Model-Checking an Alternating-time Temporal Logic with Knowledge, Imperfect Information, Perfect Recall and Communicating Coalitions[*]

Cătălin Dima

LACL, Université Paris Est-Créteil, 61 av. du G-ral de Gaulle, 94010 Créteil, France

Constantin Enea[†]

LIAFA, CNRS UMR 7089, Université Paris Diderot - Paris 7, Case 7014, 75205 Paris Cedex 13, France

Dimitar Guelev[‡]

Section of Logic, Institute of Mathematics and Informatics, Bulgarian Academy of Sciences,
Acad. G. Bonchev str., bl. 8, 1113 Sofia, Bulgaria

Abstract. We present a variant of ATL with distributed knowledge operators based on a synchronous and perfect recall semantics. The coalition modalities in this logic are based on partial observation of the full history, and incorporate a form of cooperation between members of the coalition in which agents issue their actions based on the distributed knowledge, for that coalition, of the system history. We show that model-checking is decidable for this logic. The technique utilizes two variants of games with imperfect information and partially observable objectives, as well as a subset construction for identifying states whose histories are indistinguishable to the considered coalition.

1 Introduction

Alternating-time Temporal Logic (ATL) [AHK98, AHK02] is a generalization of the Computational Tree Logic (CTL) in which path quantifiers "∃" and "∀" are replaced by *cooperation modalities* $\langle\!\langle A \rangle\!\rangle$ in which A denotes a set of *agents* who act as a *coalition*. A formula $\langle\!\langle A \rangle\!\rangle \phi$ expresses the fact that the agents in *coalition A* can cooperate to ensure that ϕ holds in an appropriate type of multiplayer game.

The precise semantics of the cooperation modalities varies depending on whether the knowledge that each agent has of the current state of the game is complete or not, and whether agents can use knowledge of the past game states when deciding on their next move or not. These alternatives are known as *complete*, resp. *incomplete information*, and *perfect*, resp. *imperfect recall*. In the case of imperfect recall further subdivisions depend on how much memory an agent is allowed for storing information on the past in addition to its possibly incomplete view of the current state. In the extreme case agents and, consequently, the strategies they can carry out, are *memoryless*.

It is known that the model-checking problem for the case of complete information is decidable in polynomial time [AHK98]. In the case of incomplete information and perfect recall model-checking is believed to be undecidable, a statement attributed to M. Yannakakis in [AHK98] for which there is no

self-contained proof that we know about. Variants of ATL with memoryless agents have been shown to have decidable model checking in [Sch04, ÅGJ07, vdHLW06]. Our earlier work [GD08] is about a special case of agents with perfect recall in which model checking is still decidable.

Incomplete information is modelled in ATL in a way which conforms with the possible worlds semantics of modal epistemic logics (cf. [FHMV04].) Therefore, it is of no surprise that the epistemic logic community contributed extensions of ATL by knowledge modalities such as *Alternating Temporal Epistemic Logic* [vdHW03]. Results on model-checking ATEL with memoryless strategies can be found in [Sch04, ÅGJ07, KP05, vdHLW06]. Results on ATL with complete information can be found in [GJ04, BJ09].

In this paper we continue our investigation of ATL with knowledge operators from [GD08], where we introduced conditions on the meaning of the cooperation modalities which make model-checking decidable. As in the previous paper, we do not restrict agents' strategies to memoryless ones, but we assume that coalition members have a communication mechanism which enables the coalitions to carry out strategies that are based on their *distributed knowledge*. (Recall that a coalition has *distributed knowledge* of fact ϕ iff ϕ is a logical consequence of the combined knowledge of the coalition members.) We assume that a coalition has a strategy to achieve a goal ϕ only if the same strategy can be used in all the cases which are indistinguishable from the actual one with respect to the distributed knowledge of the coalition. This choice is known as *de re* strategies [JA07], and rules out the possibility for a coalition to be able to achieve ϕ by taking chances, or to be able to achieve ϕ in some of the cases which are consistent with its knowledge and not in others. Therefore in our system $\langle\!\langle A \rangle\!\rangle \phi$ is equivalent to $K_A \langle\!\langle A \rangle\!\rangle \phi$ where K_A stands for the *distributed knowledge* operator (also written D_A). We call the variant of ATL which is obtained by adopting these conventions *Alternating Time Logic with Knowledge and Communicating Coalitions* and use the acronym ATL$_{iR}^D$ for it to indicate distributed knowledge, incomplete information and perfect recall.

Implementing strategies which rely on distributed knowledge requires some care. For instance, simply supplying coalition members with a mechanism to share their observations with each other would have the side effect of enhancing the knowledge at each agent's disposal upon considering the reachability of subsequent goals as part of possibly different coalitions, whereas we assume that each agent's knowledge is just what follows from its personal experience at all times. Therefore we assume that coalition activities are carried out through the guidance of corresponding *virtual supervisors* who receive the coalition members' observations and previously accumulated knowledge and in return direct their actions for as long as the coalition exists without making any additional information available.

In our previous work models are based on *interpreted systems* as known from [FHMV04]. In that setting global system states are tuples which consist of the local views of the individual agents and the satisfaction of atomic propositions at a global state need not be related to the local views in it. Unlike that, in this paper we assume that the view of each agent is described as a set of atomic propositions which the agent can observe. States which satisfy the same observable atomic propositions are indistinguishable to the agent. Observability as in interpreted systems can be simulated in this concrete observability semantics. However, the converse does not hold, see [Dim10] for details.

We prove our model-checking result by induction on the construction the formula to be checked, like in model-checking algorithms for ATL or CTL, with two significant differences. Firstly, the implicit distributed knowledge operator hidden in the coalition operator is handled by means of a "subset construction" for identifying states with indistinguishable histories, a technique used for CTLK model-checking in [Dim08]. Secondly, checking whether in a given set of indistinguishable states the coalition has a strategy to achieve goal ϕ involves building a tree automaton, which can be seen as a game between the coalition (supervisor) and the rest of the agents. This game resembles the two-player games with one

player having imperfect information from [CDHR06], but also has a notable difference: the goal of the player with imperfect information is *not fully observable*. Such a goal can be achieved *at different times* along different yet indistinguishable runs. Therefore, we have a bookkeeping mechanism for the time of achieving the goal along each run.

The tree automata we use employ only "occurrence" accepting conditions: the set of states occurring along each run of the tree is required to belong to some given set of sets of states. No Muller conditions, i.e., no restrictions on the set of states occurring *infinitely often*, are involved.

The model-checking algorithm proceeds by constructing *refinements* of the given game arena Γ, unlike in CTL and ATL model-checking where the only modifications of the given arena are the insertion of new propositional variables (corresponding to subformulas of the formula to model-check). This refinement enables telling apart classes of histories which are indistinguishable to coalition members. It involves splitting states by means of a subset construction. The technique is known from model-checking epistemic extensions of CTL or LTL with perfect recall.

The setting and techniques presented here are different from those in our previous work [GD08]. In [GD08], the knowledge modalities are required to have only argument formulas from the past subset of *LTL*. ATL_{iR}^D has only future operators. Past *LTL* operators can be added to ATL_{iR}^D in the usual way. Also, the model-checking algorithm for ATL_{iR}^D is based on tree-automata and not on the syntactical transformation of past formulas as in [GD08].

Let us also note the difference between our work and the work on ATEL: the approach proposed in ATEL is to consider that strategies are defined on *sequences of states*, which is a *perfect observability* approach. Hence, a formula of the form $\langle\!\langle Alice\rangle\!\rangle\phi$, saying that *Alice* has a strategy to ensure ϕ in a given state, refers to the situation in which *Alice* would be able to ensure ϕ if she had complete information about the system state. As in general agents do not have complete information, ATEL proposes then to use knowledge operators as a means to model imperfect information. The idea is to use formulas of the form $K_{Alice}\langle\!\langle Alice\rangle\!\rangle\phi$ to specify the fact that *Alice* knows that she can enforce ϕ in the current state.

Unfortunately, this does not solve the *unfeasible strategies* problem, studied in [GJ04]. Namely, the knowledge operator in formula $K_{Alice}\langle\!\langle Alice\rangle\!\rangle\phi$ does not give *Alice* the ability to know what action she has to apply in the current state. This is because the knowledge operator only gives evidence about the fact that *strategies exist, in all identically observable states, to ensure* ϕ, but different strategies may exist in identically observable states, and hence *Alice* might not be able to know what strategy she is to apply after some sequence of observations.

Another argument against the possibility to encode the setting from e.g. [Sch04] into the ATEL setting from [vdHW03] refers to the difficulty of giving a fixpoint definition to the operators involving $\langle\!\langle Alice\rangle\!\rangle$. The reason is that, for formulas of the form $\langle\!\langle A\rangle\!\rangle\diamond\phi$, it is possible that ϕ becomes satisfied at different times along different yet indistinguishable runs. Hence, despite that *Alice* can enforce ϕ by means of a fixed strategy, she might be unable to tell when ϕ happens. At best, in case every global state has only finitely many successors, *Alice* would eventually be able to tell that ϕ *must have been achieved*. This observation is related with the bookkeeping mechanism used in Subsection 4.2 here, in the association of a tree automaton with each subformula of the form $\langle\!\langle A\rangle\!\rangle\phi_1\mathcal{U}\phi_2$.

