Alternating-time Temporal Logic with Explicit Programs

Andreas Herzig[1], Emiliano Lorini[1], Faustine Maffre[1], Dirk Walther[2]

[1]University of Toulouse, IRIT, 118, route de Narbonne, F-31062 Toulouse Cedex 9, France
{Andreas.Herzig, Emiliano.Lorini, Faustine.Maffre}@irit.fr
[2]TU Dresden, Theoretical Computer Science, 01062 Dresden, Germany
dirk.walther@tu-dresden.de

Abstract. We introduce ATLEP, an extension of Alternating-time Temporal Logic (ATL), following the recent Alternating-time Temporal Logic with Explicit Actions (ATLEA). ATLEP adds to the language of ATL explicit programs whose agents are committed to play, formulated in an extension of Propositional Dynamic Logic (PDL) where strategies can be captured. We add combinations of commitments, inspired from the work on the Game Description Language. We also give an example on how a multi-agent game can be fully modeled with ATLEP, along with several winning strategies.

1 Introduction

ATL [1] is a logic for reasoning about strategic abilities of agents. Thanks to its language, one can express that a group of agents can or cannot ensure a property by playing some actions. ATL extends Computational-tree Temporal Logic (CTL) where each moment in time, or state, leads to several futures and formulas are expressed with existential and universal quantifications on paths (sequences of states). ATL replaces these quantifications by a *path quantifier* that is parametrized by a *coalition* (a group) of agents, which means that this coalition can "force the game" on a path ensuring a temporal property whatever the other agents do.

ATL is an interesting framework to formalize games in multi-agent systems but suffers from some weaknesses, which have led to several extensions such as ATEL [9]. One of the most recent is ATLEA [3] which tries to remedy the lack of explicit *actions* in the language. Indeed, while the semantics of ATL are rich (with actions and strategies as first-class citizens), the syntax only allows to express the existence or absence of a strategy, but does not allow to name strategies or the actions underlying them. This is what motivated the development of, first, ATL with Explicit Strategies (ATLES) [8] and then ATL with Explicit Actions (ATLEA).

Our goal here with ATL with Explicit Programs (ATLEP) is to pursue this work by generalizing actions to PDL *programs*. We can also see ATLEP as a

multi-agent version of CTL with path relativisation in the style of [5], which is a program-based extension of CTL. In comparison with that work we are now going a step further by introducing a multi-agent extension. We can also cite as a similar work the logic ADL [7], where the authors use the operators from PDL and replace the programs by groups of agents.

This paper is organized as follows: Section 2 presents the syntax and the semantics of ATLEP. Section 3 adds two new operators used to combine commitments with priorities. Section 4 gives an example of a game and several strategies that can be applied to it.

2 Syntax and Semantics

Like in ATLES and ATLEA, we index the path quantifier by *commitment functions*. Here they are defined inductively from *atomic commitments* ρ_{act} (a mapping from agents to actions), by sequential composition of commitments, non-deterministic choices between commitments, finite and infinite iteration of commitments, and tests. Informally, $\rho_{act}(a) = \omega$ (also written (a, ω) or $a \mapsto \omega$) means that agent a is committed to play action ω at the current state. Then we can compose commitments: the sequential composition $\rho_1; \rho_2$ means that agents must play according to their commitments from ρ_1, then from ρ_2, $\rho_1 \cup \rho_2$ that agents must play according to their commitments from ρ_1 or ρ_2 (non-deterministically), ρ^* that agents must play their commitments from ρ a non-deterministic but finite number of times, including zero, ρ^∞ that agents must play their commitments infinitely, and $\varphi?$ that agents are committed to test the formula φ at the current state. (We assume the last action does not take any time and is not performed by one agent but by all agents.)

The commitment function ρ indexes the ATL path quantifier $\langle\langle A \rangle\rangle$. Then, $\langle\langle A \rangle\rangle_\rho \psi$ reads "while the agents committed by ρ perform actions as specified in ρ, the agents in A have a strategy to ensure the temporal property ψ, no matter what the agents outside A do". Therefore, we reduce the set of available strategies to those compatible with commitments. For example, if we have three agents a, b and c, the formula $\langle\langle \{a, b\} \rangle\rangle_\rho \psi$ with $\rho = \{a \mapsto \omega_1\} \cup \{a \mapsto \omega_2\}; \{a \mapsto \omega_3, c \mapsto \omega_4\}$ holds at state w iff there exists a strategy for the agents a and b where a plays ω_1 or ω_2 at w, then at any next state w', ω_3, such that for every strategy for the agent c where he plays ω_4 at w', the path resulting from the chosen strategies satisfies ψ. If $\rho = \emptyset$, then the operator $\langle\langle A \rangle\rangle_\rho$ is the same as the ATL operator $\langle\langle A \rangle\rangle$.

2.1 Syntax

We fix a set Σ a set of *agents*, Π of *atomic propositions*, and Ω a set of *action names*. These sets are countably infinite and disjoint. The language of ATLEP is defined over the signature $\langle \Sigma, \Pi, \Omega \rangle$.

Definition 1 (ATLEP syntax). *The grammar of state formulas φ, path formulas ψ, action commitments ρ_{act} and commitments functions ρ is defined as follows:*

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid \langle\langle A \rangle\rangle_\rho \psi$$
$$\psi ::= \neg\psi \mid \bigcirc\varphi \mid \varphi \,\mathcal{U}\, \varphi$$
$$\rho ::= \rho_{act} \mid \rho;\rho \mid \rho \cup \rho \mid \rho^* \mid \rho^\infty \mid \varphi?$$

where p ranges over Π, A ranges over 2^Σ, and ρ_{act} ranges over partial functions from Σ to Ω with finite domain.

Note that in the state formula $\langle\langle A \rangle\rangle_\rho \psi$ there is no constraint relating the set A to the set of agents committed. We sometimes omit set parentheses as in $\langle\langle a \rangle\rangle_\rho \psi$, where a is an agent. For state formulas, the operators $\wedge$, $\rightarrow$, $\leftrightarrow$, $\top$ and $\bot$ are defined as usual with $\neg$ and $\vee$. For path formulas, the temporal operators *sometime* and *forever* are defined respectively by $\Diamond\varphi = \top\mathcal{U}\varphi$ and $\Box\varphi = \neg(\top\mathcal{U}\neg\varphi)$. As in PDL, we define the programs $skip = \top?$ and $abort = \bot?$ which respectively mean "do nothing" and "fail".

