Probabilistic Timed ATL

Wojciech Jamroga
Institute of Computer Science,
Polish Academy of Sciences
& SnT, University of Luxembourg
jamroga@ipipan.waw.pl

Marta Kwiatkowska
Institute of Computer Science, PAS
& Department of Computer Science,
University of Oxford
marta.kwiatkowska@cs.ox.ac.uk

Wojciech Penczek
Institute of Computer Science,
Polish Academy of Sciences
Warsaw, Poland
penczek@ipipan.waw.pl

Laure Petrucci
LIPN, CNRS UMR 7030,
Université Sorbonne Paris Nord
Villetaneuse, France
petrucci@lipn.univ-paris13.fr

Teofil Sidoruk
Institute of Computer Science,
Polish Academy of Sciences
Warsaw, Poland
t.sidoruk@ipipan.waw.pl

ABSTRACT

We consider strategic reasoning for multi-agent systems modelled as networks of continuous-time probabilistic timed automata (TA) with asynchronous execution (PCAMAS) in the setting of imperfect information. We define **PTATL**, a probabilistic extension of the alternating-time timed temporal logic **TATL**, which is interpreted over PCAMAS. Focusing on memoryless strategies of agents with imperfect information, both probabilistic (irP) and deterministic (irp), we establish theoretical results regarding the computational complexity of model checking for the proposed logic: between **PSPACE** and **EXPTIME** for $\mathbf{PTATL}_{\mathrm{irp}}$, and in **2EXPTIME** for $\mathbf{PTATL}_{\mathrm{irP}}$. We demonstrate the practical feasibility of verification for $\mathbf{PTATL}_{\mathrm{irp}}$ formulas through a novel proof-of-concept combination of state-of-the-art tools IMITATOR and PRISM on a scalable benchmark, with encouraging results.

KEYWORDS

multi-agent systems; probabilistic model checking; strategic ability; continuous time

ACM Reference Format:
Wojciech Jamroga, Marta Kwiatkowska, Wojciech Penczek, Laure Petrucci, and Teofil Sidoruk. 2025. Probabilistic Timed ATL. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Detroit, Michigan, USA, May 19 – 23, 2025*, IFAAMAS, 9 pages.

1 INTRODUCTION

Alternating-time temporal logics **ATL*** and **ATL** [3, 4] extend the temporal logic **CTL*** and **CTL**, resp., with the notion of *strategic ability*. These logics allow one to express properties of agents (or groups of agents) referring to what they can achieve. Such properties can be useful for specification, verification, and reasoning about agent interactions in multi-agent systems [25, 27, 31, 32, 41].

Timed extensions of strategy logics with two types of interpretations, over models of synchronous (Time) Multi-Agent Systems (MAS) and asynchronous (Time) Multi-Agent Systems (AMAS), and both discrete (D) and continuous (C) time, were investigated in [8].

Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Y. Vorobeychik, S. Das, A. Nowé (eds.), May 19 – 23, 2025, Detroit, Michigan, USA. © 2025 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

Probabilistic (untimed) extensions of **ATL** and **ATL*** (**PATL** and **PATL***) were also developed, initially in the perfect information setting [16, 17, 37], and later also under imperfect information [10, 12]. Probability is needed to model randomisation, which is often used to coordinate multi-agent systems, and to quantify failures, for example message loss on unreliable communication channels in Aloha [37] or information leakage in a security protocol. Note that time and observability are equally relevant in this setting, e.g. for specifying deadline properties [37], or modelling voter–coercer interactions in electronic voting [28].

Probabilistic real-time systems can be modelled using (networks of) probabilistic TA[1] and verified against (non-strategic) probabilistic timed temporal logic **PTCTL** and **PTCTL*** by reduction to probabilistic variants of region or zone graphs [42], though only the perfect information setting has been considered. Strategy synthesis algorithms have been formulated for probabilistic real-time multiplayer games using the coalition operator of **ATL** [35], but again only for perfect information, with no logic or complexity of model checking discussed. Solving partially observable (untimed) variants of finite [22, 23] and infinite [47] games was studied, though again without logic.

This paper extends previous works to integrate probabilistic and real-time aspects within a single coherent framework, introducing Probabilistic Timed **ATL** (**PTATL**). While adding discrete time can be viewed as a mild extension of the untimed probabilistic setting, the continuous-time setting is non-trivial and considered here for the first time. Focusing on the memoryless strategies with imperfect information, both deterministic and probabilistic, we prove complexity results for the **PTATL** model checking problem. For deterministic imperfect information strategies, we were able to adapt state-of-the-art tools IMITATOR [6] and PRISM [34] to derive a promising proof-of-concept implementation of strategy synthesis in this challenging setting.

Related work. Alternating-time Temporal Logic **ATL*** and its subset **ATL** [3] have become the most popular formalisms for reasoning about strategic abilities in multi-agent systems (MAS). Over the recent years, these logics were further extended in various directions, including (discrete) Timed **ATL** (**TATL**) [39], epistemic **ATLK** [20, 26], $\mathbf{ATL}_{\mathrm{sc}}$ with strategy contexts [14, 38], or Strategy Logic (**SL**) [15], where strategies are represented as first order variables.

[1] Since we discuss both *parametric* and *probabilistic* timed automata (TA), we will not use the acronym PTA to avoid ambiguity.

Strategic Timed **CTL** (**STCTL**), introduced in [8], can be seen as a branching-time counterpart to **TATL**.

Algorithms for stochastic model checking have been considered since the mid-1980s, initially in the context of probabilistic concurrent programs and classical **LTL** properties [46]. The specification language was first extended with probabilities in **PCTL** [21], which replaced **CTL** path quantifiers with probabilistic operators quantifying the probability of satisfaction for a set of paths (and also with discrete time bounds on their evaluation). Continuous-time **PTCTL**, interpreted over probabilistic TA, is considered in [42].

The earliest work on probabilistic verification of strategic abilities introduced **PATL** and **PATL*** [17], which were subsequently augmented with additional operators to express rewards [16] and reason about Nash equilibria [36]. Stochastic multi-player games have been extensively studied, too, including models with real time [35] and partial observability (untimed) settings [22, 23, 47]. Probabilistic Strategy Logic (**PSL**), in a similar formal setting to that considered here (stochastic MAS, albeit with synchronous execution and perfect information strategies), was proposed in [5].

Models are typically specified as Markov decision processes (MDPs), stochastic multi-player games, or as probabilistic TA. Our formalism of Asynchronous Multi-Agent Systems (AMAS) combines features of the latter two. AMAS were originally introduced in [30] (though it should be noted the modelling approach is standard and inherited from distributed and interpreted systems [44]), and subsequently extended with variables in [7], and with both discrete and continuous time in [8]. Here, we take the latter variant and lift it to the probabilistic case.

The closest related works are [11, 12], which investigate a probabilistic extension of **ATL** for agents with imperfect information. This semantics of strategic ability is better suited for most practical applications (where agents typically should not know the entire global state of the system). However, its downside is the higher computational complexity of verification, since the efficient fixpoint techniques for perfect information strategies are not applicable. As a result, agents with imperfect information are considered less often, and relevant work involving probabilistic models and logics is particularly scarce. This setting was previously considered in [24], which showed that **PATL** model checking is undecidable for strategies with imperfect information perfect recall even for single-agent coalitions; a further reduced subset of the logic is identified, however, in which the problem is decidable and in **2EXPTIME**.

2 PRELIMINARIES

In this section, we recall the formalism of Asynchronous Multi-Agent Systems (AMAS) with continuous (dense) time [8], and the logical framework for reasoning about strategic abilities of agents in such models, including the syntax and semantics of **PTATL**.

2.1 Probabilistic Continuous-time AMAS

Asynchronous Multi-Agent Systems (AMAS), introduced in [30], are a modern semantic model for the study of agents' strategies in asynchronous systems. Technically, they are similar to networks of automata that synchronise on shared actions, and interleave local transitions to execute asynchronously [19, 30, 40]. However, to deal with agent coalitions, automata semantics (e.g. for Timed Automata) must resort to algorithms and additional attributes. In contrast, by linking protocols to agents, AMAS are a natural compositional formalism to analyse multi-agent systems. In [8], AMAS were extended with time representation, following the standard notions of *timed systems* [2]. Here, we recall and further augment this formalism, lifting it to the probabilistic case. We assume the standard definition of probability distribution.

Definition 1 (Distributions). *A probability distribution over a finite, non-empty set X is a function $d : X \to [0,1]$ such that $\sum_{x\in X} d(x) = 1$. If there exists $x \in X$ such that $d(x) = 1$, we refer to d as a* Dirac distribution *or* point distribution. *The set of distributions over X is denoted by $Dist(X)$.*

Each agent has an associated set of clocks. All clocks evolve at the same rate (across agents), thus allowing for delays and instantaneous actions. $\mathbb{R}_{0+}$ denotes the set of non-negative real numbers.

