ECAI 2025
I. Lynce et al. (Eds.)

doi:10.3233/FAIA250985

Strategic Reasoning with Capacity-Constrained Agents and Imperfect Information

Gabriel Ballot[a,b,*], Vadim Malvone[a], Jean Leneutre[a] and Jingxuan Ma[b]

[a]LTCI, Télécom Paris, Institut Polytechnique de Paris, Palaiseau, France
[b]SEIDO Lab, EDF R&D, Palaiseau, France

Abstract. *Multi-Agent System* (MAS) verification comprises formal techniques to model distributed systems, express system properties, and verify them. *Capacity Alternating-time Temporal Logic* (CapATL) was recently introduced to reason about MASs where agents can have different profiles, called *capacities*. However, CapATL assumes agents know the global system state throughout the interaction, which is a strong constraint. This paper extends the concept of agent capacities to systems with imperfect information, enabling the formalisation of a wide range of systems which were previously out of reach. Our contributions are: *(i)* an extension with imperfect information of CapATL, called *Capacity Alternating-time Temporal Epistemic Logic* (CapATEL), *(ii)* the analysis and comparison of different semantics, *(iii)* a completeness result for the CapATEL model-checking problem when agents have bounded recall, and *(iv)* a cybersecurity illustration that showcases CapATEL's applicability.

1 Introduction

Modern systems grow ever more intricate due to layered architecture, legacy integration, compatibility requirements, expanded features, and security and resilience necessity. Verifying that such systems adhere to their specifications requires advanced tools, which supported the development of formal verification techniques by the scientific community. In particular, model checking [23] is a branch of formal verification that consists in modelling the system in some formalism, expressing the requirements in a logical framework, and automatically verifying that the model meets the requirements. Model checking was successfully applied to find design problems in various critical systems, notably in the NASA's Deep Space 1 plan execution module where new concurrency bugs were discovered [32].

System interconnection and dependency has further complexified the analysis of their correctness. MAS model checking extends the principles of model checking to systems where various agents with different objectives, views, or knowledge, can interact, cooperate, or hinder. *Alternating-time Temporal Logic* (ATL) [5] is a famous MAS verification framework to express strategic and temporal properties of systems modelled as *Concurrent Game Structures* (CGSs). They are transition systems where, in each state, every agent chooses an action synchronously, triggering a transition to a next state. Since the introduction of ATL in the early 2000s, various extensions have broadened the scope of systems and properties under study: with epistemic [46, 38], quantitative [3, 44], probabilistic [22, 35], continuous time [15, 24], strategy [51, 52, 42], or action specifications [6, 8, 1, 34]. In particular, CapATL [6] tackles the problem of strategic reasoning when agents may have different profiles, called *capacities*, which determine the set of actions the agent can perform.

A crucial and compelling aspect of MASs is the agents' observation asymmetry, which stems from private information, partial view, or covert behaviours. This is often modelled as an indistinguishability relation for each agent, such that related states cannot be told apart by the corresponding agent. The verification of ATL properties on CGSs with indistinguishability relations, called *imperfect information Concurrent Game Structures* (iCGSs), is notoriously undecidable in general [26]. However, the relevance of imperfect information motivates researchers to seek approximations or restrictions to reason about iCGSs: for instance, restricting the memory of agents. CapATL has been defined for *Capacity Concurrent Game Structures* (CapCGSs) with perfect state information, limiting its reach. The extension to imperfect state information would necessarily suffer from undecidability in the general case, because CapATL includes ATL. Nevertheless, restricting agents' memory introduces challenges because, in CapATL, the memory enables inferring other agents feasible profiles to act accordingly in the future. Another limitation of CapATL is that the agents' capacities are fixed at the beginning of the interaction, whereas some systems (*e.g.*, when agents can dynamically specialise) are better modelled with capacities that can evolve throughout the interaction. This article will elaborate on these semantic distinctions and tackle the imperfect information challenge.

Contributions. The contributions of this article are:

1. introducing CapATEL, which extends CapATL with imperfect information on states and actions,
2. analysing and comparing two classes of strategies and semantics, accounting for agents that keep their capacity (*static* semantics) or specialise during the interaction (*dynamic* semantics),
3. tackling the model-checking problem with bounded recall and proving its Δ_2^{P}-completeness.[1]
4. showing the applicability of CapATEL to verify and control cyber defence mechanisms.

Outline. Section 2 formalises the game structure with imperfect information and capacities. Section 3 defines CapATEL's syntax and semantics, and compares the static and dynamic semantics. Section 4 tackles the model-checking problems. Section 5 presents an application of CapATEL to cybersecurity. The related works are presented in Section 6 and Section 7 concludes this article.

[*] Corresponding Author. Email: gabriel.ballot@telecom-paris.fr.

[1] Δ_2^{P} is the class of problems that can be decided by a polynomial-time Turing machine with calls to an NPTIME oracle.

2 Game Structure

Throughout this paper, $\mathbb{N} = \{1, 2, \dots\}$ denotes the set of positive natural numbers and ω the smallest transfinite number, satisfying $i < \omega$ for all $i \in \mathbb{N}$. Let X and Y be sets. A function (respectively, partial function) f from X to Y is introduced by $f : X \to Y$ (respectively, $f : X \rightharpoonup Y$), its domain is $\mathrm{dom}(f)$. The powerset of X is denoted by $\mathcal{P}(X)$. A relation $\sim \subseteq X \times X$ is an equivalence relation if it is reflexive, symmetric, and transitive; in which case, the set $[\![x]\!]_\sim = \{x' \in X \mid x' \sim x\}$ denotes the equivalence class of $x \in X$ for $\sim$.

Imperfect information CapCGSs. Let $\mathsf{Ag} = \{1, \dots, n\}$ be a set of $n \in \mathbb{N}$ *agents*, Lb a finite set of *labels* (or atomic propositions), and Cap a finite set of *capacities*, representing agent profiles. These form the *signature* $(\mathsf{Ag}, \mathsf{Lb}, \mathsf{Cap})$. The game is formalised as an *imperfect information CapCGS* (iCapCGS), a transition system where transitions are triggered by a *joint action*—a tuple of one action per agent. In each state, agents synchronously choose an action and the game progresses to a next state based on the joint action. Repeating this yields a sequence of transitions, called a *path*. Each agent secretly commits to a capacity, restricting its actions along the path. In iCapCGSs, agents have imperfect information: certain states and joint actions are indistinguishable to them. Definition 1 formalises it.

Definition 1 (iCapCGSs)**.** An iCapCGS over $(\mathsf{Ag}, \mathsf{Lb}, \mathsf{Cap})$ with n agents is a tuple $\langle \mathsf{Ag}, \mathsf{Lb}, \mathsf{Cap}, \mathsf{St}, \mathsf{Ac}, \pi, \Gamma, \gamma, \mathsf{d}, \mathsf{o}, \sim_1, \dots, \sim_n \rangle$ *s.t.*:

- St is the finite set of *states* and Ac the finite set of *actions*,
- $\pi : \mathsf{St} \to \mathcal{P}(\mathsf{Lb})$ is the *labelling function*,
- $\Gamma : \mathsf{Ag} \to \mathcal{P}(\mathsf{Cap}) \setminus \emptyset$ assigns capacities to agents,
- $\gamma : \mathsf{Cap} \to \mathcal{P}(\mathsf{Ac}) \setminus \emptyset$ assigns *feasible* actions to capacities.
- the *protocol function* $\mathsf{d} : \mathsf{Ag} \times \mathsf{St} \to \mathcal{P}(\mathsf{Ac})$ assigns the *available* actions $\mathsf{d}(a, s)$ to all agents a in all states s, satisfying the *progression condition*: agents have feasible available actions whatever their capacity, *i.e.*, $\mathsf{d}(a, s) \cap \gamma(c) \neq \emptyset$ for all $c \in \Gamma(a)$,
- the *transition function* $\mathsf{o} : \mathsf{St} \times \mathsf{Ac}^n \rightharpoonup \mathsf{St}$ is defined for all states s and available joint actions $(\alpha_1, \dots, \alpha_n) \in \mathsf{d}(1, s) \times \dots \times \mathsf{d}(n, s)$,
- the *indistinguishability* relation $\sim_a \subseteq (\mathsf{St} \times \mathsf{St}) \cup (\mathsf{Ac}^n \times \mathsf{Ac}^n)$ of agent a, allowing same available actions in indistinguishable states: for all states $s, s' \in \mathsf{St}$, $s \sim_a s'$ implies $\mathsf{d}(a, s) = \mathsf{d}(a, s')$.

Out of context, we consider a general iCapCGS with n agents of the form $\mathcal{T} = \langle \mathsf{Ag}, \mathsf{Lb}, \mathsf{Cap}, \mathsf{St}, \mathsf{Ac}, \pi, \Gamma, \gamma, \mathsf{d}, \mathsf{o}, \sim_1, \dots, \sim_n \rangle$. A path on $\mathcal{T}$ is a countable sequence of states and joint actions $\rho = s_1 \vec{\alpha}_1 s_2 \vec{\alpha}_2 \dots$, where $s_{i+1} = \mathsf{o}(s_i, \vec{\alpha}_i[1], \dots, \vec{\alpha}_i[n])$ and $\vec{\alpha}_i \in \mathsf{d}(1, s_i) \times \dots \times \mathsf{d}(n, s_i)$ for all i. Let ρ be a path. The length of ρ is the number $|\rho| \in \mathbb{N} \cup \{\omega\}$ of states in ρ, where ω is the length of infinite paths. For $k \in \mathbb{N} \cup \{\omega\}$, $\rho_{\leq k}$ denotes ρ's longest prefix with length less than or equal to k, and, if ρ is finite, $\rho_{[-k..]}$ denotes its longest suffix with length less than or equal to k. We denote by $\rho[i]$ the i^{th} state of ρ (where $1 \leq i \leq |\rho|$), and $\vec{\alpha}[a]$ the action α_a of agent $a \in \mathsf{Ag}$ in the joint action $\vec{\alpha} = (\alpha_1, \dots, \alpha_n)$. When ρ is finite, it ends in a state denoted by $\mathrm{last}(\rho)$.

