

Expressing and Verifying Temporal and Structural Properties of Mobile Agents

Marek A. Bednarczyk*

Institute of Computer Science, Polish Academy of Sciences, Gdańsk, Poland
m.bednarczyk@ipipan.gda.pl

Wojciech Jamroga

Chair of Computational Intelligence, Clausthal University of Technology, Germany
wjamroga@in.tu-clausthal.de

Wiesław Pawłowski

Institute of Computer Science, Polish Academy of Sciences, Gdańsk, Poland
w.pawlowski@ipipan.gda.pl

Abstract. Logics for expressing properties of *Petri hypernets*, a visual formalism for modelling mobile agents, are proposed. Two classes of properties are of interest—the temporal evolution of agents and their structural correlation. In particular, we investigate how the classes can be combined into a logic capable of expressing the dynamic evolution of the structural correlation. The problem of model checking properties of a class of the logic on Petri hypernets is shown to be PSPACE-complete.

Keywords: dynamic mobile agents, Petri hypernets, temporal logics, model checking.

1. Introduction

Petri hypernets provide a new formalism for modelling mobile dynamic agents, see [1]. Hypernets, due to their Petri net heritage, offer a *visual* framework in which *individual* dynamic agents are represented. The basic assumption—*all agents are distinguishable entities*—sets Petri hypernets apart from many multi-agent formalisms.

A Petri hypernet is a collection of agents, where each individual agent is represented as a single Petri net. The collection is usually structured—some agent nets are being contained in other nets as their

*Address for correspondence: Institute of Computer Science, Polish Academy of Sciences, Gdańsk, Poland

resources. Intuitively, this leads to a *hierarchy* of agents controlling other agents. Some actions of an agent within a hypernet may be performed asynchronously. Other actions require synchronisation among agents. The above features are shared by hypernets with the **nets-within-nets** formalisms developed by Valk and his followers. Unlike in object nets, cp. [24], the evolution of the hypernet may change the position of agents within the overall hierarchy, like in the case of mobile ambients, cp. [5].

Here, we propose a logical language for reasoning about systems representable within the framework of hypernets. The language combines two families of modal operators—one family to cope with the temporal, the other to deal with the spatial (or structural) dimension. From this perspective, our approach follows [12]. Unlike Franceschet et al., however, we do not start with two logics to cope with each dimension separately, and look what we get from their combination. Instead, we interpret the language of the combination directly in a class of Kripke structures subsuming Petri hypernets. Only then we approach the problem of how the resulting logic can be seen as a combination of its two components.

For the purpose of the presentation we have chosen to represent both the temporal and the structural properties of agents in the style of Computational Tree Logic CTL. Dynamic evolution of a hypernet is represented by a transition system. Consequently, one copy of CTL is used to explain how the system changes its states. The states of hypernets are structured, e.g., they contain the information about the current hierarchy. To refer to the current agent’s sub-agents we use the classical “future tense”-like operators, while referring to its supervisors we use “past tense”-like modalities. Thus, the other copy of CTL has both types of modalities. It is a feature of Petri hypernets that the hierarchy of agents, while possibly changing from state to state, always remains a *tree*. For this reason the stronger logic brings no computational cost. The semantics of the combined logic is given with respect to a class of abstract intermediate models called agent-oriented hyper-transition systems—rather than defined directly over hypernets.

The paper is organised as follows. We begin with a short introduction to Petri hypernets and their basic properties in Section 2, and we demonstrate the ideas on an air traffic modelling example. Then, we discuss some examples of interesting tempo-structural properties that can be specified (and verified) for systems which structure of individuals evolves in time. In Sections 3.3 and 3.4, we present our intermediary models (agent-oriented hyper-transition systems), and define a logic called CTL², that we use for describing temporal and structural properties of hypernets. In section 3.5 we explain how CTL² can be seen as a combination of two logics. In Section 4, the complexity of model checking for such a combination of logics is studied. Like CTL, also CTL² can be model-checked in time linear in the length of the formula and the size of the agent-oriented hypertransition system. Moreover, the model checking problem for CTL² turns out to be PSPACE-complete in the length of the formula and the size of the *Petri hypernet* itself. Finally, we discuss limitations of the approach presented here and suggest directions for future research.

Preliminary versions of this paper appeared as [3], see also the extended version [4].

2. Hypernets — a Visual Formalism for Mobile Agents

The notion of nets introduced by C. A. Petri in 1962 offers a fundamental model of concurrent computations. It is based on two spatially distributed components: *places* and *transitions*. The places are local states in which some *resources* can be stored. Distribution of resources among the places corresponds to a global state of the net, called *marking*. The transitions can fetch the resources from some places and

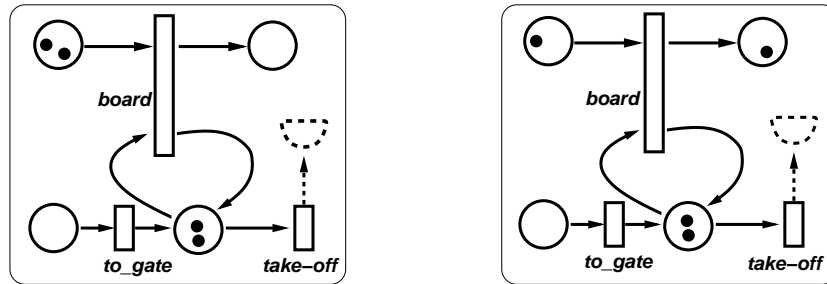


Figure 1. Boarding a plane: before (left) and after (right) event *board*.

transport them to other places—thereby changing the global state of the net. The interaction is *static*, *finitary* and *local*, i.e., each transition has a fixed and finite set of places with which it interacts, and these constitute its local environment. The structure of a net is usually represented as a directed bipartite graph, with transitions drawn as boxes, places as circles, and resources as blobs, see monograph [21] for more on Petri nets. Figure 1, for example, presents a net in two states: before and after *firing* a transition. The marking represented on the left *enables* transition *board* in several ways. That is, all the *precondition* places of *board* contain resources. Any so enabled transition can *fire*, which amounts to transporting a resource from each precondition place to every postcondition place. The result of firing the transition is depicted on the right of Figure 1.