In conclusion, we believe that there is little hope to encode the imperfect information setting studied here within the ATEL framework from [vdHW03, GJ04].

Structure of the paper The next section recalls some basic notions and notations used throughout the paper, including the tree automata that are used in the model-checking algorithm. Section 3 presents the syntax and semantics of ATL_{iR}^D. Section 4 gives the constructions involved in the model-checking algorithm: the subset construction for identifying indistinguishable histories, and then the tree automata for handling formulas of the forms $\langle\!\langle A\rangle\!\rangle p_1\mathcal{W}p_2$ and $\langle\!\langle A\rangle\!\rangle p_1\mathcal{U}p_2$, respectively. We conclude by a summary

of our result, discussion and topics of further work.

2   Preliminaries

Given a set A, A^* stands for the set of finite sequences over A. The empty sequence is denoted by ε. The prefix order between sequences is denoted by $\leq$ and the concatenation of sequences by $\cdot$. The *direct product* of a family of sets $(X_a)_{a \in A}$ is denoted by $\prod_{a \in A} X_a$. An element x of $\prod_{a \in A} X_a$ will be written in the form $x = (x_a)_{a \in A}$, where $x_a \in X_a$ for all $a \in A$. If $B \subseteq A$, then $x|_B = (x_b)_{b \in B}$ stands for the *restriction* of x to B. If the index set A is a set of natural numbers and $n \in A$, then $x|_n$ stands for $x|_{\{n\}}$. The *support* $\mathsf{supp}(f)$ of a partial function $f : A \rightharpoonup B$ is the subset of elements of A on which the function is defined.

Given a set of symbols Δ, a Δ-*labeled tree* is a partial function $t : \mathbb{N}^* \rightharpoonup \Delta$ such that

1. $\varepsilon \in \mathsf{supp}(t)$.

2. The support of t is prefix-closed: if $x \in \mathsf{supp}(t)$ and $y \leq x$, then $y \in \mathsf{supp}(t)$.

3. Trees are "full": if $xi \in \mathsf{supp}(t)$, then $xj \in \mathsf{supp}(t)$ for all $j \leq i$ too.

4. All tree branches are infinite: If $x \in \mathsf{supp}(t)$ then $x0 \in \mathsf{supp}(t)$ too.

Elements of $\mathsf{supp}(t)$ are called *nodes* of t. A *path* in t is an infinite sequence of nodes $\pi = (x_k)_{k \geq 0}$ such that for all k, x_{k+1} is an *immediate* successor of x_k, i.e. $x_{k+1} = x_k l$ for some $l \in \mathbb{N}$. Path $(x_k)_{k \geq 0}$ is *initialized* if x_0 is the tree root ε. We denote the set of labels on the path π, that is, $\{ t(x_k) \mid k \geq 0 \}$, by $t(\pi)$.

Below we use tree automata $\mathcal{A} = (Q, \Sigma, \delta, Q_0, \mathcal{F})$ in which Q is the set of *states*, Σ is the *alphabet*, $Q_0 \subseteq Q$ is the set of the *initial* states, $\delta \subseteq Q \times \Sigma \times (2^Q \setminus \varnothing)$ is the *transition relation* and the acceptance condition $\mathcal{F}$ is a subset of 2^Q.

Tree automata accept $Q \times \Sigma$-labelled trees. A tree $t : \mathbb{N}^* \rightharpoonup Q \times \Sigma$ represents an *accepting run* in $\mathcal{A}$ iff:

1. $t(\varepsilon) \in Q_0 \times \Sigma$.

2. If $x \in \mathsf{supp}(t)$, then $t(xi)|_Q \neq t(xj)|_Q$ whenever $i \neq j$, and $(t(x)|_Q, t(x)|_\Sigma, \{ t(xi)|_Q \mid xi \in \mathsf{supp}(t) \}) \in \delta$.

3. $t(\pi)|_Q \in \mathcal{F}$ for all *initialized* paths $\pi \subseteq \mathsf{supp}(t)$.

Note that we only consider automata with "occurrence" accepting conditions: an initialized path is accepting if the set of states *occurring* on the path is a member of $\mathcal{F}$, even if some of these states occur only finitely many times.

Theorem 1 ([Tho97]) *The emptiness problem for tree automata with "occurrence" accepting conditions, i.e., the problem of checking whether, given a tree automaton $\mathcal{A}$, there exists an accepting run in $\mathcal{A}$, is decidable.*

3   Syntax and semantics of ATL$_{iR}^D$

Throughout this paper we fix a non-empty finite set Ag of *agents* and, for each $a \in Ag$, a set of *atomic propositions* $Prop_a$, which are assumed to be observable to a. Given $A \subseteq Ag$, we write $Prop_A$ for $\bigcup_{a \in A} Prop_a$. We abbreviate $Prop_{Ag}$ to $Prop$.

3.1 Game arenas

Definition 2 *A* game arena *is a tuple* $\Gamma = (Ag, Q, (C_a)_{a \in Ag}, \delta, Q_0, (Prop_a)_{a \in Ag}, \lambda)$, *where*

- Ag *and* $Prop_a$, $a \in Ag$, *are as above.*
- Q *is a set of* states,
- C_a *is a finite sets of* actions *available to agent* a. *We write* C_A *for* $\prod_{a \in A} C_a$ *and* C *for* C_{Ag}.
- $Q_0 \subseteq Q$ *is the set of* initial states.
- $\lambda : Q \to 2^{Prop}$ *is the* state-labeling function.
- $\delta : Q \times C \to (2^Q \setminus \varnothing)$ *is the* transition relation.

An element $c \in C$ will be called an *action tuple*. We write $q \xrightarrow{c} r$ for *transitions* $(q, c, r) \in \delta$. We define $\lambda_A : Q \to 2^{Prop_A}$, $A \subseteq Ag$, by putting $\lambda_A(q) = \lambda(q) \cap Prop_A$. We assume that λ and λ_A are defined on subsets S of Q by putting $\lambda(S) = \bigcup_{q \in S} \lambda(q)$ for λ, and similarly for λ_A.

Given an arena Γ, a *run* ρ is a sequence of transitions $q_i' \xrightarrow{c_i} q_i''$ such that $q_{i+1}' = q_i''$ for all i. We write $\rho = (q_{i-1} \xrightarrow{c_i} q_i)_{1 \le i \le n}$, resp. $\rho = (q_{i-1} \xrightarrow{c_i} q_i)_{i \ge 1}$ for finite, resp. infinite runs. The *length* of ρ, denoted $|\rho|$, is the number of its transitions. This is ∞ for infinite runs. A run $\rho = q_0 \xrightarrow{c_1} q_1 \xrightarrow{c_2} \ldots$ is *initialized* if $q_0 \in Q_0$. $\mathsf{Runs}^f(\Gamma)$ denotes the set of initialized finite runs and $\mathsf{Runs}^\omega(\Gamma)$ denotes the set of initialized infinite runs of Γ.

Given a run $\rho = q_0 \xrightarrow{c_1} q_1 \xrightarrow{c_2} \ldots$, we denote q_i by $\rho[i]$, $i = 0, \ldots, |\rho|$, and c_{i+1} by $act(\rho, i)$, $i = 0, \ldots, |\rho| - 1$. We write $\rho[0..i]$ for the *prefix* $q_0 \xrightarrow{c_1} q_1 \xrightarrow{c_1} \ldots \xrightarrow{c_i} q_i$ of ρ of length i.

A *coalition* is a subset of Ag. Given a coalition A, $S \subseteq Q$, $c_A \in C_A$, and $Z \subseteq Prop_A$, the following set denotes the *outcome of c_A from S, labeled with Z*:

$$out(S, c_A, Z) = \{s' \in Q \mid (\exists s \in S, \exists c' \in C) c'|_A = c_A, s \xrightarrow{c'} s' \in \delta \text{ and } \lambda_A(s') = Z\}$$

whereas those from $Prop_A \setminus Z$ are false.

Runs ρ and ρ' are *indistinguishable (observationally equivalent)* to coalition A, denoted $\rho \sim_A \rho'$, if $|\rho| = |\rho'|$, $act(\rho, i)|_A = act(\rho', i)|_A$ for all $i < |\rho|$, and $\lambda_A(\rho[i]) = \lambda_A(\rho'[i])$ for all $i \le |\rho|$.

Definition 3 *A* strategy *for a coalition A is any mapping* $\sigma : (2^{Prop_A})^* \to C_A$.

We write $\Sigma(A, \Gamma)$ for the set of all strategies of coalition A in game arena Γ.