Example 1. In a cybersecurity setting, consider a machine hosting a web server that can be dynamically reconfigured by the defender (agent 1) as either an Apache server (a) or an Nginx server (n). The attacker (agent 2) can exploit the Nginx server to gain a user session (U). Additionally, the attacker can escalate privileges to root using either the Apache or Nginx server, provided a user session is active: actions A and N, respectively. The action '$\cdot$' denotes a "wait" action. The defender observes the web server type and uses probes to detect root access and the attacker learns the server type upon gaining a session. All actions are assumed to be indistinguishable. The defender has a single capacity with actions $\{\cdot, \mathsf{a}, \mathsf{n}\}$. The attacker has two capacities: c_A (allowing $\{\cdot, \mathsf{U}, \mathsf{A}\}$) and c_N (allowing $\{\cdot, \mathsf{U}, \mathsf{N}\}$). Figure 1 illustrates this scenario, with dashed lines indicating indistinguishable states and arrows potentially representing multiple joint actions.

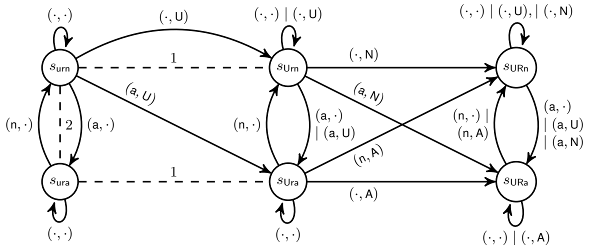

$(\cdot,\cdot)$
$(\cdot,\mathsf{U})$
$(\cdot,\cdot) \mid (\cdot,\mathsf{U})$
$(\cdot,\cdot) \mid (\cdot,\mathsf{U}), \mid (\cdot,\mathsf{N})$
s_{urn}
1
s_{Urn}
$(\cdot,\mathsf{N})$
s_{URn}
(a,U)
(a,N)
$(\mathsf{n},\cdot)$
2
$(\mathsf{a},\cdot)$
$(\mathsf{n},\cdot)$
$(\mathsf{a},\cdot) \mid (\mathsf{a},\mathsf{U})$
$(\mathsf{n},\cdot) \mid (\mathsf{n},\mathsf{A})$
$(\mathsf{a},\cdot) \mid (\mathsf{a},\mathsf{U}) \mid (\mathsf{a},\mathsf{N})$
(n,A)
s_{ura}
1
s_{Ura}
$(\cdot,\mathsf{A})$
s_{URa}
$(\cdot,\cdot)$
$(\cdot,\cdot)$
$(\cdot,\cdot) \mid (\cdot,\mathsf{A})$

Figure 1: The cybersecurity iCapCGS $\mathcal{T}_{\mathsf{cs}}$.

A *capacity assignment* in $\mathcal{T}$ is a partial function $\kappa : \mathsf{Ag} \rightharpoonup \mathsf{Cap}$ that assigns a capacity $\kappa(a) \in \Gamma(a)$ to each agent a in its domain. A capacity assignment is *complete* when its domain is Ag, and $\mathbf{C}_\mathcal{T}$ denotes the set of complete capacity assignments in $\mathcal{T}$. A path $\rho = s_1 \vec{\alpha}_1 s_2 \vec{\alpha}_2 \dots$ of length $k \in \mathbb{N} \cup \{\omega\}$ and a capacity assignment κ are *compatible* if, for all $i \in \mathbb{N}$ with $i < k$, each agent $a \in \mathrm{dom}(\kappa)$ uses a feasible action for $\kappa(a)$ at step i, *i.e.*, for all $i < k$ and $a \in \mathrm{dom}(\kappa)$, $\vec{\alpha}_i[a] \in \gamma(\kappa(a))$. We denote by $C(\rho, \mathcal{T})$ the set of complete capacity assignments compatible with the path ρ. A path is *feasible* if it has a compatible complete capacity assignment, *i.e.*, $C(\rho, \mathcal{T}) \neq \emptyset$. Finite feasible paths are also called *histories* because they describe feasible executions of the system up to some state, and $\mathcal{H}_\mathcal{T}$ denotes the set of histories. Lemma 1 follows directly from the definition of $C(\rho, \mathcal{T})$ and shows that the repetition and order of transitions in a path is irrelevant for the compatible capacity assignments.

Lemma 1. *Let $\rho = s_1 \vec{\alpha}_1 s_2 \vec{\alpha}_2 \dots$ be a path in an iCapCGS $\mathcal{T}$. We can decompose $C(\rho, \mathcal{T}) = \bigcap_{1 \leq i < |\rho|} C(s_i \vec{\alpha}_i s_{i+1}, \mathcal{T})$.*

Example 2. Let $\rho = s_{\mathsf{urn}}(\mathsf{a}, \mathsf{U}) s_{\mathsf{Ura}}(\cdot, \mathsf{A}) s_{\mathsf{URa}}$ be a path in the iCapCGS $\mathcal{T}_{\mathsf{cs}}$ from Example 1 (Figure 1). The only possible attacker capacity is c_A, as they used the privilege escalation against the Apache server (action A): $C(\rho, \mathcal{T}_{\mathsf{cs}}) = \{\kappa\}$, where $\kappa(2) = c_\mathsf{A}$.

A *strategy* is a function $s : \mathcal{H}_\mathcal{T} \to \mathsf{Ac}$ returning the action to perform for each history. A strategy s is *uniform* for an equivalence relation $\sim \subseteq \mathcal{H}_\mathcal{T} \times \mathcal{H}_\mathcal{T}$ over histories iff for all $(\rho, \rho') \in \sim$, we have $s(\rho) = s(\rho')$. A *strategy assignment* $\sigma : \mathsf{Ag} \rightharpoonup (\mathcal{H}_\mathcal{T} \to \mathsf{Ac})$ assigns a strategy to each agent of its domain, such that, for all histories ρ and agents $a \in \mathrm{dom}(\sigma)$, the action $\sigma(a)(\rho)$ is available (belongs to $\mathsf{d}(a, \mathrm{last}(\rho))$). A strategy or capacity assignment *for a coalition* $Y \subseteq \mathsf{Ag}$ is a strategy or capacity assignment with domain Y. The set $\mathsf{Out}_\mathcal{T}(s, \sigma)$ of *outcomes* of a strategy assignment σ from a state s contains the infinite feasible (compatible to at least one complete capacity assignment) paths $\rho = s_1 \vec{\alpha}_1 s_2 \vec{\alpha}_2 \dots$ such that $s_1 = s$ and, for all $i \in \mathbb{N}$ and $a \in \mathrm{dom}(\sigma)$, $\vec{\alpha}_i[a] = \sigma(a)(\rho_{\leq i})$. Notice that there may be no outcome if all paths aligning with the strategy assignment's actions are non-feasible. However, Definition 2 presents two strategy assignment constraints that correspond to agents with capacities and guarantee the existence of outcomes.

Definition 2 (Capacity-constrained strategy assignments)**.** A strategy assignment σ for Y in $\mathcal{T}$ is *static capacity-constrained* (sCC) iff there exists a capacity assignment κ for Y such that, for all agents $a \in Y$ and histories $\rho \in \mathcal{H}_\mathcal{T}$ compatible with κ, we have $\sigma(a)(\rho) \in \gamma(\kappa(a))$. It is *dynamic capacity-constrained* (dCC) iff for all agents $a \in Y$ and histories $\rho \in \mathcal{H}_\mathcal{T}$, there exists a capacity assignment κ for Y such that $\sigma(a)(\rho) \in \gamma(\kappa(a))$.

The difference between static and dynamic capacity constraints lies in the quantification order between existentiality over compatible complete capacity assignments and universality over histories and agents. In particular, an sCC strategy assignment is dCC. The progression condition (*cf.*, Definition 1) ensures the existence of sCC and dCC strategy assignments, as well as the existence of sCC and dCC strategies' outcomes. Intuitively, static capacity constraints model agents which are assigned a profile at the initialisation of the game while dynamic capacity constraints model agents that can specialise during the execution (*cf.*, Section 3).

Relation with Existing Game Structures. iCapCGSs encompass various game structures defined in the literature. When the relations $\sim_1, \ldots, \sim_n$ of an iCapCGS are identities for states and identity by projection on the agents' component for joint actions, *i.e.*, $\sim_a = \{(s, s') \in \mathsf{St} \times \mathsf{St} \mid s = s'\} \cup \{(\vec{\alpha}, \vec{\alpha}') \in \mathsf{Ac}^n \times \mathsf{Ac}^n \mid \vec{\alpha}[a] = \vec{\alpha}'[a]\}$ for all $a \in \mathsf{Ag}$, we obtain a CapCGS as defined in [6]. When there is a unique capacity c that allows every action (*i.e.*, $\gamma(c) = \mathsf{Ac}$) and all joint actions are indistinguishable by all agents, we obtain an iCGS as in [46] for instance. Finally, when all distinct states are *observable* (*i.e.*, not indistinguishable) and there is a unique capacity allowing for all actions, we obtain a CGS as in [5].

3 Capacity Alternating-time Temporal Epistemic Logic

This section defines CapATEL's syntax and semantics for static and dynamic capacity constraints, as well as bounded- and unbounded-recall. It discusses the difference between static and dynamic semantics, and shows the dynamic semantics are strictly more expressive.

Syntax. CapATEL is a logic to express strategic, temporal, and epistemic properties of systems where agents have capacities. It encompasses *Alternating-time Temporal Epistemic Logic* (ATEL) [46] and CapATL [6], and in particular, uses a single knowledge operator to reason about states' properties and capacity properties.

Definition 3 (CapATEL syntax)**.** The following grammar defines a CapATEL formula ϕ over a signature $(\mathsf{Ag}, \mathsf{Lb}, \mathsf{Cap})$, where $\ell \in \mathsf{Lb}$, $a \in \mathsf{Ag}$, $Y \subseteq \mathsf{Ag}$, and $c \in \mathsf{Cap}$:

$$\begin{aligned} \phi &::= \ell \mid \neg\phi \mid \phi \wedge \phi \mid \mathcal{K}_a(\phi) \mid \Box(\varphi) \mid \langle Y \rangle \psi \\ \psi &::= \mathcal{X}\,\phi \mid \phi\,\mathcal{U}\,\phi \mid \phi\,\mathcal{R}\,\phi \\ \varphi &::= a \mapsto c \mid \neg\varphi \mid \varphi \wedge \varphi \end{aligned}$$

Additional boolean symbols $\top$, $\bot$, $\vee$, $\to$, and $\leftrightarrow$ follow standard definitions. The *strategy operator* $\langle\cdot\rangle$ expresses that a coalition Y can enforce a *temporal* formula ψ, regardless of others' actions: $\langle Y \rangle \psi$. Its dual, $[Y]\psi$, means "Y cannot avoid ψ". Temporal formulae use $\mathcal{X}$ (next), $\mathcal{U}$ (until), and $\mathcal{R}$ (release, dual of until) operators. Derived operators include *finally* $\mathcal{F}\phi = \top\,\mathcal{U}\,\phi$ and *globally* $\mathcal{G}\,\phi = \bot\,\mathcal{R}\,\phi$. The *knowledge operator* $\mathcal{K}_a(\phi)$ means agent a knows ϕ holds; its dual, $\mathcal{M}_a(\phi) = \neg\mathcal{K}_a(\neg\phi)$, means a considers ϕ possible. A *capacity assignment formula* φ specifies valid capacity assignments. For instance, $a \mapsto c_1 \wedge (b \mapsto c_2 \vee b \mapsto c_3)$ is satisfied by assignments where a has c_1 and b has c_2 or c_3. The formula $\Box(\varphi)$ holds if all feasible capacity assignments (given the history) satisfy φ; its dual, $\Diamond(\varphi) = \neg\Box(\neg\varphi)$, holds if some assignments satisfy φ.