Since their inception, Petri nets have been generalised in many ways. Initially, the places were considered to be *facts*, e.g., in the case of *elementary* or *C/E nets*. That is, a single resource at a place would establish it, its lack would falsify it. Later, the resources also took the role of *quantities*: either natural numbers in the case of *P/T nets*, or non-negative reals in the case of *continuous* nets. In some generalisations, the resources are structured, e.g., they can be elements of some algebra or a data-type.

2.1. Petri Hypernets

In Petri hypernets, see [1, 2], *agents* are represented as structured Petri nets, and can occur as tokens in places of other agents. Thus, Petri hypernets (PHN in short) offer a *visual* formalism for modelling distributed and concurrent dynamic agent systems. Formally, a hypernet $H = \langle \mathcal{N}, m \rangle$ consists of a set \mathcal{N} of agents and a *hypermarking*. An agent $N \in \mathcal{N}$, called *open net*, consists of several *modules* which are synchronised on transitions. To ensure the preservation of the identity of agents each module has the structure of a state machine. The places of N , denoted P_N , are not shared with any other $N' \in \mathcal{N}$. We let $P_H = \bigcup_{N \in \mathcal{N}} P_N$ denote the set of places of H . A hypermarking $m : \mathcal{N} \rightarrow P_H$ is a partial function that describes the current distribution of nets as resources of other nets.

The idea to use nets as tokens has appeared quite early, and is now known as the *nets-within-nets* approach, see [24]. Intuitively, each hypermarking induces a *hierarchy*—an open net N is *lower* in the hierarchy than its *owner*, i.e., the net to which the place $m(N)$ belongs, if defined. Nets with $m(N)$ undefined are maximal in the hierarchy. In Valk’s elaboration, each token net can move between places of its current owner, but there is no proviso for token exchange among agents. Thus, a token net is bound to its owner. Agents organised in this way have a fixed, static hierarchy. Hypernets, in contrast, are capable of dynamically changing that hierarchy. In this respect they are like *mobile ambients* [5].

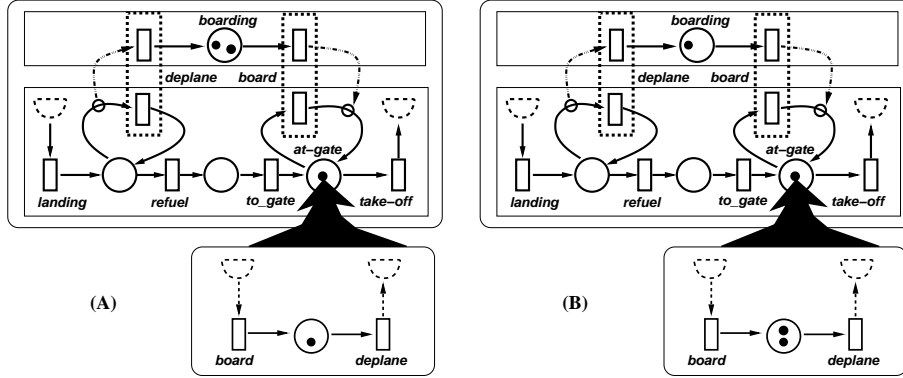


Figure 2. Airport, plane, and 3 passengers. Boarding a passenger: (A) before, (B) after.

In order to allow migration of agents from one owner to another an old concept of message passing is used in [1]. The idea is demonstrated in Figure 2A. Here, the structure of two agents: an airport on top, and a plane at the bottom is shown. The airport agent consists of two modules. The bottom one is for handling planes, and the plane agent is indeed located in one of the places. The top module is for handling passengers. The plane agent has just one module for passenger handling. There are two passengers at the airport, and one in the plane. This distribution of agents defines a hypermarking.

In hypernets, each agent can communicate with its current owner (if any), and with the agents it currently owns. To this end, some of its places are designated as *communication ports*. For instance, the airport agent in Figure 2A is *always* ready to accept a new plane agent from its hypothetical owner by engaging in transition *landing*. Similarly, the plane agent is always prepared to receive a traveller from its owner by engaging in *board* transition. Both agents offer also dual operations for sending planes (resp. travellers), up the hierarchy tree, with transitions *take-off* (resp. *deplane*). The ports for communicating with the owner are usually depicted as dashed semicircles.

Communication with the owner is simple—there is at most one assigned by the hypermarking. To send a token down, we nominate a module responsible for choosing the addressee agent. In our example, transition *board* in the traveller module of the airport has its output port attached to the output arc from the *board* transition in the plane module of the airport. Thus, the traveller token chosen for boarding in the traveller module will be sent down to the plane token chosen in the plane module for boarding, as shown in Figure 2B. Agents stay within modules of the same kind, e.g. a traveller can never enter a plane-handling module of another agent. We assume that the ownership relation induced by the initial hypermarking is a tree, and that there is only one *super-agent* containing all the other agents. It turns out that the evolution of hypernets preserves the tree-like character of the hierarchy of agents. A more detailed discussion of Petri hypernets can be found in [1].