Definition 2 (Clocks). *We denote a finite set of* clocks*, with a fixed ordering assumed for simplicity, by $\mathcal{X} = \{x_1, \dots, x_{|\mathcal{X}|}\}$, where $x_i \in \mathbb{R}_{0+}$ for all $i \in \{1, \dots, |\mathcal{X}|\}$. A* clock valuation *on $\mathcal{X}$ is an $|\mathcal{X}|$-tuple v. We denote:*

- *by $v(x_i)$ or $v(i)$, the value of clock x_i in v;*
- *by $v + \delta$, where $\delta \in \mathbb{R}_{0+}$, v' such that $v'(x) = v(x) + \delta$ for all $x \in \mathcal{X}$;*
- *by $v[X := 0]$, where $X \subseteq \mathcal{X}$, v' such that $v'(x) = 0$ for all $x \in X$, and $v'(x) = v(x)$ for all $x \in \mathcal{X} \setminus X$.*

Definition 3 (Clock constraints). *The set $C_{\mathcal{X}}$ collects all* clock constraints *over $\mathcal{X}$, defined by the grammar: $\mathtt{cc} := true \mid x_i \sim c \mid x_i - x_j \sim c \mid \mathtt{cc} \wedge \mathtt{cc}$, where $x_i, x_j \in \mathcal{X}$, $c \in \mathbb{N}$, and $\sim \in \{\leq, <, =, >, \geq\}$. For $\mathtt{cc} \in C_{\mathcal{X}}$, the satisfaction relation $\models$ is inductively defined as:*

- *$v \models true$,*
- *$v \models (x_i \sim c)$ iff $v(x_i) \sim c$,*
- *$v \models (x_i - x_j \sim c)$ iff $v(x_i) - v(x_j) \sim c$, and*
- *$v \models (\mathtt{cc} \wedge \mathtt{cc}')$ iff $v \models \mathtt{cc}$ and $v \models \mathtt{cc}'$.*

The set of all valuations satisfying $\mathtt{cc}$ is denoted by $[\![\mathtt{cc}]\!]$.

Definition 4 (PCAMAS). *A* probabilistic continuous-time AMAS (PCAMAS) *consists of n agents $\mathcal{A} = \{1, \dots, n\}$, each associated with a 9-tuple $AG_i = (L_i, \iota_i, Act_i, P_i, \mathcal{X}_i, \mathcal{I}_i, T_i, PV_i, V_i)$ including:*

- *a finite non-empty set of* local states $L_i = \{l_i^1, l_i^2, \dots, l_i^{n_i}\}$;
- *an* initial local state $\iota_i \in L_i$;
- *a finite non-empty set of* local actions $Act_i = \{a_i^1, a_i^2, \dots, a_i^{m_i}\}$;
- *a* local protocol $P_i : L_i \to 2^{Act_i} \setminus \{\emptyset\}$;
- *a set of* clocks $\mathcal{X}_i$;
- *a* local invariant $\mathcal{I}_i : L_i \to C_{\mathcal{X}_i}$;
- *a (partial)* probabilistic local transition function $T_i : L_i \times Act_i \times C_{\mathcal{X}_i} \times 2^{\mathcal{X}_i} \rightharpoonup Dist(L_i)$ *such that $T_i(l_i, a, \mathtt{cc}, X) \in Dist(L_i)$ iff $a \in P_i(l_i)$, $\mathtt{cc} \in C_{\mathcal{X}_i}$, and $X \subseteq \mathcal{X}_i$;*
- *a finite non-empty set of* local propositions $PV_i = \{\mathrm{p}_\mathrm{i}^1, \dots, \mathrm{p}_\mathrm{i}^{r_i}\}$;
- *a* local valuation function $V_i : L_i \to 2^{PV_i}$.

For a local transition $t := l \xrightarrow{a,\mathtt{cc},X} (l', d(l'))$ for a distribution $d \in Dist(L_i)$, l is the *source* state, a is the executed *action*, which has probability $d(l')$ of moving to *target* state l', clock condition $\mathtt{cc}$ is called a *guard*, and X is the set of clocks to be *reset*. Note that T_i is defined on local actions only. This is reflected in the definition of PCAMAS models, or Interleaved Interpreted Systems [30, 40].

Definition 5 (Model). *The* model *of a PCAMAS is an 8-tuple* $M = (\mathcal{A}, S, \iota, Act, \mathcal{X}, \mathcal{I}, T, V)$*, including:*

- *the set of* agents $\mathcal{A} = \{1, \ldots, n\}$;
- *the set of* global states $S = \prod_{i=1}^{n} L_i$;
- *the* initial global state $\iota = (\iota_1, \ldots, \iota_n) \in S$;
- *the set of* actions $Act = \bigcup_{i \in \mathcal{A}} Act_i$;
- *the set of* clocks $\mathcal{X} = \bigcup_{i \in \mathcal{A}} \mathcal{X}_i$;
- *the* invariant $\mathcal{I}(s) = \bigwedge_{i \in \mathcal{A}} \mathcal{I}_i(s^i)$*, where* s^i *is* i*'s local state of* s;
- *the* probabilistic global transition function $T : S \times Act \times C_{\mathcal{X}} \times 2^{\mathcal{X}} \to Dist(S)$*, such that* $T(s, a, \bigwedge_{i \in Agent(a)} \mathtt{cc}_i, X) = d$ *iff:*
  $\forall s' \in S, \quad d(s') = \prod_{i \in Agent(a)} d_i((s')^i)$,
  where $Agent(a) = \{i \in \mathcal{A} | a \in Act_i\}$*, and* $d_i = T_i(s^i, a, \mathtt{cc}_i, X_i)$;
- *the* valuation function $V \colon S \to 2^{PV}$*, where* $PV = \bigcup_{i=1}^{n} PV_i$*, s.t. we have* $V(s) = (V_1(s^1), \ldots, V_n(s^n))$ *for each* $s \in S$.

Intuitively, an action a shared by agents $i \in \{1, \ldots, n\}$, whose transitions T_i define respectively $m_1, \ldots, m_n$ local successors upon executing a, induces $m_1 \cdot \ldots \cdot m_n$ possible global state successors. Each one is assigned the product of probabilities of all local successors involved.

The continuous (dense) semantics of time defines *concrete states* as tuples of global states and non-negative real clock valuations.

Definition 6 (PACTS). *The* concrete model *of a PCAMAS model* $M = (\mathcal{A}, S, \iota, \mathcal{X}, \mathcal{I}, T, V)$ *is given by its* Probabilistic Asynchronous Continuous Transition System *(PACTS): a 5-tuple* $(\mathcal{A}, \mathcal{CS}, q_\iota, \rightarrow_c, V_c)$*, including:*

- *the set of* agents $\mathcal{A} = \{1, \ldots, n\}$;
- *the set of* concrete states $\mathcal{CS} = S \times \mathbb{R}_{0+}^{|\mathcal{X}|}$;
- *the* concrete initial state $q_\iota = (\iota, v) \in \mathcal{CS}$*, s.t.* $\forall x_i \in \mathcal{X}, v(x_i) = 0$;
- *the* probabilistic transition relation $\rightarrow_c \subseteq \mathcal{CS} \times ((\mathbb{R}_{0+} \times \mathcal{CS}) \cup (Act \times Dist(S)))$*, defined by time- and probabilistic action-successors as follows:*
  - $(s, v) \xrightarrow{\delta}_c (s, v + \delta)$ *such that* $\delta \in \mathbb{R}_{0+}$ *and* $v, v + \delta \in [\![\mathcal{I}(s)]\!]$,
  - $(s, v) \xrightarrow{a,p}_c (s', v')$ *iff there are* $a \in Act$, $p \in [0, 1]$, $\mathtt{cc} \in C_{\mathcal{X}}$, $X \subseteq \mathcal{X}$ *such that:* $v \in [\![\mathtt{cc}]\!]$, $v \in [\![\mathcal{I}(s)]\!]$, $v' = v[X := 0]$, $v' \in [\![\mathcal{I}(s')]\!]$*, and* $p = T(s, a, \mathtt{cc}, X)(s')$;
- *the* valuation function $V_c(s, v) = V(s)$.

Intuitively, delays $\xrightarrow{\delta}_c$ increase the clock valuation(s) by a given δ but do not change the global state, while probabilistic actions $\xrightarrow{a,p}_c$ move to a successor state, possibly resetting some clocks. For $q = (s, v)$, we use $state(q)$ for s and $vclock(q)$ for v.

Note that untimed probabilistic AMAS (PAMAS) are a special case of Def. 4, where $\mathcal{X}_i = \emptyset$ for all $i \in \mathcal{A}$. Since the set of clocks is empty, the concrete model contains only action transitions, and thus it is identical to the model itself.

Executions of PACTS (i.e. concrete PCAMAS models) are interleaved sequences of concrete states and delays or probabilistic action transitions between them, defined as follows.

Definition 7 (Execution). *An execution of PACTS from a concrete state* $q_0 = (s_0, v_0)$ *is an infinite sequence of interleaved timed- and probabilistic action steps* $\rho = q_0, \delta_0, q_0', a_0, q_1, \delta_1, q_1', a_1, \ldots$*, where* $q_i = (s_i, v_i)$, $q_i' = (s_i, v_{i+1})$*, such that for each* $i \geq 0$ *we have:* $q_i \xrightarrow{\delta_i}_c q_i'$, $q_i' \xrightarrow{a_i, p_i}_c q_{i+1}$*, for some* $\delta_i \in \mathbb{R}_{0+}$, $a_i \in Act$*, and* $p_i > 0$.

