Model Checking Logics for Communicating Sequential Agents*

Michaela Huhn[1] and Peter Niebert[2] and Frank Wallner[3]

[1] Institut für Rechnerentwurf und Fehlertoleranz (Prof. D. Schmid), Univ. Karlsruhe, Postf. 6980, D-76128 Karlsruhe, huhn@ira.uka.de
[2] VERIMAG, 2, av. de Vignate, 38610 Gières, France, niebert@imag.fr
[3] Institut für Informatik, Technische Universität München, D-80290 München, wallnerf@in.tum.de

Abstract. We present a model checking algorithm for $\mathcal{L}_{CSA}$, a temporal logic for *communicating sequential agents* (CSAs) introduced by Lodaya, Ramanujam, and Thiagarajan. $\mathcal{L}_{CSA}$ contains temporal modalities indexed with a local point of view of one agent and allows to refer to properties of other agents according to the *latest gossip* which is related to *local knowledge*.
The model checking procedure relies on a modularisation of $\mathcal{L}_{CSA}$ into temporal and gossip modalities. We introduce a hierarchy of formulae and a corresponding hierarchy of equivalences, which allows to compute for each formula and finite state distributed system a finite multi modal Kripke structure, on which the formula can be checked with standard techniques.

1 Introduction

A reasonable and lucid way of formally treating distributed systems is to consider them as a fixed collection of sequential components (agents) which can operate independently as well as cooperate by exchanging information. There is an increasing awareness, both in theory and practice, of the benefits of specifying the requirements of such systems by *localised*, component based formalisms, that allow to refer to properties of the individual components.

The operational models for localised specification usually consist of *local* temporal orders (sequences in the linear time case, trees in branching time) together with an interrelation between these orders, descended from communication [LRT92,Ram95]. The most established models for the linear time case are partial orders, whereas in the branching time setting, *(prime) event structures* or closely related models like *occurrence nets* [NPW80,Win87] have been recognised to be a suitable formalism. In these models, partial orders are extended by an additional conflict relation, representing the moments of choice.

* This work was partially supported by the DFG within SFB 342 (A3), and within the priority program "*Design and design methodology of embedded systems*", and by the EEC program on *Training and Mobility in Research* (TMR). The work was done while the second author was affiliated with the Univ. of Hildesheim, Germany.

Investigating partial order models has attained the interest of researchers for mainly two reasons: There is no distinction among computations that are equal up to possible total orderings of independent actions, which makes it a faithful and natural formalism for representing concurrency. Furthermore, restricting the attention to local states mitigates one of the most tackled difficulty of model checking, the so-called *state explosion problem*, which results from an explicit computation of the global state space of a distributed system.

For a component-oriented specification of behaviour, local linear time temporal logics have been investigated by Thiagarajan in [Thi94,Thi95] and Niebert [Nie98]. Local branching time logics were introduced in [LT87,LRT92,HNW98b]. While for the linear time case there now exist sound model checking procedures based on automata [Thi94,Nie98], only recently the model checking problem for local branching time logics has been inspected [Pen97,HNW98b].

In this paper, we investigate model checking for a local branching time logic defined by Lodaya, Ramanujam and Thiagarajan in [LRT92], here called $\mathcal{L}_{CSA}$, which is intended to specify the behaviour of *communicating sequential agents* (CSAs). It allows a component i to refer to local properties of another component j according to the *latest gossip* (in [LRT92] also called local knowledge), i.e., the most recent j-local state that causally precedes the current i-local state.

Based on net unfoldings [Eng91] and McMillan's *finite prefix* construction [McM92], we solve the model checking problem for $\mathcal{L}_{CSA}$, which remained open since [LRT92].

McMillan's prefix has successfully been applied to alleviate state explosion in many verification problems, for instance deadlock detection [McM92], and model checking S4 [Esp94], LTL [Wal98], and the distributed μ-calculus [HNW98b]. All of the previous problems principally can be solved with conventional state space exploration, but often with an exponentially higher effort.

The focus of this paper is to show decidability of model checking $\mathcal{L}_{CSA}$. Generalising the techniques of [HNW98b], we demonstrate that the unfolding approach is very suitable for model checking a wider class of local logics, for which previously the problem appeared to be too difficult.