Example 3. Following from Example 1 (Figure 1), the formula $\phi = \langle 1 \rangle\,\mathcal{G}(\neg\ell_{\mathsf{R}} \vee \mathcal{K}_1(\Box(2 \mapsto c_{\mathsf{A}})))$, where ℓ_{R} labels states with attacker root sessions (s_{URa} and s_{URn}), intuitively means that the defender can prevent attacker root sessions unless the defender knows that the attacker capacity is c_{A}.

CapATEL extends CapATL by decoupling knowledge and capacity operators. For example, the formula $\mathcal{K}^a_{\mathsf{cap}}(\varphi)$ from [6] becomes $\mathcal{K}_a(\Box(\varphi))$ in CapATEL. This allows nested epistemic reasoning, *e.g.*, $\mathcal{K}_a(\mathcal{K}_b(\phi))$, which CapATL cannot express. The explicit quantification with $\Box$ further enhances expressiveness. Overall, CapATEL naturally unifies capacity-based and epistemic reasoning.

Semantics. Games with imperfect information and more than two agents are generally undecidable [26]. Decidability can be restored by restricting agent memory—a concept we extend to CapATEL. In CapATEL, memory not only improves state awareness (as in ATL) but also helps infer other agents' capacities. Various memory bounds exist: memoryless agents act based on the current state [46], bounded-recall agents remember the last k states [11], finite-memory agents use finite-state transducers [50], and natural strategy agents follow condition-action tables [39]. Since memoryless agents in CapATEL learn nothing about capacities, we focus on bounded recall.

For recall parameter $k \in \mathbb{N} \cup \{\omega\}$ and agent a's indistinguishability relation $\sim_a$ in $\mathcal{T}$, we define $\sim^k_a \subseteq \mathcal{H}_{\mathcal{T}} \times \mathcal{H}_{\mathcal{T}}$ on histories. Two histories ρ and ρ' satisfy $\rho \sim^k_a \rho'$ iff: *(i)* a has the same set of compatible capacity assignments in both, *(ii)* $\rho_{[-k..]}$ and $\rho'_{[-k..]}$ have equal length, and *(iii)* their states and joint actions in these suffixes are pairwise $\sim_a$-equivalent. Thus, $\sim^k_a$ captures agent a's observation with capacity awareness and k-bounded recall. When $k = \omega$, the agent is *memoryful*; when $k = 1$, *memoryless*.

Definition 4 (CapATEL semantics)**.** Let $\mathcal{T}$ be an iCapCGS, ϕ and $\bar{\phi}$ two CapATEL formulae, ψ a temporal formula, ϑ and $\bar{\vartheta}$ two capacity assignment or CapATEL formulae, all on the signature $(\mathsf{Ag}, \mathsf{Lb}, \mathsf{Cap})$. Let ρ be an infinite feasible path, $i \in \mathbb{N}$, $\kappa \in \mathbf{C}_{\mathcal{T}}$, $\ell \in \mathsf{Lb}$, $a \in \mathsf{Ag}$, and $Y \subseteq \mathsf{Ag}$. For $x \in \{s, d\}$ and $k \in \mathbb{N} \cup \{\omega\}$, the following satisfaction relation defines CapATEL k-recall x-semantics:

- $(\mathcal{T}, \rho, i, \kappa) \models^k_x \ell$ iff $\ell \in \pi(\rho[i])$,
- $(\mathcal{T}, \rho, i, \kappa) \models^k_x \mathcal{K}_a(\phi)$ iff, for all infinite feasible paths η such that $\eta_{\leq i} \sim^k_a \rho_{\leq i}$, we have $(\mathcal{T}, \eta, i, \kappa) \models^k_x \phi$,
- $(\mathcal{T}, \rho, i, \kappa) \models^k_x \Box(\varphi)$ iff $\forall \kappa' \in C(\rho_{\leq i}, \mathcal{T}), (\mathcal{T}, \rho, i, \kappa') \models^k_x \varphi$,
- $(\mathcal{T}, \rho, i, \kappa) \models^k_x a \mapsto c$ iff $\kappa(a) = c$,
- $(\mathcal{T}, \rho, i, \kappa) \models^k_x \neg\vartheta$ iff $(\mathcal{T}, \rho, i, \kappa) \not\models^k_x \vartheta$,
- $(\mathcal{T}, \rho, i, \kappa) \models^k_x \vartheta \wedge \bar{\vartheta}$ iff $(\mathcal{T}, \rho, i, \kappa) \models^k_x \vartheta$ and $(\mathcal{T}, \rho, i, \kappa) \models^k_x \bar{\vartheta}$,
- $(\mathcal{T}, \rho, i, \kappa) \models^k_x \langle Y \rangle \psi$ iff there is an xCC strategy assignment σ for Y that assigns $\sim^k_a$-uniform strategies to each agent $a \in Y$—called a winning strategy assignment—such that, for all outcomes $\eta \in \mathsf{Out}_{\mathcal{T}}(\rho[i], \sigma)$, we have $(\mathcal{T}, \eta, 1, \kappa) \models^k_x \psi$,
- $(\mathcal{T}, \rho, i, \kappa) \models^k_x \mathcal{X}\,\phi$ iff $(\mathcal{T}, \rho, i+1, \kappa) \models^k_x \phi$,
- $(\mathcal{T}, \rho, i, \kappa) \models^k_x \phi\,\mathcal{U}\,\bar{\phi}$ iff there is $j \geq i$ such that $(\mathcal{T}, \rho, j, \kappa) \models^k_x \phi$ and, for all k where $i \leq k < j$, we have $(\mathcal{T}, \rho, k, \kappa) \models^k_x \bar{\phi}$,
- $(\mathcal{T}, \rho, i, \kappa) \models^k_x \phi\,\mathcal{R}\,\bar{\phi}$ iff either *(i)* for all $j \geq i$, $(\mathcal{T}, \rho, j, \kappa) \models^k_x \bar{\phi}$, or *(ii)* there exists $j \geq i$ such that $(\mathcal{T}, \rho, j, \kappa) \models^k_x \phi \wedge \bar{\phi}$ and, for all k where $i \leq k < j$, we have $(\mathcal{T}, \rho, k, \kappa) \models^k_x \bar{\phi}$.

In the following, a general semantic parameter x is either static or dynamic and a general recall parameter k is either bounded or unbounded. For a CapATEL formula ϕ, the satisfaction relation $(\mathcal{T}, \rho, i, \kappa) \models^k_x \phi$ depends only on $\mathcal{T}$, ϕ, and the history $\rho_{\leq i}$. Thus, we write $(\mathcal{T}, \rho_{\leq i}) \models^k_x \phi$, and in particular, $(\mathcal{T}, s) \models^k_x \phi$ for a state s. For capacity formulae φ, $(\mathcal{T}, \rho, i, \kappa) \models^k_x \varphi$ depends only on $\mathcal{T}$, φ, and κ, so we write $(\mathcal{T}, \kappa) \models^k_x \varphi$. Each nested strategy operator resets capacity assignments and forgets history.

Example 4. Consider the formula ϕ from Example 3 and the iCapCGS $\mathcal{T}_{\mathsf{cs}}$ from Figure 1. We have $(\mathcal{T}_{\mathsf{cs}}, s_{\mathsf{urn}}) \models^1_d \phi$, with a winning strategy σ *s.t.* $\sigma(1)(s_{\mathsf{urn}}) = \mathsf{a}$ and $\sigma(1)(s_{\mathsf{ura}}) = \cdot$. Remark that, by uniformity and 1-recall, we have $\sigma(1)(s_{\mathsf{urn}}(\mathsf{a}, \mathsf{U})s_{\mathsf{Ura}}) = \cdot$ as well.

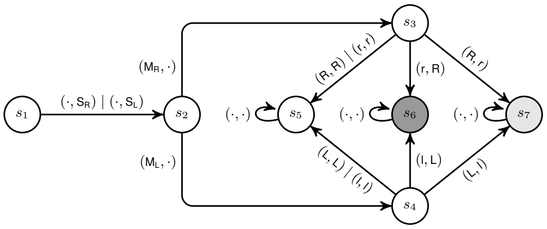

s_1
$(\cdot, \mathsf{S_R}) \mid (\cdot, \mathsf{S_L})$
s_2
$(\mathsf{M_R}, \cdot)$
$(\mathsf{M_L}, \cdot)$
s_3
s_4
s_5
s_6
s_7
$(\mathsf{R}, \mathsf{R}) \mid (\mathsf{r}, \mathsf{r})$
(r, R)
(R, r)
$(\mathsf{L}, \mathsf{L}) \mid (\mathsf{l}, \mathsf{l})$
(l, L)
(L, l)
$(\cdot, \cdot)$
$(\cdot, \cdot)$
$(\cdot, \cdot)$

Figure 2: The arm-wrestling iCapCGS $\mathcal{T}_{\mathsf{aw}}$.

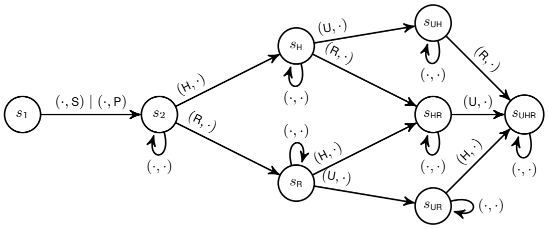


Figure 3: The smart grid integrity access control iCapCGS $\mathcal{T}_{\mathsf{ac}}$.

