

Strategy Logic, Imperfect Information, and Hyperproperties

Raven Beutner, Bernd Finkbeiner

CISPA Helmholtz Center for Information Security, Germany

Abstract

Strategy logic (SL) is a powerful temporal logic that enables first-class reasoning over strategic behavior in multi-agent systems (MAS). In many MASs, the agents (and their strategies) cannot observe the global state of the system, leading to many extensions of SL centered around imperfect information, such as *strategy logic with imperfect information* (SL_{ii}). Along orthogonal lines, researchers have studied the combination of strategic behavior and hyperproperties. Hyperproperties are system properties that relate multiple executions in a system and commonly arise when specifying security policies. Hyper Strategy Logic (HyperSL) is a temporal logic that combines quantification over strategies with the ability to express hyperproperties on the executions of different strategy profiles. In this paper, we study the relation between SL_{ii} and HyperSL. Our main result is that both logics (restricted to formulas where no state formulas are nested within path formulas) are *equivalent* in the sense that we can encode SL_{ii} instances into HyperSL instances and vice versa. For the former direction, we build on the well-known observation that imperfect information is a hyperproperty. For the latter direction, we construct a self-composition of MASs and show how we can simulate hyperproperties using imperfect information.

1 Introduction

Multi-agent systems (MAS) are ubiquitous in our everyday lives, necessitating the need for formal guarantees on their behavior. In MASs, we typically reason about the ability of groups of agents, which requires reasoning about *strategies*. This led to the development of powerful temporal logics like ATL/ATL* (Alur, Henzinger, and Kupferman 2002) and Strategy Logic (SL) (Chatterjee, Henzinger, and Piterman 2010; Mogavero et al. 2014). While the former can implicitly reason about strategic ability (e.g., a group of agents has *some* strategy to enforce a certain goal), the latter features *explicit* quantification over strategy, allowing the same strategy to be used in multiple contexts, which is critical to express important properties like Nash equilibria.

Strategy Logic With Imperfect Information In plain SL, the strategies of agents can observe the entire state of the MAS (Mogavero et al. 2014). In most models of real-world situations, this is unrealistic, i.e., an agent typically acts on some local sensing ability and must thus act under incomplete information. This observation led to many logics that

can reason about strategic behavior under imperfect information (Knight and Maubert 2019; Belardinelli et al. 2017; Berthon, Maubert, and Murano 2017). A particularly powerful logic among these is SL with imperfect information (SL_{ii}) (Berthon et al. 2017), which extends SL with the ability to quantify over strategies with a given observation model. For example,

$$\exists x^o. \forall y^{o'}. (1 \blacktriangleright x)(2 \blacktriangleright y)(3 \blacktriangleright y) \text{ GF } goal$$

expresses that there exists some strategy with observation model o (formally, o is associated with an indistinguishability relation on states of the MAS), such that for every strategy y under observation o' , the play where agent 1 plays strategy x , and agents 2 and 3 play y (i.e., the play under strategy profile $(1 \mapsto x, 2 \mapsto y, 3 \mapsto y)$) satisfies $\text{GF } goal$.

Hyper Strategy Logic Along orthogonal lines, SL has been extended with the concept of *hyperproperties* (Clarkson and Schneider 2008), i.e., properties that relate multiple paths in a system. In plain SL, we can use the same strategy in different situations, but each strategy profile is evaluated against an LTL formula. As a result, we can only express properties on individual strategy profiles and then reason about Boolean combinations of these properties. For many properties (e.g., security and robustness policies), we need to compare *multiple* executions to, e.g., see how different high-security inputs impact the low-security observations of a system. HyperSL (Beutner and Finkbeiner 2024a), extends SL with the concept of path variables (similar to logics like HyperCTL* (Clarkson et al. 2014)). For example,

$$\exists x. \forall y. \forall z. ((\neg g_{\pi_2}) \cup g_{\pi_1}) \left[\begin{array}{l} \pi_1 : (1 \mapsto x, 2 \mapsto x, 3 \mapsto z) \\ \pi_2 : (1 \mapsto x, 2 \mapsto y, 3 \mapsto z) \end{array} \right]$$

states that there exists some strategy x (under full information), such that for all y, z , the strategy profile $(1 \mapsto x, 2 \mapsto x, 3 \mapsto z)$ reaches a goal g at least as fast as profile $(1 \mapsto x, 2 \mapsto y, 3 \mapsto z)$. To express this, we construct *two* paths π_1, π_2 using different strategy profiles, and, within the LTL body, can refer to atomic propositions on both paths.

The Connection At first glance, SL_{ii} and HyperSL appear orthogonal. The former is centered around the information

of agents, whereas the latter is targeted at quantitative properties and security policies. However, in this paper, we show that both logics (restricted to formulas where no state formulas are nested within path formulas) are equally expressive in the sense that we can translate model-checking instances between both. This connection allows for a unified study of (imperfect) knowledge and hyperproperties, and potentially allows the transfer of decidability results between SL_{ii} and HyperSL (model-checking is undecidable for both logics). To translate SL_{ii} into HyperSL (Section 3), we build on the well-known observation that a strategy acting under imperfect knowledge *is* a hyperproperty, i.e., the strategy should pick the same action on all *pairs* of executions that appear indistinguishable. For the latter direction (Section 4), we simulate hyperproperties within SL_{ii} . Our key observation here is that we can construct the self-composition of an MAS and, using imperfect information, simulate multiple executions within this composition.

2 Preliminaries

We let AP be a fixed finite set of atomic propositions and fix a finite set of agents $Agts = \{1, \dots, n\}$.