2.2 Concurrent Game Structure with action Names

Formulas are evaluated on CGSN (*Concurrent Game Structure with action Names*) that are defined just like in ATLEA. They are CGS from ATL that additionally interpret action names as moves.

Definition 2 (CGSN). *Let $S = \{1, ..., n\} \subset \Sigma$ with $n \geq 1$ be a finite set of agents, $P \subset \Pi$ be a finite set of propositions and $O \subset \Omega$ be a finite set of action names. A CGSN $\mathcal{C}$ for the signature $\langle S, P, O\rangle$ is a tuple $\mathcal{C} = \langle W, V, M, Mov, E, \|.\|\rangle$ with:*

- *W a finite, non-empty set of* states*, or worlds;*
- *$V : W \longrightarrow 2^P$ a* valuation function *which associates to every state the set of propositions true at this state;*
- *M a finite, non-empty set of* moves*;*
- *$Mov : W \times S \longrightarrow 2^M \setminus \emptyset$ a function such that for every state $w \in W$ and every agent $a \in S$, $Mov(w, a) \subseteq M$ is the non-empty set of* moves available *for a at w;*
- *$E : W \times M^S \longrightarrow W$ is a* transition function *mapping a world w and a move profile $\overrightarrow{m} = \langle m_1, ..., m_n\rangle$ (containing one move for each agent) to the world $E(w, \overrightarrow{m}) \in W$;*[1]
- *$\|.\| : O \longrightarrow M$ is a* denotation function *mapping action names in O to moves in M, such that a move can interpret several action names.*

[1] Note that the function E is defined over all move profiles, including the non-available ones (according to the Mov function). We choose this to make the definition more compact; we deal with the non-available move profiles later.

An action ω is *available* to agent a in the state w if $\|\omega\| \in Mov(w,a)$. The denotation function $\|.\|$ is a mapping from action names to actions. In the following, we do not distinguish between these two terms and designate both as "actions".

2.3 Strategies and compatibility with commitments

In order to interpret formulas, we need a notion of *compatibility* between strategies and programs. Indeed, if an agent is committed to a program, we should only consider strategies compatible with this commitment as available strategies. In this section, we define several necessary notions to express this compatibility.

Move profiles. We define move profiles wrt. a CGSN. A move profile $\overrightarrow{m}$ is a tuple of actions, one for each agent in S. A *partial* move profile $\overrightarrow{m}_A$ for the coalition A is a tuple of actions, one for each agent from A. The action for the agent a in a move profile $\overrightarrow{m}$ is denoted by $\overrightarrow{m}(a)$; in a partial move profile $\overrightarrow{m}_A$, it is denoted by $\overrightarrow{m}_A(a)$ (for $a \in A$). A move profile $\overrightarrow{m}_S$ is the same as a (total) move profile $\overrightarrow{m}$. The move profile $\overrightarrow{m}_\emptyset$ is an empty tuple $\langle\rangle$. We define the set of (total) move profiles as M^S and the set of partial move profiles for A as M^A. Therefore a (finite or infinite) sequence of move profiles $\overrightarrow{M} = \overrightarrow{m}^1\overrightarrow{m}^2...$ can be considered as a word on the alphabet M^S, and a sequence of partial move profiles $\overrightarrow{M}_A = \overrightarrow{m}^1_A\overrightarrow{m}^2_A...$ can be considered as a word on the alphabet M^A. The empty sequence is denoted by ε. The sequence of move profiles $\overrightarrow{M}_S$ for the set S is an equivalent notation for the sequence $\overrightarrow{M}$. The length of a sequence $\overrightarrow{M}$, denoted by $length(\overrightarrow{M})$ is equal to the number of move profiles composing it.

A partial move profile $\overrightarrow{m}_A$ is *included* in a partial move profile $\overrightarrow{m}_B$, noted $\overrightarrow{m}_A \subseteq \overrightarrow{m}_B$, iff $A \subseteq B$ and $\forall a \in A, \overrightarrow{m}_A(a) = \overrightarrow{m}_B(a)$. The empty tuple $\langle\rangle$ is included in every move profile. Then a sequence $\overrightarrow{M}_A = \overrightarrow{m}^1_A\overrightarrow{m}^2_A...$ is a *prefix* of a sequence $\overrightarrow{M}_B = \overrightarrow{m}^1_B\overrightarrow{m}^2_B...$, noted $\overrightarrow{M}_A \sqsubseteq \overrightarrow{M}_B$, iff $length(\overrightarrow{M}_A) \leq length(\overrightarrow{M}_B)$ and $\forall i < length(\overrightarrow{M}_A) + 1, \overrightarrow{m}^i_A \subseteq \overrightarrow{m}^i_B$.[2]

A move profile is used to determine the successor of a state with the transition function E. The set of available move profiles in the state w is defined as $\mathsf{prof}(w) = \{\overrightarrow{m} : \forall a \in S, \overrightarrow{m}(a) \in Mov(w,a)\}$. In the same way, the set of available partial move profiles for A in the state w is defined as $\mathsf{prof}_A(w) = \{\overrightarrow{m}_A : \forall a \in A, \overrightarrow{m}_A(a) \in Mov(w,a)\}$. The *completion* of a partial move profile $\overrightarrow{m}_A$ is the set of (total) move profiles $\mathsf{compl}(w, \overrightarrow{m}_A) = \{\overrightarrow{m} : \overrightarrow{m} \in \mathsf{prof}(w)$ and $\forall a \in A, \overrightarrow{m}(a) = \overrightarrow{m}_A(a)\}$. Observe that the completion $\mathsf{compl}(w, \overrightarrow{m}_A)$ is available at w iff $\overrightarrow{m}_A$ is available at w. The completion of the move profile $\overrightarrow{m}_\emptyset$ at w is, by definition, equal to $\mathsf{prof}(w)$.

The set of *possible successors* of w is the set of states $E(w, \overrightarrow{m})$ with $\overrightarrow{m} \in \mathsf{prof}(w)$. We define the function $E(w, \overrightarrow{m}_A)$ (with $\overrightarrow{m}_A$ a partial move profile), which gives the set of possible successors of w such that the agents in A play

[2] We write $i < length(\overrightarrow{M}_A) + 1$ to include the case where $\overrightarrow{M}_A$ is infinite.

actions from $\overrightarrow{m}_A$, defined as $E(w, \overrightarrow{m}_A) = \{E(w, \overrightarrow{m}) : \overrightarrow{m} \in \mathsf{compl}(w, \overrightarrow{m}_A)\}$. If at least one action in $\overrightarrow{m}_A$ is not available at w, then $E(w, \overrightarrow{m}_A)$ will be empty. Also, $E(w, \overrightarrow{m}_S)$ will contain at most one world.