The relation between static and dynamic semantics. In the static semantics, $\langle Y \rangle \psi$ means there exists a capacity assignment for Y that guarantees ψ, regardless of other agents' capacities or actions—effectively choosing a fixed winning assignment upfront. In contrast, the dynamic semantics allow agents in Y to adapt dynamically: they may use any action from any compatible capacities at each step. For example, if an action α belongs to both c_1 and c_2, an agent can initially use α and later pick actions specific to either capacity, depending on the state. For instance, the dynamic semantics models:

1. An agent that upgrades, specialises, or purchases equipment during the execution, in order to access new actions.
2. An agent whose actions alter its clearance level, access rights, or permission, resulting in the loss of access to other actions.

Examples 5 and 6 illustrate the difference between s- and d-semantics through an arm-wrestling and a smart grid access control scenario.

Example 5. Agents 1 and 2 compete in an arm-wrestling match. Each is either right- or left-handed, represented by capacities c_{R} and c_{L}, respectively. A win occurs when a strong arm meets a weak one; same-arm matches result in a draw. The game begins with agent 2 initiating a handshake using either the right ($\mathsf{S_R}$) or left hand ($\mathsf{S_L}$), revealing their handedness. Agent 1 then chooses an arm for the match ($\mathsf{M_R}$ or $\mathsf{M_L}$). This is modelled in the iCapCGS $\mathcal{T}_{\mathsf{aw}}$ (Figure 2), where state s_6 is labelled w_2 (agent 2 wins), and s_7 is w_1 (agent 1 wins). Actions are $\cdot$ (waiting), R (use strong right arm), r (use weak right arm), L (use strong left arm), and l (use weak left arm). The right-handed capacity allows $\{\cdot, \mathsf{M_R}, \mathsf{M_L}, \mathsf{S_R}, \mathsf{R}, \mathsf{l}\}$; the left-handed one allows $\{\cdot, \mathsf{M_R}, \mathsf{M_L}, \mathsf{S_L}, \mathsf{r}, \mathsf{L}\}$. The game is fully observable. Despite seeing agent 2's handedness, agent 1 cannot guarantee a win if their own handedness is fixed, as a draw may result. Formally, $(\mathcal{T}_{\mathsf{aw}}, s_1) \models_s^k \neg\langle 1 \rangle \mathcal{F}(w_1)$. However, under dynamic semantics with $k \geq 2$, we have $(\mathcal{T}_{\mathsf{aw}}, s_1) \models_d^k \langle 1 \rangle \mathcal{F}(w_1)$, since agent 1 may adapt their handedness after the handshake—a counterintuitive ability.

Example 6. A smart grid features a server (agent 1) and a client (agent 2). The client requests either a safety report (S) or a performance report (P). The safety report requires updating a high-integrity log (U) using either historical data (H) or recent unverified inputs (R). The performance report uses both data sources but does not alter the log. The server enforces an integrity policy inspired by Biba [13]: new processes have high integrity, lose it when accessing unreliable data (R), and need high integrity to update the log (U). In the iCapCGS $\mathcal{T}_{\mathsf{ac}}$ (Figure 3), the client has a capacity per request: c_{S} (allows $\cdot$, S) and c_{P} (allows $\cdot$, P). The server has a high-integrity capacity c_h (allows $\cdot$, U, H) and low-integrity capacity c_l (allows $\cdot$, H, R). The system is fully observable. Let $\phi_{\mathsf{S}} = \square(2 \mapsto c_{\mathsf{S}})$ and $\phi_{\mathsf{P}} = \square(2 \mapsto c_{\mathsf{P}})$, which are true when the respective request is made. Let $\phi_{\mathsf{S}}^* = \ell_{\mathsf{U}} \wedge (\ell_{\mathsf{H}} \vee \ell_{\mathsf{R}})$ and $\phi_{\mathsf{P}}^* = \ell_{\mathsf{H}} \wedge \ell_{\mathsf{R}}$, denoting successful responses (where a state s_{XY} has labels $\{\ell_{\mathsf{X}}, \ell_{\mathsf{Y}}\}$). Then, $\phi = \langle 1 \rangle \mathcal{F}((\phi_{\mathsf{S}} \rightarrow \phi_{\mathsf{S}}^*) \wedge (\phi_{\mathsf{P}} \rightarrow \phi_{\mathsf{P}}^*))$ asserts the server can eventually satisfy the client. Intuitively, if the client requests a safety report, the server must avoid degrading its integrity: using verified historical logs (H) to update the safety log (U). For the performance report, the server can use recent (R) and historical (H) data without concern for integrity degradation. This corresponds to the dynamic semantics because the integrity level is updated after the client's request. Indeed, $(\mathcal{T}_{\mathsf{ac}}, s_1) \models_d^k \phi$ holds (when $k \geq 3$), but $(\mathcal{T}_{\mathsf{ac}}, s_1) \not\models_s^k \phi$ because the desired integrity level cannot be guessed before the request.

The following compares the expressiveness of s- and d-semantics. A semantics defined by $\models$ is at least as expressive as one defined by $\models'$ on the same language iff for every formula ϕ', there exists ϕ such that for all models $\mathcal{M}$, $\mathcal{M} \models \phi \Leftrightarrow \mathcal{M} \models' \phi'$. Since sCC strategy assignments are a special case of dCC ones, Lemma 2 holds.

Lemma 2. *For all iCapCGSs $\mathcal{T}$, histories $\rho \in \mathcal{H}_{\mathcal{T}}$, and CapATEL formulae ϕ, if $(\mathcal{T}, \rho) \models_s^k \phi$ then $(\mathcal{T}, \rho) \models_d^k \phi$.*

From this, we can derive a translation from strategic formulae interpreted with static semantics to strategic formulae interpreted with dynamic semantics. Proposition 1 formalises this translation.

Proposition 1. *Let $\mathcal{T}$ be an iCapCGS, $\rho \in \mathcal{H}_{\mathcal{T}}$, $Y \in \mathcal{P}(\mathsf{Ag}) \setminus \emptyset$, K the set of all $\kappa : \mathsf{Ag} \rightarrow \mathsf{Cap}$, $\phi_\kappa = \diamond \left(\bigwedge_{a \in Y} a \mapsto \kappa(a)\right)$ for $\kappa \in K$, and ϕ_1, ϕ_2 two CapATEL formulae without strategy operators. Then,*

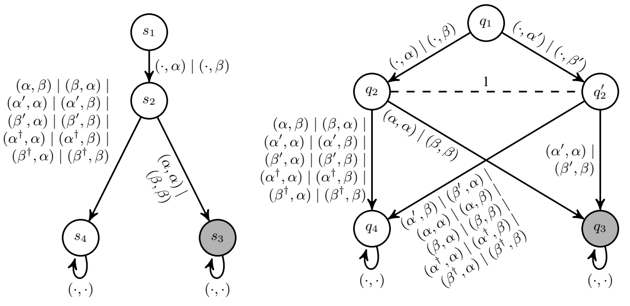
- $(\mathcal{T}, \rho) \models_s^k \langle Y \rangle \mathcal{X} \phi_1$ *iff* $(\mathcal{T}, \rho) \models_d^k \langle Y \rangle \mathcal{X} \phi_1$
- $(\mathcal{T}, \rho) \models_s^k \langle Y \rangle \phi_1 \mathcal{U} \phi_2$ *iff* $(\mathcal{T}, \rho) \models_d^k \bigvee_{\kappa \in K} \langle Y \rangle \phi_1 \mathcal{U} (\phi_2 \wedge \phi_\kappa)$
- $(\mathcal{T}, \rho) \models_s^k \langle Y \rangle \phi_1 \mathcal{R} \phi_2$ *iff* $(\mathcal{T}, \rho) \models_d^k \bigvee_{\kappa \in K} \langle Y \rangle \phi_1 \mathcal{R} (\phi_2 \wedge \phi_\kappa)$

Proof. The first case ($\mathcal{X}$ operator) holds because agents perform a single action each. We focus on the $\mathcal{U}$ operator (and the $\mathcal{R}$ operator is similar). For all sCC strategy assignments σ_s in $\mathcal{T}$, there is a complete capacity assignment $\kappa \in \mathbf{C}_{\mathcal{T}}$ such that the outcomes of σ_s in $\mathcal{T}$ are all compatible with κ (*i.e.*, every prefix of each outcome verifies ϕ_κ). So, if σ_s is winning for $(\mathcal{T}, \rho) \models_s^k \langle Y \rangle \phi_1 \mathcal{U} \phi_2$, then $(\mathcal{T}, \rho) \models_s^k \langle Y \rangle \phi_1 \mathcal{U} (\phi_2 \wedge \phi_\kappa)$, and by Lemma 2, $(\mathcal{T}, \rho) \models_d^k \langle Y \rangle \phi_1 \mathcal{U} (\phi_2 \wedge \phi_\kappa)$. Conversely, suppose we can take $\kappa \in \mathbf{C}_{\mathcal{T}}$ such that σ_d is winning for $(\mathcal{T}, \rho) \models_d^k \langle Y \rangle \phi_1 \mathcal{U} (\phi_2 \wedge \phi_\kappa)$. Then, κ belongs to $\mathbf{C}_{\mathcal{T}}$ and we can build an sCC strategy assignment σ_s from σ_d with the same outcomes. So, σ_s is winning for $(\mathcal{T}, \rho) \models_d^k \langle Y \rangle \phi_1 \mathcal{U} (\phi_2 \wedge \phi_\kappa)$, and consequently, $(\mathcal{T}, \rho) \models_s^k \langle Y \rangle \phi_1 \mathcal{U} \phi_2$. □

Proposition 1 considers a non-empty Y, but trivially, $\langle \emptyset \rangle \psi$ is equivalent in static and dynamic semantics. Moreover, the static and dynamic distinction affects only strategic subformulae satisfaction. Applying Proposition 1 recursively to strategic subformulae, we prove Corollary 1, which states that the dynamic semantics are at least as expressive as the static semantics for a given recall.