Note that, instead of describing strategies for coalitions as tuples of strategies for their individual members with every member choosing its actions using just its own view of the past, we assume a *joint* strategy in which the actions of every coalition member depend on the combined view of the past of all the members. We may therefore assume that the coalition is guided by a supervisor who receives the members' view of the current state, and, in return, advices every coalition member of its next action. The supervisor sends no other information. We refer the reader to a short discussion in the last section, on this supervisor interpretation of joint strategies.

Finite sequences of subsets of $Prop_A$ will be called *A-histories*.

Strategy σ for coalition A is *compatible* with a run $\rho = q_0 \xrightarrow{c_1} q_1 \xrightarrow{c_2} \ldots$ if

$$\sigma(\lambda_A(\rho[0]) \cdots \lambda_A(\rho[i])) = c_{i+1}|_A$$

for all $i \le |\rho|$. Obviously if σ is compatible with run ρ then it is compatible with any run that is indistinguishable from ρ to A.

3.2 ATL_{iR}^D defined

The syntax of ATL_{iR}^D formulas ϕ can be defined by the grammar

$$\phi ::= p \mid \phi \wedge \phi \mid \neg\phi \mid \langle\!\langle A \rangle\!\rangle \bigcirc \phi \mid \langle\!\langle A \rangle\!\rangle \phi \, \mathcal{U} \, \phi \mid \langle\!\langle A \rangle\!\rangle \phi_1 \, \mathcal{W} \, \phi_2 \mid K_A \phi$$

where p ranges over the set *Prop* of atomic propositions, and A ranges over the set of subsets of *Ag*.

Below it becomes clear that admitting $\mathcal{W}$ as a basic temporal connective allows us to introduce all the remaining combinations of $\langle\!\langle A \rangle\!\rangle$ and its dual $[\![A]\!]$ and the temporal connectives as syntactic sugar (see [BJ09, LMO08] for more details). Satisfaction of ATL_{iR}^D formulas is defined with respect to a given arena Γ, a run $\rho \in \mathsf{Runs}^\omega(\Gamma)$ and a position i in ρ by the clauses:

- $(\Gamma, \rho, i) \vDash p$ if $p \in \lambda(\rho[i])$.

- $(\Gamma, \rho, i) \vDash \phi_1 \wedge \phi_2$, if $(\Gamma, \rho, i) \vDash \phi_1$ and $(\Gamma, \rho, i) \vDash \phi_2$.

- $(\Gamma, \rho, i) \vDash \neg\phi$ if $(\Gamma, \rho, i) \nvDash \phi$.

- $(\Gamma, \rho, i) \vDash \langle\!\langle A \rangle\!\rangle \bigcirc \phi$ if there exists a strategy $\sigma \in \Sigma(A, \Gamma)$ such that $(\Gamma, \rho', i+1) \vDash \phi$ for all runs $\rho' \in \mathsf{Runs}^\omega(\Gamma)$ which are compatible with σ and satisfy $\rho'[0..i] \sim_A \rho[0..i]$.

- $(\Gamma, \rho, i) \vDash \langle\!\langle A \rangle\!\rangle \phi_1 \, \mathcal{U} \, \phi_2$ iff there exists a strategy $\sigma \in \Sigma(A, \Gamma)$ such that for every run $\rho' \in \mathsf{Runs}^\omega(\Gamma)$ which is compatible with σ and satisfies $\rho'[0..i] \sim_A \rho[0..i]$ there exists $j \geq i$ such that $(\Gamma, \rho', j) \vDash \phi_2$ and $(\Gamma, \rho', k) \vDash \phi_1$ for all $k = i, \ldots, j-1$.

- $(\Gamma, \rho, i) \vDash \langle\!\langle A \rangle\!\rangle \phi_1 \, \mathcal{W} \, \phi_2$ iff there exists a strategy $\sigma \in \Sigma(A, \Gamma)$ such that for every run $\rho' \in \mathsf{Runs}^\omega(\Gamma)$ which is compatible with σ and satisfies $\rho'[0..i] \sim_A \rho[0..i]$ one of the two situations occur:
    1. Either there exists $j \geq i$ such that $(\Gamma, \rho', j) \vDash \phi_2$ and $(\Gamma, \rho', k) \vDash \phi_1$ for all $k = i, \ldots, j-1$.
    2. Or $(\Gamma, \rho', k) \vDash \phi_1$ for all $k \geq i$.

- $(\Gamma, \rho, i) \vDash K_A \phi$ iff $(\Gamma, \rho', i) \vDash \phi$, for all runs $\rho' \in \mathsf{Runs}^\omega(\Gamma)$ which satisfy $\rho'[0..i] \sim_A \rho[0..i]$.

The rest of the combinations between the temporal connectives and the cooperation modalities $\langle\!\langle A \rangle\!\rangle$ and $[\![A]\!]$ are defined as follows:

$$
\begin{array}{ll}
P_A \phi = \neg K_A \neg \phi & [\![A]\!]\bigcirc\phi = \neg\langle\!\langle A \rangle\!\rangle\bigcirc\neg\phi \\
[\![A]\!]\phi\,\mathcal{U}\,\psi = \neg\langle\!\langle A \rangle\!\rangle(\neg\psi\,\mathcal{W}\,\neg\psi\wedge\neg\varphi) & [\![A]\!]\phi\,\mathcal{W}\,\psi = \neg\langle\!\langle A \rangle\!\rangle(\neg\psi\,\mathcal{U}\,\neg\psi\wedge\neg\varphi) \\
\langle\!\langle A \rangle\!\rangle\Diamond\phi = \langle\!\langle A \rangle\!\rangle\mathit{true}\,\mathcal{U}\,\phi & \langle\!\langle A \rangle\!\rangle\Box\phi = \langle\!\langle A \rangle\!\rangle\phi\,\mathcal{W}\,\mathit{false} \\
[\![A]\!]\Diamond\phi = [\![A]\!]\mathit{true}\,\mathcal{U}\,\phi & [\![A]\!]\Box\phi = [\![A]\!]\phi\,\mathcal{W}\,\mathit{false}
\end{array}
$$

A formula ϕ is *valid in a game arena* Γ, written $\Gamma \vDash \phi$, if $(\Gamma, \rho, 0) \vDash \phi$ for all $\rho \in \mathsf{Runs}^\omega(\Gamma)$. The *model-checking problem* for ATL_{iR}^D is to decide whether $\Gamma \vDash \phi$ for a given formula ϕ and arena Γ.

Example 4 *Alice and Bob, married, work in the same company. When they arrive at work, they are assigned (by some non-modeled agent) one of the tasks x or y. These tasks need different periods of time to be executed: tx time units for x and ty time units for y, where tx < ty. The assignment is always such that task y cannot be assigned to both Alice and Bob. After they finished executing their task, Alice and Bob have two objectives: (1) to pick their child from the nursery, and (2) to do the shopping. The supermarket closes early, so the one who does the longest task cannot do the shopping. So Alice and Bob need to exchange information about their assigned task in order to fix who's to do the shopping and who's to pick the child from the nursery.*

Figure 1 pictures the game arena representing this system. The actions Alice and Bob can do are: g for going at work, e for working on their task, tc for taking the child, ds for doing the shopping, and i for idling. The atomic proposition xx denotes the assignment of task x to both Alice and Bob, xy denotes assignment of task x to Alice and task y to Bob, and yx denotes assignment of task y to Alice and task x to Bob. All these atomic propositions are not observable by the two agents. The atomic propositions x_a and y_a are observed only by Alice, and the atomic propositions x_b and y_b are observed only by Bob. All these four atoms denote the fact that the respective person has to execute task x or y. Furthermore, the atomic propositions tx_a and ty_a which are observed only by Alice, and tx_b and ty_b, which are observed only by Bob, denote the fact that the respective person has been working for tx or ty time units. The atomic propositions c, s, can be observed by both Alice and Bob and denote the fact that the child was picked from the nursery, and, respectively, that the fridge is full with food from the supermarket. An arc labeled by two vectors of actions, e.g. (tc, ds) (ds, tc), denotes two arcs with the same origin and the same destination, each one of them labeled by one of the vectors.

We suppose that the game arena contains a sink state *which is the output of all the transitions not pictured in Figure 1 (for instance, both agents idling in state q_6 brings the sistem to the sink state). Also, we suppose that all the states except for the sink state are labeled by some atomic proposition valid visible to Alice.*

An interesting property for this system is that Alice and Bob can form a coalition in order to pick their child and do their shopping (if we ignore the sink state)– that is, the following formula is true:

$$\phi = \langle\!\langle \{Alice, Bob\} \rangle\!\rangle (valid \ \mathcal{U} \ c \wedge s)$$

Note that Alice and Bob need a strategy which must include some communication during its execution, which would help each of them to know who is assigned which task during the day, and hence who cannot do the shopping. Note also that the model incorporates some timing information, such that the two agents need a strategy with perfect recall in order to reach their goal: after working tx time units both Alice and Bob must use their observable past to remember if they have finished working. Finally, note that, if we consider that strategies for coalitions are tuples of strategies for individual members, as in [AHK98, Sch04] then the formula ϕ is false: whatever decision Alice and Bob take together, in the morning, about who is to pick the child, who is to do shopping, and in what observable circumstances (but without exchanging any information), can be countered by the task assignment, which would bring Alice and Bob at the end of the day either with an empty fridge or the child spending his night at the nursery.