Remark 2.1. Let us remark that the number of hypermarkings reachable from the initial one is more than exponential in general. Assume that there are k traveller agents at place *boarding* of the airport agent in Figure 2, and ℓ planes ready to pick them up at place *at-gate*. Then each traveller can board any plane, which gives at least ℓ^k different hypermarkings. The travellers are represented by empty hypernets; the planes also have constant size (cf. Figure 2). Let n be the number of places in the whole hypernet. Taking $k = \ell = n/c$ for a suitable constant c gives us at least $2^{O(n \log n)}$ states in the resulting

transition system. Conversely, we can distribute $k + \ell$ agents among their n places in at most $n^{\ell+k}$ ways, which proves $2^{O(n \log n)}$ to be an upper bound too.

3. Properties of Agents in Petri Hypernets

Hypernets seem to provide a natural formalism for modelling and designing systems of mobile agents. Hypernets will remain, however, only a modelling tool until a language is developed in which one can express properties of agents. Here, we focus on two dimensions of such a specification formalism. The *temporal* dimension captures how the system evolves in time. The *spatial*, or *structural* dimension refers to the agents' position within other agents. From this perspective, our approach resembles [12].

A hypermarking in a hypernet associates to each token net a local place of its owner. Here, we neglect such details and concentrate on the induced information about the subsumption of agents to other agents, i.e., the tree hierarchies of agents. It is important to keep in mind that the evolving subsumption hierarchy among agents is always a tree, cp. [1].

Remark 3.1. In order to make basic statements concerning the two aspects of multi-agent systems, we assume that we deal with *labeled* Petri hypernets, in which agents and places can be labelled with the names of atomic propositions. Namely, we assume that the set Π of atomic propositions is partitioned into two disjoint sets: Π_{ag} (agent labels) and Π_{pl} (place labels), $\Pi = \Pi_{ag} \cup \Pi_{pl}$, and that there are two labelling functions given $\lambda_{ag} : \Pi_{ag} \rightarrow \mathcal{P}(\mathcal{N})$ and $\lambda_{pl} : \Pi_{pl} \rightarrow \mathcal{P}(P_H)$. *Agent proposition* $p_{ag} \in \Pi_{ag}$ will hold in the context of each agent labelled with p_{ag} in H (e.g., proposition `airportGD` can be used to designate the Gdańsk airport). *Place proposition* $p_{pl} \in \Pi_{pl}$ will hold in the context of a hypermarking in which every place labelled with p_{pl} is inhabited by some token (e.g., proposition `gate7` can be used to designate the situations in which the gate no. 7 is non-empty). See also Section 4.1.

3.1. Temporal Evolution of Systems

There are a number of modal logics that address how a system can evolve in time [9]. The temporal aspect of the system is usually modelled with a *transition system*, in which nodes represent possible situations or *states* of the system, and arcs or *transitions* show how states can change in a single step. In the case of a PHN, states can be naturally thought of as hypermarkings; to determine the outgoing transitions for state q representing hypermarking m , we take arcs to all the states that represent hypermarkings that can be obtained from m by firing a single transaction. Several operators are used to capture temporal properties in transition systems: \bigcirc (*next*), \diamond (*sometime*), \square (*always*) and \mathcal{U} (*until*). Furthermore, there are many alternative courses of action (paths) that can actually happen in the future. The linear-time temporal logic LTL treats each path as a separate alternative model of time [14]. The branching-time logic CTL [7, 8, 9] collects all possible paths in a single model, and adds explicit *path quantifiers*: A (*for all paths*) and E (*there is a path*) to the language.

In this paper, we focus on CTL-like formalisms, with their explicit way of addressing tree-like semantic structures. CTL comes in two variants: in “vanilla” CTL, every occurrence of a temporal operator is preceded by exactly one path quantifier. In CTL* no such restriction is imposed. CTL* is more expressive of the two (it strictly subsumes both CTL and LTL), but the “vanilla” version has slightly simpler semantics and some nice computational properties (e.g. model checking is linear in the size of the model

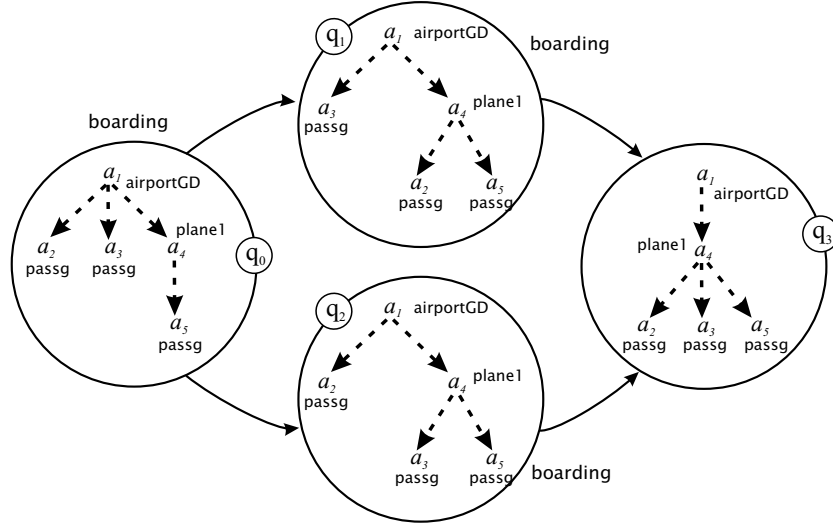


Figure 3. A hierarchy of agents evolving in time

and the length of the formula). Of course, one may use LTL, CTL*, or a more expressive logic like ATL* or μ -calculus instead. Essentially, the choice is a matter of taste, expressive power and complexity.