We extend the function E to finite sequences of partial profiles as follows:

$$E(w, \overrightarrow{M}_A) = \begin{cases} \{w\} & \text{if } \overrightarrow{M}_A = \varepsilon \\ \{w' : \text{there is a } w'' \in E(w, \overrightarrow{m}^1_A), w' \in E(w'', \overrightarrow{M}'_A)\} & \text{if } \overrightarrow{M}_A = \overrightarrow{m}^1_A \cdot \overrightarrow{M}'_A \end{cases}$$

If an action is not available in one of the move profiles of the sequence $\overrightarrow{M}_A$, then the result will be an empty set. If profiles are total then this function will return at most one state.

Strategies. A strategy for an agent a is a function f_a mapping a world w to an available action $f_a(w) \in Mov(w, a)$; a strategy for a coalition (a set of agents) $A \subseteq S$ is a function F_A mapping each agent $a \in A$ to his strategy $F_A(a)$. The function $F_A(w)$ maps a world to a partial move profile $\overrightarrow{m}_A$ such that for all $a \in A, \overrightarrow{m}_A(a) = F_A(a)(w)$.

The set $\mathsf{out}(w, F_A)$ of *outcomes* of a strategy F_A from the world w is the set of infinite sequences of move profiles corresponding to the actions chosen in F_A by each agent from A. Formally, it is the set of sequences:

$$\begin{aligned} \mathsf{out}(w, F_A) = \ & \{\overrightarrow{m}^1\overrightarrow{m}^2... : \exists w_0 w_1 ... \text{ with } w_0 = w \text{ and} \\ & E(w_i, \overrightarrow{m}^{i+1}) = w_{i+1} \text{ and } \overrightarrow{m}^{i+1} \in \mathsf{compl}(w_i, F_A(w_i))\} \end{aligned}$$

Traces of commitments. The domain of an atomic commitment ρ_{act}, denoted by $\mathsf{dom}(\rho_{act})$, is the set of all agents committed by it.

A *trace* is a (finite or infinite) sequence of (total or partial) move profiles. The set $\mathsf{tr}(w, \rho, A)$ is the set of sequences of move profiles describing actions from ρ for the committed agents from A if they begin their commitment at w. It is inductively defined as follows:[3]

$$\begin{aligned} \mathsf{tr}(w, \rho_{act}, A) &= \{\overrightarrow{m}_A : \forall a \in A, \overrightarrow{m}_A(a) \in Mov(w, a) \text{ and} \\ &\quad \text{if } a \in \mathsf{dom}(\rho_{act}) \text{ then } \overrightarrow{m}_A(a) = \|\rho_{act}(a)\|\} \\ \mathsf{tr}(w, \rho_1; \rho_2, A) &= \{\overrightarrow{M}^1_A \cdot \overrightarrow{M}^2_A : \overrightarrow{M}^1_A \in \mathsf{tr}(w, \rho_1, A), \text{ finite and } \overrightarrow{M}^2_A \in \mathsf{tr}(E(w, \overrightarrow{M}^1_A), \rho_2, A)\} \\ &\quad \cup \{\overrightarrow{M}^1_A : \overrightarrow{M}^1_A \in \mathsf{tr}(w, \rho_1, A), \text{ infinite}\} \\ \mathsf{tr}(w, \rho_1 \cup \rho_2, A) &= \mathsf{tr}(w, \rho_1, A) \cup \mathsf{tr}(w, \rho_2, A) \\ \mathsf{tr}(w, \rho^*, A) &= \{\overrightarrow{M}^1_A \cdot ... \cdot \overrightarrow{M}^n_A : n \in \mathbb{N}, \forall i \leq n, \overrightarrow{M}^i_A \in \mathsf{tr}(E(w, \overrightarrow{M}^1_A \cdot ... \cdot \overrightarrow{M}^{i-1}_A), \rho, A) \\ &\quad \text{and } \forall i \leq n-1, \overrightarrow{M}^i_A \text{ finite}\} \\ \mathsf{tr}(w, \rho^\infty, A) &= \{\overrightarrow{M}^1_A \cdot \overrightarrow{M}^2_A \cdot ... : \forall i, \overrightarrow{M}^i_A \in \mathsf{tr}(E(w, \overrightarrow{M}^1_A \cdot ... \cdot \overrightarrow{M}^{i-1}_A), \rho, A), \text{ finite}\} \\ &\quad \cup \{\overrightarrow{M}_A : \overrightarrow{M}_A \in \mathsf{tr}(w, \rho, A), \text{ infinite}\} \\ \mathsf{tr}(w, \varphi?, A) &= \{\varepsilon\} \end{aligned}$$

[3] Note that we have a world w as a parameter of tr while it does not appear in the definitions. However, this will be needed for combined commitments (Section 3).

Informally: the trace of an atomic commitment ρ_{act} corresponds to the set of move profiles where agents from A choose the action they are committed to (the others can choose any action); a sequential composition corresponds to the concatenation of all possible sequences from the first and the second commitment, if the first is finite (if the first is infinite, we do not concatenate the second); a non-deterministic choice corresponds to the union; the non-deterministic repetition and infinite repetition corresponds to the concatenation of finite sequences (in finite repetition, only the last one can be infinite). The test does not take any time, therefore it is represented by the empty sequence.

Let us consider some special cases. If the commitment function ρ is empty, it is equivalent to the atomic program *skip* and in this case, $\mathsf{tr}(w, skip, A) = \{\varepsilon\}$. If the set A is empty, then for each ρ_{act} composing ρ, $\mathsf{tr}(w, \rho_{act}, A) = \{\overrightarrow{m}_{\emptyset}\} = \{\langle\rangle\}$. Therefore, $\mathsf{tr}(w, \rho, A)$ contains sequences of empty tuples. Finally, if ρ does not contain any agent from A, then for each ρ_{act} composing ρ, $\mathsf{tr}(w, \rho_{act}, A) = M^A$ and therefore, $\mathsf{tr}(w, \rho, A)$ contains sequences of various lengths, but for every move profile of every sequence, there exists other sequences of the same length where all other possible move profiles from M^A are placed at the same position.