4   Model-checking ATL$_{iR}^{D}$

The model-checking procedure for ATL$_{iR}^{D}$ builds on model checking techniques for CTL with knowledge modalities and ATL with complete information. It works by recursion on the construction of formulas. Given a formula ϕ with a cooperation modality as the main connective, the procedure involves refining the given arena Γ to an arena $\widehat{\Gamma}$ in which the state space can be partitioned into states which satisfy ϕ and states which do not.

The idea is to have, after the splitting, an equivalence relation $\equiv_A$ on the states of the resulting game arena $\widehat{\Gamma}$, such that $\widehat{q_1} \equiv_A \widehat{q_2}$ iff $\widehat{q_1}$ and $\widehat{q_2}$ are reachable through the same histories, as seen by A.

The construction of the refined state space is inspired by the usual construction of a game with perfect information for solving two-player games with one player having imperfect information, see [CDHR06]. However the construction is more involved, because, contrary to [CDHR06] the objectives here may not be observable by the coalition.
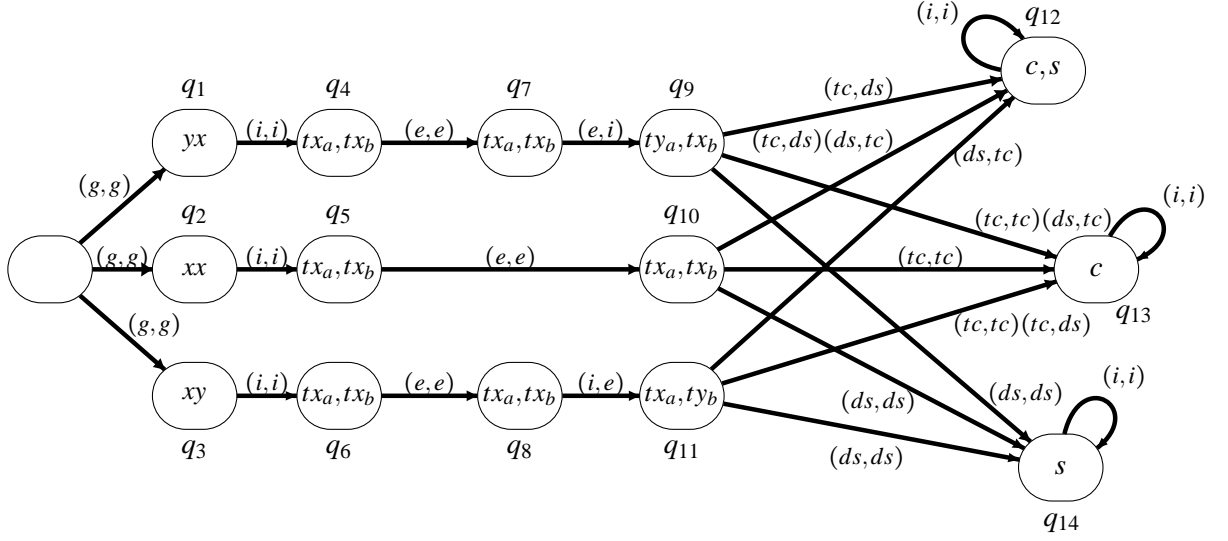
Figure 1: A game arena for Example 4

4.1   The state-splitting construction.

Given a game arena $\Gamma = (Ag, Q, (C_a)_{a \in Ag}, \delta, Q_0, (Prop_a)_{a \in Ag}, \lambda)$ and a coalition A, we construct a new game arena $\widehat{\Gamma_A} = (Ag, \widehat{Q}, (C_a)_{a \in Ag}, \widehat{\delta}, \widehat{Q_0}, (Prop_a)_{a \in Ag}, \widehat{\lambda})$, as follows:

- $\widehat{Q} = \{(q, S) \mid S \subseteq Q, q \in S \text{ and for all } s \in S, \lambda_A(s) = \lambda_A(q)\}$;

- $\widehat{Q_0} = \{(q_0, S_0) \mid q_0 \in Q_0 \text{ and } S_0 = \{s \in Q_0 \mid \lambda_A(s) = \lambda_A(q_0)\}\}$;

- $\widehat{\lambda}(q, S) = \lambda(q)$ for all $(q, S) \in \widehat{Q}$.

- $(q, S) \xrightarrow{c} (q', S') \in \widehat{\delta}$ if and only if the following properties hold:

  - $(q, S), (q', S') \in \widehat{Q}$ and $c \in C$;
  - $q \xrightarrow{c} q' \in \delta$;
  - $S' = out(S, c|_A, \lambda_A(q'))$.

The intended equivalence on states is then the following: $\widehat{q} \equiv_A \widehat{q'}$ if and only if there exists $S \subseteq Q$ with $\widehat{q} = (q, S)$ and $\widehat{q'} = (q', S)$.

Every run $\rho \in \mathsf{Runs}^{\omega}(\Gamma)$, $\rho = (q_{i-1} \xrightarrow{c_i} q_i)_{i \geq 1}$, has a unique corresponding run $\widehat{\rho} \in \mathsf{Runs}^{\omega}(\widehat{\Gamma_A})$, $\widehat{\rho} = ((q_{i-1}, S_{i-1}) \xrightarrow{c_i} (q_i, S_i))_{i \geq 1}$. This is because q_0 unambiguously determines S_0 and, recursively, S_{i-1} uniquely determines S_i, for any $i \geq 1$. The converse holds too, that is, to each run $\overline{\rho} = ((q_{i-1}, S_{i-1}) \xrightarrow{c_i} (q_i, S_i))_{i \geq 1}$ in $\mathsf{Runs}^{\omega}(\widehat{\Gamma_A})$, corresponds a unique run $\rho = (q_{i-1} \xrightarrow{c_i} q_i)_{i \geq 1}$ such that $\widehat{\rho} = \overline{\rho}$. Furthemore, every strategy for A in Γ is also a strategy for A in $\widehat{\Gamma_A}$.

Proposition 5   *1. If ρ and ρ' are runs in Γ of the same length, then $\rho \sim_A \rho'$ iff $\widehat{\rho} \sim_A \widehat{\rho'}$.*

  2. *If $B \subseteq Ag$, $\sigma \in \Sigma(B, \Gamma) = \Sigma(B, \widehat{\Gamma_A})$, and $\rho \in \mathsf{Runs}^{\omega}(\Gamma)$, then σ is compatible with ρ iff σ is compatible with $\widehat{\rho}$.*

  3. *If $\rho \in \mathsf{Runs}^{\omega}(\Gamma)$, $p \in Prop$ and $i \geq 0$, then $(\Gamma, \rho, i) \vDash K_A p$ is equivalent to both $(\widehat{\Gamma}, \widehat{\rho}, i) \vDash K_A p$, and to $p \in \lambda(s)$ for all s in the second component of $\widehat{\rho}[i]$.*

4. *If $\rho \in \mathsf{Runs}^{\omega}(\Gamma)$, ϕ is an arbitrary ATL^D_{iR} formula and $i \geq 0$, then*

$$(\Gamma, \rho, i) \vDash \phi \text{ iff } (\widehat{\Gamma_A}, \widehat{\rho}, i) \vDash \phi$$

Proof: (1), (2) and (3) follow directly from definition. (4) is proved by structural induction on ϕ. For example,

- if $\phi = K_B \psi$, for some $B \subseteq Ag$, then $(\widehat{\Gamma_A}, \widehat{\rho}, 0) \vDash \phi$ iff $(\widehat{\Gamma_A}, \widehat{\rho'}, 0) \vDash \psi$ for all $\widehat{\rho'} \in \mathsf{Runs}^{\omega}(\widehat{\Gamma_A})$ such that $\lambda_B(\widehat{\rho'}[0]) = \lambda_B(\widehat{\rho}[0])$. By the induction hypothesis, this is equivalent to $(\Gamma, \rho', 0) \vDash \psi$ for all $\rho' \in \mathsf{Runs}^{\omega}(\Gamma)$ such that $\lambda_B(\rho'[0]) = \lambda_B(\rho[0])$. The latter is equivalent to $(\Gamma, \rho, 0) \vDash \phi$.