3.2. Structural Properties of Systems

Temporal logic allows for reasoning about possible evolution of abstract properties of systems that can be addressed through atomic propositions. In the context of mobile agents, agents locations and overall structure is also very interesting; if the system may evolve in time, we are interested in the structure of agents at each given moment. In the case of Petri hypernets, one can think here about properties that relate to the current subsumption structure of agents. However, rather than refer to such statements via atomic propositions, it is more practical and elegant to express the relevant structural properties by modal means. Moreover, the agents' subsumption structure defined by a hypermarking in a PHN is always a *tree*, which suggests that a CTL-like logic can be a good formal tool to capture its properties. To this end, we define new path quantifiers E_s (*there is a subsumption path*) and A_s (*for all subsumption paths*), plus spatial operators: \odot (*the agent one level down in the hierarchy*), \square (*all agents down, including the current agent*), \downarrow (*all agents down until some property holds*), \ominus (*the agent one level up*), \square (*all agents up, including the current agent*), and \uparrow (*all agents up until some property holds*), analogous to the temporal operators of CTL.

Note that in the logic of subsumption one should be able to reason not only about the “owned” agents, but also about the “owners”. Adding past tense-style operators can, in general, pose complexity problems with respect to CTL model checking, cf. [23]. However, our subsumption hierarchies are always finite trees, so no additional computational cost is involved.

Note also that, for every agent a , there is always exactly one maximal path upwards. Consequently, superposition path quantifiers are redundant.

3.3. Combining the Temporal and Structural Dimensions

In a Petri hypernet, every hypermarking defines a global state of the system, and induces subsumption structure among agents. As the hypermarking changes (because a transaction has been fired), the subsumption hierarchy may change as well (see Figure 3). The picture is similar to the well-known BDI logics [20]. There, temporal structures were embedded in epistemic positions of agents. Here, spatial agent structures are embedded in temporal positions of the system (i.e. states). It has been shown by Schild [22] that the semantics of BDI can be equivalently defined in terms of conventional Kripke structures with two binary modal relations, on which some structural restrictions are imposed. We follow that idea and introduce a class of models analogous to Schild's situation structures, that can be used to represent the temporal and structural dimensions of agents in a PHN. We define *agent-oriented hyper-transition systems* (AOHS in short) as 6-tuples: $M = \langle Q, q_0, \text{Agt}, \mathcal{R}, \mathcal{S}, \pi \rangle$, where:

- Q is a nonempty set of (temporal) *states* of the system, and $q_0 \in Q$ is a distinguished *initial state*.
- Agt is a nonempty, finite set of *agents*.
- The set of possible worlds, or *agent-oriented states*, is defined as $\Omega = Q \times \text{Agt}$.
- $\mathcal{R}, \mathcal{S} \subseteq \Omega \times \Omega$ are two modal relations. \mathcal{R} is the *temporal relation* that defines possible transitions between states, $\langle q, a \rangle \mathcal{R} \langle q', a' \rangle$ meaning that agent a at state q can evolve in a single step into agent a' at state q' . $\langle q, a \rangle \mathcal{S} \langle q', a' \rangle$ means that agent a at state q subsumes (or controls) agent a' at state q' . We require relations \mathcal{R}, \mathcal{S} to satisfy the following structural conditions:
 1. $\langle q, a \rangle \mathcal{S} \langle q', a' \rangle \Rightarrow q = q'$: agents subsume each other *within states*,
 2. $\langle q, a \rangle \mathcal{R} \langle q', a' \rangle \Rightarrow a = a'$: \mathcal{R} models evolution of each agent *separately*,
 3. $\langle q, a_1 \rangle \mathcal{R} \langle q', a_1 \rangle \Rightarrow \langle q, a_2 \rangle \mathcal{R} \langle q', a_2 \rangle$: the temporal evolution is the same for all agents (time is global).
 4. \mathcal{S} forms a tree at every temporal state.
- Finally, $\pi : \Pi \rightarrow \mathcal{P}(\Omega)$ is a valuation of atomic propositions from a given set Π . Thus, for every proposition $p \in \Pi$, $\pi(p)$ defines the set of agent-oriented states in which p holds.

The notion of a *reachable* world is defined as usual. Note that, due to (3), for every reachable agent-oriented state $\langle q, a \rangle$, we have $\langle q, a' \rangle$ also being reachable for any agent a' . This captures the “no creation, no destruction of agents” principle. Figure 4 presents an AOHS that models the evolving hierarchy of agents from Figure 3. We will refer to this transition system as M_1 in further examples.

A ρ -*path* in M is a sequence of agent-oriented states that follow relation ρ , that is, a sequence $\omega_0 \omega_1 \omega_2 \dots$ such that $\omega_i \rho \omega_{i+1}$ for every $i = 0, 1, 2, \dots$. In what we consider, ρ can be one of the three relations: $\mathcal{R}, \mathcal{S}, \mathcal{S}^{-1}$ (thus, we can have paths that refer to an agent's evolution in time, agent subsumption chains, and agent superposition chains). A *full ρ -path*, or *run*, is a path that is either infinite or ends with a state ω that is blocked: $\omega \rho \omega'$ for no ω' . Finally, we denote the i th state in path Λ by $\Lambda[i]$ (starting from $i = 0$).