Compatibility. We can now define the compatibility of a strategy with a commitment: the strategy F_A is *compatible* with the commitment ρ at world w iff for every $\overrightarrow{m}^1\overrightarrow{m}^2... \in \mathsf{out}(w, F_A)$, there exists a (finite or infinite) sequence $\overrightarrow{m}^1_A\overrightarrow{m}^2_A... \in \mathsf{tr}(w, \rho, A)$ such that $\overrightarrow{m}^1_A\overrightarrow{m}^2_A... \sqsubseteq \overrightarrow{m}^1\overrightarrow{m}^2....$ Intuitively, for every possible outcome of the strategy, we can find a sequence in the trace of the commitment which is a prefix of the outcome.

Paths. An infinite sequence $\lambda = w_0w_1w_2...$ of worlds is called a *computation* if for each position $i \geq 0$, w_{i+1} is a possible successor of w_i. We denote by $\lambda[i]$ the i-th component w_i in λ and by $\lambda[0, i]$ the finite initial sequence $w_0...w_i$ of λ.

The set $\mathsf{paths}(w, \overrightarrow{M}_A)$ with w a world and $\overrightarrow{M}_A$ a sequence of partial move profiles, denotes the set of computations beginning at w and which can be reached if agents in A play the actions described in the move profiles from $\overrightarrow{M}_A$. More formally:

$$\mathsf{paths}(w, \overrightarrow{M}_A) =$$
$$\begin{cases} \{\lambda : \lambda[0] = w\} & \text{if } \overrightarrow{M}_A = \varepsilon \\ \{w_0w_1w_2... : w_0 = w \text{ and } w_1w_2... \in \mathsf{paths}(E(w, \overrightarrow{m}^1_A), \overrightarrow{M}'_A)\} & \text{if } \overrightarrow{M}_A = \overrightarrow{m}^1_A \cdot \overrightarrow{M}'_A \end{cases}$$

We extend this to a set of sequences of move profiles $\mathcal{M}_A$ as $\mathsf{paths}(w, \mathcal{M}_A) = \bigcup_{\overrightarrow{M}_A \in \mathcal{M}_A} \mathsf{paths}(w, \overrightarrow{M}_A)$.

The commitment function ρ is *feasible for a coalition A at a world w* iff $\mathsf{paths}(w, \mathsf{tr}(w, \rho, A)) \neq \emptyset$. More generally, a commitment ρ is *feasible for a coalition A* iff it is feasible for A at every world (for each $w \in W$ we have $\mathsf{paths}(w, \mathsf{tr}(w, \rho, A)) \neq \emptyset$). We also say that a commitment ρ is *deterministic*

for a coalition A iff for every $w \in W$, $\mathsf{paths}(w, \mathsf{tr}(w, \rho, A))$ is either empty or a singleton. If a commitment is deterministic and feasible for A, then it leads to exactly one computation everywhere for this coalition.

2.4 Truth conditions

The satisfaction relation, with $\mathcal{C}$ a CGSN and w a world, is defined as follows:

$$\begin{array}{c}\mathcal{C}, w \models \langle\!\langle A \rangle\!\rangle_\rho \psi \text{ iff there exists a strategy } F_A \text{ compatible with } \rho \text{ at } w \\ \text{such that for every strategy } F_{S\setminus A} \text{ compatible with } \rho \text{ at } w, \\ \text{it holds that } \mathcal{C}, \lambda \vDash \psi \text{ where } \{\lambda\} = \mathsf{paths}(w, \mathsf{out}(w, F_A \cup F_{S\setminus A})).\end{array}$$

That means, $\langle\!\langle A \rangle\!\rangle_\rho \psi$ is true at w iff it is possible for agents from A to find a strategy compatible with their commitments such that for each strategy from their opponents compatible with their commitments, it ensures the path formula. If a commitment forces actions which are not available, then the formula cannot be satisfied. Atomic propositions, Boolean and temporal operators have the same satisfaction relation as in ATL.

We can express the *availability* (and unavailability) of programs with ATLEP formulas. As in ATLEA, if $\langle\!\langle a \rangle\!\rangle_{a \mapsto \omega} \bigcirc \top$ is true at w then the program composed of one action ω is available for agent a at world w (written $\|\omega\| \in Mov(w, a)$). In ATLEP, we have $\mathcal{C}, w \models \langle\!\langle A \rangle\!\rangle_\rho \bigcirc \top$ iff $\mathsf{paths}(w, \mathsf{tr}(w, \rho, A)) \neq \emptyset$ (ρ is feasible for the players from A at the world w). The *unavailability* is therefore expressed by the negation of this formula: $\mathcal{C}, w \models \neg\langle\!\langle A \rangle\!\rangle_\rho \bigcirc \top$ iff $\mathsf{paths}(w, \mathsf{tr}(w, \rho, A)) = \emptyset$.

2.5 Validity

Validity and satisfiability are defined as usual. We now discuss some ATLEP validities. First of all, we note that $\langle\!\langle A \rangle\!\rangle_\emptyset$ is equivalent to the standard ATL operator $\langle\!\langle A \rangle\!\rangle$. Therefore, all validities from ATL are also applicable here if $\rho = \emptyset$.

We extend the domain of an atomic commitment to programs as follows:[4]

- $\mathsf{dom}(\rho_1; \rho_2) = \mathsf{dom}(\rho_1 \cup \rho_2) = \mathsf{dom}(\rho_1) \cup \mathsf{dom}(\rho_2)$;
- $\mathsf{dom}(\rho^*) = \mathsf{dom}(\rho^\infty) = \mathsf{dom}(\rho)$;
- $\mathsf{dom}(\varphi?) = S$.

To express new ATLEP validities, we define the *strength* of a commitment function. A commitment function ρ is stronger than another commitment function ρ', denoted by $\rho \geq \rho'$, iff $\forall \mathcal{C}, \forall w \in W, \forall \overrightarrow{M}_{\mathsf{dom}(\rho)} \in \mathsf{tr}(w, \rho, \mathsf{dom}(\rho)), \exists \overrightarrow{M}_{\mathsf{dom}(\rho')} \in \mathsf{tr}(w, \rho', \mathsf{dom}(\rho'))$ such that $\overrightarrow{M}_{\mathsf{dom}(\rho)} \sqsupseteq \overrightarrow{M}_{\mathsf{dom}(\rho')}$. Clearly, $\geq$ is a partial preorder.

Proposition 1. *The following formulas are* ATLEP *valid.*

[4] One might argue that the domain of a test should be the empty set, however setting it to S is necessary for technical reasons that will become clear in Section 3.