Concurrent Game Structures As the underlying model of MASs, we use concurrent game structures (CGS) (Alur, Henzinger, and Kupferman 2002). A CGS is a tuple $\mathcal{G} = (S, s_0, \mathbb{A}, \kappa, L)$ where S is a finite set of states, $s_0 \in S$ is an initial state, $\mathbb{A}$ is a finite set of actions, $\kappa : S \times (Agts \rightarrow \mathbb{A}) \rightarrow S$ is a transition function, and $L : S \rightarrow 2^{AP}$ is a labeling function. The transition function takes a state s and an action profile $\vec{\alpha} : Agts \rightarrow \mathbb{A}$ and returns a unique successor state $\kappa(s, \vec{\alpha})$. A strategy in $\mathcal{G}$ is a function $f : S^+ \rightarrow \mathbb{A}$, mapping finite plays to actions. We denote the set of all strategies in $\mathcal{G}$ with $Str(\mathcal{G})$. Once we fix a strategy for each agent, we obtain a unique path in the CGS (cf. the full version (Beutner and Finkbeiner 2025c)).

SL_{ii} SL_{ii} (Berthon et al. 2017) extends plain SL by quantifying over strategies with a given observation model. Let $\mathcal{G} = (S, s_0, \mathbb{A}, \kappa, L)$ be a fixed CGS, and let Obs be a fixed finite set of so-called observations. An observation family $\{\sim_o\}_{o \in Obs}$ associates each $o \in Obs$ with an equivalence relation $\sim_o \subseteq S \times S$. For a strategy with observation o , two states $s \sim_o s'$ appear identical. This naturally extends to finite plays: Two finite plays $p, p' \in S^+$ are o -indistinguishable, written $p \sim_o p'$, if $|p| = |p'|$ and for each $0 \leq i < |p|$, $p(i) \sim_o p'(i)$. An o -strategy is a function $f : S^+ \rightarrow \mathbb{A}$ that cannot distinguish between o -indistinguishable plays, i.e., for all $p, p' \in S^+$ with $p \sim_o p'$ we have $f(p) = f(p')$. We denote with $Str(\mathcal{G}, o)$ the set of all o -strategies in $\mathcal{G}$. Now, assume that $\mathcal{X} = \{x, y, \dots\}$ is a set of *strategy variables*. We consider SL_{ii} formulas that are generated by the following grammar:

$$\begin{aligned} \psi &:= a \mid \neg\psi \mid \psi \wedge \psi \mid \mathbf{X}\psi \mid \psi \mathbf{U} \psi \\ \varphi &:= \psi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \forall x^o. \varphi \mid \exists x^o. \varphi \mid (i \blacktriangleright x) \varphi \end{aligned}$$

where $a \in AP$, $x \in \mathcal{X}$, $i \in Agts$, and $o \in Obs$ is an observation. We use the usual Boolean connectives $\vee, \rightarrow, \leftrightarrow$, and

constants $\top, \perp$, as well as the derived LTL operators *eventually* $\mathbf{F}\psi$, *globally* $\mathbf{G}\psi$, and *weak until* $\psi_1 \mathbf{W} \psi_2$. Note that we do not allow nested state formulas within path formulas. In SL_{ii} , quantification ($\forall x^o$ and $\exists x^o$) ranges over a strategy with fixed observation o . The agent binding construct $(i \blacktriangleright x)\varphi$, then evaluates φ after binding agent i to some previously quantified strategy x (cf. (Mogavero et al. 2014)). The semantics of SL_{ii} is defined as expected: We maintain a partial mapping $\Delta : \mathcal{X} \rightarrow Str(\mathcal{G})$ that maps strategy variables to strategies, and a mapping $\Theta : Agts \rightarrow Str(\mathcal{G})$ mapping agents to strategies. Whenever we evaluate $\forall x^o$ or $\exists x^o$, we quantify over a strategy in $Str(\mathcal{G}, o)$ and add it to Δ ; when evaluating an agent binding $(i \blacktriangleright x)$, we update Θ by mapping agent i to strategy $\Delta(x)$. Once we reach a path formula ψ , we check if the path resulting from the strategy profile Θ satisfies the LTL formula ψ . We give the full semantics in the full version (Beutner and Finkbeiner 2025c). We write $(\mathcal{G}, \{\sim_o\}_{o \in Obs}) \models_{SL_{ii}} \varphi$ if φ holds in $\mathcal{G}, \{\sim_o\}_{o \in Obs}$.

Hyper Strategy Logic HyperSL (Beutner and Finkbeiner 2024a) is centered around the idea of combining strategic reasoning (as possible in SL) with the ability to express hyperproperties (as possible in logics such as HyperCTL* (Clarkson et al. 2014)). In addition to the strategy variables $\mathcal{X}$, we assume that $\mathcal{V} = \{\pi, \pi_1, \dots\}$ is a set of *path variables*. Path and state formulas in HyperSL are generated by the following grammar:

$$\begin{aligned} \psi &:= a_\pi \mid \neg\psi \mid \psi \wedge \psi \mid \mathbf{X}\psi \mid \psi \mathbf{U} \psi \\ \varphi &:= \forall x. \varphi \mid \exists x. \varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \psi[\pi_k : \vec{x}_k]_{k=1}^m \end{aligned}$$

where $a \in AP$, $\pi, \pi_1, \dots, \pi_m \in \mathcal{V}$ are path variables, $x \in \mathcal{X}$, and $\vec{x}_1, \dots, \vec{x}_m : Agts \rightarrow \mathcal{X}$ are strategy profiles that assign a strategy variable to each agent.