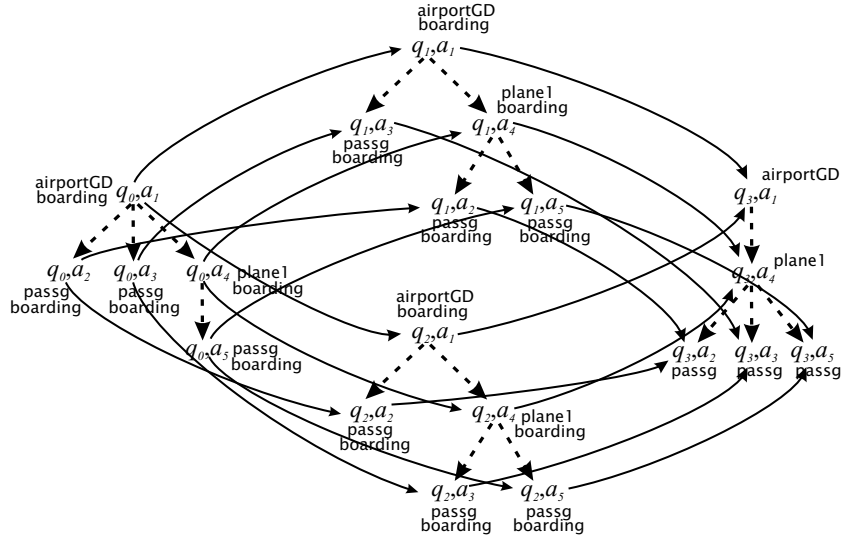


Figure 4. The “flattened” Kripke structure with two accessibility relations

3.4. The Logic of CTL²

We now formally introduce a logic, in which both the temporal and the structural dimensions are captured by CTL-style modalities. The logic of CTL² (with respect to a set of atomic propositions Π) is defined by the following grammar:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid E\bigcirc\varphi \mid E\Box\varphi \mid E\varphi\mathcal{U}\varphi \mid A\bigcirc\varphi \mid A\Box\varphi \mid A\varphi\mathcal{U}\varphi \mid E_s\bigcirc\varphi \mid E_s\Box\varphi \mid E_s\varphi\downarrow\varphi \mid A_s\bigcirc\varphi \mid A_s\Box\varphi \mid A_s\varphi\downarrow\varphi \mid \uparrow\varphi \mid \Box\varphi \mid \varphi\uparrow\varphi,$$

where $p \in \Pi$. The semantics of CTL² is defined as follows:

$M, \omega \models E\bigcirc\varphi$	iff there is a full \mathcal{R} -path Λ (\mathcal{S} -path, \mathcal{S}^{-1} -path, respectively)
$M, \omega \models E_s\bigcirc\varphi$	with the beginning state $\Lambda[0] = \omega$, such that $M, \Lambda[1] \models \varphi$;
$M, \omega \models \uparrow\varphi$	
$M, \omega \models E\Box\varphi$	iff there is a full \mathcal{R} -path Λ (\mathcal{S} -path, \mathcal{S}^{-1} -path, respectively)
$M, \omega \models E_s\Box\varphi$	with the beginning state $\Lambda[0] = \omega$, such that $M, \Lambda[i] \models \varphi$
$M, \omega \models \Box\varphi$	for all $i \geq 0$;
$M, \omega \models E\varphi\mathcal{U}\psi$	iff there is a \mathcal{R} -path Λ (\mathcal{S} -path, \mathcal{S}^{-1} -path, resp.) with
$M, \omega \models E_s\varphi\downarrow\psi$	$\Lambda[0] = \omega$, such that there is $i \geq 0$ with $M, \Lambda[i] \models \psi$
$M, \omega \models \varphi\uparrow\psi$	and for all j such that $0 \leq j < i$ we have $M, \Lambda[j] \models \varphi$.

The semantics of universal path quantifiers A and A_s is defined analogously. Additionally, the “some-time” operator is defined as: $\Diamond\varphi \equiv \top\mathcal{U}\varphi$. Formula φ is *valid* in model M iff $M, \langle q_0, a \rangle \models \varphi$ for every $a \in \text{Agt}$.

Example 3.1. Consider system M_1 from Figure 4. Now, formula $\neg E_s\bigcirc\top$ expresses the property that the “current” agent is empty, i.e. it contains no further agents. This property always holds for every

passenger agent: that is, $\text{passg} \rightarrow A \square \neg E_s \odot \top$ is valid in M_1 . Other formulae, valid in M_1 , are: $A \square (\text{boarding} \wedge \text{airportGD} \rightarrow E_s \odot \text{passg})$: while boarding, there are passengers at the Gdansk airport, and $A \square \uparrow A_s \downarrow (\text{passg} \rightarrow A \diamond \odot \text{plane1})$: all passengers are going to sit in plane 1 eventually.

Example 3.2. Suppose that the Gdańsk airport and the Milan airport are labelled with agent propositions airportGD and airportMIL , respectively. Let plane be a proposition that labels all the airplane agents. Now, $\text{passg} \rightarrow A \square (\odot \text{airportGD} \rightarrow E \diamond \odot (\text{plane} \wedge A \diamond \odot \text{airportMIL}))$ says that a passenger, whenever in Gdansk, can possibly board a connection that is bound for Milan.