- if $\phi = \langle\!\langle B \rangle\!\rangle \psi_1 \mathcal{U} \psi_2$, for some $B \subseteq Ag$, then $(\widehat{\Gamma_A}, \widehat{\rho}, i) \vDash \phi$ iff there exists a strategy $\sigma \in \Sigma(B, \widehat{\Gamma})$ such that for every run $\widehat{\rho'} \in \mathsf{Runs}^{\omega}(\widehat{\Gamma})$ which is compatible with σ and satisfies $\widehat{\rho'}[0..i] \sim_A \widehat{\rho}[0..i]$ there exists $j \geq i$ such that $(\widehat{\Gamma}, \widehat{\rho'}, j) \vDash \psi_2$ and $(\widehat{\Gamma}, \widehat{\rho'}, k) \vDash \psi_1$ for all $k = i, \dots, j-1$. Let $\rho'' \in \mathsf{Runs}^{\omega}(\Gamma)$ be a run compatible with σ such that $\rho''[0..i] \sim_A \rho[0..i]$. We have that $\widehat{\rho''}[0..i] \sim_A \rho[0..i] \sim_A \widehat{\rho}[0..i]$ and by (2), $\widehat{\rho''}$ is compatible with σ. Consequently, there exists $j \geq i$ such that $(\widehat{\Gamma}, \widehat{\rho''}, j) \vDash \psi_2$ and $(\widehat{\Gamma}, \widehat{\rho''}, k) \vDash \psi_1$ for all $k = i, \dots, j-1$. By the induction hypothesis, we obtain that $(\Gamma, \rho'', j) \vDash \psi_2$ and $(\Gamma, \rho'', k) \vDash \psi_1$ for all $k = i, \dots, j-1$ which implies $(\Gamma, \rho, i) \vDash \phi$. For the other implication we can proceed in a similar manner.

⊣

Remark 6 *Item (3) from Proposition 5 gives the state partitioning procedure for knowledge operators: we may partition the state space of $\widehat{\Gamma_A}$ as $\widehat{Q} = \widehat{Q}^{K_A p} \cup \widehat{Q}^{\neg K_A p}$, where*

$$\widehat{Q}^{K_A p} = \{(q, S) \in \widehat{Q} \mid (\forall s \in S)(p \in \lambda(s) = \lambda(q))\} \tag{1}$$

$$\widehat{Q}^{\neg K_A p} = Q \smallsetminus \widehat{Q}^{K_A p} \tag{2}$$

Example 7 *The arena $\widehat{\Gamma_{\{Alice,Bob\}}}$ corresponding to Γ from Figure 1 is obtained by replacing each state q with:*

- $(q, \{q\})$, *if $q \notin \{q_1, q_2, q_3\}$,*

- $(q, \{q_1, q_2, q_3\})$, *otherwise.*

The states $(q, \{q_1, q_2, q_3\})$ with $q \in \{q_1, q_2, q_3\}$ denote the fact that, from the point of view of Alice and Bob, q is reachable through the same history as the states q_1, q_2, and q_3.

4.2 The state labeling constructions

Our next step is to describe how, given an arena Γ and a coalition A, the states $(q, S) \in \widehat{Q}$ of $\widehat{\Gamma_A}$ can be labelled with the ATL^D_{iR} formulas which they satisfy in case the considered formulas have one of the forms $\langle\!\langle A \rangle\!\rangle \bigcirc p$, $\langle\!\langle A \rangle\!\rangle p_1 \mathcal{U} p_2$ and $\langle\!\langle A \rangle\!\rangle p_1 \mathcal{W} p_2$.

The three cases are different. Formulas of the form $\langle\!\langle A \rangle\!\rangle \bigcirc p$ are the simplest to handle. To do formulas of the forms $\langle\!\langle A \rangle\!\rangle p_1 \mathcal{U} p_2$ and $\langle\!\langle A \rangle\!\rangle p_1 \mathcal{W} p_2$, we build appropriate tree automata.

Case $\langle\!\langle A \rangle\!\rangle \bigcirc p$: We partition the state space of $\widehat{\Gamma_A}$ in $\widehat{Q}^{\langle\!\langle A \rangle\!\rangle \bigcirc p}$ and $\widehat{Q}^{\neg \langle\!\langle A \rangle\!\rangle \bigcirc p}$, where

$$\widehat{Q}^{\langle\!\langle A \rangle\!\rangle \bigcirc p} = \big\{(q, S) \in \widehat{Q} \mid \exists c \in C_A \text{ s.t. } \forall S' \subseteq Q, \ \forall r \in S, \ \forall r' \in S', \ \forall c' \in C,$$

$$\text{if } (r, S) \xrightarrow{c'} (r', S') \text{ and } c'\big|_A = c \text{ then } p \in \widehat{\lambda}(r')\big\} \tag{3}$$

$$\widehat{Q}^{\neg \langle\!\langle A \rangle\!\rangle \bigcirc p} = \widehat{Q} \smallsetminus \widehat{Q}^{\langle\!\langle A \rangle\!\rangle \bigcirc p} \tag{4}$$

Case $\langle\!\langle A\rangle\!\rangle p_1\,\mathcal{U}\,p_2$**:**   We build a tree automaton whose states represent histories which are indistinguish-able to A in a finitary way. A special mechanism is needed for checking whether the objective $p_1\,\mathcal{U}\,p_2$ is satisfied on all paths of an accepted tree. The main difficulty lies in the fact that the objective need not be observable by coalition A because neither p_1 nor p_2 are required to belong to $Prop_A$. Hence there can be behaviours ρ and ρ' such that $\rho'[0..i] \sim_A \rho[0..i]$ and (ρ,i) satisfies $p_1\,\mathcal{U}\,p_2$ but (ρ',i) does not.

Therefore, given some group of states R representing some history, we need to keep track of the subset R' of states in R for which the obligation $p_1\,\mathcal{U}\,p_2$ was not yet satisfied on their history. All the states in R' must be labeled with p_1, and we need to find outgoing transitions in the automaton that ensure the obligation to have $p_1\,\mathcal{U}\,p_2$ on all paths leaving R'. On the other hand, states in $R\smallsetminus R'$ are assumed to have histories in which $p_1\,\mathcal{U}\,p_2$ has been " achieved" in the past, and, therefore, are "free" from the obligation to fulfill $p_1\,\mathcal{U}\,p_2$.

Let $(q,S)\in\widehat{Q}$. Formally, the tree automaton is $\breve{\mathcal{A}}_{(q,S)} = (\breve{Q}, C_A, \breve{\delta}, \breve{Q}_0, \breve{\mathcal{F}})$ where:

- $\breve{Q}$ contains $\bot$, assumed to signal failure to fulfil $p_1\,\mathcal{U}\,p_2$, and all the sets of pairs (R_1,R_2) with:
  - $R_1 \subseteq R_2 \subseteq Q$,
  - $\forall r_1,r_2\in R_2, \lambda_A(r_1)=\lambda_A(r_2)$, and $\forall r_1\in R_1, p_2\notin\lambda(r_1)\wedge p_1\in\lambda(r_1)$,
- The initial state $\breve{Q}_0$ is defined by:
  1. if there exists $s\in S$ for which $\lambda(s)\cap\{p_1,p_2\}=\varnothing$ then $\breve{Q}_0=\bot$.
  2. otherwise, we denote $Q[p_2]=\big\{q\in Q\mid p_2\in\lambda(q)\big\}$ and we put $Q_0=\{(S\smallsetminus Q[p_2],S)\}$.
- $\breve{\delta}:\breve{Q}\times C_A\to 2^{\breve{Q}}\smallsetminus\varnothing$ is defined as follows: first, for any $c_A\in C_A$, $\delta((\bot,c_A))=\{\bot\}$. Then, for each $(R_1,R_2)\in\breve{Q}\smallsetminus\{\bot\}$ and $c_A\in C_A$, two situations may occur:
  1. If there exist $r_1\in R_1$, $(r,R)\in\widehat{Q}$ and $c\in C$ such that $(r_1,R_2)\overset{c}{\to}(r,R)\in\widehat{\delta}$, $c\big|_A=c_A$ and $\{p_1,p_2\}\cap\lambda(r)=\varnothing$, then $\delta\big((R_1,R_2),c_A\big)=\{\bot\}$.
  2. Otherwise,

$$\delta\big((R_1,R_2),c_A\big) = \big\{(out(R_1,c_A,Z)\smallsetminus Q[p_2], out(R_2,c_A,Z)) \mid Z\subseteq Prop_A, out(R_2,c_A,Z)\neq\varnothing\big\}$$

     That is, each transition from (R_1,R_2) labeled with c_A must embody sets of states representing all the variants of observations which occur as outcomes of the action tuple c_A from R_2, paired with the subset of states in which the $p_1\,\mathcal{U}\,p_2$ obligation is not fulfiled.
- The acceptance condition is

$$\breve{\mathcal{F}} = \big\{\mathcal{R}\mid\mathcal{R}\subseteq\breve{Q} \text{ with } (\varnothing,R)\in\mathcal{R}, \text{ for some } R\subseteq Q\big\}.$$

  That is, $\breve{\mathcal{A}}_{\widehat{q}}$ accepts only trees in which each path reaches some node containing the empty set as first state label.