Corollary 1. *For $k \in \mathbb{N} \cup \{\omega\}$, the CapATEL k-recall dynamic semantics is at least as expressive as the k-recall static semantics.*

Conversely, one could try to simulate dynamic strategies with static strategies using nested strategy operators and capacity constraints, because the coalition can change its capacity assignment for


s_1
$(\cdot,\alpha) \mid (\cdot,\beta)$
s_2
$(\alpha,\beta) \mid (\beta,\alpha) \mid$
$(\alpha',\alpha) \mid (\alpha',\beta) \mid$
$(\beta',\alpha) \mid (\beta',\beta) \mid$
$(\alpha^\dagger,\alpha) \mid (\alpha^\dagger,\beta) \mid$
$(\beta^\dagger,\alpha) \mid (\beta^\dagger,\beta)$
$(\alpha,\alpha) \mid$
(β,β)
s_4
s_3
$(\cdot,\cdot)$
$(\cdot,\cdot)$
q_1
$(\cdot,\alpha) \mid (\cdot,\beta)$
$(\cdot,\alpha') \mid (\cdot,\beta')$
q_2
1
q'_2
$(\alpha,\beta) \mid (\beta,\alpha) \mid$
$(\alpha',\alpha) \mid (\alpha',\beta) \mid$
$(\beta',\alpha) \mid (\beta',\beta) \mid$
$(\alpha^\dagger,\alpha) \mid (\alpha^\dagger,\beta) \mid$
$(\beta^\dagger,\alpha) \mid (\beta^\dagger,\beta)$
$(\alpha,\alpha) \mid (\beta,\beta)$
$(\alpha',\alpha) \mid$
(β',β)
$(\alpha',\beta) \mid (\beta',\alpha) \mid$
$(\alpha,\alpha) \mid (\alpha,\beta) \mid$
$(\beta,\alpha) \mid (\beta,\beta) \mid$
$(\alpha^\dagger,\alpha) \mid (\alpha^\dagger,\beta) \mid$
$(\beta^\dagger,\alpha) \mid (\beta^\dagger,\beta)$
q_4
q_3
$(\cdot,\cdot)$
$(\cdot,\cdot)$

(a) iCapCGS $\mathcal{T}_1$.

(b) iCapCGS $\mathcal{T}_2$.

Figure 4: Two equivalent iCapCGSs for the static semantics.

each nested strategy operator. However, this simulates non-uniform dynamic strategies, which does not correspond to the dynamic semantics. In fact, the dynamic semantics are strictly more expressive than the static semantics for a given recall, *i.e.*, the dynamic semantics are at least as expressive as the static ones, but not the converse.

Theorem 1. *For $k \in \{2, \dots, \omega\}$, the CapATEL k-recall dynamic semantics is strictly more expressive than the k-recall static semantics.*

Proof. By Corollary 1, the dynamic semantics are at least as expressive as the static ones. We need to show that the converse does not hold. Consider the iCapCGSs $\mathcal{T}_1$ and $\mathcal{T}_2$ as depicted in Figure 4. The gray states s_3 and q_3 are labelled with $\{\ell\}$ and other states have no label. The two agents have the same two feasible capacities: c_α enabling $\{\cdot, \alpha, \alpha', \alpha^\dagger\}$ and c_β enabling $\{\cdot, \beta, \beta', \beta^\dagger\}$. The indistinguishability relation of agent 2 is identity (*i.e.*, they have full observability) in both CapCGS, and agent 1 has full observability on $\mathcal{T}_1$ but cannot distinguish q_3 from q'_3, $(\cdot, \alpha)$ from $(\cdot, \alpha')$, and $(\cdot, \beta)$ from $(\cdot, \beta')$ in $\mathcal{T}_2$. Intuitively, we can see that the subgraph from s_2 is exactly the same as the one from q_2 and q'_2—up to inverting the primed and non-primed versions of α and β for agent 1, which does not change "much" because they belong to the same capacity of agent 1. The key difference between $\mathcal{T}_1$ and $\mathcal{T}_2$ is that agent 1 knows agent 2's capacity and the next state after a transition from s_1 in $\mathcal{T}_1$, while agent 1 knows agent 2's capacity *but not* the next state from q_1 in $\mathcal{T}_2$.

For $k \in \{2, \dots, \omega\}$, $(\mathcal{T}_1, s_1) \models^k_d \langle 1 \rangle \mathcal{F}(\ell)$ but $(\mathcal{T}_2, q_1) \not\models^k_d \langle 1 \rangle \mathcal{F}(\ell)$. So the dynamic semantics can differentiate the two CapCGSs with a recall greater than 1. However, we can show that for all CapATEL formulae ϕ, $(\mathcal{T}_1, s_1) \models^k_s \phi$ iff $(\mathcal{T}_2, q_1) \models^k_s \phi$. □

Remark. Interpreting ATEL k-recall semantics on iCapCGSs via their induced iCGSs makes it incomparable with CapATEL's static and dynamic k-recall semantics. ATEL cannot distinguish iCapCGSs with the same underlying iCGS, while CapATEL generally can. Conversely, consider two iCapCGSs with one agent and two states: s_1 (unlabelled) and s_2 (labelled ℓ), each with a self-loop on action α. In $\mathcal{T}_2$, add a transition from s_1 to s_2 with action β. With a single capacity allowing only α, $\langle 1 \rangle \mathcal{X}\, \ell$ distinguishes $(\mathcal{T}_1, s_1)$ from $(\mathcal{T}_2, s_2)$ in ATEL, though CapATEL considers them equivalent. We believe that restricting protocol functions to actions allowed by some agent capacity renders CapATEL strictly more expressive than ATEL. Alternatively, if the capacity–action map γ is part of the logic (e.g., in $\langle Y \rangle_\gamma$), then CapATEL is strictly more expressive than ATEL.

4 Model Checking

For a semantic parameter $x \in \{s, d\}$, a recall parameter $k \in \mathbb{N} \cup \{\omega\}$, a number of agents $n \in \mathbb{N}$, a number of capacities $m \in \mathbb{N}$, the CapATEL model-checking problem $\textsc{CapAtelMC}[x, k, n, m]$ has for inputs an iCapCGS $\mathcal{T}$, a CapATEL formula ϕ on the same signature with n agents and m capacities, and a state s of $\mathcal{T}$; it returns whether $(\mathcal{T}, s) \models^k_x \phi$. The general bounded-recall model-checking problem $\textsc{CapAtelBrMC}[x]$ has for inputs an iCapCGS $\mathcal{T}$, a CapATEL formula ϕ on the same signature, a state s of $\mathcal{T}$, and a bounded-recall parameter $k \in \mathbb{N}$; it returns whether $(\mathcal{T}, s) \models^k_x \phi$. As CapATEL includes ATL, iCapCGSs are more general structures than iCGSs (*cf.*, Section 2), and ATL is undecidable on iCGS [26] with at least 3 agents, we have the following negative result.

Corollary 2. *For $x \in \{s, d\}$, $n \geq 3$, and $m \in \mathbb{N}$, the model-checking problem $\textsc{CapAtelMC}[x, \omega, n, m]$ is undecidable.*

In order to restore decidability, we focus on the k-recall semantics as in [11]. However, the recall plays two roles in CapATEL. First, the recall enables agents to have a better understanding of the current state, as in ATL with imperfect information. Second, the recall enables agents to remember the possible capacities of other agents, which is a distinctive aspect of CapATEL. As such, we cannot ignore what happened before the last k states as in [11]. However, by Lemma 1, we can abstract the whole history in a finite structure which does not depend on the order and multiplicity of transitions.

Let $\mathcal{T} = \langle \mathsf{Ag}, \mathsf{Lb}, \mathsf{Cap}, \mathsf{St}, \mathsf{Ac}, \pi, \Gamma, \gamma, \mathsf{d}, \mathsf{o}, \sim_1, \dots, \sim_n \rangle$ be an iCapCGS, $\phi = \langle Y \rangle \psi$ be a CapATEL formula without nested strategy operator, and $k \in \mathbb{N}$. Let $\approx_k$ denote the k-length suffix equivalence relation on histories, *i.e.*, $\rho \approx_k \eta$ iff $\rho_{[-k..]} = \eta_{[-k..]}$. We build the iCGS $\mathcal{S}(\mathcal{T}, k) = \langle \mathsf{Ag}, \mathsf{St}', \mathsf{Lb}', \pi', \mathsf{Ac}, \mathsf{d}', \mathsf{o}', \sim'_1, \dots, \sim'_n \rangle$ where:

- $\mathsf{St}' = \{([\![\rho]\!]_{\approx_k}, C(\rho, \mathcal{T})) \mid \rho \in \mathcal{H}_\mathcal{T}\}$,
- $\mathsf{Lb}' = \mathsf{Lb} \cup \{\ell_\kappa \mid \kappa \in \mathbf{C}_\mathcal{T}\}$,
- for $\rho \in \mathcal{H}_\mathcal{T}$, $\pi'(([\![\rho]\!]_{\approx_k}, C(\rho, \mathcal{T}))) = \pi(\text{last}(\rho)) \cup \{\ell_\kappa \mid \kappa \in C(\rho, \mathcal{T})\}$,
- for $\rho \in \mathcal{H}_\mathcal{T}$, $\mathsf{d}'(a, ([\![\rho]\!]_{\approx_k}, C(\rho, \mathcal{T}))) = \mathsf{d}(a, \text{last}(\rho)) \cap \{\kappa(a) \mid \kappa \in C(\rho, \mathcal{T})\}$,
- for $\rho \in \mathcal{H}_\mathcal{T}$ with $s = \text{last}(\rho)$, $q = ([\![\rho]\!]_{\approx_k}, C(\rho, \mathcal{T}))$, and $\vec{\alpha} \in \mathsf{d}'(1, q) \times \cdots \times \mathsf{d}'(n, q)$, we have $\mathsf{o}'(q, \vec{\alpha}[1], \dots, \vec{\alpha}[n]) = ([\![\rho\vec{\alpha}s']\!]_{\approx_k}, C(\rho\vec{\alpha}s', \mathcal{T}))$ where $s' = \mathsf{o}(s, \vec{\alpha}[1], \dots, \vec{\alpha}[n])$,
- for $a \in \mathsf{Ag}$, and two states $q = ([\![\rho]\!]_{\approx_k}, C(\rho, \mathcal{T}))$ and $q' = ([\![\eta]\!]_{\approx_k}, C(\eta, \mathcal{T}))$, we have $q \sim'_a q'$ iff the induced relation $\sim^k_a$ on $\mathcal{T}$ verifies $\rho \sim^k_a \eta$.