3.5. CTL² as a Combination of Logics

CTL² defined by the above monolithic definition can also be explained as a certain *combination* of the two components describing the *temporal* and the *spatial* dimensions of Petri hypernets, respectively. Many forms of (modal) logic combinations have been proposed in the literature: most importantly *fusion* [16], *product* or *join* [15], and *fibring* [13]. As it turns out, to adequately capture the semantics of CTL², it is best to apply a method which lays somewhere between the product and the fibring constructions. Below we show how to obtain CTL² as a combination of its temporal and spatial CTL-like components. We denote these components by CTL_T² and CTL_S² respectively. We concentrate on the semantic part of the construction, since at the syntactic level the “interleaved formulae” of CTL² are obtained by simply taking the union of the formation rules for CTL_T² and CTL_S² (as in the case of both product and fibring).

For any CTL_T² model $M = \langle Q, q_0, \mathcal{R}, \pi \rangle$ let us consider an arbitrary family (fibring function) $\mathcal{F}_M = \langle \langle A_M, \mathcal{S}^q, \pi^q \rangle \mid q \in Q \rangle$ of CTL_S² models. Please note, that we assume that all the models in \mathcal{F}_M share the same set of states. Using the model M and the fibring function \mathcal{F}_M we can define a *combined model* as a tuple $\langle Q \times A, q_0, \overline{\mathcal{R}}, \overline{\mathcal{S}}, \overline{\pi} \rangle$, where

- $\overline{\mathcal{R}} = \{ \langle \langle q_1, a \rangle, \langle q_2, a \rangle \rangle \mid q_1 \mathcal{R} q_2 \}$
- $\overline{\mathcal{S}} = \{ \langle \langle q, a_1 \rangle, \langle q, a_2 \rangle \rangle \mid a_1 \mathcal{S}^q a_2 \}$
- $\overline{\pi}(p) = \begin{cases} \pi(p) \times A_M & \text{for } p \in \Pi_T \\ \{ \langle q, a \rangle \mid a \in \pi^q(p) \wedge q \in Q \} & \text{for } p \in \Pi_S \end{cases}$

where Π_T and Π_S denote the appropriate sets of temporal and spatial/structural atomic propositions. The issue of a particular choice of atomic propositions was already discussed in Remark 3.1.

We take the class of all *combined models* as the class of models of the resulting logic. From the construction it is clear that combined models are equivalent to the *agent-oriented hyper-transition systems* as defined in Section 3.3. The semantic interpretation of formulae of CTL² is obtained as the union of semantic rules for CTL_T² and CTL_S².

4. Model Checking Properties of Mobile Agents

The *model checking* problem asks whether a given formula φ holds in a given model M and state q . Any CTL model checking algorithm can be easily adapted to handle CTL² formulae. Thus, the results for CTL model checking (cf. [23]) apply to CTL² as well.

Proposition 4.1. Let l be the length of a CTL² formula φ , and \overline{m} be the cardinality of the “densest” modal relation in an agent-oriented hyper-transition system M . Model checking φ in M is PTIME-complete and can be done in time $O(\overline{m}l)$.

We will see in Section 4.1 that the result actually promises less than it suggests, because the AOHS derived from Petri hypernets are usually large. However, this problem is well known for the original CTL too: CTL models are usually exponentially large in terms of a higher-level description of the problem domain—yet model checking turns out to be feasible in many cases [18, 19].

4.1. From Hypernets to Hyper-transition Systems

Let $H = \langle \mathcal{N}, m \rangle$ be a hypernet. CTL² properties are interpreted over the AOHS that includes all reachable markings of H , and transitions that can occur in H . Formally, $aohs(H) = \langle Q_H, q_0, \mathbb{A}gt_H, \mathcal{R}_H, \mathcal{S}_H, \pi_H \rangle$, is called an agent-oriented hypertransition system associated with H , and defined as follows.

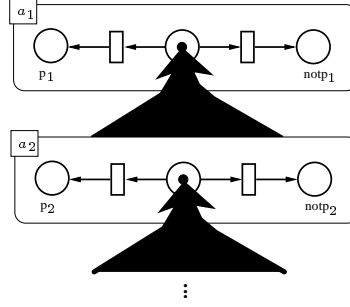
- $\mathbb{A}gt_H = \mathcal{N}$ (agents are identified with nets of H);
- Q_H and \mathcal{R}_H are defined recursively: (1) $m \in Q_H$, (2) if a transaction can be fired in hypernet $\langle \mathcal{N}, m_1 \rangle$, $m_1 \in Q_H$, yielding $\langle \mathcal{N}, m_2 \rangle$, then: (i) $m_2 \in Q_H$ and (ii) $\langle m_1, a \rangle \mathcal{R}_H \langle m_2, a \rangle$ for all $a \in \mathbb{A}gt_H$;
- $\langle m, a_1 \rangle \mathcal{S}_H \langle m, a_2 \rangle$ iff $m(a_1) \in P_{a_2}$, where P_a is the set of places of agent a ;
- $q_0 = m$: the initial state is defined by the current hypermarking m ;
- if $p \in \Pi_{ag}$ then $\pi(p) = Q_H \times \lambda_{ag}(p)$: agent labels in H yield agent-identifying propositions in $aohs(H)$;
- If $p \in \Pi_{pl}$ then $\pi(p) = \{m \mid \forall \mathfrak{p} \in \lambda_{pl}(p) \exists N \in \mathcal{N} . m(N) \in \mathfrak{p}\} \times \mathbb{A}gt_H$. That is, p distinguishes those hypermarkings, according to which every place from $\lambda_{pl}(p)$ is inhabited by some token.