1. $\langle\!\langle A\rangle\!\rangle_\rho \bigcirc \top \quad$ *for* $\mathsf{dom}(\rho) \cap A = \emptyset$;
2. $\neg\langle\!\langle A\rangle\!\rangle_\rho \bigcirc \bot \quad$ *for* $\mathsf{dom}(\rho) \subseteq A$;
3. $(\langle\!\langle A_1\rangle\!\rangle_\rho \bigcirc \varphi_1 \wedge \langle\!\langle A_2\rangle\!\rangle_\rho \bigcirc \varphi_2) \to \langle\!\langle A_1 \cup A_2\rangle\!\rangle_\rho \bigcirc (\varphi_1 \wedge \varphi_2) \quad$ *for* $A_1 \cap A_2 \subseteq \mathsf{dom}(\rho)$;
4. $\langle\!\langle A\rangle\!\rangle_\rho \psi \to \langle\!\langle A\rangle\!\rangle_{\rho'} \psi \quad$ *for* $\rho' \geq \rho$ *and* $(\mathsf{dom}(\rho') \setminus \mathsf{dom}(\rho)) \cap A = \emptyset$;
5. $\langle\!\langle A\rangle\!\rangle_{\rho'} \psi \to \langle\!\langle A\rangle\!\rangle_\rho \psi \quad$ *for* $\rho' \geq \rho$ *and* $\mathsf{dom}(\rho') \setminus \mathsf{dom}(\rho) \subseteq A$;
6. $\langle\!\langle A \cup \{a\}\rangle\!\rangle_\rho \psi \to \langle\!\langle A\rangle\!\rangle_\rho \psi \quad$ *for* $a \in \mathsf{dom}(\rho)$;
7. $\langle\!\langle A\rangle\!\rangle_{\rho_{act};\rho} \Box\varphi \leftrightarrow \varphi \wedge \langle\!\langle A\rangle\!\rangle_{\rho_{act}} \bigcirc \langle\!\langle A\rangle\!\rangle_\rho \Box\varphi$;
8. $\langle\!\langle A\rangle\!\rangle_{\rho_1 \cup \rho_2} \psi \leftrightarrow \langle\!\langle A\rangle\!\rangle_{\rho_1} \psi \vee \langle\!\langle A\rangle\!\rangle_{\rho_2} \psi \quad$ *for* $\mathsf{dom}(\rho_1 \cup \rho_2) \subseteq A$;
9. $\langle\!\langle A\rangle\!\rangle_{\rho_1 \cup \rho_2} \psi \leftrightarrow \langle\!\langle A\rangle\!\rangle_{\rho_1} \psi \wedge \langle\!\langle A\rangle\!\rangle_{\rho_2} \psi \quad$ *for* $\mathsf{dom}(\rho_1 \cup \rho_2) \cap A = \emptyset$;
10. $\langle\!\langle A\rangle\!\rangle_{\rho_1^\infty;\rho_2} \psi \leftrightarrow \langle\!\langle A\rangle\!\rangle_{\rho_1^\infty} \psi$;
11. $\langle\!\langle A\rangle\!\rangle_{\rho_{act}^\infty} \psi \leftrightarrow \langle\!\langle A\rangle\!\rangle_{\rho_{act}} \bigcirc \langle\!\langle A\rangle\!\rangle_{\rho_{act}^\infty} \psi$;
12. $\langle\!\langle A\rangle\!\rangle_{\varphi?;\rho} \psi \leftrightarrow \varphi \wedge \langle\!\langle A\rangle\!\rangle_\rho \psi$.

Items 1 and 2 corresponds to the ($\top$) and ($\bot$) axioms from ATL (and their extensions from ATLEA). Item 3 is the superadditivity axiom from ATL as seen in ATLEA. It says how two coalitions (with actions from their commons members fixed) can join their forces. Item 4 and 5 are respectively about increasing the commitment of opponents (a coalition can achieve the same goal if agents from outside are more committed) and releasing the commitment of proponents (a coalition can achieve the same goal if agents inside it are less committed, that means, we release some constraints). Item 6 is the same as in ATLEA, about committed proponents becoming opponents. Equivalences 7-12 have no ATLEA counterpart. Item 7 gives a equivalence for the sequential compositions, in the case where the sequence begins with an atomic action and the path formula must hold forever. For non-deterministic choice and test, items 8, 9 and 12 give more general formulas. Item 10 is about the infinite commitments and the useleness of a commitment after an infinite commitment, and item 11 is about the decomposition of an infinite repetition of atomic commitments.

3 Combining commitments

We have seen that in our language, commitments are partial functions forcing agents to play specified actions. An interesting idea may be to combine these commitments to produce new, more complicated strategies. To do this, we use the operators defined in [10], where a language based on GDL (Game Description Language) is developed to reason about strategies. The authors combine them with the *prioritized disjunction* $\triangledown$ and the *prioritized conjunction* $\triangle$. Informally, the first gives a choice between strategies: try to apply every strategy, beginning with the highest priority, until one can be applied; the second one combines actions from strategies: try to apply as many strategies as possible at the same time; if there is a conflict, remove the lowest priority, then the second lowest priority, etc. We adapt these operators to our framework and define prioritized disjunction and conjunction on commitments.

Definition 3. *(Prioritized commitments) Prioritized commitments are defined as an extension of commitments:*

$$\rho ::= \rho_{act} \mid \rho;\rho \mid \rho \cup \rho \mid \rho^* \mid \rho^\infty \mid \varphi? \mid \rho\triangledown \ldots \triangledown\rho \mid \rho \triangle \ldots \triangle \rho$$

State formulas, path formulas and action commitments are as before. We refer to that language as the extended language for ATLEP.

With this definition, and unlike the original connectives, we allow the composition of conjunctions and disjunctions, like in $\langle\langle A\rangle\rangle_{(\rho_1 \triangledown \rho_2) \vartriangle \rho_3} \psi$, and the melting with program connectives; for example, $\langle\langle A\rangle\rangle_{(\rho_1 \triangledown \rho_2);\rho_3} \psi$ is syntactically allowed.