Proposition 2. *For all $k \in \mathbb{N}$ and iCapCGS $\mathcal{T}$ with n agents and m capacities, $\mathcal{S}(\mathcal{T}, k)$ is well defined and $|\mathcal{S}(\mathcal{T}, k)| = \mathcal{O}(|\mathcal{T}|^{knm})$.*

Proof. Let ρ and η be two histories such that $\rho \approx_k \eta$ and $C(\rho, \mathcal{T}) = C(\eta, \mathcal{T})$. We have $\text{last}(\rho) = \text{last}(\eta)$, so π' and d' are well defined. By Lemma 1, for any joint action $\vec{\alpha}$ and state s in $\mathcal{T}$, $C(\rho\vec{\alpha}s', \mathcal{T}) = C(\eta\vec{\alpha}s', \mathcal{T})$, so o' is well defined too. The relations $\sim'_a$ are well defined because $\approx_k \subseteq \sim^k_a$. Finally, the protocol function gives the same action set to an agent in its indistringuishable states. On the one hand, the number of equivalence classes for $\approx_k$ is in $\mathcal{O}(|\mathcal{T}|^k)$. On the second hand, for any history ρ in $\mathcal{T}$, the set of feasible complete capacity assignments $C(\rho, \mathcal{T})$ is isomorphic to $C_1 \times \cdots \times C_n$ where $C_a = \{\kappa(a) \mid \kappa \in C(\rho, \mathcal{T})\}$ for all agents a. Indeed, the feasible capacities of an agent depend only on its own actions in the history. As such, we have $|\mathcal{S}(\mathcal{T}, k)| = \mathcal{O}(|\mathcal{T}|^{knm})$. □

The iCGS $\mathcal{S}(\mathcal{T}, k)$ abstracts the observable histories in single states to leverage memoryless ATEL. However, CapATEL expresses properties about arbitrary long histories (*e.g.*, $\Box\,(\varphi)$). Fortunately, we only need to keep track of the feasible capacity assignments induced by these histories: this is why states have the form $([\![\rho]\!]_{\approx_k}, C(\rho, \mathcal{T}))$ for all histories ρ. Theorem 2 states the complexity class of the CapATEL model-checking problems with bounded recall.

Theorem 2. *For $x \in \{s, d\}$, $n \geq 2$, $m \in \mathbb{N}$, and a recall parameter $k \in \mathbb{N}$, the problem* CAPATELMC$[x, k, n, m]$ *is Δ_2^{P}-complete and* CAPATELBRMC$[x]$ *is in the Δ_2^{E} class.*[2]

Proof. The lower bound of CAPATELMC$[x, k, n, m]$ comes from the Δ_2^{P}-completeness of ATL with imperfect information and two memoryless agents [37]. For the upper bounds, we need to prove that checking $(\mathcal{T}, s) \models_x^k \langle Y \rangle \psi$, where ψ has no strategy operator, is in NPTIME for a fixed k and n or in NEXPTIME otherwise. By induction on the whole formula, this gives the Δ_2^{P} and Δ_2^{E} upper-bounds. Suppose $(\mathcal{T}, s) \models_x^k \langle Y \rangle \psi$ where ψ has no strategy operator. The winning k-recall strategy assignment in $\mathcal{T}$ induces a memoryless uniform (for each $\sim'_a$ with $a \in Y$) strategy assignment σ in $\mathcal{S}(\mathcal{T}, k)$, which uses the same sequence of actions. Moreover, if $x = s$, we can take the capacity assignment κ for Y in $\mathcal{T}$, which makes the winning strategy assignment sCC. The strategy assignment σ in $\mathcal{S}(\mathcal{T}, k)$ (and κ, in case $x = s$) is a nondeterministic certificate that $(\mathcal{T}, s) \models_x^k \langle Y \rangle \psi$. The certificate verification procedure consists in exploring the projection of the strategy on $\mathcal{S}(\mathcal{T}, k)$, and asserting that: *(i)* σ assigns a $\sim'_a$-uniform strategy to each agent $a \in Y$ and, if $x = s$, the actions are feasible for $\kappa(a)$, and *(ii)* the temporal ATEL formula ψ' holds, where ψ' is obtained by replacing subformulae of the from $\Box(\varphi)$ in ψ by $\bigwedge_{\kappa \in K} \neg \ell_\kappa$ with $K = \{\kappa \in \mathbf{C}_\mathcal{T} \mid (\mathcal{T}, \kappa) \not\models_x^k \varphi\}$. As $|\mathcal{S}(\mathcal{T}, k)| = \mathcal{O}(|\mathcal{T}|^{knm})$ (Proposition 2), the certificate size and verification time is polynomial for a fixed k, n, and m, and exponential otherwise. □

The Δ_2^{P}-completeness matches the complexity of bounded-recall ATEL [46, 11] while offering greater expressive power, since CapATEL includes ATEL. However, our Δ_2^{P}-completeness result holds only for a fixed number of agents and capacities, whereas ATEL is Δ_2^{P}-complete both when the number of agents is fixed or is a problem parameter. The thesis [17] presents algorithms for ATEL model checking with uniform strategies, tackling the challenge of uniform strategy enumeration. In practice, techniques such as on-the-flight model checking would improve CapATEL verification by avoiding building explicitely $\mathcal{S}(\mathcal{T}, k)$—as many states might be irrelevant.

5 Illustrative Example: Moving Target Defences

Cyber attacks typically follow a structured process (*cf.*, cyber-kill-chain [36]), starting with reconnaissance to explore the *attack surface* (*e.g.*, configurations, channels, persistent data, APIs) and identify vulnerabilities. Static systems and defences benefit attackers by enabling them to meticulously plan their attacks. In contrast, *Moving Target Defences* (MTDs) [31, 43, 25] reconfigure systems dynamically to alter the attack surface, disrupt attacker knowledge, and improve security. A company's cybersecurity team aims to deploy autonomous MTDs to defend its infrastructure. These MTDs rely on *Intrusion Detection Systems* (IDSs), which monitor system parameters but only partially, introducing imperfect information. The team has identified atomic attacks and attacker profiles (*e.g.*, beginner, hacktivist, insider) through risk analysis.

Model. We consider finite sets or resources $\mathbf{R}$ (*e.g.*, machines, networks, users, files), parameters $\mathbf{P}$ (*e.g.*, configuration and monitoring flags, addresses, versions), parameter values $\mathbf{V}$, autonomous MTDs $\mathbf{M} = \{1, \ldots, n-1\}$, and IDSs $\mathbf{I}$. Each resource $r \in \mathbf{R}$ has parameters $\mathrm{prms}(r) \subseteq \mathbf{P}$ and parameters $p \in \mathbf{P}$ have possible values $\mathrm{val}(p) \subseteq \mathbf{V}$. An MTD m can modify its *moving parameter* $\mathrm{mp}(m) \in \mathbf{P}$. However, different MTD implementations allow different accessible values: we denote by $\mathrm{impl}(m) = \{V_1, V_2, \ldots\}$ the set of possible implementations where each implementation $V_i \subseteq \mathrm{val}(\mathrm{mp}(m))$ is a set of possible values. An MTD m monitors the system through a set $\mathrm{ids}(m) \subseteq \mathbf{I}$ of IDSs, where an IDS $i \in \mathbf{I}$ monitors the value of a parameter $\mathrm{prm}(i) \in \mathbf{P}$. An atomic attack $a = (R_i, \mathrm{pre}, R_t, \mathrm{post})$ has preconditions a partial assignment $\mathrm{pre} : \mathbf{P} \rightharpoonup \mathbf{V}$ and requires $R_i \subseteq \mathbf{R}$ under attacker control. Upon success, it grants the attacker with $R_t \subseteq \mathbf{R}$ and updates parameters via $\mathrm{post} : \mathbf{P} \rightharpoonup \mathbf{V}$. We denote by $\mathbf{A}$ the set of atomic attacks.

States, actions, and transitions. The system state is a tuple (χ, R) where $\chi : \mathbf{P} \to \mathbf{V}$ is a parameter valuation and $R \subseteq \mathbf{R}$ are attacker-controlled resources. For instance, $\chi(\mathsf{Machine12Address}) = \mathsf{10.40.0.15}$ means that the machine 12 has address 10.40.0.15, $\chi(\mathsf{ConfigFileOwner}) = \mathsf{user3}$ means that user 3 owns the configuration file, and $\mathsf{user3} \in R$ means that the attacker has compromised user 3. A state (χ, R) has labels $\{\ell_{p,\chi(p)} \mid p \in \mathbf{P}\} \cup \{\ell_r \mid r \in \mathbf{R}\}$. The possible attacker-defender interaction is modelled by an iCapCGS with such states and agents $\mathsf{Ag} = \mathbf{M} \cup \{n\}$, where n is the attacker. In each state (χ, R), each MTD m either updates the moving parameter $\mathrm{mp}(m)$ with one of the values $v \in \mathrm{val}(\mathrm{mp}(m))$ (denoted by action v) or leaves it unchanged (action $\cdot$); and, the attacker either performs an atomic attack $a = (R_i, \mathrm{pre}, R_t, \mathrm{post}) \in \mathbf{A}$ whenever $R_i \subseteq R$ or waits (action $\cdot$). Given the joint action $\vec{\alpha}$ in state (χ, R), we update the state as follows. First, for each MTD m from 1 to $n-1$ which does an action $v \in \mathbf{V}$, we replace the value for parameter $\mathrm{mp}(m)$ by v in the valuation. Second, if the attacker does an attack $a = (R_i, \mathrm{pre}, R_t, \mathrm{post}) \in \mathbf{A}$ and, for all $p \in \mathrm{dom}(\mathrm{pre})$, we have $\mathrm{pre}(p) = \chi(p)$, we update R as $R \cup R_t$ and we update the parameter valuation with the values from post. We obtain a new state (χ', R').

Indistinguishability. An MTD m cannot distinguish (χ, R) and (χ', R') if all $\mathrm{ids}(m)$ observe the same values (*i.e.*, $\chi(\mathrm{prm}(i)) = \chi'(\mathrm{prm}(i))$ for all $i \in \mathrm{ids}(m)$) and attacker cannot distinguish them if $R = R'$ and $\chi(p) = \chi'(p)$ for all $p \in \mathrm{prms}(r)$ and $r \in R$ (*i.e.*, the attacker knows the resources they control).

Capacities. Each MTD m has a capacity c_V for each $V \in \mathrm{impl}(m)$, allowing actions $V \cup \{\cdot\}$. The attacker has one capacity c_p per profile $p \in \mathsf{Pr}$ (where Pr is the set of attacker profiles), granting access to attacks for that profile.