In HyperSL, we can quantify over strategies (under full information), and evaluate LTL formulas on *multiple* paths at the same time (thus expressing hyperproperties). Formally, $\psi[\pi_k : \vec{x}_k]_{k=1}^m$ expresses a hyperproperty over m paths in the CGS, where the k th path (bound to π_k) is the unique play where each agent i plays strategy $\vec{x}_k(i)$. In the LTL formula ψ , we then use path-variable-indexed atomic propositions: The formula a_π holds iff AP a holds on the path bound to path variable π . HyperSL thus allows us to express *temporal* properties on multiple strategy profiles at the same time. We refer the reader to (Beutner and Finkbeiner 2024a) for details. The semantics of HyperSL is defined as expected: Similar to SL_{ii} , we collect all strategies in a mapping $\Delta : \mathcal{X} \rightarrow Str(\mathcal{G})$. When evaluating $\psi[\pi_k : \vec{x}_k]_{k=1}^m$, we then define π_k to be the unique path where each agent $i \in Agts$ plays strategy $\Delta(\vec{x}_k(i))$, and evaluate the LTL formula on the resulting m paths $\pi_1, \dots, \pi_m$. We give a full semantics in the full version (Beutner and Finkbeiner 2025c). We write $\mathcal{G} \models \varphi$ if $\mathcal{G}$ satisfies φ .

3 Encoding SL_{ii} Into HyperSL

Note that the underlying game structure differs between both logics (HyperSL is defined on plain CGSs and SL_{ii} on CGSs with an observation model), so our encodings have to modify both the formula and the underlying system.

In this section, we show that we can encode SL_{ii} into HyperSL by recalling the encoding from (Beutner and Finkbeiner 2024a). Our encoding is based on the well-known observation that acting under imperfect information is a hyperproperty (Bozzelli, Maubert, and Pinchinat 2015).

Injective Labeling and Action Recording Strategies are defined as functions $S^+ \rightarrow \mathbb{A}$, and $\sim_o$ is defined as a relation on states, i.e., both are defined directly on components of the game structure. In contrast, *within* our logic, we only observe the evaluation of the atomic propositions. In the first step, we thus modify the game structure to provide sufficient information within its atomic propositions.

Definition 1. A CGS $\mathcal{G} = (S, s_0, \mathbb{A}, \kappa, L)$ is injectively labeled (IL) if $L : S \rightarrow 2^{AP}$ is injective. The CGS is action recording (AR) if for each agent $i \in Agts$ and every action $\alpha \in \mathbb{A}$, there exists an AP $\langle i, \alpha \rangle \in AP$ that holds in a state iff agent i played action α in the last step.

We can easily modify any CGS to be IL and AR (note that SL_{ii} model-checking is undecidable):

Lemma 1. Given an SL_{ii} MC instance $(\mathcal{G}, \{\sim_o\}_{o \in Obs}, \varphi)$ there exists an effectively computable SL_{ii} instance $(\mathcal{G}', \{\sim'_o\}_{o \in Obs}, \varphi')$ where (1) $\mathcal{G}'$ is IL and AR, and (2) $(\mathcal{G}, \{\sim_o\}_{o \in Obs}) \models_{SL_{ii}} \varphi$ iff $(\mathcal{G}', \{\sim'_o\}_{o \in Obs}) \models_{SL_{ii}} \varphi'$.

Now assume some fixed CGS $\mathcal{G} = (S, s_0, \mathbb{A}, \kappa, L)$, observation family $\{\sim_o\}_{o \in Obs}$, and SL_{ii} formula φ . Using Lemma 1, we assume, w.l.o.g., that $\mathcal{G}$ is IL and AR.

Enforcing Imperfect Information First, we construct a formula that identifies pairs of states that are indistinguishable according to $\sim_o$. For $o \in Obs$, we define the HyperSL path formula ind_o over path variables π_1, π_2 as follows:

$$ind_o := \bigvee_{(s, s') \in \sim_o} \left(\bigwedge_{a \in L(s)} a_{\pi_1} \wedge \bigwedge_{a \in AP \setminus L(s)} \neg a_{\pi_1} \wedge \bigwedge_{a \in L(s')} a_{\pi_2} \wedge \bigwedge_{a \in AP \setminus L(s')} \neg a_{\pi_2} \right)$$

It is easy to see that on any injectively labeled game structure ind_o holds if the two paths bound to π_1, π_2 are $\sim_o$ -related in their first state. Given an observation $o \in Obs$, and strategy variable $x \in \mathcal{X}$, we define a formula $ii_o(x)$ that holds on a strategy iff this strategy is an o -strategy (in all reachable situations and for all agents) as follows:

$$ii_o(x) := \forall y_1, \dots, y_n, y'_1, \dots, y'_n. \bigwedge_{i=1}^n \psi_o^i \left[\begin{array}{l} \pi_1 : (y_1, \dots, y_{i-1}, x, y_{i+1}, \dots, y_n) \\ \pi_2 : (y'_1, \dots, y'_{i-1}, x, y'_{i+1}, \dots, y'_n) \end{array} \right]$$

where

$$\psi_o^i := \left(\bigwedge_{\alpha \in \mathbb{A}} \langle i, \alpha \rangle_{\pi_1} \leftrightarrow \langle i, \alpha \rangle_{\pi_2} \right) \mathbf{W} (\neg ind_o).$$

The path formula ψ_o^i compares two paths π_1, π_2 and states that as long as prefixes of those two paths are o -indistinguishable (i.e., ind_o holds in each step), the action

selected by agent i is the same on both prefixes (using the fact that the structure records actions). As we do not know which agents might end up playing strategy x , we assert that x behaves as an o -strategy for all agents. For each $i \in Agts$, we thus compare two paths where i plays x , but all other agents play some arbitrary strategy, and assert that ψ_o^i holds for those two paths. Strategy x must thus respond with the same action on any two o -indistinguishable prefixes; in all reachable situations for all agents.