We extend the definition of dom in the obvious way:

$$\mathsf{dom}(\rho_1 \triangledown \ ... \ \triangledown \rho_m) = \mathsf{dom}(\rho_1 \ \vartriangle \ ... \ \vartriangle \ \rho_m) = \bigcup_{1 \leq i \leq m} \mathsf{dom}(\rho_i)$$

In the semantics, we define the set $\mathsf{tr}(w, \rho, A)$ of sequences of move profiles resulting if we want the execute the prioritized commitments ρ for committed agents from A, beginning at w, as follows:

$$\mathsf{tr}(w, \rho_1 \triangledown \ ... \ \triangledown \rho_m, A) = \begin{cases} \mathsf{tr}(w, \rho_1, A) & \text{if } m = 1 \text{ or} \\ & \mathsf{paths}(w, \mathsf{tr}(w, \rho_1, A)) \neq \emptyset \\ \mathsf{tr}(w, \rho_2 \triangledown \ ... \ \triangledown \rho_m, A) & \text{otherwise} \end{cases}$$

$$\mathsf{tr}(w, \rho_1 \ \vartriangle \ \ ... \ \ \vartriangle \ \rho_m, A) = \begin{cases} \bigcap_{1 \leq i \leq m} \mathsf{tr}(w, \rho_i, A) & \text{if } m = 1 \text{ or} \\ & \bigcap_{1 \leq i \leq m} \mathsf{paths}(w, \mathsf{tr}(w, \rho_i, A)) \neq \emptyset \\ \mathsf{tr}(w, \rho_1 \triangledown \ ... \ \triangledown \rho_{m-1}, A) & \text{otherwise} \end{cases}$$

This means that, if we have a prioritized disjunction of commitments, we will try each one, beginning from the left, until we found one that is feasible. If we have a prioritized conjunction, we will try to apply them all, then if we have a conflict (two different actions for the same agent at the same state), we will try again without the rightmost one, and so on.

Moreover, we expand the notions of *deterministic* and *feasible* for prioritized commitments:

- $\rho_1 \triangledown \ ... \ \triangledown \rho_m$ is deterministic for A if for each k such that $1 \leq k \leq m$, ρ_k is deterministic for A;
- $\rho_1 \ \vartriangle \ \ ... \ \ \vartriangle \ \rho_m$ is deterministic for A if ρ_1 is deterministic for A;
- $\rho_1 \triangledown \ ... \ \triangledown \rho_m$ is feasible for A if there exists k with $1 \leq k \leq m$ such that ρ_k is feasible for A;
- $\rho_1 \ \vartriangle \ \ ... \ \ \vartriangle \ \rho_m$ is feasible for A if ρ_1 is feasible for A.

Intuitively, a prioritized disjunction is deterministic if each commitment is deterministic. Indeed, if each ρ_i is deterministic, then it will lead to zero or one path. So, we will try to apply ρ_1: if it gives one path, then we choose it and the disjunction is deterministic; if not, we try ρ_2, and so on. If we reach ρ_m, we apply it and since it is deterministic, the prioritized disjunction will be deterministic. A prioritized conjunction will be deterministic if ρ_1 is deterministic because whatever the number of commitments we take into account, we will always intersect the resulting set of paths with the paths for ρ_1, which leads to zero or one paths. On the other side, a prioritized disjunction is feasible if there exists a k for which ρ_k is feasible (if there is more than one, we choose the

leftmost one). A prioritized conjunction is feasible if ρ_1 is since if no other ρ_i is feasible, we will choose ρ_1.

The truth condition stays the same, except that we can now interpret prioritized commitments thanks to the function tr.

With these new elements, we can add some validities to the language.

Proposition 2. *The following formulas are* ATLEP *valid.*

13. $\langle\!\langle A\rangle\!\rangle_{\rho_1}\psi \to \langle\!\langle A\rangle\!\rangle_{\rho_1 \triangledown \rho_2}\psi$*;*
14. $\langle\!\langle A\rangle\!\rangle_{\rho_1}\psi \to \langle\!\langle A\rangle\!\rangle_{\rho_1 \vartriangle \rho_2}\psi$ *for* $A \cap \mathsf{dom}(\rho_2) = \emptyset$*;*
15. $\langle\!\langle A\rangle\!\rangle_{\rho_1 \vartriangle \rho_2}\psi \to \langle\!\langle A\rangle\!\rangle_{\rho_1}\psi$ *for* $\mathsf{dom}(\rho_2) \subseteq A$*;*
16. $\langle\!\langle A\rangle\!\rangle_{\{a_1 \mapsto \omega_1\} \vartriangle \ldots \vartriangle \{a_m \mapsto \omega_m\}}\psi \to \langle\!\langle A\rangle\!\rangle_{\{a_1 \mapsto \omega_1, \ldots, a_m \mapsto \omega_m\}}\psi$
with $\forall i, j$ *such that* $1 \leq i, j \leq m$*, if* $i \neq j$ *then* $a_i \neq a_j$*.*

Item 13 is about the disjunction: if the first commitment of a prioritized disjunction is possible, then it will always be chosen and so, adding commitments will be useless. Item 14 and 15 are about the conjunction: we can try to combine a commitment with new ones and ensure the same result if the new commitments are not about the proponents; on the other side, we can remove commitments as long as they do not release constraints on opponents. Item 16 connects prioritized disjunction of actions with sets of atomic commitments. Note that the implication is invalid if there are i, j with $a_i = a_j$: then that agent is overcommitted and the right hand side of the implication is false, while the left hand side is not necessarily so. Note also that it does not matter here whether the a_i are members of A or not.

4 Reasoning about strategies: *the game of Nim*

We now illustrate ATLEP with the turn-based multiple player game of Nim.

4.1 Modeling the game

We model a version of the *game of Nim* for $n \geq 2$ players. We have a row of $m \geq 1$ objects, for example chips, and at his turn, a player can remove from 1 to k (with $k \geq 1$) chips. The winner is the player removing the last chip (the game cannot end with a tie: one player always wins and the others lose).

The version with 2 players is well-known in game theory because there exists a simple winning strategy (for the first or the second player, depending on the initial number of chips m). We will see it later.

We assume players move in the order of their number, from player 1, to player n, then player 1 again and so on. Let us model this game, which we will call $Nim_{n,m,k}$, by a CGSN $\mathcal{C}_{Nim_{n,m,k}}$, with the signature $\langle S, P, O\rangle$ as follows:

- $S = \{1, ..., n\}$, with $n > 1$;

- $P = \{c_0, ..., c_m, turn_1, ..., turn_n, wins_1, ..., wins_n, multiple\}$, with c_i meaning that there remain i chips in the row, $turn_a$ meaning that it is a's turn, $wins_a$ that player a has won, and *multiple* meaning that the number of chips left is a multiple of $k+1$ (useful for strategies);
- $O = \{take_1, ..., take_k, idle\}$ with $take_i$ the action of taking i chips from the row and *idle* the action of waiting.

We also define $terminal \equiv \bigvee_{a \in S} wins_a$.

A state of this game is determined by the number of chips left and the player which has to move. Therefore we denote states by $w_{a,ncl}$, with a the current player and ncl the number of chips left in the row. Then the initial state is denoted by $w_{1,m}$.