Objectives. The defence team aims to avoid bad configurations $\chi_1, \ldots, \chi_k$ (partial valuations) until an MTD identifies the attacker profile. Let $\phi_{\mathsf{bc}} = \bigvee_{1 \leq i \leq k} \bigwedge_{p \in \mathrm{dom}(\chi_i)} \ell_{p,\chi_i(p)}$, which is true iff the state has bad configuration, and $\phi_{\mathsf{id}} = \bigvee_{m \in \mathbf{M}} \bigvee_{p \in \mathsf{Pr}} \mathcal{K}_m (\Box (n \mapsto c_p))$, which is true when an MTD can tell the attacker profile. The overall objective can be $\phi = \langle \mathbf{M} \rangle (\neg \phi_{\mathsf{bc}}) \, \mathcal{U} \, (\neg \phi_{\mathsf{bc}} \wedge \phi_{\mathsf{id}})$. If ϕ holds under static semantics, fixed implementations for each MTD suffice for defence and profile identification. Under dynamic semantics, implementations could adapt at runtime, enabling further flexibility. Depending on specific use cases, CapATEL enables specifying various other defence objectives with strategic, temporal, epistemic, and profile aspects.

Example 7. The cybersecurity scenario from Examples 1, 2, 3, and 4 (Figure 1) instantiates this model. It includes one resource, the server ($\mathbf{R} = \{\mathsf{srv}\}$), and three parameters ($\mathbf{P} = \{\mathsf{web}, \mathsf{ownU}, \mathsf{ownR}\}$), all linked to the server ($\mathrm{prms}(\mathsf{srv}) = \mathbf{P}$). The web server can be Apache or Nginx ($\mathrm{val}(\mathsf{web}) = \{\mathsf{a}, \mathsf{n}\}$), and the session flags can be active or inactive ($\mathrm{val}(\mathsf{ownU}) = \{\mathsf{u}, \mathsf{U}\}$, $\mathrm{val}(\mathsf{ownR}) = \{\mathsf{r}, \mathsf{R}\}$). There is one MTD with web as its moving parameter and a single implementation. It relies on an IDS i monitoring the root session status ($\mathrm{prm}(i) =$

[2] Δ_2^{E} is the class of problems that can be decided by an polynomial-time Turing machine with calls to an NEXPTIME oracle. [33]

ownR). The initial exploit is $\mathsf{U} = (\emptyset, \mathsf{web} \mapsto \mathsf{a}, \{\mathsf{srv}\}, \mathsf{ownU} \mapsto \mathsf{U})$ and the privilege escalation atomic attack with Nginx and Apache server are respectively $\mathsf{N} = (\{\mathsf{srv}\}, (\mathsf{web}, \mathsf{ownU}) \mapsto (\mathsf{n}, \mathsf{U}), \emptyset, \mathsf{ownR} \mapsto \mathsf{R})$, and $\mathsf{A} = (\{\mathsf{srv}\}, (\mathsf{web}, \mathsf{ownU}) \mapsto (\mathsf{a}, \mathsf{U}), \emptyset, \mathsf{ownR} \mapsto \mathsf{R})$, where the $\mapsto$ notation implicitly defines partial assignments.

6 Related Work

ATL-related logics with imperfect information are an active area of research due to both their practical relevance and theoretical complexity. Indistinguishability relations were already introduced in the original ATL paper in 2002 [5]. The epistemic extension, ATEL, was formalised in 2003 using non-uniform strategies [48], and constraints to enforce uniform strategies were added in 2004 [46, 38]. ATL with imperfect information and perfect recall was shown to be undecidable in 2011 [26]. Various restrictions were introduced to regain decidability, such as bounded recall [11], finite-memory agents [50], natural strategies [39], structural constraints [12], or communicating coalitions [27]. An alternative, sound but incomplete, is approximate verification [11]. Public actions, where all agents observe the actions performed, allow for decidable semantics in both ATL [8] and *Strategy Logic* (SL) [10]. In this paper, we consider agents with bounded recall for CapATEL, where capacity is inherently linked to histories and requires a distinct treatment from bounded-recall ATL.

Capacities were introduced in [6], and *Alternating-time Temporal Logic with Stochastic Abilities* (ATL-SA) extends this concept with probability [7]. Both frameworks consider only static, capacity-constrained strategy assignments. The notion of capacity helps define strategy: an agent with capacity constraints has a strategy for achieving a goal only if the strategy uses actions feasible under some capacity profile. In the *Logic of Capabilities* [49], the operator $A_a\alpha$ indicates whether agent a can perform action α in the current state. However, this logic does not allow for varying capacities within a single agent. In CapATEL, agent actions induce a lasting capacity commitment, differing from the Logic of Capabilities. In the *Situation Calculus* [41], a *situation* is defined by the sequence of actions from an initial state. Given a situation s, the predicate $Poss(a, \alpha, s)$ is true when agent a can use the action α. However, this does not explicitely capture multiple capacity profiles or commitment over time as in CapATEL. Explicit strategy manipulation appears in *ATL with explicit strategies* (ATLES) [51], which introduces a strategy commitment operator, and *ATL with irrevocable strategies* (IATL) [52], which enforces strategy persistence across nested coalitions. *ATL with actions* (ATL-A) [1], *ATEL with actions* (ATEL-A) [1], and *ATL with explicit actions* (ATLEA) [34] explicitly define the set of actions available to an agent. These are only binding for the next step, whereas CapATEL induces a commitment for the entire interaction, thus increasing expressivity. Making action restrictions part of the logic itself allows greater expressiveness, which CapATEL could match if action and capacity mappings were embedded in formulae rather than in game structures. *Coalition Action Logic* (CAL) [14] and *First-Order Coalition Logic* (FOCL) [20] feature explicit quantification over actions but are limited to reasoning about immediate successor states. They do not support lasting capacity commitments.

Norms and social laws in MAS research seek a balance between fully centralised agent control and total agent autonomy. Norms offer coordination rules (e.g., "cars drive on the right") to prevent chaos while maintaining individual agent freedom, thus enabling decentralisation [47]. Originally, norms were defined as tuples (α, ϕ), meaning action α is forbidden in states satisfying ϕ [47]. In [2], norms are combined with *Computation Tree Logic* (CTL) [28] to express obligatory and permitted outcomes. In [16], both norms and preferences are encoded as *Linear Temporal Logic* (LTL) [45] formulae. In [4], norms are dynamic and adapt based on actions taken in each state. Agent capabilities can be interpreted as implicit norms that agents privately follow, though this is not addressed in the existing literature, to the best of our knowledge.

A related research direction investigates dynamic game structures. In *Obstruction Logic* (OL) [18], agents can remove outgoing edges (up to a cost) from the current state, and another agent chooses among the remaining options. These changes last only one round. OL has been extended systems with an arbitrary number of agents [19, 21] and real-time systems [40]. Other approaches to dynamic games include *Dynamic Dictatorial Coalition Logic* (DDCL) [29] and *Logic for ATL Model Building* (LAMB) [30], which allow permanent changes to states and transitions. Permanent edge removal can be linked to CapATEL in perfect information settings, but OL does not feature permanent removal and does not link it to agent profiles.

7 Conclusion

This paper introduces CapATEL, a logic to express strategic, temporal, and epistemic properties of MASs where agents have capacities which restrict the set of feasible actions. CapATEL unifies CapATL [6] and ATEL [46] and our game structure, iCapCGS, encodes capacities and imperfect state and action information, which subsumes iCGSs and CapCGSs. We distinguish between static and dynamic capacity constraints for strategy assignments: the first models agents with a unique profile through the interaction while the second allows for agent specialisation. We prove that the dynamic semantics are strictly more expressive than the static ones, for a given recall. Moreover, this article studies the model-checking problem of CapATEL. It is undecidable for memoryful agents, but we prove its Δ_2^{P}-completeness for a fixed number of capacities and agents with a given bounded recall. Finally, we show the applicability of CapATEL in the cybersecurity context, with dynamic defences such as MTDs.

Future works include defining a different memory model for agents. Indeed, knowledge about feasible capacities are inferred from observed transitions, which may have happened far in the past. Agents would probably gain from memorising key moments in the past, that discriminate the most between capacities. This research direction can be inspired from natural strategies [39] and finite-state transducers [50]. Another future work is the extension to complex temporal properties by allowing a free interleaving of temporal subformulae as in ATL* [5], or allowing complex strategy quantification as in SL with simple goals [9]. The capacities are defined as subsets of feasible actions, but we cound push further the theory of capacities. For instance, capacities could be defined in terms of logical formulae, which would model agent profiles more accurately. Finally, the link between capacities and social norms can be studied further. In particular, social norms have been defined thourgh different logical framework and combined with various extrernal logics, sometimes with a mechanism design approach [4, 16].

Acknowledgements

This work was carried out within SEIDO Lab, a joint research laboratory covering research topics in the field of smart grids, *e.g.*, distributed intelligence, service collaboration, cybersecurity, and privacy. It involves researchers from academia (Télécom Paris, Télécom SudParis, CNRS LAAS) and industry (EDF R&D).

References

[1] T. Ågotnes. Action and knowledge in alternating-time temporal logic. *Synth.*, 149(2):375–407, 2006. doi: 10.1007/S11229-005-3875-8.

[2] T. Ågotnes, W. van der Hoek, J. A. Rodríguez-Aguilar, C. Sierra, and M. J. Wooldridge. A temporal logic of normative systems. In D. Makinson, J. Malinowski, and H. Wansing, editors, *Towards Mathematical Philosophy*, volume 28 of *Trends in logic*. Springer, 2009.

[3] N. Alechina, B. Logan, N. H. Nga, and A. Rakib. Resource-bounded alternating-time temporal logic. In *AAMAS'10*, pages 481–488, Richland, SC, May 2010. IFAAMAS. ISBN 9780982657119.

[4] N. Alechina, G. D. Giacomo, B. Logan, and G. Perelli. Automatic synthesis of dynamic norms for multi-agent systems. In G. Kern-Isberner, G. Lakemeyer, and T. Meyer, editors, *KR'22*, 2022.

[5] R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49(5):672–713, Sept. 2002. ISSN 0004-5411. doi: 10.1145/585265.585270.

[6] G. Ballot, V. Malvone, J. Leneutre, and Y. Laarouchi. Strategic reasoning under capacity-constrained agents. In M. Dastani, J. S. Sichman, N. Alechina, and V. Dignum, editors, *AAMAS'24*, pages 123–131. ACM, 2024. doi: 10.5555/3635637.3662859.