The Translation Using $ii_o(x)$ as a building block, we can translate SL_{ii} into HyperSL. Let $\hat{\pi} \in \mathcal{V}$ denote some *fixed* path variable. SL_{ii} path formulas are then translated directly into HyperSL path formulas by indexing APs with $\hat{\pi}$:

$$\begin{aligned} \langle a \rangle &:= a_{\hat{\pi}} \\ \langle \neg \psi \rangle &:= \neg \langle \psi \rangle \\ \langle \psi_1 \wedge \psi_2 \rangle &:= \langle \psi_1 \rangle \wedge \langle \psi_2 \rangle \\ \langle X \psi \rangle &:= X \langle \psi \rangle \\ \langle \psi_1 \cup \psi_2 \rangle &:= \langle \psi_1 \rangle \cup \langle \psi_2 \rangle \end{aligned}$$

To translate state formulas, we track the current agent binding – which SL_{ii} formalizes via agent bindings of the form $(i \blacktriangleright x)$ – using a partial strategy profile $\vec{x} : Agts \rightarrow \mathcal{X}$ (we write $\emptyset$ for the empty profile). We can then use $\vec{x}$ to construct the unique path $\hat{\pi}$ whenever we encounter a path formula:

$$\begin{aligned} \langle \varphi_1 \wedge \varphi_2 \rangle^{\vec{x}} &:= \langle \varphi_1 \rangle^{\vec{x}} \wedge \langle \varphi_2 \rangle^{\vec{x}} \\ \langle \varphi_1 \vee \varphi_2 \rangle^{\vec{x}} &:= \langle \varphi_1 \rangle^{\vec{x}} \vee \langle \varphi_2 \rangle^{\vec{x}} \\ \langle \psi \rangle^{\vec{x}} &:= \langle \psi \rangle[\hat{\pi} : \vec{x}] \\ \langle (i \blacktriangleright x) \varphi \rangle^{\vec{x}} &:= \langle \varphi \rangle^{\vec{x}[i \mapsto x]} \\ \langle \forall x^o. \varphi \rangle^{\vec{x}} &:= \forall x. ii_o(x) \rightarrow \langle \varphi \rangle^{\vec{x}} \\ \langle \exists x^o. \varphi \rangle^{\vec{x}} &:= \exists x. ii_o(x) \wedge \langle \varphi \rangle^{\vec{x}} \end{aligned}$$

Note that HyperSL does not allow implications between state formulas, but we can convert $\varphi_1 \rightarrow \varphi_2$ to $\neg \varphi_1 \vee \varphi_2$, and push the negation into the path formulas.

Theorem 1. For any SL_{ii} instance $((\mathcal{G}, \{\sim_o\}_{o \in Obs}), \varphi)$ where $\mathcal{G}$ is IL and AR (cf. Lemma 1), we have $(\mathcal{G}, \{\sim_o\}_{o \in Obs}) \models_{SL_{ii}} \varphi$ iff $\mathcal{G} \models \langle \varphi \rangle^{\emptyset}$.

To see the above, observe that for any strategy $f \in Str(\mathcal{G})$ and state $s \in S$, f satisfies ii_o in state s iff f is an o -strategy in all *reachable situations* starting from s . Phrased differently, for any strategy f that satisfies ii_o there exists some proper o -strategy $f' \in Str(\mathcal{G}, o)$ that agrees with f in all reachable situations. As any strategy will only be queried on plays that are compatible with the strategy itself, this suffices to encode the SL_{ii} semantics.

Encoding Size We can analyze the size of our formula encoding. Formula ind_o is of size $\mathcal{O}(|AP| \cdot |\sim_o|)$, and ii_o is of size $\mathcal{O}(|Agts| \cdot (|\mathbb{A}| + |ind_o|))$. The translation for path formulas is linear, and for each quantifier we add one instance of ii_o , so the size of $\langle \varphi \rangle^{\emptyset}$ is bounded by

$$\mathcal{O}(|\varphi| \cdot |Agts| \cdot (|\mathbb{A}| + |AP| \cdot \max_{o \in Obs} |\sim_o|)).$$

Note that this assumes the worst case, where ind_o enumerates all pairs in $\sim_o$. In most cases, we can identify indistinguishable states easily (by, e.g., looking at a particular AP), reducing the size of ind_o significantly.

4 Encoding HyperSL Into SL_{ii}

In this section, we consider the reverse problem and encode a HyperSL MC instance into an equivalent SL_{ii} MC instance. Let $\mathcal{G} = (S, s_0, \mathbb{A}, \kappa, L)$ be a fixed CGS and φ be a fixed HyperSL formula. The idea of our translation is to model a hyperproperty on a self-composition of the game structure (Barthe, D'Argenio, and Rezk 2011). That is, we construct a new game structure that simulates multiple copies of $\mathcal{G}$ in parallel. Intuitively, each copy will correspond to one of the paths we construct in the HyperSL formula. We then ensure that an agent does not observe the state of other copies by using the observation available in SL_{ii} . We assume, w.l.o.g., that the set of path variables $\mathcal{V}$ is finite. Similar to Lemma 1, we make use of the fact that we can assume w.l.o.g. that the game structure is IL and AR:

Lemma 2. *Given a HyperSL MC instance $(\mathcal{G}, \varphi)$, there exists an effectively computable MC instance $(\mathcal{G}', \varphi')$ where (1) $\mathcal{G}'$ is IL and AR, and (2) $\mathcal{G} \models \varphi$ iff $\mathcal{G}' \models \varphi'$.*

Self-Composition We first formally define the self-composition of a game structure. In our construction, we identify each copy in the self-composition by a path variable. States in the composition are, therefore, functions $\mathcal{V} \rightarrow S$ that assign a state to each path variable.