We also introduce the functions $previous(a, n)$ and $next(a, n)$, returning respectively the player acting before a and the player acting after a, formally defined as:

- $previous(a, n) = \begin{cases} a-1 & \text{if } a > 1 \\ n & \text{if } a = 1 \end{cases}$
- $next(a, n) = \begin{cases} a+1 & \text{if } a < n \\ 1 & \text{if } a = n \end{cases}$

Then we define $\mathcal{C}_{Nim_{n,m,k}} = \langle W, V, M, Mov, E, ||.|| \rangle$ as:

- $W = \{w_{a,ncl} : 1 \leq a \leq n, 0 \leq ncl \leq m\}$;[5]
- V (the valuation function) such that:
  - $c_i \in V(w_{a,ncl})$ iff $i = ncl$;
  - $turn_b \in V(w_{a,ncl})$ iff $b = a$;
  - $wins_b \in V(w_{a,ncl})$ iff $b = previous(a, n)$ and $ncl = 0$;
  - $terminal \in V(w_{a,ncl})$ iff $ncl = 0$;
  - $multiple \in V(w_{a,ncl})$ iff $ncl \pmod{k+1} = 0$.
- Mov (the legality function), defined as:
  - $Mov(w_{a,ncl}, b) = \begin{cases} \{idle\} & \text{if } terminal \in V(w_{a,ncl}) \text{ or } b \neq a \\ \{take_i : 1 \leq i \leq \min(ncl, k)\} & \text{otherwise} \end{cases}$
- E (the transition function), defined for the $n+1$ possible move profiles $\overrightarrow{m}$:
  - $E(w_{a,ncl}, \overrightarrow{m}) = \begin{cases} w_{next(a,n),ncl-i} & \text{if } \overrightarrow{m}(a) = take_i \text{ and } \forall a' \neq a,\ \overrightarrow{m}(a') = idle \\ w_{a,ncl} & \text{otherwise} \end{cases}$

4.2 Individual strategies

First, we can find some simple strategies for individuals agents.

The most obvious is to finish the game instantly if this is possible, using the following strategy:

[5] Note that, because each player is forced to pick at least one chip, some worlds are unreachable, like $w_{2,m}$, $w_{3,m}$ or $w_{3,m-1}$, etc.

– $winInst_a$ (win instantly if possible):

$$\rho_{winInst_a} = \bigcup_{1 \leq i \leq k} (\langle\!\langle a \rangle\!\rangle_{\{a \mapsto take_i\}} \bigcirc wins_a)?; \{a \mapsto take_i\}$$

This will be feasible only if the number of chips left is less than k during a's turn, so this strategy is not feasible for a everywhere. However, and despite the non-deterministic choice, $winInst_a$ is deterministic (if there is a way to win, it will always be unique).

Another strategy of a is to try to ensure that a can play once more; if possible, remove a number of chips such that even if all other players remove the maximum number of chips, we can play again. To capture this, we first define the state formula:

$$othersWin_a \equiv \langle\!\langle S \setminus \{a\} \rangle\!\rangle_{\emptyset} \Big(\Big(\bigvee_{a' \in S \setminus \{a\}} wins_{a'} \Big) \, \mathcal{U} \, turn_a \Big)$$

meaning that players other than a have a strategy so that one of them wins before a can play again. Then we define the following strategy:

– $playAgain_a$ (play such that a can play next turn if possible):

$$\rho_{playAgain_a} = \bigcup_{1 \leq i \leq k} (\langle\!\langle a \rangle\!\rangle_{\{a \mapsto take_i\}} \bigcirc \neg othersWin_a)?; \{a \mapsto take_i\}$$

This commitment is neither feasible (it is not feasible if there is less than $k(n-1)+1$ chips left) nor deterministic (we may have several choices if there is more than $k(n-1)+1$ chips left) for a. The commitment $playAgain_a$ is interesting because it takes into account the other players' strategies through the test (in $othersWin_a$). We add another strategy which takes a specified number of chips:

– $remove_{a,i}$ (remove i chips):

$$\rho_{remove_{a,i}} = \{a \mapsto take_i\}$$

We can combine $playAgain_a$ and $remove_{a,i}$ to obtain:

– $playAgainMax_a$ (take as much as possible chips such that a can play next turn):

$$\rho_{playAgainMax_a} = (\rho_{playAgain_a} \vartriangle \rho_{remove_{a,k}}) \, \triangledown ... \triangledown (\rho_{playAgain_a} \vartriangle \rho_{remove_{a,1}})$$

This strategy (take as many as possible chips such that we can play again) is deterministic. We finally add a simple strategy, consisting in removing the minimal number of chips, that means, one:

– $removeMin_a$ (remove 1 chip):

$$\rho_{removeMin_a} = \{a \mapsto take_1\}$$

All these commitments are not always feasible for a, so we finally add a last strategy, consisting in simply doing nothing (which will be used later, at almost every combination, so that the strategy is feasible during another player's turn):

– $idle_a$ (wait for your turn):

$$\rho_{idle_a} = \{a \mapsto idle\}$$

With all these, we can define the prioritized commitment $individual_a$ as:

$$\rho_{individual_a} = \rho_{winInst_a} \triangledown \rho_{playAgainMax_a} \triangledown \rho_{removeMin_a} \triangledown \rho_{idle_a}$$

This commitment is deterministic (because every element of the disjunction is) and feasible for a: if it is a's turn and there is at least one chip, he can always play $take_1$; if not, or if the game has ended, then he can always play $idle$. If one player follows this, he will try to win instantly if this is possible, if not, he will try to remove the maximum number of chips such that he can play at his next turn (to get closer the winning state) if this is possible, if not, to remove only 1 chip, to keep the opponents away from the winning state; if none of these options is available, then it is not his turn or the game is in a winning state, and he just waits.

This strategy is a winning strategy in some cases.

Proposition 3. *If $k \geq m$, then we have:*

$$\mathcal{C}_{Nim_{n,m,k}}, w_{1,m} \models \langle\!\langle 1 \rangle\!\rangle_{\rho^{\infty}_{individual_1}} \Diamond wins_1$$

(the strategy $individual_1$ is a winning strategy for the first player).

If $k \geq m$, then $winInst_1$ is feasible: $\langle\!\langle a \rangle\!\rangle_{\{a \mapsto take_m\}} \bigcirc wins_a$ is true in the initial state (a can remove all the m chips at once and win), so a will play $take_m$ and win.