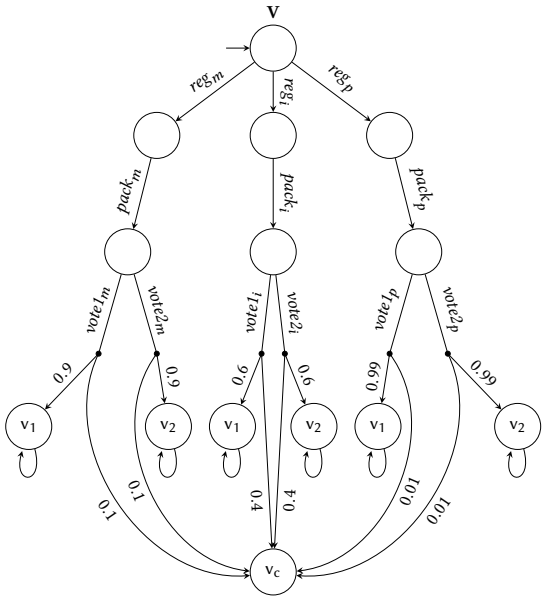

Figure 1: Voter V in the PCAMAS from Example 1.

We denote a finite prefix (or a *history*) of an execution by h, the last concrete state of h by $last(h)$, and the set of all histories by H.

Example 1. *Consider a simple model inspired by Estonian election procedures, previously featured in [8], in which a voter V selects one of three modalities (vote by mail, over the internet, or at a polling station), receives the appropriate voting package from the election authority EA (e.g. a postal ballot, e-voting access credentials, or address of the local election office), and proceeds to vote for a candidate. We augment the original model by adding a chance of coercion with different probabilities depending on the chosen voting modality. Agents comprising the PCAMAS for this scenario are depicted in Figs. 1 and 2.*

2.2 Strategies and Outcomes

Strategies are conditional plans that dictate the (potentially probabilistic) choice of each agent $i \in \mathcal{A}$ in each possible situation, that is, a function $\sigma_i \colon H \to Dist(Act_i)$. In deterministic MAS, the strategy type is usually classified based on the agents' *state information*: perfect (I) vs. imperfect (i), and their *recall of state history*: perfect (R) vs. no recall (r) [45]. Here we add yet another angle, namely probabilistic (P) vs. non-probabilistic (p) strategies. To that end, we extend the standard notation as follows:

Definition 8 (irP-strategy). *A* probabilistic memoryless imperfect information (irP) strategy *for agent* $i \in \mathcal{A}$ *is a function* $\sigma_i \colon L_i \to Dist(Act_i)$ *such that for each* $a \in Act_i$, $l \in L_i$*, and* $d \in Dist(Act_i)$*: if* $d(a) > 0$*, then* $a \in P_i(l)$.

Probabilistic strategies indicate the probability of choosing each action from a given location, as opposed to probabilities of transitions resulting from choosing the action as in the model definition (Def. 5). Moreover, non-probabilistic (or deterministic) memoryless imperfect information strategies, denoted by irp, are a special case of the above where only point distributions are considered.

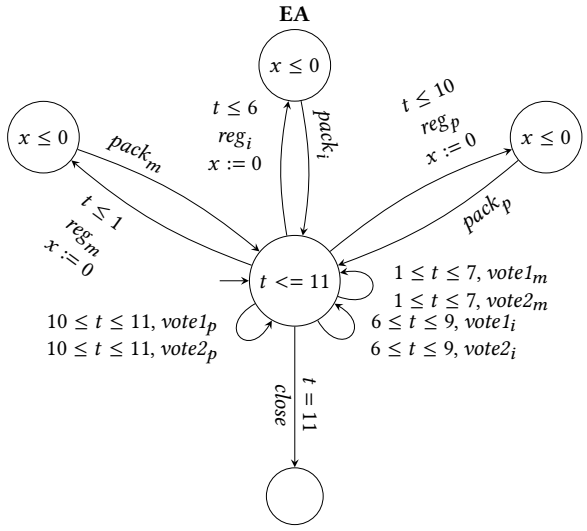

Figure 2: Election authority *EA* in the PCAMAS from Ex. 1.

Definition 9 (irp-strategy). *An* irp-strategy *for agent $i \in \mathcal{A}$ is an* irP*-strategy σ_i such that each $\sigma_i(l)$ is a point distribution.*

In the following, we will parameterise strategy type with $Y \in \{\text{irP}, \text{irp}\}$ wherever both variants are applicable. Str_i^Y denotes the set of all Y-strategies of agent i. A *joint Y-strategy* σ_A of coalition $A \subseteq \mathcal{A}$ is a tuple of Y-strategies, one for each agent $i \in A$.

Definition 10 (Outcome paths). *Let $Y \in \{\text{irP}, \text{irp}\}$, $A \subseteq \mathcal{A}$, M be a PCAMAS model, and ρ be an execution of the concrete model of M. The set of* outcome paths *of Y-strategy σ_A in state q_0 of the concrete model of M, denoted by $Out_M^Y(q_0, \sigma_A)$, collects all executions $\rho = q_0, \delta_0, q_0', a_0, \ldots$, where $q_j' = (s_j', v_j')$, such that for each $i, j \geq 0$ if $i \in Agent(a_j) \cap A$, then we have $\sigma_i(s_j'^{i})(a_j) > 0$.*

To enable quantitative reasoning about the probabilities of **PTATL** properties, we define the probability measure over outcome paths.

Let $A \subseteq \mathcal{A}$, M be a PCAMAS model, and σ_A an Y-strategy, where $Y \in \{\text{irP}, \text{irp}\}$. Note that the outcome paths of σ_A can also be seen as the outcome paths of a general strategy σ for all agents $\mathcal{A}$, such that $\sigma(h) = \sigma_A(last(h))$, while opponents in $\bar{A} = \mathcal{A} \setminus A$ are not restricted to memoryless strategies and can make any probabilistic choices $\sigma_{\bar{A}}$ based on state history and time. Following the standard construction [13, 18, 33], used e.g. in [12], we have that the set of outcome paths $Out_M^Y(q_0, \sigma_A)$ induces an infinite-state Markov chain $\mathcal{MC}_M^Y(q, \sigma_A) = (St, Pr)$, defined as follows. States $St \subseteq H$ are the finite prefixes (i.e. histories) of the paths in $Out_M^Y(q_0, \sigma_A)$. Distribution $Pr \in Dist(H \times H)$ is defined as $Pr(h, hq') =$

$$\begin{cases} \sum_{\substack{a \in Act,\\ cc \in C_X,\\ X \in \mathcal{X}}} \sigma(h)(a) \cdot T(last(h), a, cc, X)(q') & \text{if } last(h) \xrightarrow{a,p}_c q' \\ 1 & \text{if } last(h) \xrightarrow{\delta}_c q'. \end{cases}$$

for some $h, hq' \in H$, $a \in Act$, $p \in [0,1]$, and $\delta \in \mathbb{R}_{0+}$. Moreover, $\mathcal{MC}_M^Y(q, \sigma_A)$ induces a canonical probability space on its infinite paths, and thus also on $Out_M^Y(q_0, \sigma_A)$.

For history $h \in St$, by a *cylinder set* $C(h)$ we mean the set of all infinite paths in $\mathcal{MC}_M^Y(q, \sigma_A)$ (which can be identified with the outcome paths of strategy σ_A) that start with prefix h. From [9, 46], we have that there is a unique smallest σ-algebra Σ^h that contains all cylinder sets $C(h)$. Let $h = q_0, \delta_0, q_0', a_0, \ldots, q_m, \delta_m, q_m', a_m$. Following [9], there exists a unique probability measure $Prob_h$ on Σ^h, defined as $Prob_h(C(h)) = \prod_{k \in [0,m]} Pr(h_k, h_{k+1})$, where $h_k = q_0, \delta_0, q_0', a_0, \ldots, q_k, \delta_k, q_k', a_k$.

Definition 11 (Outcome). *Let M be a model, σ_A a joint Y-strategy and $\mathcal{MC}_M^Y(q, \sigma_A) = (St, Pr)$ the Markov chain induced by the outcome paths of σ_A in concrete state $q \in \mathcal{CS}$. The* outcome *of σ_A in q, denoted by $out_M^Y(q, \sigma_A)$, is the set of all probability measures $Prob_h$, for $h \in St$.*

We will use $\mu_M^Y(q, \sigma_A)$ to iterate over the elements of $out_M^Y(q, \sigma_A)$.

