  - Budgets (actions with costs).

    [AAAI-23 - Demri & Fervari]
  - Non-deterministic plans, state-dependent costs/plans.

    [Future work - Demri & Fervari]

Recent work

Model-Checking for Ability-Based Logics with Constrained Plans

Stéphane Demri[1], **Raul Fervari**[2,3]

[1]Université Paris-Saclay, CNRS, ENS Paris-Saclay,
Laboratoire Méthodes Formelles, 91190, Gif-Sur-Yvette, France
[2]FAMAF, Universidad Nacional de Córdoba & CONICET, Argentina
[3]Guangdong Technion - Israel Institute of Technology, China

Abstract

We investigate the complexity of the model-checking problem for a family of modal logics capturing the notion of "knowing how". We consider the most standard ability-based knowing how logic, for which we show that model-checking is PSpace-complete. By contrast, a multi-agent variant based on an uncertainty relation between plans in which uncertainty is encoded by a regular language, is shown to admit a PTime model-checking problem. We extend with budgets the above-mentioned ability-logics, as done for ATL-like logics. We show that for the former logic enriched with budgets, the complexity increases to at least ExpSpace-hardness, whereas for the latter, the PTime bound is preserved. Other variant logics are discussed along the paper.

Introduction

Knowing How Logics. The epistemic concept of "knowing how" has received considerable attention lately, as a new

and Weld 1998)). Thus, "knowing how" is given by the abilities described by the graph. The simplicity of the logical language is partly reflected by the fact that formulae of the form $\mathsf{Kh}(p, q)$ are global, no action symbol appears in formulae, a single agent is considered, and no "knowing that" modality is present. A complete axiomatisation is provided in (Wang 2018b) but more importantly, such a work has been a source of inspiration for many others. Some variants include: multiple agents, other classes of plans, or admit "knowing that" operators (see e.g. (Fervari et al. 2017; Li and Wang 2021b)). Other approaches, related to strategic games and coalitions, have been studied in (Naumov and Tao 2018c,a,b). Finally, the logic studied in (Areces et al. 2021) (called herein $\mathcal{L}^{\mathrm{U}}$) is based on a notion of indistinguishability over plans. Arguably, such a proposal provides a more epistemic view of knowing how than other approaches.

Substantial progress has been already done related to philosophical motivations, axiom systems and combinations with other epistemic operators. However, much less contri-

**Accepted for publication at the 37th AAAI Conference on Artificial Intelligence (AAAI-23), 2023.*

Thanks!