Definition 2. Define $\mathcal{G}_{\mathcal{V}} := (\mathcal{V} \rightarrow S, \prod_{\pi \in \mathcal{V}} s_0, \mathbb{A}, \kappa', L')$ as the CGS over atomic propositions $AP_{\mathcal{V}} := \{a@_{\pi} \mid a \in AP, \pi \in \mathcal{V}\}$ and agents $Agt_{\mathcal{V}} := \{i@_{\pi} \mid i \in Agts, \pi \in \mathcal{V}\}$ where $\kappa' : (\mathcal{V} \rightarrow S) \times (Agt_{\mathcal{V}} \rightarrow \mathbb{A}) \rightarrow (\mathcal{V} \rightarrow S)$ is defined as follows (where $\vec{\alpha} : Agts_{\mathcal{V}} \rightarrow \mathbb{A}$):

$$\kappa' \left(\prod_{\pi \in \mathcal{V}} s_{\pi}, \vec{\alpha} \right) := \prod_{\pi \in \mathcal{V}} \kappa \left(s_{\pi}, \prod_{i \in Agts} \vec{\alpha}(i@_{\pi}) \right)$$

and $L' \left(\prod_{\pi \in \mathcal{V}} s_{\pi} \right) := \bigcup_{\pi \in \mathcal{V}} \{a@_{\pi} \mid a \in L(s_{\pi})\}$.

In $\mathcal{G}_{\mathcal{V}}$, states are functions $\mathcal{V} \rightarrow S$ (recall that $\mathcal{V}$ and S are both finite), and the initial state is the function $\prod_{\pi \in \mathcal{V}} s_0$, mapping all path variables to the initial state s_0 . Each state in $\mathcal{G}_{\mathcal{V}}$ records a state for every $\pi \in \mathcal{V}$ (which we refer to as the π -copy), so each path in $\mathcal{G}_{\mathcal{V}}$ defines paths in $\mathcal{G}$ for all path variables. Agents in $\mathcal{G}_{\mathcal{V}}$ are of the form $i@_{\pi} \in Agts_{\mathcal{V}}$, and APs are of the form $a@_{\pi} \in AP_{\mathcal{V}}$, i.e., they are indexed by path variables. Intuitively, for each $\pi \in \mathcal{V}$, the agents $\{i@_{\pi} \mid i \in Agts\}$ are responsible for updating the system copy for π , so their strategies define the π -path. The transition function thus takes a state $\prod_{\pi \in \mathcal{V}} s_{\pi}$ (where, for all $\pi \in \mathcal{V}$, s_{π} is the current state of the π -copy) and an action vector $\vec{\alpha} : Agts_{\mathcal{V}} \rightarrow \mathbb{A}$, and returns a new state $\prod_{\pi \in \mathcal{V}} s'_{\pi}$. Here, each s'_{π} is defined as $\kappa(s_{\pi}, \prod_{i \in Agts} \vec{\alpha}(i@_{\pi}))$, i.e., we use $\mathcal{G}$'s transition function κ and construct the action profile $Agts \rightarrow \mathbb{A}$ by mapping each agent $i \in Agts$ (in $\mathcal{G}$) to the action chosen by agent $i@_{\pi}$ (in $\mathcal{G}_{\mathcal{V}}$).

We turn $\mathcal{G}_{\mathcal{V}}$ into a game structure under imperfect information by setting $Obs = \{o_{\pi} \mid \pi \in \mathcal{V}\}$ and defining

$$\sim_{o_{\pi}} := \left\{ \left(\prod_{\pi \in \mathcal{V}} s_{\pi}, \prod_{\pi \in \mathcal{V}} s'_{\pi} \right) \mid s_{\pi} = s'_{\pi} \right\}.$$

Two states $\prod_{\pi \in \mathcal{V}} s_{\pi}$ and $\prod_{\pi \in \mathcal{V}} s'_{\pi}$ thus appear indistinguishable under observation o_{π} if the state of the π -copy agrees. A strategy under observation o_{π} can thus only observe the π -copy and not base its decision on the other system copies.

Translating Path Formulas We can now translate the HyperSL formula into an equivalent SL_{ii} formula over the self-composed game structure. The idea of our construction is to interpret path π in the HyperSL formula as the path traversed by the π -copy in $\mathcal{G}_{\mathcal{V}}$. In HyperSL path formulas, we, therefore, replace every path-variable-indexed atomic formula a_{π} (where $a \in AP$) with the atomic proposition $a@_{\pi} \in AP_{\mathcal{V}}$ in $\mathcal{G}_{\mathcal{V}}$:

$$\begin{aligned} \llbracket a_{\pi} \rrbracket &:= a@_{\pi} \\ \llbracket \neg \psi \rrbracket &:= \neg \llbracket \psi \rrbracket \\ \llbracket \psi_1 \wedge \psi_2 \rrbracket &:= \llbracket \psi_1 \rrbracket \wedge \llbracket \psi_2 \rrbracket \\ \llbracket X \psi \rrbracket &:= X \llbracket \psi \rrbracket \\ \llbracket \psi_1 \cup \psi_2 \rrbracket &:= \llbracket \psi_1 \rrbracket \cup \llbracket \psi_2 \rrbracket \end{aligned}$$