Note that, in a pair (R_1,R_2) representing an element in $\breve{Q}$, the first component R_1 represents the subset of states of R_2 whose history has not yet accomplished $p_1\,\mathcal{U}\,p_2$. Hence, a tree node with label $(\varnothing,R)$ signals that the obligation $p_1\,\mathcal{U}\,p_2$ is accomplished for all histories ending in R.

Note also that, whenever the successors of (R_1,R_2) labeled c_A do not contain a state labeled by $\bot$, we have that, for any $Z\subseteq Prop_A$ and any $s\in out(R_2,c_A,Z)$, $p_1\in\lambda(s)$ or $p_2\in\lambda(s)$.

We may then prove the following result:

Proposition 8 *For any run $\widehat{\rho}\in\mathrm{Runs}^\omega(\widehat{\Gamma_A})$ and position i on the run for which $\rho[i]=\widehat{q}=(q,S)$,*

$$(\widehat{\Gamma_A},\rho,i)\vDash\langle\!\langle A\rangle\!\rangle p_1\,\mathcal{U}\,p_2 \text{ if and only if } L(\breve{\mathcal{A}}_{\widehat{q}})\neq\varnothing$$

Proof: ($\Rightarrow$) Suppose that $(\widehat{\Gamma_A}, \rho, i) \vDash \langle\!\langle A \rangle\!\rangle p_1 \mathcal{U} p_2$. Then, there exists $\sigma \in \Sigma(A, \widehat{\Gamma_A})$ such that for any $\rho' \in \mathrm{Runs}^\omega(\widehat{\Gamma_A})$ compatible with σ and for which $\rho'[0..i] \sim_A \rho[0..i]$ we have $(\widehat{\Gamma_A}, \rho', i) \vDash p_1 \mathcal{U} p_2$.

Let $t : \mathbb{N}^* \rightharpoonup \check{Q} \times C_A$ be a tree constructed recursively as follows:

- The root of the tree is $t(\varepsilon) = ((S \smallsetminus Q[p_2], S), c) \in \check{Q}_0$ where $c = \sigma(\lambda_A(\rho[0]) \dots \lambda_A(\rho[i]))$. Note that, by hypothesis, $\perp \notin \check{Q}_0$.

- Suppose we have build the tree up to level $j \geq 0$. Let $t(x) = ((R_1, R_2), c_A)$ be a node on the jth level, where $x \in \mathrm{supp}(t) \cap \mathbb{N}^j$. Consider some order on the set $\check{\delta}(t(x)) = \check{\delta}((R_1, R_2), c_A) = \{(R_1^1, R_2^1), \dots, (R_1^k, R_2^k)\}$ for some $k \geq 1$. The successors of $t(x)$ will be labeled with the elements of this set, each one in pair with an action symbol in C_A – action symbol which is chosen as follows: Denote $(x_p)_{1 \leq p \leq j}$ the initialized path in t which ends in x. For each $1 \leq l \leq k$, put

$$c_l = \sigma(\lambda_A(t(x_1)|_{\check{Q}}) \dots \lambda_A(t(x_k)|_{\check{Q}}) \lambda_A(R_2^l)).$$

  Then, for all $1 \leq l \leq k$ we put $t(xl) = ((R_1^l, R_2^l), c_l)$.

Suppose that $L(\check{\mathcal{A}}_{\widehat{q}}) = \varnothing$. This implies that t is not an accepting run in $\mathcal{A}$. Consequently, there exists an infinite path $\pi = (x_k)_{k \geq 0}$, where $x_k \in \mathbb{N}^k$, in t which does not satisfy any acceptance condition in $\check{\mathcal{F}}$. We have two cases:

1. π contains states different from $(\varnothing, R)$, for any $R \subseteq Q$, it reaches state $\perp$ and then loops in this state forever, or

2. π contains a cycle passing through states which are all different from $(\varnothing, R)$ or $\perp$, for any $R \subseteq Q$.

For the first case, let α be the length of the maximal prefix of π containing only states different from $\perp$. Let $t(x_k) = ((R_1^k, R_2^k), c_A^k)$, for any $0 \leq k < \alpha$. By the definition of t, we have that $\sigma(\lambda_A(R_2^0) \dots \lambda_A(R_2^k)) = c_A^k$, for any $0 \leq k < \alpha$.

Let $\rho' = \left((q_{k-1}, R_{k-1}) \xrightarrow{c_k} (q_k, R_k)\right)_{k \geq 1}$ be an infinite run in $\widehat{\Gamma_A}$ such that:

- $\rho'[0..i] \sim_A \rho[0..i]$ and $q_i \in R_1^0$,

- $q_{i+k} \in R_1^k$ and $R_{i+k} = R_2^k$, for all $\alpha > k \geq 1$,

- note that, by definition of π, $\check{\delta}((R_1^{\alpha-1}, R_2^{\alpha-1}), c_A^{\alpha-1}) = \perp$. We define $(q_{i+\alpha}, R_{i+\alpha}) \in \widehat{Q}$ such that $(q_{i+\alpha-1}, R_2^{\alpha-1}) \xrightarrow{c} (q_{i+\alpha}, R_{i+\alpha}) \in \widehat{\delta}$, for some $c \in C$ with $c|_A = c_A^{\alpha-1}$, and $\{p_1, p_2\} \cap \lambda(q_{i+\alpha}) = \varnothing$.

By definition of t, this run exists and it is compatible with σ. Also, starting with position i, ρ' contains a sequence of states labeled by p_1 but not by p_2 followed by a state which is not labeled by p_1 or p_2. Consequently, $(\widehat{\Gamma_A}, \rho', i) \not\vDash p_1 \mathcal{U} p_2$ which contradicts the hypothesis.

Similarly, for the second case above, we can construct a run ρ' in $\widehat{\Gamma_A}$ compatible with σ such that $\rho'[0..i] \sim_A \rho[0..i]$ and $(\widehat{\Gamma_A}, \rho', i) \not\vDash \Diamond p_2$. Consequently, $(\widehat{\Gamma_A}, \rho', i) \not\vDash p_1 \mathcal{U} p_2$ which contradicts the hypothesis.

($\Leftarrow$) Assume that t is a tree accepted by $\check{\mathcal{A}}_{\widehat{q}}$. We will construct inductively a strategy σ which is compatible with $\rho[0..i]$ and satisfies the required conditions for witnessing that $(\widehat{\Gamma_A}, \widehat{\rho}, i) \vDash \langle\!\langle A \rangle\!\rangle p_1 \mathcal{U} p_2$.

Suppose that the run ρ is $\rho = (\widehat{q_{j-1}} \xrightarrow{c_j} \widehat{q_j})_{j \geq 1}$. First, we may define σ for sequences of elements in 2^{Prop_A} of length at most i: for any A-history of length less than or equal to i, $w \in (2^{Prop_A})^*$, $|w| = j$ with $1 \leq j \leq i$, we put

$$\sigma(w) = \begin{cases} c_j|_A & \text{if } w = \widehat{\lambda_A}(\widehat{q_0}) \dots \widehat{\lambda_A}(\widehat{q_{j-1}}) \\ \text{arbitrary} & \text{otherwise} \end{cases}$$

For defining σ on sequences of length greater than i, let's denote first $w_{\widehat{\rho}} = \widehat{\lambda}_A(\widehat{q_0})\ldots\widehat{\lambda}_A(\widehat{q_{i-1}})$. Also, given a sequence of subsets of $Prop_A$, $z = (Z_1 \cdot \ldots \cdot Z_k) \in (2^{Prop_A})^*$ and a node $x \in \mathrm{supp}(t)$, we say that z *labels a path from ε to x in t* if the A-history along the path from ε to x in t is exactly z, that is,

$$\forall y \leq x, \forall 0 \leq j \leq |x|, \text{ if } |y| = j \text{ then } \widehat{\lambda}_A\big(t(y)\big|_1\big) = Z_{j+1}.$$

Then, for all $k \geq 1$ we put:

$$\sigma(w_{\widehat{\rho}} Z_1 \ldots Z_k) = \begin{cases} t(x)\big|_2 & \text{if } (Z_1,\ldots,Z_k) \text{ labels a path from } \varepsilon \text{ to } x \text{ in } t \\ \text{arbitrary} & \text{otherwise} \end{cases}$$

To prove that σ is a strategy that witnesses for $(\widehat{\Gamma_A},\widehat{\rho},i) \vDash \langle\!\langle A \rangle\!\rangle p_1 \mathcal{U} p_2$, take some run ρ' compatible with σ and for which $\rho'[0..i] \sim_A \rho[0..i]$. We may prove that, if we denote the run as $\rho' = (\widehat{q'_{j-1}} \xrightarrow{c'_j} \widehat{q'_j})_{j \geq 1}$, with $\widehat{q'_j} = (r_j, S_j) \in \widehat{Q}$, and we also denote $Z_j = \widehat{\lambda}_A(\widehat{q'_j})$, then:

- there exists a path $(x_{j-i})_{j \geq i}$ in t with $t(x_{j-i})\big|_1 = (R_1^j, S_j)$ and $t(x_0)\big|_1 = (R_1^0, S_i)$, for some $R_1^k \subseteq Q$, $0 \leq k$.