2.3 Syntax and Semantics of PTATL

Assume a countable set PV of atomic propositions, and a finite set $\mathcal{A}$ of agents. The syntax of Probabilistic Timed **ATL** (**PTATL**) is defined as:

$$\varphi ::= \mathsf{p} \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\langle A \rangle\rangle^{\bowtie z} \varphi \, \mathsf{U}_I \varphi \mid \langle\langle A \rangle\rangle^{\bowtie z} \varphi \, \mathsf{R}_I \varphi,$$

where $\mathsf{p} \in PV$, $A \subseteq \mathcal{A}$, $\bowtie \in \{\leq, <, >, \geq\}$, $z \in [0,1]$, and $I \subseteq \mathbb{R}_{0+}$ is an interval with bounds $[n, n']$, $[n, n')$, $(n, n']$, (n, n'), (n, ∞), or $[n, \infty)$, for $n, n' \in \mathbb{N}$. The operator $\langle\langle A \rangle\rangle^{\bowtie z}$ states that coalition A has a strategy to enforce the temporal property that follows with a probability in relation $\bowtie$ with constant z. U ("strong until") and R ("release") are standard temporal operators. G ("always"), F ("eventually"), and Boolean connectives can be derived as usual.

Example 2. *The formula $\varphi = \langle\langle V \rangle\rangle^{\geq 0.8} \mathsf{F}_{[0,8]} \mathsf{v}_1$ states that V has a strategy to successfully vote for candidate 1 (i.e. avoid being coerced to pick c instead) within 8 time units, with probability at least 0.8.*

Below, we give the continuous-time semantics of **PTATL**.

Definition 12 (**PTATL** Semantics). *Let $Y = \{\text{irP}, \text{irp}\}$, $M = (\mathcal{A}, \mathcal{CS}, q_\iota, \rightarrow_c, V_c)$ be a PACTS, $q_0 = (s_0, v_0) \in \mathcal{CS}$ a concrete state, $A \subseteq \mathcal{A}$, φ, ψ be* **PTATL** *formulae, $\rho = q_0, \delta_0, q_0', \alpha_0, \ldots$ an execution of M where $q_k = (s_k, v_{k+1})$. The Y-semantics of* **PTATL** *is given as:*

- $M, (s,v) \models \mathsf{p}$ *iff* $\mathsf{p} \in V_c(s,v)$,
- $M, (s,v) \models \neg\varphi$ *iff* $M, (s,v) \not\models \varphi$,
- $M, (s,v) \models \varphi \wedge \psi$ *iff* $M, (s,v) \models \varphi$ *and* $M, (s,v) \models \psi$,
- $M, (s,v) \models \langle\langle A \rangle\rangle^{\bowtie z} \gamma$ *iff there exists a joint Y-strategy σ_A such that for all $\mu_M^Y((s,v)), \sigma_A) \in out_M^Y((s,v), \sigma_A)$ we have $\mu_M^Y((s,v)), \sigma_A)(\{\rho \mid M, \rho \models \gamma\}) \bowtie z$, where*
  - $M, \rho \models \gamma_1 U_I \gamma_2$ *iff there is $r \in I$ such that: $M, \pi_\rho(r) \models \gamma_2$ and for all $0 \leq r' < r$: $M, \pi_\rho(r') \models \gamma_1$.*
  - $M, \rho \models \gamma_1 R_I \gamma_2$ *iff for all $r \in I$: $M, \pi_\rho(r) \models \gamma_2$ or there is $0 \leq r' < r$: $M, \pi_\rho(r') \models \gamma_1$.*

In the above, by π_ρ we mean the dense path corresponding to ρ, i.e. a mapping from $\mathbb{R}_{0+}$ to a set of concrete states [43], given by $\pi_\rho(r) = (s_i, v_i + \delta)$, for $r = \Sigma_{j=0}^{i-1} \delta_j + \delta$, where $i \geq 0$ and $0 \leq \delta < \delta_i$.

3 MODEL CHECKING PROBABILISTIC TATL

In this section, we discuss theoretical results regarding the model checking problem for **PTATL** with continuous time, considering both probabilistic (irP) and deterministic (irp) strategies. Then, we

introduce our approach to practical model checking, applicable to the latter semantics of strategic ability.

3.1 Complexity Results for TATL

In previous works, **TATL** has only been considered in the discrete-time semantics. Therefore, we begin by establishing the complexity of model checking **TATL** with continuous time (**TATL**$^{\mathrm{C}}$).

Proposition 1. **TATL** *semantically subsumes* **TCTL**.

Proof. The cases of propositions and Boolean connectives are straightforward. Consider **TCTL** formulas $\varphi_1 = \forall\gamma$ and $\varphi_2 = \exists\gamma$, where $\gamma = \psi_1 \mathsf{U}_I \psi_2$ or $\gamma = \psi_1 \mathsf{R}_I \psi_2$. Using the empty coalition $\emptyset$ and the grand coalition $\mathcal{A}$, φ_1 and φ_2 can be equivalently expressed in **TATL** as $\varphi_1' = \langle\langle\emptyset\rangle\rangle\gamma$ and $\varphi_2' = \neg\langle\langle\mathcal{A}\rangle\rangle\neg(\gamma)$, respectively, where $\neg(\psi_1 \mathsf{U}_I \psi_2) = (\neg\psi_1)\,\mathsf{R}_I(\neg\psi_2)$ and $\neg(\psi_1 \mathsf{R}_I \psi_2) = (\neg\psi_1)\,\mathsf{U}_I(\neg\psi_2)$. Note that the strategic modality generalises quantifiers $\forall$ and $\exists$, allowing for quantification over strategies of coalitions between these extremes; such properties cannot be expressed in **TCTL**. □

Theorem 2. *Model checking* **TATL**$_{\mathrm{ir}}{}^{\mathrm{C}}$ *is* **PSPACE***-complete.*

Proof. The lower bound follows from Proposition 1 and the **PSPACE**-completeness of **TCTL** model checking [1].

The upper bound follows from the previous results for **STCTL**, whose model checking is **PSPACE**-complete [8, Theorem 4.5], and which subsumes **TATL** [8, Proposition 5.3]. □

3.2 PTATL with Deterministic Strategies

We begin, similarly to [11], by looking at the special case where coalitions can only use deterministic (non-randomized) strategies, i.e. considering the verification of **PTATL**$_{\mathrm{irp}}$. Below, we fix some additional notations.

- Let $reg(M,\varphi)$ denote the set of regions in the detailed region graph for model M and **PTATL** formula φ, obtained by the standard construction in [1, 43]. Moreover, for a clock valuation v, we will denote its region in $reg(M,\varphi)$ by $reg(v)$. Note that, while the construction was originally defined for models and formulas of **TCTL**, the addition of strategic operators and probabilistic constraints does not affect it, so we apply it for the **TCTL** formula obtained by replacing the strategy operators with $\forall$ in φ.
- Observe also that, for each $r \in reg(M,\varphi), v, v' \in r, s \in S$, and each state subformula ψ of φ, we have $M,(s,v) \models \psi$ iff $M,(s,v') \models \psi$. To prove it, consider (s,v) and (s,v') with $v, v' \in r$. Then, we can show that for each execution $\rho = q_0, \delta_0, q_0', a_0, q_1, \delta_1, q_1', a_1, \ldots$, starting at $q_0 = (s,v)$ there is a corresponding execution $\rho' = u_0, \delta_0', u_0', a_0', u_1, \delta_1', u_1', a_1', \ldots$ starting at $q_0' = (s,v')$ s.t. for each $i \geq 0$ we have $a_i = a_i'$, $state(q_i) = state(u_i)$; $vclock(q_i) \in r_i$ and $vclock(u_i) \in r_i$ for some $r_i \in reg(M,\varphi)$, and $state(q_i') = state(u_i')$; $vclock(q_i') \in r_i'$ and $vclock(u_i') \in r_i'$ for some $r_i' \in reg(M,\varphi)$. This follows by induction from the fact that the same probabilistic actions are enabled at q_i' and u_i' and the same delays can be taken at q_i and u_i, for each $i \geq 0$. The base case follows from the fact that $v, v' \in r$. Therefore, by induction on the complexity of a formula φ, we can show that $M,(s,v) \models \psi$ iff $M,(s,v') \models \psi$, for each subformula ψ of φ.
- An *extended model* $\hat{M}$ for φ extends model M by some fresh atomic propositions $\hat{PV}$ and a valuation function $\hat{V} : (PV \cup \hat{PV}) \to S \times reg(M,\varphi)$. That is, the propositional variables are now evaluated w.r.t. the current state and the equivalence class for clock valuations. We assume $\hat{V}(\mathrm{p}) = \{(s,r) \mid \mathrm{p} \in V(s), r \in reg(M,\varphi)\}$ for $\mathrm{p} \in PV$.
- By $Int(\varphi)$, we denote the number of intervals I in a state (or path) formula φ. We also say that φ is *flat* if it contains no proper subformula with a strategic operator.