Strategy Equality For state formulas, we need some additional gadget. In the HyperSL formula, each strategy variable x can be used on multiple paths. Yet, in $\mathcal{G}_{\mathcal{V}}$, each strategy only acts in a fixed copy (determined by its observation). We, therefore, translate the quantification over a strategy variable x in the HyperSL formula into quantification over $|\mathcal{V}|$ -many variables $\{x_{\pi} \mid \pi \in \mathcal{V}\}$ in SL_{ii} . Each strategy variable x_{π} will act in the π -copy of $\mathcal{G}_{\mathcal{V}}$ and thus be assigned observation o_{π} . Whenever strategy variable x is used in the HyperSL formula to construct path $\pi \in \mathcal{V}$, our translation uses strategy variable x_{π} . Consequently, we need to ensure that the strategies bound to $\{x_{\pi} \mid \pi \in \mathcal{V}\}$ (who all will act in different copies of $\mathcal{G}_{\mathcal{V}}$) denote the same strategy, i.e., respond with the same action to the same prefix (in different copies of $\mathcal{G}_{\mathcal{V}}$). For $x \in \mathcal{X}$ and $\pi, \pi' \in \mathcal{V}$, we define formula $eq(x_{\pi}, x_{\pi'})$ as follows:

$$\bigvee_{j \in Agts, \pi \in \mathcal{V}} y_{j, \pi}^{o_{\pi}} \bullet \bigvee_{j \in Agts, \pi' \in \mathcal{V}} (j@_{\pi} \triangleright y_{j, \pi'}) \bigwedge_{i_1, i_2 \in Agts} (i_1@_{\pi} \triangleright x_{\pi})(i_2@_{\pi'} \triangleright x_{\pi'}) \psi_{i_1, i_2}^{eq},$$

where ψ_{i_1, i_2}^{eq} is defined as follows:

$$\begin{aligned} \psi_{i_1, i_2}^{eq} &:= \left(X \bigwedge_{\alpha \in \mathbb{A}} \langle i_1, \alpha \rangle @_{\pi} \leftrightarrow \langle i_2, \alpha \rangle @_{\pi'} \right) \\ &\quad \mathbf{W} \left(\bigvee_{a \in AP} a@_{\pi} \not\leftrightarrow a@_{\pi'} \right). \end{aligned}$$

Here, we write $\bullet$ as an abbreviation for the concatenation of multiple agent bindings in SL_{ii} .

Formula $eq(x_\pi, x_{\pi'})$ expresses that x_π and $x_{\pi'}$ denote the same strategy, even though x_π operates in the π -copy and $x_{\pi'}$ operates in the π' -copy. The underlying idea is similar to the one used in Section 3: we use the fact that $\mathcal{G}$ is IL and AR and can thus reason about the action selection of agents within our logic. In $eq(x_\pi, x_{\pi'})$, we consider every agent $j \in Agts$ and every path variable $\pi \in \mathcal{V}$, quantify universally over a strategy $y_{j,\pi}$ with observation o_π , and then bind agent $j@ \pi$ in $\mathcal{G}_\mathcal{V}$ to strategy $y_{j,\pi}$. We then consider any pair of agents $i_1, i_2 \in Agts$, and want to express that if i_1 plays x_π in the π -copy (so agent $i_1@ \pi$ plays x_π) and i_2 plays $x_{\pi'}$ in the π' -copy (so agent $i_2@ \pi'$ plays $x_{\pi'}$), the same prefix (in the π and π' copies) results in the same action selected by $i_1@ \pi$ and $i_2@ \pi'$. We thus re-bind agents $i_1@ \pi$ and $i_2@ \pi'$ to strategies x_π and $x_{\pi'}$, respectively. The resulting strategy profile now explores all possible reachable situations in which agent i_1 in the π -copy plays strategy x_π , and agent i_2 in the π' -copy plays strategy $x_{\pi'}$. The path formula ψ_{i_1, i_2}^{eq} then asserts that as long as the prefix in the π and π' copy is equal (using the fact that $\mathcal{G}$ is IL), the action selected by $i_1@ \pi$ and $i_2@ \pi'$ is the same (using the fact that $\mathcal{G}$ is AR). Note that we assumed that $\mathcal{G}$ is AR and thus includes APs of the form $\langle i, \alpha \rangle$ for each agent i and action α , so $\mathcal{G}_\mathcal{V}$ includes APs of the form $\langle i, \alpha \rangle@ \pi$.

Translating State Formulas Using the $eq(\cdot, \cdot)$ construction, we can translate HyperSL state formulas:

$$\begin{aligned} \llbracket \forall x. \varphi \rrbracket &:= \bigvee_{\pi \in \mathcal{V}} x_\pi^{o_\pi} \cdot \left(\bigwedge_{\pi, \pi' \in \mathcal{V}} eq(x_\pi, x_{\pi'}) \right) \rightarrow \llbracket \varphi \rrbracket \\ \llbracket \exists x. \varphi \rrbracket &:= \bigvee_{\pi \in \mathcal{V}} x_\pi^{o_\pi} \cdot \left(\bigwedge_{\pi, \pi' \in \mathcal{V}} eq(x_\pi, x_{\pi'}) \right) \wedge \llbracket \varphi \rrbracket \\ \llbracket \varphi_1 \wedge \varphi_2 \rrbracket &:= \llbracket \varphi_1 \rrbracket \wedge \llbracket \varphi_2 \rrbracket \\ \llbracket \varphi_1 \vee \varphi_2 \rrbracket &:= \llbracket \varphi_1 \rrbracket \vee \llbracket \varphi_2 \rrbracket \\ \llbracket \psi[\pi_k : \vec{x}_k]_{k=1}^m \rrbracket &:= \left(\bigbullet_{i \in Agts, k=1}^m (i@ \pi_k \blacktriangleright (\vec{x}_k(i))_{\pi_k}) \right) \llbracket \psi \rrbracket \end{aligned}$$