- for all $j \geq i+1$, $c'_j\big|_A = \sigma(Z_0\ldots Z_{j-1}) = t(x_{j-i-1})\big|_2$.

This property follows by induction on j, and ends the proof of our theorem. $\dashv$

Case $\langle\!\langle A \rangle\!\rangle p_1 \mathcal{W} p_2$: The construction is almost entirely the same as for the previous case, the only difference being the accepting condition. For this case, the condition from the until case is relaxed: any path of an accepting tree may still only have labels of the type (R_1, R_2) denoting the fact that all the runs that are simulated by the path and lead to a member of R_1 are only labeled with p_1. But we no longer require that, on each path, a label of the type $(\varnothing, R)$ occurs. This is due to the fact that $p_1 \mathcal{W} p_2$ does not incorporate the obligation to reach a point where p_2 holds, runs on which p_1 holds forever are also acceptable.

So, formally, the construction for $\langle\!\langle A \rangle\!\rangle p_1 \mathcal{W} p_2$ is the following: $\widetilde{\mathcal{A}}_{(q,S)} = (\breve{Q}, C_A, \breve{\delta}, \breve{Q}_0, \widetilde{\mathcal{F}})$ where $\breve{Q}, \breve{Q}_0$ and $\breve{\delta}$ are the same as in the construction for $\langle\!\langle A \rangle\!\rangle p_1 \mathcal{W} p_2$, while the acceptance condition is the following:

$$\widetilde{\mathcal{F}} = \{\mathcal{R} \mid \mathcal{R} \subseteq \breve{Q}\}.$$

The following result can be proved similarly to Proposition 8.

Proposition 9 *For any run $\widehat{\rho} \in \mathrm{Runs}^\omega(\widehat{\Gamma_A})$ and position i on the run for which $\rho[i] = \widehat{q} = (q, S)$,*

$$(\widehat{\Gamma_A}, \rho, i) \vDash \langle\!\langle A \rangle\!\rangle p_1 \mathcal{W} p_2 \text{ if and only if } L(\widetilde{\mathcal{A}}_{\widehat{q}}) \neq \varnothing$$

Example 10 *For our running example, the tree automaton constructed from the arena $\widehat{\Gamma_{Alice,Bob}}$ (given in Example 7), for the state $(q_0, \{q_0\})$ and the formula $\phi_1 = \langle\!\langle \{Alice, Bob\} \rangle\!\rangle \Diamond (c \wedge s)$ is pictured in Figure 2. Note that it accepts an infinite tree such that all its paths contain the state $(\varnothing, \{q_{12}\})$ but never reach $\bot$. Moreover, this tree defines a strategy for the coalition $\{Alice, Bob\}$ to reach the goal $c \wedge s$.*
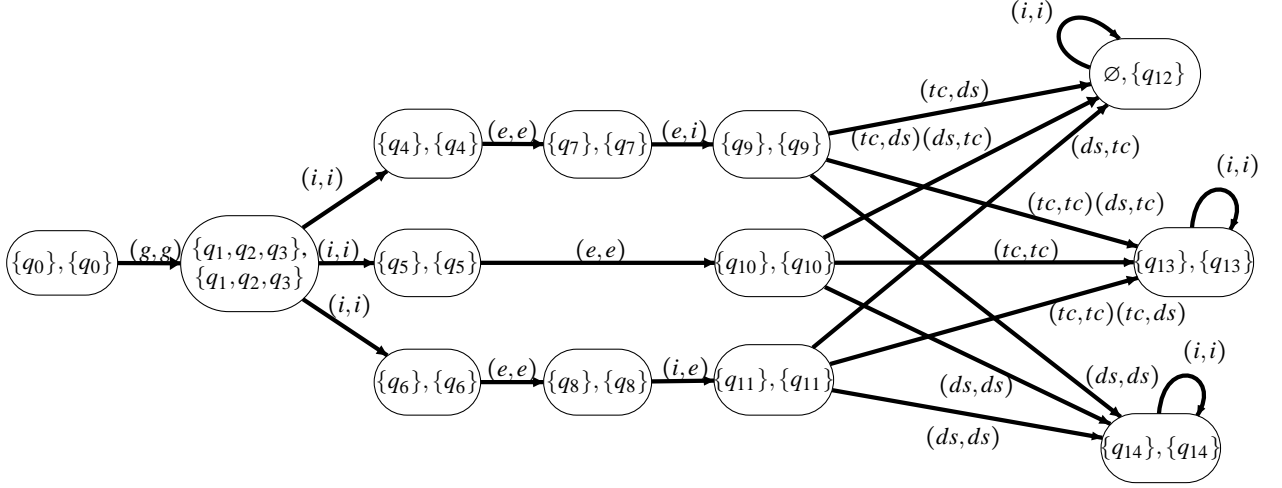
Figure 2: A tree automaton for the game arena in Figure 1

4.3 The model-checking algorithm

Our algorithm for the model-checking problem for ATL_{iR}^{D} works by structural induction on the formula ϕ to be model-checked. The input of the algorithm is a game arena $\Gamma = (Q, C, \delta, Q_0, Prop, \lambda)$ and an enumeration $\Phi = \{\phi_1, \ldots, \phi_n\}$ of the subformulas of ϕ, in which $\phi = \phi_n$ and ϕ_i is a subformula of ϕ_j only if $i < j$. The algorithm determines whether ϕ holds at all the initial states of Γ. It works by constructing a sequence of arenas $\Gamma_k = (Q_k, C, \delta_k, Q_0^k, Prop_k, \lambda_k)$, $k = 0, \ldots, n$, with $\Gamma_0 = \Gamma$. The formula ϕ is assumed to be written in terms of the agents from Ag and the atomic propositions from $Prop = \bigcup_{a \in Ag} Prop_a$ of Γ. The atomic propositions of $\Gamma_1, \ldots, \Gamma_n$ include those of Γ and n fresh atomic propositions p_{ϕ_k}, $k = 1, \ldots, n$, which represent the labelling of the states of these arenas by the corresponding formulas ϕ_k. For any $1 \le k \le n$, upon step k the algorithm constructs Γ_k from Γ_{k-1} and calculates the labelling of its states with formula ϕ_k. $Prop_k = Prop \cup \Phi_k$ where Φ_k denotes $\{p_{\phi_1}, \ldots, p_{\phi_k}\}$, $k = 0, \ldots, n$. The state labelling function λ_k is defined so that equivalence $p_{\phi_k} \Leftrightarrow \phi_k$ is valid in Γ_k. Therefore, we define the formula $\chi_k = \phi_k [\phi_{k-1}/p_{\phi_{k-1}}], \ldots, [\phi_1/p_{\phi_1}]$ which has at most one connective of the form $\langle\!\langle A \rangle\!\rangle \bigcirc$, $\langle\!\langle A \rangle\!\rangle \mathcal{U}$, $\langle\!\langle A \rangle\!\rangle \mathcal{W}$ or K_A. The algorithm computes the states that should be labeled by p_{ϕ_k} using the formula χ_k which is equivalent to ϕ_k. The fresh propositions $p_{\phi_1}, \ldots, p_{\phi_n}$ are not assumed to be observable by any particular agent. Therefore the requirements $Prop_k = \bigcup_{a \in Ag} Prop_{a,k}$ on arenas are not met by $\Gamma_1, \ldots, \Gamma_n$, but this is of no consequence.

Let us note the need to switch, at each step, from analyzing Γ_k to analyzing Γ_{k+1}. This is needed as Γ_{k+1} only has the necessary information about the identically-observable histories, needed in the semantics of coalition operators.

In case ϕ_k is atomic, $\Gamma_k = (Q_{k-1}, C, \delta_{k-1}, Q_0^{k-1}, Prop_k, \lambda_k)$ where $\lambda_k(q) \cap Prop_{k-1} = \lambda_{k-1}(q)$ and $p_{\phi_k} \in \lambda_k(q)$ iff $\phi_k \in \lambda_{k-1}(q)$. In case ϕ_k is not atomic, the construction of Γ_k depends on the main connective of χ_k:

1. Let χ_k be a boolean combination of atoms from $Prop_{k-1}$. Then $\Gamma_k = (Q_{k-1}, C, \delta_{k-1}, Q_0^{k-1}, Prop_k, \lambda_k)$ where $\lambda_k(q) \cap Prop_{k-1} = \lambda_{k-1}(q)$ and $p_{\phi_k} \in \lambda_k(q)$ iff the boolean formula $\bigwedge_{p \in \lambda_{k-1}(q)} p$ implies χ_k.