Proposition 3. *Let φ be a* **PTATL**$_{\mathrm{irp}}$ *formula, $\psi \equiv \langle\langle A\rangle\rangle^{\bowtie z}\gamma$ a flat subformula of φ, M a probabilistic timed model (as in Definition 5), s a state in M, and $r \in reg(M,\varphi)$ a region in the detailed region graph for model M,φ. Checking whether $\forall_{v\in r}\, M,(s,v) \models \psi$ can be done in $2^{O((|S|+|\mathcal{X}|+Int(\varphi))\log(|T|+|\mathcal{I}|))}$ steps.*

Proof. Observe after [11, 12] that, in finite games, the opponents always have a deterministic best-response strategy to any given (deterministic or probabilistic) strategy σ_A of the coalition. The following procedure is used to check if $\forall_{v\in r}\, M,(s,v) \models \langle\langle A\rangle\rangle^{\bowtie z}\gamma$:

- **repeat**
  - Guess an irp-strategy σ_A for coalition A;
  - **for** every Irp-strategy $\sigma_{\mathcal{A}\setminus A}$ **do**
    - $M' :=$ M pruned according to $(\sigma_A, \sigma_{\mathcal{A}\setminus A})$;
    - $M'' :=$ the detailed region graph model of M';
    - $win :=$ output of **PCTL** model checking of $M'', (s,v) \models \mathbb{P}^{\bowtie z}\gamma$, where $\mathbb{P}^{\bowtie z}\gamma$ expresses that the probability measure of the set of paths satisfying γ is in relation $\bowtie$ with z;
- **until** $win = true$ or no more irp-strategies for A;
- Return(win).

Note: (a) the combined **repeat**/**for** loops iterate $O(|Act|^{|\mathcal{A}|\cdot|S|}) \leq O(|T|^{|S|})$ times; (b) pruning takes $O(|S| \cdot |Act| \cdot |\mathcal{A}|) = O(|T|)$ steps; (c) construction of the region graph takes $O((|S| + |T| + |\mathcal{I}|)^{|\mathcal{X}|+Int(\varphi)}) = O((|T| + |\mathcal{I}|)^{|\mathcal{X}|+Int(\varphi)})$ steps, and the resulting graph has as many states [1, 43]; (d) model checking **PCTL** is cubic in the number of states in the model [21]. Thus, the above algorithm runs in time $O\big(|T|^{|S|} \cdot \big(|T| + (|T| + |\mathcal{I}|)^{|\mathcal{X}|+Int(\varphi)} + (|T| + |\mathcal{I}|)^{3(|\mathcal{X}|+Int(\varphi))}\big)\big) = O\big(|T|^{|S|} \cdot (|T| + |\mathcal{I}|)^{3(|\mathcal{X}|+Int(\varphi))}\big) = 2^{O((|S|+|\mathcal{X}|+Int(\varphi))\log(|T|+|\mathcal{I}|))}$. □

Theorem 4. *Model checking* **PTATL**$_{\mathrm{irp}}$ *is* **PSPACE***-hard.*

Proof. The lower bound follows by a reduction of **TCTL** model checking, which is **PSPACE**-complete [1]. Given are: a model M, a state s in M, and a **TCTL** formula φ. Before we present the reduction, we make some important observations.

Note that M can be seen as a stochastic model with only Dirac probability distributions for transitions. Moreover, in finite games, the opponents always have a deterministic best-response strategy to any given strategy σ_A of the coalition. Thus, $M,(s,v) \models_{\mathbf{TCTL}} \forall\gamma$ **iff** the agents in $\emptyset$ enforce γ on all paths **iff** they do so against all the probabilistic responses from $\mathcal{A}$. Since the set of best responses includes deterministic strategies of $\mathcal{A}$ in the deterministic CGS[2] M, this is equivalent to saying that $M,(s,v) \models_{\mathbf{PTATL}_{\mathrm{irp}}} \langle\langle\emptyset\rangle\rangle^{\geq 1}\gamma$.

Based on that, we present the reduction. In order to model-check φ, we: (i) transform φ so that it contains only universal path quantifiers $\forall$, (ii) replace each $\forall$ with $\langle\langle\emptyset\rangle\rangle^{\geq 1}$, and (iii) verify the resulting **PATL**$_{\mathrm{irp}}$ formula, which completes the reduction. □

[2] Concurrent Game Structure [12].

Theorem 5. *Model checking* $\mathbf{PTATL}_{\mathrm{irp}}$ *is in* **EXPTIME** *w.r.t. the size of the model and the length of the formula.*

Proof. For the upper bound, we do a variant of *global model checking*, computing the set of states and clock valuations in which the input formula is satisfied. Note that this set is usually infinite, but it can be represented by the corresponding finite set of (state,region) pairs from the extended model [1, 43]. For the nested formulas, we proceed recursively (bottom-up). The resulting algorithm is as follows.

- Construct the region graph for M, φ, and let $Reg = reg(M, \varphi)$;
- Let $\hat{M}$ be the extended model of M for φ with $\hat{PV} = \emptyset$;
- **while** φ is not atomic **do**
  - Take any flat non-atomic state subformula ψ of φ;
  - **if** $\psi \equiv \langle\langle A\rangle\rangle^{\bowtie z}\gamma$ **then** use the procedure of Proposition 3 to compute $Sat(\psi) = \{(s, r) \in S \times Reg \mid \forall_{v\in r}\ M, (s, v) \models \psi\}$;
  - **else** compute $Sat(\psi)$ in the standard way (ψ must be a boolean combination of atomic propositions);
  - Add a fresh atomic proposition p_ψ to $\hat{PV}$ with $\hat{V}(\mathrm{p}_\psi) = Sat(\psi)$;
  - Replace every occurrence of ψ in φ by p_ψ;
- **if** $(s, reg(v)) \in \hat{V}(\mathrm{p}_\varphi)$ **then** return *true* else return *false*.

It halts in $O((|T|+|\mathcal{I}|)^{|\mathcal{X}|+Int(\varphi)})+|\varphi|\cdot O((|T|+|\mathcal{I}|)^{|\mathcal{X}|+Int(\varphi)})\cdot O((|T|+|\mathcal{I}|)^{|S|+|\mathcal{X}|+Int(\varphi)}) = 2^{O((|S|+|\mathcal{X}|+Int(\varphi))\log(|T|+|\mathcal{I}|))}$ steps, and thus in deterministic exponential time, by the complexity of region graphs and their construction [1, 43], and from Prop. 3. □

3.3 PTATL with Probabilistic Strategies

Now, we consider the general case where coalitions can use randomized strategies, similarly to [12].

Theorem 6. *Model checking* $\mathbf{PTATL}_{\mathrm{irP}}$ *is* **PSPACE***-hard.*

Proof. We proceed by a reduction of **TCTL** model checking, which is **PSPACE**-complete [1]. Similarly to Theorem 4, we have $M, (s, v) \models_{\mathrm{TCTL}} \forall\gamma$ **iff** $M, (s, v) \models_{\mathbf{PTATL}_{\mathrm{irP}}} \langle\langle\emptyset\rangle\rangle^{\geq 1}\gamma$, and the same reduction can be used to complete the proof. □

Proposition 7. *Let* φ *be a* $\mathbf{PTATL}_{\mathrm{irP}}$ *formula,* $\psi \equiv \langle\langle A\rangle\rangle^{\bowtie z}\gamma$ *a flat subformula of* ψ*,* M *a probabilistic timed model,* s *a state in* M*, and* $r \in reg(M, \varphi)$ *a region in the detailed region graph for model* M, φ*. Checking whether* $\forall_{v\in r}\ M, (s, v) \models \psi$ *can be done in* $2^{2^{O((|\mathcal{X}|+Int(\varphi))\log(|T|+|\mathcal{I}|))}}$ *steps.*

Proof. Similarly to Proposition 3, we observe that in finite games the opponents always have a deterministic best-response strategy to any given strategy σ_A of the coalition. Thus, we can use the following procedure to check if $\forall_{v\in r}\ M, (s, v) \models \langle\langle A\rangle\rangle^{\bowtie z}\gamma$:

- Transform M, φ to M', φ' by the standard construction that removes each interval I from φ and implements it via an auxiliary clock y, auxiliary action a_y resetting y, and the proposition $p_{y\in I}$ in M' (see the construction in [43], p. 172). Note that φ' is (syntactically) a formula of **PATL**;
- Construct M'' as the detailed region graph for M. Note also that M'' can be seen as a finite stochastic iCGS[3];
- Use the procedure of [12, Proposition 1, Proposition 2 & Proposition 3] to check if $M'', r \models_{\mathbf{PATL}_{\mathrm{irP}}} \varphi'$, and return the outcome.

[3]Concurrent Game Structure with imperfect information [12].

Note: construction of the region graph takes $O((|T|+|\mathcal{I}|)^{|\mathcal{X}|+Int(\varphi)})$ steps, and the resulting graph has $|S_{M''}| = O((|T|+|\mathcal{I}|)^{|\mathcal{X}|+Int(\varphi)})$ states [1, 43]; model checking of flat $\mathbf{PATL}_{\mathrm{irP}}$ formulas is in time $2^{O(|\mathcal{A}|\cdot|Act|\cdot|S_{M''}|\cdot\log(|\mathcal{A}|\cdot|Act|\cdot|S_{M''}|)}$ [12], and therefore also in $2^{2^{O((|\mathcal{X}|+Int(\varphi))\log(|T|+|\mathcal{I}|))}}$. □

Theorem 8. *Model checking* $\mathbf{PTATL}_{\mathrm{irP}}$ *is in* **2EXPTIME** *w.r.t. to the size of the model and the length of the formula.*

Proof. We use an analogous algorithm to that of Theorem 5, which runs in deterministic time of $|\varphi|\cdot 2^{2^{O((|\mathcal{X}|+Int(\varphi))\log(|T|+|\mathcal{I}|))}}$. □

The following is a straightforward corollary.