Here, we again write $\bigbullet$ as an abbreviation for the concatenation of multiple agent bindings in SL_{ii} . Whenever we translate quantification over variable x , we instead quantify over $|\mathcal{V}|$ -many variables $\{x_\pi \mid \pi \in \mathcal{V}\}$ with the appropriate observation and make sure that they all denote the same strategy by using $eq(\cdot, \cdot)$. When we translate $\psi[\pi_k : \vec{x}_k]_{k=1}^m$, we fix strategies for all agents $i@ \pi \in Agts_\mathcal{V}$, and evaluate the translated path formula $\llbracket \psi \rrbracket$. The key idea here is to bind each agent $i@ \pi_k$ to the strategy that corresponds to the strategy $\vec{x}_k(i)$ in the HyperSL formula, i.e., the strategy that takes the role of agent i in the π_k -copy of $\mathcal{G}_\mathcal{V}$ (i.e., the strategy of $i@ \pi_k$) should equal the strategy that agent i uses to construct path π_k in the HyperSL formula. In our encoding, we translate each strategy quantifier over x in HyperSL, into $|\mathcal{V}|$ -many strategies $\{x_\pi \mid \pi \in \mathcal{V}\}$ in SL_{ii} , where each x_π acts in the π -copy of $\mathcal{G}_\mathcal{V}$. Consequently, the strategy variable that corresponds to $\vec{x}_k(i)$ in the π_k -copy is the variable $(\vec{x}_k(i))_{\pi_k}$. For example, if $\vec{x}_k(i) = z$, we bind agent $i@ \pi_k$ to strategy variable z_{π_k} .

Example 1. Consider the HyperSL formula

$$\exists x, y. \forall z. (\mathbf{G}(a_{\pi_1} \rightarrow b_{\pi_2}))[\pi_1 : (x, y), \pi_2 : (z, x)]$$

Using our translation, and after removing non-needed variables, we obtain the following formula on $\mathcal{G}_{\{\pi_1, \pi_2\}}$:

$$\begin{aligned} &\exists x_{\pi_1}^{o_{\pi_1}}, x_{\pi_2}^{o_{\pi_2}}, y_{\pi_1}^{o_{\pi_1}}. \forall z_{\pi_2}^{o_{\pi_2}}. eq(x_{\pi_1}, x_{\pi_2}) \wedge \\ &(1@ \pi_1 \blacktriangleright x_{\pi_1})(2@ \pi_1 \blacktriangleright y_{\pi_1})(1@ \pi_2 \blacktriangleright z_{\pi_2})(2@ \pi_2 \blacktriangleright x_{\pi_2}) \\ &\mathbf{G}(a@ \pi_1 \rightarrow b@ \pi_2). \end{aligned}$$

Theorem 2. Let $(\mathcal{G}, \varphi)$ be a HyperSL instance, where $\mathcal{G}$ is IL and AR. Then $\mathcal{G} \models \varphi$ iff $(\mathcal{G}_\mathcal{V}, \{\sim_{o_\pi}\}_{o_\pi \in Obs}) \models_{SL_{ii}} \llbracket \varphi \rrbracket$.

Encoding Size We can, again, analyze the size of our encoding. The size of $eq(\cdot, \cdot)$ is of order

$$\mathcal{O}(|Agts| \cdot |\mathcal{V}| + |Agts|^2 \cdot (|\mathbb{A}| + |AP|)).$$

The translation of path formulas is linear, for each quantifier we add $|\mathcal{V}|^2$ -many $eq(\cdot, \cdot)$ constraints, and for each nested path formula we add $|\mathcal{V}| \cdot |Agts|$ -many agent bindings. The overall size of $\llbracket \varphi \rrbracket$ is thus bounded by

$$\mathcal{O}(|\varphi| \cdot |\mathcal{V}|^3 \cdot |Agts|^2 \cdot (|\mathbb{A}| + |AP|)).$$

As expected for a self-composition, $\mathcal{G}_\mathcal{V}$ has $\mathcal{O}(|S|^{|\mathcal{V}|})$ states (Barthe, D'Argenio, and Rezk 2011).

5 Related Work

There exist many extensions of ATL* and SL to reason about imperfect information (Belardinelli, Lomuscio, and Malvone 2019; Dima and Tiplea 2011; Jamroga, Malvone, and Murano 2019; Knight and Maubert 2019; Belardinelli et al. 2017; Berthon, Maubert, and Murano 2017; Huang and van der Meyden 2014; Huang and van der Meyden 2018) or hyperproperties (Beutner and Finkbeiner 2021; Beutner and Finkbeiner 2023; Beutner and Finkbeiner 2024b; Beutner and Finkbeiner 2024a). The connection between knowledge and hyperproperties has been studied extensively (Rabe 2016; Coenen et al. 2020; Beutner et al. 2023; Beutner and Finkbeiner 2025a; Beutner and Finkbeiner 2025b). The work most closely related to ours is the study of (Bozzelli, Maubert, and Pinchinat 2015), who show that LTL_K and a fragment of HyperCTL* are equally expressive (on standard Kripke structures). We study the relation of hyperproperties and knowledge in the setting of MASs and strategies. For example, unlike LTL_K , SL_{ii} does not use a knowledge operator that we can directly encode. Instead, SL_{ii} quantifies over strategies with a given observation relation.