Note that the place propositions from Π_{pl} are purely *temporal* properties which only depend on the current temporal state. On the other hand, the agent propositions (elements of Π_{ag}) are purely *structural* properties depending on the current agent—the spatial/structural state.

Proposition 4.2. $aohs(H)$ is an AOHS for every PHN H .

Proposition 4.3. For a hypernet H with n places in total, $aohs(H)$ includes at most $O(n)2^{O(n \log n)}$ agent-oriented states, and the bound is tight (cf. Remark 2.1).

It seems worth pointing out that the size of the $aohs(H)$ becomes polynomial when the number of agents is fixed or bounded. Unfortunately, decomposing complex systems into *many* individual agents is an obvious methodology and in fact it is one of the main advantages that Petri hypernets can offer.

Figure 5. A hypernet generating an arbitrary assignment for p_1, \dots, p_k

4.2. Model Checking Petri Hypernets

Having defined the correspondence between AOHS and (labelled) PHN, we can say what it means that a CTL² formula holds for net a in a hypernet $H = \langle \mathcal{N}, m \rangle$.

Definition 4.1. $H, a \models \varphi$ iff $aohs(H), \langle m, a \rangle \models \varphi$.

Proposition 4.4. Model checking CTL² formulae over PHN is PSPACE-hard.

Proof. We show this through a reduction of the QSAT problem (satisfiability for quantified Boolean formulae). In QSAT, we are given k propositional variables p_1, \dots, p_k partitioned into sets P_1, P_2, \dots , and a formula $\Phi \equiv \exists P_1 \forall P_2 \exists P_3 \dots \varphi$, where φ is a Boolean combination of p_1, \dots, p_k . We construct a hypernet H that includes $k + 1$ agents, each next agent placed inside the previous one. The task of agent a_i for $i = 1, \dots, k$ is to “declare” variable p_i true or false; the agent does it by moving “his” sub-agent to a place where proposition p_i (or proposition $\text{not}p_i$) holds. The structure of the net is depicted in Figure 5. We define auxiliary formulae $\text{ready}_i \equiv \bigwedge_{j=1}^i (p_j \vee \text{not}p_j) \wedge \bigwedge_{j=i+1}^k \neg(p_j \vee \text{not}p_j)$, saying that agents a_1, \dots, a_i have done their job, and the rest have not.

We also assume that φ has been transformed so that negations apply to atomic propositions only. We propose the following translation of Φ :

$$\begin{aligned} tr(\exists p_i \Psi) &= E\bigcirc(\text{ready}_i \wedge tr(\Psi)) & tr(\forall p_i \Psi) &= A\bigcirc(\text{ready}_i \rightarrow tr(\Psi)) \\ tr(\varphi \wedge \psi) &= tr(\varphi) \wedge tr(\psi) & tr(\varphi \vee \psi) &= tr(\varphi) \vee tr(\psi) \\ tr(p_i) &= p_i & tr(\neg p_i) &= \text{not}p_i \end{aligned}$$

Note that Φ holds iff $H, a_1 \models tr(\Phi)$. Moreover, H includes only $O(k)$ places, and the length of $tr(\Phi)$ is $O(k^2 + |\varphi|)$, which concludes the proof. \square

For the upper bound, let n_K, m_K be the numbers of nodes and transitions in a Kripke structure (e.g. an AOHS). It has been proved in [17] (Theorem 5.7) that formulae of CTL can be model-checked in space $O(l \log^2((n_K + m_K)l))$. By Proposition 4.3, CTL² can be model-checked in space $O(l^3 n_H^4)$, where n_H is the number of places in the underlying hypernet H . As [17] presents an analogous result concerning CTL* model checking, and LTL can be seen as a fragment of CTL*, we can state the following.

Theorem 4.1. Model checking CTL² over Petri hypernets is PSPACE-complete. The same complexity bounds apply if we use LTL or CTL* to reason about the temporal and/or structural dimension of Petri hypernets.

The PSPACE-completeness is actually what one should expect from a problem of this kind [10, 11]. We note that the problem is no worse than for the logic of mobile ambients [6], and for the standard temporal logics CTL, LTL and CTL* (when the input is given as a combination of smaller components [23]). This hints that automatic verification of agent properties via Petri hypernets and modal logics like CTL² should be feasible after all.

5. Conclusions

We have shown that one indeed, as advocated in [12], can use a combination of two modal logics, one to cope with the temporal, the other with the structural aspects, in order to specify the properties of mobile agents. The logic we have obtained cannot be seen as one of the simple combinations considered in [12], though. Instead, we blend some aspects of *product* and *fibring*. We treat our choice of logics to address the temporal and the spatial dimension as somewhat arbitrary. In a way, one can treat the actual logics as parameters of the resulting framework. In this paper, we used CTL to address both dimensions, but most other temporal logics can be used instead in a natural way. It can be interesting to study what properties can be specified using other, more expressive logics – most notably μ -calculus, ATL, and logics of strategic ability under incomplete information.

In CTL², atomic propositions cope purely with one of the aspects, either temporal or spatial. In the case of hypernets it seems very natural to consider atomic propositions of the form: “the agent is at gate G7”, which refer to both the current temporal state and the current agent. Thus, finer-grained models and a more general combination of logics seem desirable. This is another promising direction for future research. This shows also that Petri hypernets offer a rich modelling framework, one which has not been matched yet by a sufficiently expressive logical counterpart.