2. Let χ_k be $K_A p$ for some $p \in Prop_{k-1}$. Consider the arena $\widehat{(\Gamma_{k-1})_A}$ defined as in Subsection 4.1. Then $\Gamma_k = (\widehat{Q}_{k-1}, C, \widehat{\delta}_{k-1}, \widehat{Q}_0^{k-1}, Prop_k, \lambda_k)$ where $\lambda_k(q) \cap Prop_{k-1} = \widehat{\lambda}_{k-1}(q)$ and $p_{\phi_k} \in \lambda_k(q)$ iff $q \in \widehat{Q}_{k-1}^{K_A p}$, where $\widehat{Q}_{k-1}^{K_A p}$ is defined in (1).

3. Let χ_k be $\langle\!\langle A \rangle\!\rangle \bigcirc p$ for some $p \in Prop_{k-1}$. Consider $\widehat{(\Gamma_{k-1})_A}$. Then $\Gamma_k = (\widehat{Q}_{k-1}, C, \widehat{\delta}_{k-1}, \widehat{Q}_0^{k-1}, Prop_k, \lambda_k)$ where $\lambda_k(q) \cap Prop_{k-1} = \widehat{\lambda}_{k-1}(q)$ and $p_{\phi_k} \in \lambda_k(q)$ iff $q \in \widehat{Q}_{k-1}^{\langle\!\langle A \rangle\!\rangle \bigcirc p}$, where $\widehat{Q}_{k-1}^{\langle\!\langle A \rangle\!\rangle \bigcirc p}$ is defined in (3).

4. Let χ_k be $\langle\!\langle A \rangle\!\rangle p_1 \mathcal{U} p_2$ for some $p_1, p_2 \in Prop_{k-1}$. Consider $\widehat{(\Gamma_{k-1})_A}$ again and, for each state $\widehat{q} \in \widehat{Q}_{k-1}$, construct the tree automaton $\check{\mathcal{A}}_{\widehat{q}}$. Then put $\Gamma_k = (\widehat{Q}_{k-1}, C, \widehat{\delta}_{k-1}, \widehat{Q}_0^{k-1}, Prop_k, \lambda_k)$ where $\lambda_k(q) \cap Prop_{k-1} = \widehat{\lambda}_{k-1}(q)$ and $p_{\phi_k} \in \lambda_k(q)$ iff $L(\check{\mathcal{A}}_{\widehat{q}}) \neq \varnothing$.

5. Finally, let χ_k be $\langle\!\langle A \rangle\!\rangle p_1 \mathcal{W} p_2$ for some $p_1, p_2 \in Prop_{k-1}$. Consider $\widehat{(\Gamma_{k-1})_A}$ again and, for each state $\widehat{q} \in \widehat{Q}_{k-1}$, construct the tree automaton $\widetilde{\mathcal{A}}_{\widehat{q}}$. Then put $\Gamma_k = (\widehat{Q}_{k-1}, C, \widehat{\delta}_{k-1}, \widehat{Q}_0^{k-1}, Prop_k, \lambda_k)$ where $\lambda_k(q) \cap Prop_{k-1} = \widehat{\lambda}_{k-1}(q)$ and $p_{\phi_k} \in \lambda_k(q)$ iff $L(\widetilde{\mathcal{A}}_{\widehat{q}}) \neq \varnothing$.

The following result is a direct consequence of Propositions 5, 8, and 9.

Theorem 11 *Let $\Gamma_n = (Q_n, C, \delta_n, Q_0^n, Prop_n, \lambda_n)$ be the last game arena obtained in the algorithm described above. Then,*

$$p_\phi \in \lambda_n(q), \text{ for all states } q \in Q_0^n \quad \textit{iff} \quad (\Gamma, \rho, 0) \vDash \phi, \text{ for all runs } \rho \in \mathsf{Runs}^\omega(\Gamma).$$

5   Concluding remarks

We have presented a model-checking technique for ATL$_{iR}^D$, a variant of the Alternating Temporal Logic with Knowledge, in which coalitions may coordinate their actions, based on their distributed knowledge of the system state. The technique is based on a state labeling algorithm which involves tree automata for identifying states to be labeled with cooperation modality subformulas, and a state splitting construction which serves for identifying (finite classes of) histories which are indistinguishable to some coalition.

According to our semantics, while distributed knowledge is used for constructing coalition strategies, it is assumed that the individual agents in the coalition gain no access to that knowledge as a side effect of their cooperation. That is why the proposed semantics corresponds to coalitions being organised under *virtual supervisors* who guide the implementation of strategies by receiving reports on observations of the coalitions' members and, in return, just directing the members' actions without making any other knowledge available to them.

The possibility of a subsequent increase of individual knowledge as a side effect of the use of distributed knowledge for coordinated action, which we avoid by introducing virtual supervisors, becomes relevant only in settings such as that of ATL with incomplete information. This possibility appears to be an interaction between the understanding of distributed knowledge as established in non-temporal epistemic logic and temporal settings. This is just one of the numerous subtle interpretation issues which were created by the straightforward introduction of modalities from non-temporal epistemic logic and cooperation modalities into temporal logics. For an example of another such issue, a semantics for ATL in which agents, once having chosen a strategy for achieving a certain main goal, cannot revise it upon considering the reachability of subgoals, was proposed and studied in [ÅGJ07].

The state labeling algorithm suggests that tree automata with partial observations and with partially-observable objectives might be useful to study. We believe that the two state-labeling constructions can be generalized to such automata, giving us also a decision method for the "starred" version of ATL$_{iR}^D$.

Acknowledgments

References

[ÅGJ07]   Th. Ågotnes, V. Goranko, and W. Jamroga. Alternating-time temporal logics with irrevocable strategies. In *Proceedings of TARK'07*, pages 15–24, 2007.

[AHK98]   R. Alur, Th. Henzinger, and O. Kupferman. Alternating-time temporal logic. In *Proceedings of COMPOS'97*, volume 1536 of *LNCS*, pages 23–60. Springer Verlag, 1998.

[AHK02]   R. Alur, Th. Henzinger, and O. Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49(5):672–713, 2002.

[BJ09]   N. Bulling and W. Jamroga. Model checking ATL$^+$ is harder than it seemed. Technical Report IfI-09-13, 2009.

[CDHR06]   K. Chatterjee, L. Doyen, Th.A. Henzinger, and J.-F. Raskin. Algorithms for omega-regular games with imperfect information. In *Proceedings of CSL'06*, volume 4207 of *Lecture Notes in Computer Science*, pages 287–302. Springer, 2006.

[Dim08]   C. Dima. Revisiting satisfiability and model-checking for CTLK with synchrony and perfect recall. In *Proceedings of the 9th International Workshop on Computational Logic in Multi-Agent Systems (CLIMA IX)*, volume 5405 of *LNAI*, pages 117–131, 2008.

[Dim10]   C. Dima. Non-axiomatizability for linear temporal logic of knowledge with concrete observability, 2010. Submitted.

[FHMV04]   R. Fagin, J. Halpern, Y. Moses, and M. Vardi. *Reasoning about knowledge*. The MIT Press, 2004.

[GD08]   D.P. Guelev and Catalin Dima. Model-checking strategic ability and knowledge of the past of communicating coalitions. In *Proceedings of DALT 2008*, volume 5397 of *Lecture Notes in Computer Science*, pages 75–90. Springer, 2008.

[GJ04]   V. Goranko and W. Jamroga. Comparing semantics of logics for multi-agent systems. *Synthese*, 139(2):241–280, 2004.

[GvD06]   V. Goranko and G. van Drimmelen. Complete axiomatization and decidability of alternating-time temporal logic. *Theoretical Computer Science*, 353(1-3):93–117, 2006.

[JA07]   W. Jamroga and Th. Agotnes. Constructive knowledge: What agents can achieve under imperfect information. *Journal of Applied Non-Classical Logics*, 17(4):423–475, 2007.

[KP05]   M. Kacprzak and W. Penczek. Fully symbolic unbounded model checking for alternating-time temporal logic. *Autonomous Agents and Multi-Agent Systems*, 11(1):69–89, 2005.

[LMO08]   Fr. Laroussinie, N. Markey, and Gh. Oreiby. On the expressiveness and complexity of ATL. *Logical Methods in Computer Science*, 4(2), 2008.

[Sch04]   P.-Y. Schobbens. Alternating-time logic with imperfect recall. *Electronic Notes in Theoretical Computer Science*, 85(2):82–93, 2004.

[Tho97]   W. Thomas. Languages, automata, and logic. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 3, Beyond Words, pages 389–455. Springer Verlag, 1997.

[vdHLW06]   W. van der Hoek, A. Lomuscio, and M. Wooldridge. On the complexity of practical ATL model checking. In *Proceedings of AAMAS 2006*, pages 201–208. ACM, 2006.

[vdHW03]   W. van der Hoek and M. Wooldridge. Cooperation, knowledge, and time: Alternating-time temporal epistemic logic and its applications. *Studia Logica*, 75(1):125–157, 2003.