[7] G. Ballot, V. Malvone, J. Leneutre, J. Ma, and M. Leslous. Alternating-time temporal logic with stochastic abilities. In *AAMAS'25*, pages 214–222, Richland, SC, 2025. IFAAMAS. ISBN 9798400714269.

[8] F. Belardinelli, A. Lomuscio, A. Murano, and S. Rubin. Verification of multi-agent systems with imperfect information and public actions. In K. Larson, M. Winikoff, S. Das, and E. H. Durfee, editors, *AAMAS'17*, pages 1268–1276, 2017.

[9] F. Belardinelli, W. Jamroga, D. Kurpiewski, V. Malvone, and A. Murano. Strategy logic with simple goals: Tractable reasoning about strategies. In S. Kraus, editor, *IJCAI'19*, 2019. doi: 10.24963/ijcai.2019/13.

[10] F. Belardinelli, A. Lomuscio, A. Murano, and S. Rubin. Verification of multi-agent systems with public actions against strategy logic. *AIJ*, 285:103302, 2020. doi: 10.1016/J.ARTINT.2020.103302.

[11] F. Belardinelli, A. Lomuscio, V. Malvone, and E. Yu. Approximating perfect recall when model checking strategic abilities: Theory and applications. *JAIR*, 73:897–932, 2022. doi: 10.1613/jair.1.12539.

[12] R. Berthon, B. Maubert, and A. Murano. Decidability results for ATL* with imperfect information and perfect recall. In *AAMAS'17*, pages 1250–1258, Richland, SC, 2017. IFAAMAS.

[13] K. J. Biba. Integrity considerations for secure computer systems. Technical report, Apr. 1977.

[14] S. Borgo. Coalitions in action logic. In M. M. Veloso, editor, *IJCAI'07*, pages 1822–1827, 2007.

[15] T. Brihaye, V. Bruyere, and J.-F. Raskin. On optimal timed strategies. In P. Pettersson and W. Yi, editors, *FORMATS*, pages 49–64, Berlin, Heidelberg, 2005. Springer. ISBN 978-3-540-31616-9.

[16] N. Bulling and M. Dastani. Norm-based mechanism design. *AIJ*, 239: 97–142, 2016. doi: 10.1016/J.ARTINT.2016.07.001.

[17] S. Busard. *Symbolic Model Checking of Multi-Modal Logics: Uniform Strategies and Rich Explanations.* PhD thesis, Catholic University of Louvain, Louvain-la-Neuve, Belgium, 2017.

[18] D. Catta, J. Leneutre, and V. Malvone. Obstruction Logic: A Strategic Temporal Logic to Reason About Dynamic Game Models. In *ECAI'23*, pages 365–372, Kracow, Poland, 2023.

[19] D. Catta, J. Leneutre, V. Malvone, and A. Murano. Obstruction alternating-time temporal logic: A strategic logic to reason about dynamic models. In M. Dastani, J. S. Sichman, N. Alechina, and V. Dignum, editors, *AAMAS'24*, pages 271–280. IFAAMAS, 2024.

[20] D. Catta, R. Galimullin, and A. Murano. First-order coalition logic. *CoRR*, abs/2505.06960, 2025. doi: 10.48550/ARXIV.2505.06960.

[21] D. Catta, J. Leneutre, V. Malvone, and J. Ortiz Vega. Coalition obstruction temporal logic: A new obstruction logic to reason about demon coalitions. In *IJCAI'25*, 2025.

[22] T. Chen and J. Lu. Probabilistic alternating-time temporal logic and model checking algorithm. In *FSKD'07*, pages 35–39, Haikou, China, 2007. IEEE. ISBN 978-0-7695-2874-8. doi: 10.1109/FSKD.2007.458.

[23] E. M. Clarke, T. A. Henzinger, H. Veith, R. Bloem, et al. *Handbook of Model Checking*. Springer, 2018. ISBN 978-3-319-10575-8.

[24] A. David, P. G. Jensen, K. G. Larsen, A. Legay, D. Lime, M. G. Sørensen, and J. H. Taankvist. On time with minimal expected cost! In F. Cassez and J.-F. Raskin, editors, *Automated Technology for Verification and Analysis*. Springer, 2014. ISBN 978-3-319-11936-6.

[25] R. S. Dewar. Active cyber defense. Report, Zurich, 2017.

[26] C. Dima and F. L. Tiplea. Model-checking ATL under imperfect information and perfect recall semantics is undecidable. *CoRR*, abs/1102.4225, 2011.

[27] C. Dima, C. Enea, and D. Guelev. Model-checking an alternating-time temporal logic with knowledge, imperfect information, perfect recall and communicating coalitions. *Theoretical Computer Science*, 25:103–117, June 2010. doi: 10.4204/eptcs.25.12.

[28] E. A. Emerson and E. M. Clarke. Using branching time temporal logic to synthesize synchronization skeletons. *Sci. Comput. Program.*, 2(3): 241–266, 1982. ISSN 0167-6423.

[29] R. Galimullin and T. Ågotnes. Dynamic coalition logic: Granting and revoking dictatorial powers. In S. Ghosh and T. Icard, editors, *LORI'21*, volume 13039, pages 88–101. Springer, 2021.

[30] R. Galimullin, M. Gladyshev, M. Mittelmann, and N. Motamed. Changing the rules of the game: Reasoning about dynamic phenomena in multi-agent systems. In *AAMAS'25*, pages 829–838, Richland, SC, 2025. IFAAMAS. ISBN 9798400714269.

[31] A. K. Ghosh, D. Pendarakis, and W. H. Sanders. *Moving Target Defense Co-Chair's Report-National Cyber Leap Year Summit 2009*. FNITRD, Washington, 2009.

[32] K. Havelund, M. Lowry, and J. Penix. Formal analysis of a space-craft controller using spin. *IEEE Trans. Softw. Eng.*, 27(8):749–765, 2001.

[33] L. A. Hemachandra. The strong exponential hierarchy collapses. *JCSS*, 39(3):299–322, 1989. doi: 10.1016/0022-0000(89)90025-1.

[34] A. Herzig, E. Lorini, and D. Walther. Reasoning about actions meets strategic logics. In D. Grossi, O. Roy, and H. Huang, editors, *Logic, Rationality, and Interaction*, pages 162–175. Springer, 2013.

[35] X. Huang, K. Su, and C. Zhang. Probabilistic alternating-time temporal logic of incomplete information and synchronous perfect recall. In J. Hoffmann and B. Selman, editors, *AAAI'12*, number 1, pages 765–771, Sept. 2012. doi: 10.1609/aaai.v26i1.8214.

[36] E. M. Hutchins, M. J. Cloppert, R. M. Amin, et al. Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. *Leading Issues in Information Warfare & Security Research*, 1(1):80, 2011.

[37] W. Jamroga and J. Dix. Model checking abilities of agents: A closer look. *Theor. Comput. Syst.*, 42(3):366–410, Apr. 2008. ISSN 1433-0490. doi: 10.1007/S00224-007-9080-Z.

[38] W. Jamroga and W. van der Hoek. Agents that know how to play. *Fundam. Inform.*, 63:185–219, 2004. 2-3.

[39] W. Jamroga, V. Malvone, and A. Murano. Natural strategic ability under imperfect information. In E. Elkind, M. Veloso, N. Agmon, and M. E. Taylor, editors, *AAMAS'19*, Montreal, QC, Canada, 2019.

[40] J. Leneutre, V. Malvone, and J. Ortiz. Timed obstruction logic: A timed approach to dynamic game reasoning. In *AAMAS'25*, pages 1272–1281, Richland, SC, 2025. IFAAMAS. ISBN 9798400714269.

[41] Y. Lespérance, H. J. Levesque, F. Lin, and R. B. Scherl. Ability and knowing how in the situation calculus. *Studia Logica*, 66(1), 2000.

[42] F. Mogavero, A. Murano, G. Perelli, and M. Y. Vardi. Reasoning about strategies: On the model-checking problem. *ACM Transactions on Computational Logic*, 15:1–47, 2014. ISSN 1529-3785.

[43] R. E. Navas, F. Cuppens, N. Boulahia Cuppens, L. Toutain, and G. Z. Papadopoulos. MTD, where art thou? a systematic review of moving target defense techniques for IoT. *IEEE Internet of Things Journal*, 2021. doi: 10.1109/JIOT.2020.3040358.

[44] H. N. Nguyen and A. Rakib. Probabilistic resource-bounded alternating-time temporal logic. In *AAMAS'19*, pages 2141–2143, Richland, SC, May 2019. IFAAMAS. ISBN 9781450363099.

[45] A. Pnueli. The temporal logic of programs. In *SFCF'77*, pages 46–57. IEEE, IEEE, Sept. 1977. doi: 10.1109/SFCS.1977.32.

[46] P.-Y. Schobbens. Alternating-time logic with imperfect recall. *Electron. Notes Theor. Comput. Sci.*, 85:82–93, 2004. ISSN 1571-0661. doi: 10.1016/S1571-0661(05)82604-0. LCMAS'03.

[47] Y. Shoham and M. Tennenholtz. On social laws for artificial agent societies: Off-line design. *AIJ*, 73:231–252, 1995.

[48] W. van der Hoek and M. J. Wooldridge. Cooperation, knowledge, and time: Alternating-time temporal epistemic logic and its applications. *Stud Logica*, 75(1):125–157, 2003. doi: 10.1023/A:1026185103185.

[49] W. van der Hoek, B. van Linder, and J.-J. C. Meyer. A logic of capabilities. In A. Nerode and Y. V. Matiyasevich, editors, *Logical Foundations of Computer Science*, pages 366–378, Berlin, Heidelberg, 1994. Springer. ISBN 978-3-540-48442-4.

[50] S. Vester. Alternating-time temporal logic with finite-memory strategies. In G. Puppis and T. Villa, editors, *GandALF'13*, volume 119, pages 194–207, 2013. doi: 10.4204/EPTCS.119.17.

[51] D. Walther, W. van der Hoek, and M. Wooldridge. Alternating-time temporal logic with explicit strategies. In *TARK'07*, pages 269–278, New York, NY, USA, June 2007. ACM. ISBN 9781450378413. doi: 10.1145/1324249.1324285.

[52] T. Ågotnes, V. Goranko, and W. Jamroga. Alternating-time temporal logics with irrevocable strategies. In *TARK'07*, pages 15–24, New York, NY, USA, June 2007. ACM. ISBN 9781450378413.