There are a number of interesting questions to investigate. One would be to link current approach with research in the multi-agent systems area. For instance, by forgetting about the identity of agents we should be able to obtain a multi-agent system in which two agents can be distinguished only by their actions. Then, a natural problem of *synthesis* arises—given a multi-agent system is it possible to realise it as Petri hypernet? In fact, translation in both directions would be interesting. Even the simpler forgetful translation from Petri hypernets to multi-agent systems could help relating logics in both areas.

A tool for editing and simulating Petri hypernets has already been developed by Artur Siekielski and Jakub Sławiński, students at the Gdańsk University. The tool has been recently extended by a simple model-checker for verifying CTL² formulae. The development of more effective tool is under construction.

References

- [1] Bednarczyk, M., Bernardinello, L., Pawłowski, W., Pomello, L.: Modelling Mobility with Petri Hypernets, *Proceedings of WADT 2004*, 3423, 2004.
- [2] Bednarczyk, M. A., Bernardinello, L., Pawłowski, W., Pomello, L.: Petri hypernets explained, 2006, Submitted to ATPN’06.
- [3] Bednarczyk, M. A., Jamroga, W., Pawłowski, W.: Expressing and Verifying Temporal and Structural Properties of Mobile Agents, *Proceedings of the Concurrency, Specification and Programming Workshop CS&P’05* (L. Czaja, Ed.), 2005.

- [4] Bednarczyk, M. A., Jamroga, W., Pawłowski, W.: *Expressing and Verifying Temporal and Structural Properties of Mobile Agents*, Technical Report IfI-05-09, Clausthal University of Technology, 2005.
- [5] Cardelli, L., Gordon, A. D.: Mobile Ambients., *Electr. Notes Theor. Comput. Sci.*, **10**, 1997.
- [6] Charatonik, W., Dal-Zilio, S., Gordon, A. D., Mukhopadhyay, S., Talbot, J.-M.: Model checking mobile ambients., *Theor. Comput. Sci.*, **308**(1-3), 2003, 277–331.
- [7] Clarke, E., Emerson, E.: Design and Synthesis of Synchronization Skeletons Using Branching Time Temporal Logic, *Proceedings of Logics of Programs Workshop*, 131, 1981.
- [8] Emerson, E., Halpern, J.: "Sometimes" and "Not Never" Revisited: On Branching versus Linear Time Temporal Logic, *Journal of the ACM*, **33**(1), 1986, 151–178.
- [9] Emerson, E. A.: Temporal and Modal Logic, in: *Handbook of Theoretical Computer Science* (J. van Leeuwen, Ed.), vol. B, Elsevier Science Publishers, 1990, 995–1072.
- [10] Esparza, J.: Decidability of Model Checking for Infinite-State Concurrent Systems., *Acta Inf.*, **34**(2), 1997, 85–107.
- [11] Esparza, J.: Decidability and Complexity of Petri Net Problems. An introduction, in: *Lectures on Petri Nets I: Basic Models. Adv. in Petri Nets* (G. Rozenberg, W. Reisig, Eds.), vol. 1491 of *LNCS*, Springer Verlag, 1998, 374–428.
- [12] Franceschet, M., Montanari, A., de Rijke, M.: Model Checking for Combined Logics with an Application to Mobile Systems., *Autom. Softw. Eng.*, **11**(3), 2004, 289–321.
- [13] Gabbay, D. M.: *Fibring Logics*, vol. 38 of *Oxford Logic Guides*, Oxford University Press, 1998.
- [14] Gabbay, D. M., Pnuelli, A., Shelah, S., Stavi, J.: On the Temporal Analysis of Fairness, *Proceedings of POPL'80*, 1980.
- [15] Gabbay, D. M., Shehtman, V.: Products of modal logics, Part 1, *Logic Journal of the IGPL*, **6**(1), 1998, 73–146.
- [16] Kracht, M., Wolter, F.: Properties Of Independently Axiomatizable Bimodal Logics, *Journal Of Symbolic Logic*, **54** (4), 1991, 1469–1485.
- [17] Kupferman, O., Vardi, M., Wolper, P.: An automata-theoretic approach to branching-time model checking, *Journal of the ACM*, **47**(2), 2000, 312–360.
- [18] McMillan, K.: *Symbolic Model Checking: An Approach to the State Explosion Problem*, Kluwer Academic Publishers, 1993.
- [19] McMillan, K.: Applying SAT Methods in Unbounded Symbolic Model Checking, *Proceedings of CAV'02*, 2404, 2002.
- [20] Rao, A., Georgeff, M.: Modeling Rational Agents within a BDI-Architecture, *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning*, 1991.
- [21] Reisig, W.: *Petri Nets*, vol. 4 of *EATCS Monographs on Theoretical Computer Science*, Springer-Verlag, 1985.
- [22] Schild, K.: On the Relationship Between BDI Logics and Standard Logics of Concurrency, *Autonomous Agents and Multi Agent Systems*, 2000, 259–283.
- [23] Schnoebelen, P.: The Complexity of Temporal Model Checking, *Advances in Modal Logics, Proceedings of AiML 2002*, World Scientific, 2003.
- [24] Valk, R.: Petri Nets as Token Objects: An Introduction to Elementary Object Nets, *Applications and Theory of Petri Nets 1998, Proceedings* (W. van der Aalst, E. Best, Eds.), 1420, Springer-Verlag, 1998.