4.3 Two-player classic strategy for coalitions

The two-player version of this game has often been studied and there exists a simple winning strategy for it [2]. The idea is simple: if possible, take a number of chips such that the number of remaining chips is equal to a multiple of $k+1$. If the number of remaining chips is different from a multiple of $k+1$, this is always possible: if $ncl \pmod{k+1} = p$ with $p \neq 0$, then we can always remove p chips to reach a state with $ncl \pmod{k+1} = 0$ since $1 \leq p \leq k$. If a player manages to do this once, then his opponent cannot reach a world with the same statement, and so the first player can maintain this state until the end of the game, and win when $ncl = 0$. In our language, this strategy $modulo_a$ for a player a can be expressed as:

– $modulo_a$ (if possible, reach a state with a number of chips multiple of $k+1$):

$$\rho_{modulo_a} = \neg multiple?; \{a \mapsto take_{ncl \pmod{k+1}}\} \cup multiple?; \{a \mapsto take_1\}$$

If the initial number of chips m is not a multiple of $k+1$, and if a follows it every time he can play $take_i$, this commitment is deterministic and feasible (whatever the second player chooses, he cannot reach a state with *multiple* true) and leads to a winning state for the first player. If m is a multiple of $k+1$, then this commitment is also feasible and deterministic (actually, this second part of the commitment will never be used in the following formulas, since we will never commit agents to this strategy if they are in a case where *multiple* is true).

This strategy does not transfer directly to n players. But an idea is, if n is even, to form two coalitions of the same size, like $A = \{1, 3, 5, ..., n-1\}$ and $A' = S \setminus A = \{2, 4, 6, ..., n\}$, to "simulate" a two-player game. Then the previous commitment can be applied.

We define the strategy $classic_A$, with A a coalition of agents, as:

$$\rho_{classic_A} = \underset{a \in A}{\triangle} (\rho_{modulo_a} \nabla \rho_{idle_a})$$

meaning that each agent from a coalition A tries to play the specified $take_i$, but if this is not possible (because it is another player's turn or the game has ended), then he just waits (as seen before for the individual strategy). This combination is deterministic and feasible for our coalition A if m is not a multiple of $k+1$, because 1 can follow ρ_{modulo_1} and the other agents from A cannot play $take_i$, so they follow ρ_{idle_a}, then at the next turn *multiple* is true, so all players from A follow ρ_{idle_a} (while 2 plays $take_1$), then 3 follows ρ_{modulo_3} while the others follow ρ_{idle_a}, and so on. Therefore players from A can always follow ρ_{modulo_a} at their turn and wait when *multiple* is true (during the turn of an agent from A') or when $take_i$ is not feasible (when another agent from A plays). Then all commitments from the prioritized conjunction $\rho_{classic_A}$ will always be applied together. Symmetrically, $\rho_{classic_{A'}}$ is deterministic and feasible for A'.

With these commitments, we can formulate the two-player strategy of one coalition or the other. Using the result from the two-player game of Nim, we can came to the following conclusion.

Proposition 4. *If m is not a multiple of $k+1$, then we have:*

$$\mathcal{C}_{Nim_{n,m,k}}, w_{1,m} \models \langle\langle A \rangle\rangle_{\rho^{\infty}_{classic_A}} \Diamond \left(\bigvee_{a \in A} wins_a \right)$$

(the strategy $classic_A$ is a winning strategy for the coalition A). If m is a multiple of $k+1$, then we have:

$$\mathcal{C}_{Nim_{n,m,k}}, w_{1,m} \models \langle\langle A' \rangle\rangle_{\rho^{\infty}_{classic_{A'}}} \Diamond \left(\bigvee_{a' \in A'} wins_{a'} \right)$$

(the strategy $classic_{A'}$ is a winning strategy for the coalition A').

A winning strategy for a coalition is a strategy leading to a winning state for at least one member of the coalition.

We could also consider strategies for "consecutive coalitions", that is, the first $p \leq n$ players will form a coalition against the last players. With a certain number of chips left, one coalition could inevitably win or simply win by removing as many chips as possible.

5 Conclusion

We have extended ATLEA by using programs instead of actions. This allows to commit agents further in the future and even to repeat some actions indefinitely, that means, to express (memoryless) strategies. We have then adapted a program operator stemming from the Game Description Language GDL and have demonstrated its usefulness by means of an example.

In the future, we would like to add an epistemic dimension to this framework in order to directly express uniform strategies with the help of commitments, that is, strategies for which agents choose the same action at indistinguishable worlds.

Acknowledgments

We gratefully acknowledge the anonymous reviewers for their comments. The last author acknowledges the support of the German Research Foundation (DFG) within the Cluster of Excellence Center for Advancing Electronics Dresden (cfAED).

References

1. Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49(5):672–713, 2002.
2. Elwyn Berlekamp, John H. Conway, and Richard K. Guy. *Winning Ways for Your Mathematical Plays (4 vols.)*. A. K. Peters, 2nd edition, 2000.
3. Andreas Herzig, Emiliano Lorini, and Dirk Walther. Reasoning about actions meets strategic logics. In Davide Grossi, Olivier Roy, and Huaxin Huang, editors, *International Workshop on Logic, Rationality and Interaction*, 2013.
4. Annela R. Kelly. One-pile misre Nim for three or more players. *International Journal of Mathematics and Mathematical Sciences*, 2006.
5. Martin Lange and Markus Latte. A CTL-based logic for program abstractions. In *WoLLIC*, volume 6188, pages 19–33, 2010.
6. Fabio Mogavero, Aniello Murano, and Moshe Y. Vardi. Reasoning about strategies. In Kamal Lodaya and Meena Mahajan, editors, *FSTTCS*, volume 8 of *LIPIcs*, pages 133–144, 2010.
7. Nicolas Troquard and Dirk Walther. Alternating-time dynamic logic. In Wiebe van der Hoek, Gal A. Kaminka, Yves Lespérance, Michael Luck, and Sandip Sen, editors, *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 473–480. IFAAMAS, 2010.
8. Dirk Walther, Wiebe van der Hoek, and Michael Wooldridge. Alternating-time temporal logic with explicit strategies. In *Proceedings of the 11th Conference on Theoretical Aspects of Rationality and Knowledge*, pages 269–278, 2007.
9. Michael Wooldridge and Wiebe van der Hoek. Cooperation, knowledge, and time: Alternating-time temporal epistemic logic and its applications. *Studia Logica*, 75(1):125–157, 2003.
10. Dongmo Zhang and Michael Thielscher. Representing and reasoning about game strategies. *Under review process of J. Philosophical Logic*, 2014.