Theorem 9. *If the total number of clocks in the model and the intervals in the formula is bounded, then model checking* $\mathbf{PTATL}_{\mathrm{irP}}$ *is in* **EXPTIME** *w.r.t. to the model size and the formula length.*

Finally, we note undecidability of the *satisfiability* problem of **PTATL**, which is also a straightforward corollary of prior results.

Theorem 10. *The satisfiability problem is undecidable for both* $\mathbf{PTATL}_{\mathrm{irp}}$ *and* $\mathbf{PTATL}_{\mathrm{irP}}$.

Proof. Since the problem of **TCTL** satisfiability is undecidable [1], and **TCTL** is subsumed by **TATL** (cf. Proposition 1), clearly the problem is also undecidable for extensions of the latter: $\mathbf{PTATL}_{\mathrm{irp}}$ and its more general case $\mathbf{PTATL}_{\mathrm{irP}}$. □

3.4 Practical Model Checking of PTATL

The problem is to determine, given a PCAMAS $\mathcal{S}$, and a $\mathbf{PTATL}_{\mathrm{irp}}$ formula $\varphi = \langle\langle A\rangle\rangle^{\bowtie z}\gamma$, whether the formula is satisfied in the initial state $\iota = (s, v)$ of model M of $\mathcal{S}$.

While several tools can be used for model checking subsets of $\mathbf{PTATL}_{\mathrm{irp}}$, none of them supports the full language of the logic. IMITATOR [6] allows for real-time and parametric verification, and its parametric verification engine can be leveraged to encode agents' irp-strategies, see [8]. PRISM [34] is a state-of-the-art probabilistic model checker, but does not explicitly support the **ATL** coalition operator for probabilistic TA or imperfect information strategies of agents. Therefore, to demonstrate the practical feasibility of $\mathbf{PTATL}_{\mathrm{irp}}$ model checking, we combine these two verifiers.

We proceed as summarised in Algorithm 1. The network of non-probabilistic, *parametric* TA $\mathcal{S}'$ is obtained from $\mathcal{S}$ by replacing each transition $T_i(l_i, a, \mathtt{cc}, X) = d$ with several non-probabilistic copies $T_i(l_i, a, \mathtt{cc}, X) = l_i'$, one for each l_i' such that $d(l_i') > 0$. Moreover, for each coalition agent $i \in A$, we add one *parameter* per local state, whose value will be tested in the guards of outgoing transitions from that location (e.g. compared to 1 for the first transition, 2 for the second, and so on, see Fig. 3). Multiple copies originating from the same probabilistic transition in $\mathcal{S}$ are compared to the same value as they correspond to the same choice in an irp-strategy.

Note that we supply a non-strategic formula $\varphi' = \gamma$ to IMITATOR, which does not explicitly support the modality. Instead, possible strategies are encoded using parameters in the tool's input parametric TA $\mathcal{S}'$. Because IMITATOR may synthesise multiple candidate strategies for the (non-probabilistic) model of $\mathcal{S}'$, we need to prepare as many corresponding variants $\mathcal{S}_{var}$ to be checked with PRISM. However, it is enough that one of them produces a positive result for the original formula φ to be satisfied in $\mathcal{S}$.

Algorithm 1 MCHECKPTATL$_{\text{irp}}(M, (s,v), \langle\langle A\rangle\rangle^{\bowtie z}\gamma)$

- Obtain $\mathcal{S}'$ by replacing each probabilistic local transition in $\mathcal{S}$ with several non-probabilistic copies.
- Encode strategies in $\mathcal{S}$ by adding parameters checked in guards of all transitions of agents in the coalition A, as defined in [8].
- Run IMITATOR for the network of parametric TA $\mathcal{S}'$ and $\varphi' = \gamma$.
- If $\mathcal{S}' \not\models \varphi'$, then **return** $\mathcal{S} \not\models \varphi$ (since there is no strategy for coalition A regardless of the probabilities on transitions).
- Otherwise, if $\mathcal{S}' \models \varphi'$,
  for each set of param. constraints *var* synthesised by IMITATOR:
  - Obtain $\mathcal{S}_{var}$ by pruning all transitions of coalition agents in $\mathcal{S}$ that are not consistent with the strategy corresponding to *var*.
  - Run PRISM for the network of probabilistic TA $\mathcal{S}_{var}$ and formula $\varphi'' = \mathsf{P}_{\mathsf{max=?}}[\gamma]$, yielding the value $P_{var} \in [0,1]$.
  - If $P_{var} \bowtie z$, then **return** $\mathcal{S} \models \varphi$ (since the strategy was already fixed by pruning all other transitions).
- If there is no *var* such that $P_{var} \bowtie z$, then **return** $\mathcal{S} \not\models \varphi$.

Note also that the PRISM input property $\varphi'' = \mathsf{P}_{\mathsf{max=?}}[\gamma]$ is a *quantitative* query yielding the maximum probability for **PTCTL** formula γ, see [42]. It corresponds to the original **PTATL** formula φ, since **TATL** subsumes **TCTL** (cf. Prop. 1), and the coalition's strategy is already fixed in $\mathcal{S}_{var}$.

Theorem 11. *Algorithm 1 is sound and complete.*

Proof. Let $\varphi = \langle\langle A\rangle\rangle^{\bowtie z}\gamma$. Note that any irp-strategy for the coalition A that allows one to enforce γ in the input PCAMAS model M of $\mathcal{S}$ will also be synthesised by IMITATOR in the non-probabilistic model M' of $\mathcal{S}'$, since the latter overapproximates M by removing probabilities on transitions. Therefore, no strategies are missed by the algorithm.

The obtained set of "candidate" strategies is then exhaustively checked by PRISM, guaranteeing that it returns $M, (s,v) \models \varphi$ iff a strategy exists that enforces γ with probability in the relation $\bowtie$ with the constant z. □

4 EXPERIMENTAL RESULTS

To demonstrate the practical feasibility of **PTATL** model checking, and to assess the scalability of our approach, we used the voting scenario from Example 1 as a benchmark, scaling it with both the number of voters n and the number of candidates c. Propositions $\mathsf{v}_{\mathsf{i,j}}$ are now indexed with $i \in \{1,\dots,n\}$ and $j \in \{1,\dots,c\}$. The input formula is $\varphi = \langle\langle V_1\rangle\rangle^{\geq 0.8}\mathsf{F}_{[0,8]}\mathsf{v}_{1,1}$, which states that the first voter has a strategy to successfully vote for the first candidate within 8 time units, with a probability of at least 0.8. Note that we cannot currently synthesise *probabilistic* strategies, because our approach relies on a parametric engine, and neither tool supports probabilities to the extent needed. More precisely, PRISM's probabilistic TA engine does not support parametric expressions in guards, only in transition probabilities. On the other hand, IMITATOR has robust functionality for verification with time parameters, but not with probabilities. Therefore, in these experimental results we are only considering deterministic strategies of agent V_1.

4.1 Generating Test Instances

Our prototype implementation is a tool chain supported by a generator to automate the process, which proceeds as follows. First, a non-probabilistic IMITATOR model is prepared, containing duplicates of transitions wherever probabilistic execution can occur in the PCAMAS of Figs. 1 and 2. Moreover, it includes *parameters* added in each local state of agent V_1, see Fig. 3. Checked in guards of outgoing transitions, they ensure only one transition can be chosen from each local state (thus encoding irp-strategies).

```
loc v10: invariant True
    when regm1=1 & regi1=0 & regp1=0 sync regm1 goto m1; (* postal vote *)
    when regm1=0 & regi1=1 & regp1=0 sync regi1 goto i1; (* internet vote *)
    when regm1=0 & regi1=0 & regp1=1 sync regp1 goto p1; (* polling station *)
loc m1: invariant True
    when packm1=1 sync packm1 goto rm1; (* receive package for postal vote *)
loc i1: invariant True
    when packi1=1 sync packi1 goto ri1; (* receive credentials for e-voting *)
loc p1: invariant True
    when packp1=1 sync packp1 goto rp1; (* receive address of polling station *)
loc rm1: invariant True
    when vm11=1 & vm12=0 sync vm11 goto voted11; (* two copies of each trans. *)
    when vm11=1 & vm12=0 sync vm11 goto coerced1;
    when vm11=0 & vm12=1 sync vm12 goto voted12;
    when vm11=0 & vm12=1 sync vm12 goto coerced1;
loc ri1: invariant True
    when vi11=1 & vi12=0 sync vi11 goto voted11; (* two copies of each trans. *)
    when vi11=1 & vi12=0 sync vi11 goto coerced1;
    when vi11=0 & vi12=1 sync vi12 goto voted12;
    when vi11=0 & vi12=1 sync vi12 goto coerced1;
loc rp1: invariant True
    when vp11=1 & vp12=0 sync vp11 goto voted11; (* two copies of each trans. *)
    when vp11=1 & vp12=0 sync vp11 goto coerced1;
    when vp11=0 & vp12=1 sync vp12 goto voted12;
    when vp11=0 & vp12=1 sync vp12 goto coerced1;
```

Figure 3: Parameters checked in guards of the coalition agent V_1's outgoing transitions in the IMITATOR model (excerpt).