6 Conclusion

In this paper, we have established the first formal connection between imperfect information and hyperproperties in the context of strategy logic. Our results shed new light on the intricate connection between knowledge and hyperproperties and allow for the transfer of tools and techniques. For future work, it is interesting to check how existing verification tools for imperfect information (Jamroga et al. 2019; Lomuscio, Qu, and Raimondi 2009; Kurpiewski, Jamroga, and Knapik 2019) perform on the encoding from HyperSL, and if the decidable fragments of HyperSL correspond to interesting classes of SL_{ii} properties.

Acknowledgments

This work was supported by the European Research Council (ERC) Grant HYPER (101055412), and by the German Research Foundation (DFG) as part of TRR 248 (389792660).

References

- Alur, R.; Henzinger, T. A.; and Kupferman, O. 2002. Alternating-time temporal logic. *J. ACM*.
- Barthe, G.; D’Argenio, P. R.; and Rezk, T. 2011. Secure information flow by self-composition. *Math. Struct. Comput. Sci.*
- Belardinelli, F.; Lomuscio, A.; Murano, A.; and Rubin, S. 2017. Verification of multi-agent systems with imperfect information and public actions. In *Conference on Autonomous Agents and Multiagent Systems, AAMAS 2017*.
- Belardinelli, F.; Lomuscio, A.; and Malvone, V. 2019. An abstraction-based method for verifying strategic properties in multi-agent systems with imperfect information. In *Conference on Artificial Intelligence, AAAI 2019*.
- Berthon, R.; Maubert, B.; Murano, A.; Rubin, S.; and Vardi, M. Y. 2017. Strategy logic with imperfect information. In *Symposium on Logic in Computer Science, LICS 2017*.
- Berthon, R.; Maubert, B.; and Murano, A. 2017. Decidability results for ATL* with imperfect information and perfect recall. In *Conference on Autonomous Agents and Multiagent Systems, AAMAS 2017*.
- Beutner, R., and Finkbeiner, B. 2021. A temporal logic for strategic hyperproperties. In *International Conference on Concurrency Theory, CONCUR 2021*.
- Beutner, R., and Finkbeiner, B. 2023. HyperATL*: A logic for hyperproperties in multi-agent systems. *Log. Methods Comput. Sci.*
- Beutner, R., and Finkbeiner, B. 2024a. Hyper strategy logic. In *International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2024*.
- Beutner, R., and Finkbeiner, B. 2024b. On alternating-time temporal logic, hyperproperties, and strategy sharing. In *Conference on Artificial Intelligence, AAAI 2024*.
- Beutner, R., and Finkbeiner, B. 2025a. On conformant planning and model-checking of $\exists^*\forall^*$ hyperproperties. In *European Conference on Artificial Intelligence, ECAI 2025*.
- Beutner, R., and Finkbeiner, B. 2025b. On hyperproperty verification, quantifier alternations, and games under partial information. In *Formal Methods in Computer-Aided Design, FMCAD 2025*.
- Beutner, R., and Finkbeiner, B. 2025c. Strategy logic, imperfect information, and hyperproperties. *CoRR*.
- Beutner, R.; Finkbeiner, B.; Frenkel, H.; and Metzger, N. 2023. Second-order hyperproperties. In *International Conference on Computer Aided Verification, CAV 2023*.
- Bozzelli, L.; Maubert, B.; and Pinchinat, S. 2015. Unifying hyper and epistemic temporal logics. In *International Conference on Foundations of Software Science and Computation Structures, FoSSaCS 2015*.
- Chatterjee, K.; Henzinger, T. A.; and Piterman, N. 2010. Strategy logic. *Inf. Comput.*
- Clarkson, M. R., and Schneider, F. B. 2008. Hyperproperties. In *Computer Security Foundations Symposium, CSF 2008*.
- Clarkson, M. R.; Finkbeiner, B.; Koleini, M.; Micinski, K. K.; Rabe, M. N.; and Sánchez, C. 2014. Temporal logics for hyperproperties. In *International Conference on Principles of Security and Trust, POST 2014*.
- Coenen, N.; Finkbeiner, B.; Hahn, C.; and Hofmann, J. 2020. The hierarchy of hyperlogics: A knowledge reasoning perspective.
- Dima, C., and Tiplea, F. L. 2011. Model-checking ATL under imperfect information and perfect recall semantics is undecidable. *CoRR*.
- Huang, X., and van der Meyden, R. 2014. A temporal logic of strategic knowledge. In *International Conference on Principles of Knowledge Representation and Reasoning, KR 2014*.
- Huang, X., and van der Meyden, R. 2018. An epistemic strategy logic. *ACM Trans. Comput. Log.*
- Jamroga, W.; Knapik, M.; Kurpiewski, D.; and Mikulski, L. 2019. Approximate verification of strategic abilities under imperfect information. *Artif. Intell.*
- Jamroga, W.; Malvone, V.; and Murano, A. 2019. Natural strategic ability under imperfect information. In *International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2019*.
- Knight, S., and Maubert, B. 2019. Dealing with imperfect information in strategy logic. *CoRR*.
- Kurpiewski, D.; Jamroga, W.; and Knapik, M. 2019. STV: model checking for strategies under imperfect information. In *International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2019*.
- Lomuscio, A.; Qu, H.; and Raimondi, F. 2009. MCMAS: A model checker for the verification of multi-agent systems. In *International Conference on Computer Aided Verification, CAV 2009*.
- Mogavero, F.; Murano, A.; Perelli, G.; and Vardi, M. Y. 2014. Reasoning about strategies: On the model-checking problem. *ACM Trans. Comput. Log.*
- Rabe, M. N. 2016. *A temporal logic approach to information-flow control*. Ph.D. Dissertation, Saarland University.