On this parametric model, the (non-probabilistic) formula $\varphi' = \langle\langle V_1\rangle\rangle\mathsf{F}_{[0.8]}\mathsf{v}_{1,1}$ is verified. IMITATOR's command line arguments *-merge=none* and *-comparison=equality* allow one to skip comparing time zones (which are disjoint in our case and so cannot be merged anyway). The IMITATOR output lists all valuations of parameters such that φ' is satisfied. In other words, we obtain a superset of all possible irp-strategies of agent V_1 that ensure the timing constraint, but not necessarily the probabilistic requirement.

```
// constants synthesised by IMITATOR (fixing strategy):
const int regp1 = 0; const int regi1 = 1; const int regm1 = 0;
[...]
module V1 [...]
  [reg1m] (locV1=0) & (regm1=1) -> (locV1'=1);
  [reg1i] (locV1=0) & (regi1=1) -> (locV1'=2);
  [reg1p] (locV1=0) & (regp1=1) -> (locV1'=3);
```

Figure 4: Fixing a strategy in the PRISM model (excerpt) by defining constants synthesised by IMITATOR.

To proceed with probabilistic verification, a set of PRISM probabilistic TA models is then generated, each corresponding to one strategy synthesised by IMITATOR, and with all choices of the coalition agent V_1 *fixed* according to that strategy. Technically, this is

done by taking the IMITATOR output, i.e. sets of constraints where parameters are valued either 0 or 1, and attaching them as definitions of *constants* at the beginning of each PRISM model. These constants are checked in the guards of V_1's transitions, preventing the execution of those not consistent with the currently evaluated candidate strategy, see Fig. 4.

The property $\mathsf{P}_{\mathsf{max=?}}[\mathsf{F}^{<=8}\mathsf{v}_{1,1}]$ is verified using PRISM, with the command line argument *-ptamethod digital* that specifies its *digital clocks* engine should be used. This reduces model checking of probabilistic TA to MDP model checking for a restricted class of **PCTL** properties. Note that this property corresponds to the original **PTATL** formula φ, since **TATL** subsumes **TCTL** (see Proposition 1), and the coalition's strategy is already fixed in the PRISM model by disabling the execution of any transition not consistent with it. Therefore, the output is the maximum probability of achieving $\mathsf{F}_{[0;8]}\mathsf{v}_{1,1}$ when following the currently fixed strategy. This value is compared against $z = 0.8$ given in the formula φ. If greater than or equal to the latter, the process is stopped and subsequent strategies (fixed in the remaining PRISM models) are not checked, as φ is already known to be satisfied in the input PCAMAS.

4.2 Results and Discussion

All experiments were performed on a Windows 10 PC with a 4.0 Ghz Intel Core i9-9900KS CPU (8 physical cores, 16 threads) and 64 GB RAM, running the IMITATOR and PRISM binaries via Windows Subsystem for Linux (WSL). The results reported in Table 1 include, in addition to model checking time with both verifiers, also the generation time of all models (which was always negligible compared to the former). As per the default WSL configuration, the running process was terminated if its memory usage exceeded half of the total available, indicated by *memout* in the table.

	c = 1	c = 2	c = 3	c = 4	c = 5	c = 6
n = 1	1.2	1.2	1.2	1.3	1.5	1.5
n = 2	2.6	2.9	4.0	6.4	9.5	15.4
n = 3	4.5	16.5	45.3	125.9	302.8	610.5
n = 4	40.3	181.2	854.2	3083.4	8107.4	memout
n = 5	344.6	3647.2	memout			
n = 6	7177.0	memout				

Table 1: Results for the voting benchmark with n voters and c candidates. All times in seconds.

As expected, the results scale much worse with the number n of voters vs. that of candidates c. Each additional voter introduces another component of the PCAMAS that needs to synchronise with the election authority, significantly increasing the number of global states and transitions in the model (exacerbated by additional interleavings due to asynchronous execution). On the other hand, adding more candidates only requires a modest number of new transitions in existing modules, plus corresponding parameters and constants (in the IMITATOR and PRISM models, respectively).

In all instances, the input formula φ is satisfied in the model for V_1's strategy σ_{V1} that registers for a vote by mail (i.e. selects reg_m in the initial local state) and then selects $vote1_m$ when choosing the candidate.[4] Note that another strategy permitting a vote for candidate 1 within the interval $[0; 8]$ is σ'_{V_1}, in which registration for e-voting (action reg_i) is chosen in the initial state instead, followed by $vote1_i$. As such, IMITATOR synthesises two sets of constraints, corresponding to strategies σ_{V1} and σ'_{V_1}, respectively. However, the latter only gives a maximum probability of 0.6 and thus is discarded upon verifying the PRISM probabilistic TA model with fixed σ'_{V_1}.

We note that the timing and probability constraints in the input PCAMAS and formula can significantly affect the efficiency of our approach. In particular, if the *timing* constraints are strict (in the sense they can only be met in very few executions of the model), then only a small number of candidate strategies will be synthesised by IMITATOR, significantly speeding up the process of verification as fewer calls to PRISM will be needed. On the other hand, the opposite is true for *probability* constraints: the more often $P_{var} \bowtie z$ is satisfied, the fewer times PRISM needs to run as it is sufficient to find just one candidate strategy that meets the constraint. Note also that by not halting the computation at that point, *all* possible strategies can be easily synthesised using our method, if needed.

5 CONCLUSIONS AND FUTURE WORK

In this paper, we established theoretical complexity results for the model checking of probabilistic, continuous-time **ATL** (**PTATL**). Moreover, by combining two state-of-the-art tools IMITATOR and PRISM, we demonstrated for the first time the practical feasibility of verifying specifications that feature probability, real time, and strategic ability of asynchronous agents with imperfect information.

It is important to note the semantic side effects that arise in strategic reasoning, discussed extensively in [29], are applicable also to PCAMAS. While it should be relatively straightforward to account for these issues, we leave this adaptation for future work due to space constraints and instead focus here on putting forward novel theoretical and practical results. Another important direction is to investigate the feasibility of practical model checking of $\mathbf{PTATL}_{\mathrm{irP}}$, i.e. with probabilistic strategies, which are lacking at present. Finally, since $\mathbf{STCTL}_{\mathrm{ir}}$ is more expressive than $\mathbf{TATL}_{\mathrm{ir}}$ while retaining the same model checking complexity, it would be interesting to study an analogous extension of the former logic, i.e. **PSTCTL**, and in particular to determine whether it enjoys the same advantage when probabilistic models and strategies are considered.

ACKNOWLEDGMENTS

This work was supported by: CNRS IRP "Le Trójkąt", NCBR Poland & FNR Luxembourg under the PolLux/FNR-CORE project SpaceVote (POLLUX-XI/14/SpaceVote/2023 and C22/IS/17232062/SpaceVote), the ANR-22-CE48-0012 project BISOUS, the PHC Polonium project MoCcA (BPN/BFR/2023/1/00045). Marta Kwiatkowska contributed while on sabbatical and acknowledges funding from the ERC under the EU's Horizon 2020 research and innovation programme (FUN2MODEL, grant agreement No. 834115). For the purpose of open access, and in fulfilment of the obligations arising from the grant agreement, the authors have applied CC BY 4.0 license to any Author Accepted Manuscript version arising from this submission.

[4] Formally, an irp-strategy assigns an action to *each* local state of an agent, as per Def. 9. For clarity, we omit here V_1's choices in states that will never be visited in the strategy's outcome paths (or only have a single successor).

REFERENCES

[1] Rajeev Alur, Costas Courcoubetis, and David L. Dill. 1993. Model-Checking in Dense Real-Time. *Inf. Comput.* 104, 1 (1993), 2–34.

[2] Rajeev Alur and David L. Dill. 1990. Automata for Modeling Real-Time Systems. In *Proceedings of ICALP'90 (Lecture Notes in Computer Science, Vol. 443)*. Springer, 322–335.

[3] Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. 1997. Alternating-Time Temporal Logic. In *Proceedings of the 38th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE Computer Society Press, 100–109.

[4] Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. 2002. Alternating-Time Temporal Logic. *J. ACM* 49 (2002), 672–713.

[5] Benjamin Aminof, Marta Kwiatkowska, Bastien Maubert, Aniello Murano, and Sasha Rubin. 2019. Probabilistic Strategy Logic. In *Proceedings of IJCAI 2019*. ijcai.org, 32–38.

[6] Étienne André. 2021. IMITATOR 3: Synthesis of Timing Parameters Beyond Decidability. In *Proceedings of CAV 2021 (Lecture Notes in Computer Science, Vol. 12759)*. Springer, 552–565.

[7] Jaime Arias, Carlos E. Budde, Wojciech Penczek, Laure Petrucci, Teofil Sidoruk, and Mariëlle Stoelinga. 2020. Hackers vs. Security: Attack-Defence Trees as Asynchronous Multi-agent Systems. In *Proceedings of ICFEM 2020*. Springer, 3–19.

[8] Jaime Arias, Wojciech Jamroga, Wojciech Penczek, Laure Petrucci, and Teofil Sidoruk. 2023. Strategic (Timed) Computation Tree Logic. In *Proceedings of AAMAS'23*. ACM, 382–390.

[9] Christel Baier and Joost-Pieter Katoen. 2008. *Principles of Model Checking*. MIT Press.

[10] Francesco Belardinelli, Wojciech Jamroga, Damian Kurpiewski, Vadim Malvone, and Aniello Murano. 2019. Strategy Logic with Simple Goals: Tractable Reasoning about Strategies. In *Proceedings of IJCAI'19*. ijcai.org, 88–94.

[11] Francesco Belardinelli, Wojciech Jamroga, Munyque Mittelmann, and Aniello Murano. 2023. Strategic Abilities of Forgetful Agents in Stochastic Environments. In *Proceedings KR 2023*. 726–731.

[12] Francesco Belardinelli, Wojciech Jamroga, Munyque Mittelmann, and Aniello Murano. 2024. Verification of Stochastic Multi-Agent Systems with Forgetful Strategies. In *Proceedings of AAMAS 2024*. ACM, 160–169.

[13] Raphaël Berthon, Nathanaël Fijalkow, Emmanuel Filiot, Shibashis Guha, Bastien Maubert, Aniello Murano, Laureline Pinault, Sophie Pinchinat, Sasha Rubin, and Olivier Serre. 2020. Alternating Tree Automata with Qualitative Semantics. *ACM Trans. Comput. Logic* 22, 1, Article 7 (2020), 24 pages.

[14] Thomas Brihaye, Arnaud Da Costa Lopes, François Laroussinie, and Nicolas Markey. 2009. ATL with Strategy Contexts and Bounded Memory. In *Proceedings of LFCS (Lecture Notes in Computer Science, Vol. 5407)*. Springer, 92–106.

[15] Krishnendu Chatterjee, Thomas A. Henzinger, and Nir Piterman. 2007. Strategy Logic. In *Proceedings of CONCUR 2007*. 59–73.

[16] Taolue Chen, Vojtech Forejt, Marta Kwiatkowska, David Parker, and Aistis Simaitis. 2013. Automatic Verification of Competitive Stochastic Systems. *Formal Methods in System Design* 43, 1 (2013), 61–92.

[17] Taolue Chen and Jian Lu. 2007. Probabilistic Alternating-time Temporal Logic and Model Checking Algorithm. In *Proceedings of FSKD 2007*. 35–39.

[18] Edmund M. Clarke, Orna Grumberg, Daniel Kroening, Doron A. Peled, and Helmut Veith. 2018. *Model Checking, 2nd Edition*. MIT Press.

[19] Ronald Fagin, Joseph Y. Halpern, Yoram Moses, and Moshe Y. Vardi. 1995. *Reasoning about Knowledge*. MIT Press.

[20] Dimitar P. Guelev and Catalin Dima. 2012. Epistemic ATL with Perfect Recall, Past and Strategy Contexts. In *Proceedings of CLIMA-XIII (Lecture Notes in Computer Science, Vol. 7486)*. Springer, 77–93. https://doi.org/10.1007/978-3-642-32897-8_7

[21] Hans Hansson and Bengt Jonsson. 1994. A Logic for Reasoning about Time and Reliability. *Formal Aspects of Computing* 6, 5 (1994), 512–535.

[22] Karel Horák and Branislav Bošanskỳ. 2019. Solving Partially Observable Stochastic Games with Public Observations. In *Proceedings of AAAI'19*, Vol. 33. AAAI Press, 2029–2036.

[23] Karel Horák, Branislav Bošanskỳ, Vojtěch Kovařík, and Christopher Kiekintveld. 2023. Solving Zero-sum One-sided Partially Observable Stochastic Games. *Artificial Intelligence* 316 (2023), 103838.

[24] Xiaowei Huang, Kaile Su, and Chenyi Zhang. 2012. Probabilistic Alternating-Time Temporal Logic of Incomplete Information and Synchronous Perfect Recall. In *Proceedings of AAAI'12*. 765–771.

[25] Xiaowei Huang and Ron van der Meyden. 2014. Symbolic Model Checking Epistemic Strategy Logic. In *Proceedings of AAAI'14*. 1426–1432.

[26] Wojciech Jamroga and Nils Bulling. 2011. Comparing Variants of Strategic Ability. In *Proc. of IJCAI 2011*. IJCAI/AAAI, 252–257.

[27] Wojciech Jamroga, Beata Konikowska, and Wojciech Penczek. 2016. Multi-Valued Verification of Strategic Ability. In *Proceedings of AAMAS 2016*. 1180–1189.

[28] Wojtek Jamroga, Lukasz Masko, Lukasz Mikulski, Witold Pazderski, Wojciech Penczek, Teofil Sidoruk, and Damian Kurpiewski. 2022. Verification of Multi-Agent Properties in Electronic Voting: A Case Study. In *Proceedings of AiML 2022*. College Publications, 531–556.

[29] Wojciech Jamroga, Wojciech Penczek, and Teofil Sidoruk. 2021. Strategic Abilities of Asynchronous Agents: Semantic Side Effects and How to Tame Them. In *Proceedings of KR 2021*. 368–378.

[30] Wojciech Jamroga, Wojciech Penczek, Teofil Sidoruk, Piotr Dembinski, and Antoni W. Mazurkiewicz. 2020. Towards Partial Order Reductions for Strategic Ability. *JAIR* 68 (2020), 817–850.

[31] Magdalena Kacprzak and Wojciech Penczek. 2004. Unbounded Model Checking for Alternating-Time Temporal Logic. In *Proceedings of AAMAS 2004*. IEEE Computer Society, 646–653.

[32] Magdalena Kacprzak and Wojciech Penczek. 2005. Fully Symbolic Unbounded Model Checking for Alternating-time Temporal Logic. *Autonomous Agents and Multi-Agent Systems* 11, 1 (2005), 69–89.

[33] John G. Kemeny, J. Laurie Snell, and Anthony W. Knapp. 1976. *Denumerable Markov Chains*. Springer, Chapter Stochastic Processes, 40–57.

[34] Marta Kwiatkowska, Gethin Norman, and David Parker. 2011. PRISM 4.0: Verification of Probabilistic Real-time Systems. In *Proc. 23rd International Conference on Computer Aided Verification (CAV'11) (LNCS, Vol. 6806)*. Springer, 585–591.

[35] Marta Kwiatkowska, Gethin Norman, and David Parker. 2019. Verification and Control of Turn-Based Probabilistic Real-Time Games. *Lecture Notes in Computer Science* 11760, 379–396.

[36] Marta Kwiatkowska, Gethin Norman, David Parker, and Gabriel Santos. 2019. Equilibria-Based Probabilistic Model Checking for Concurrent Stochastic Games. In *Proceedings of FM 2019 (Lecture Notes in Computer Science, Vol. 11800)*. Springer, 298–315.

[37] Marta Kwiatkowska, Gethin Norman, David Parker, and Gabriel Santos. 2021. Automatic Verification of Concurrent Stochastic Systems. *Formal Methods in System Design* 58, 1 (2021), 188–250.

[38] François Laroussinie and Nicolas Markey. 2015. Augmenting ATL with Strategy Contexts. *Information and Computation* 245 (2015), 98–123.

[39] François Laroussinie, Nicolas Markey, and Ghassan Oreiby. 2006. Model-Checking Timed ATL for Durational Concurrent Game Structures. In *Proceedings of FORMATS'06 (Lecture Notes in Computer Science, Vol. 4202)*. Springer, 245–259.

[40] Alessio Lomuscio, Wojciech Penczek, and Hongyang Qu. 2010. Partial Order Reductions for Model Checking Temporal-Epistemic Logics over Interleaved Multi-Agent Systems. *Fundam. Informaticae* 101, 1-2 (2010), 71–90.

[41] A. Lomuscio, H. Qu, and F. Raimondi. 2015. MCMAS: An Open-Source Model Checker for the Verification of Multi-Agent Systems. *International Journal on Software Tools for Technology Transfer* 24 (2015), 84–90. Available online.

[42] Gethin Norman, David Parker, and Jeremy Sproston. 2013. Model Checking for Probabilistic Timed Automata. *Formal Methods in System Design* 43, 2 (2013), 164–190.

[43] Wojciech Penczek and Agata Pólrola. 2006. *Advances in Verification of Time Petri Nets and Timed Automata: A Temporal Logic Approach*. Studies in Computational Intelligence, Vol. 20. Springer.

[44] Lutz Priese. 1983. Automata and Concurrency. *Theoretical Computer Science* 25, 3 (1983), 221 – 265.

[45] Pierre-Yves Schobbens. 2004. Alternating-Time Logic with Imperfect Recall. In *Proceedings of LCMAS'03*, Vol. 85(2). 1–12.

[46] Moshe Y. Vardi. 1985. Automatic Verification of Probabilistic Concurrent Finite-State Programs. In *Proceedings of FOCS'85*. IEEE Computer Society, 327–338.

[47] Rui Yan, Gabriel Santos, Gethin Norman, David Parker, and Marta Kwiatkowska. 2024. Partially Observable Stochastic Games with Neural Perception Mechanisms. In *Proceedings of FM'24*. Springer, 363–380.