Technically, we proceed as follows: We lift the semantics of $\mathcal{L}_{CSA}$ from CSAs onto net unfoldings, and factorise the net unfolding with respect to an equivalence relation satisfying two key properties: It is a congruence for the $\mathcal{L}_{CSA}$-specification to be checked, and it has finite index. Via this factorisation, the $\mathcal{L}_{CSA}$ model checking problem can be transformed into a model checking problem for a multi modal logic on a finite transition system constructed upon a modified McMillan prefix, using the defined equivalence relation as cutoff condition. With an appropriate interpretation of the $\mathcal{L}_{CSA}$ modalities, standard model checking algorithms, e.g. [CES86], can be applied on this transition system.

The approach follows the lines of [HNW98b], but whereas the focus there was to derive an algorithm for calculating the transition system, the main difficulty here is to develop an appropriate equivalence relation. The modalities of the distributed μ-calculus of [HNW98b] are purely future oriented, while the past and also the gossip modalities of $\mathcal{L}_{CSA}$ may lead to rather complex patterns within the past of a configuration. As a consequence, the coarsest equivalence preserving *all* $\mathcal{L}_{CSA}$ properties has non-finite index and it is not possible to construct a single (finite-state) transition system representing all $\mathcal{L}_{CSA}$ properties of a particular finite state distributed system. However, a single $\mathcal{L}_{CSA}$ formula has a limited power of referring to the past so that we can construct an equivalence depending on the formula. For this purpose, we introduce a syntactic hierarchy of formulae and a corresponding equivalence hierarchy. The construction of these equivalences and the proof of their soundness are both complex, and the resulting model checking complexity of the construction given here is high.

The technical presentation of the paper relies on notions from Petri net theory, mainly to correspond directly to McMillan's prefix. Note however, that the entire method can easily be restated for other formalisms, like e.g. asynchronous automata, coupled finite state machines, and so forth.

The paper is structured as follows. In Section 2 we introduce distributed net systems, and their unfoldings as semantic model of branching behaviour. In Section 3 we introduce the logic $\mathcal{L}_{CSA}$ and our slightly generalised version $\mathcal{L}$. In Section 4 we present McMillan's finite prefix, and parameterise its definition by an abstract equivalence relation. Then we develop an appropriate equivalence for $\mathcal{L}$. In Section 5 we use this equivalence to compute a finite state transition system, on which the model checking problem for $\mathcal{L}$ can be solved by conventional model checkers. In Section 6, we discuss our results and indicate future work.

2 Distributed net systems and their unfoldings

Petri nets. Let P and T be disjoint, finite sets of *places* and *transitions*, generically called *nodes*. A *net* is a triple $N = (P, T, F)$ with a *flow relation* $F \subseteq (P{\times}T) \cup (T{\times}P)$. The *preset* of a node x is defined as ${}^\bullet x := \{y \in P \cup T \mid yFx\}$ and its *postset* as $x^\bullet := \{y \in P \cup T \mid xFy\}$. The preset (resp. postset) of a set X of nodes is the union of the presets (resp. postsets) of all nodes in X.

A *marking* of a net is a mapping $M : P \to \mathbb{N}_0$. If $M(p) = n$, we say that p contains n *tokens* at M. A *net system* $\Sigma = (N, M_0)$ consists of a net N, and an *initial marking* M_0. The marking M *enables* the transition t if every place in the preset of t contains at least one token. In this case the transition can *occur*. If t occurs, it removes one token from each place $p \in {}^\bullet t$ and adds one token to each place $p' \in t^\bullet$, yielding a new marking M'. We denote this occurrence by $M \xrightarrow{t} M'$. If there exists a chain $M_0 \xrightarrow{t_1} M_1 \xrightarrow{t_2} \ldots \xrightarrow{t_n} M_n$ for $n \geq 0$, then the marking M_n is a *reachable marking*.

We will restrict our attention to *1-safe net systems*, in which every reachable marking M puts at most one token on each place, and thus can be identified by the subset of places that contain a token, i.e., $M \subseteq P$.

In the last years, 1-safe net systems have become a significant model [CEP95]. In [NRT90] it has been shown that an instance of 1-safe nets, called *Elementary Net Systems*, correspond to other models of concurrency, such as (Mazurkiewicz) traces and prime event structures. They can naturally be interpreted as a synchronised product of several finite automata, and thus they are frequently used as a convenient formalism for modelling distributed systems. In the following we will exploit this compositional view by considering the notion of *locations*.

Distributed net systems. Let us introduce the formalism for describing distributed systems. Clearly, the *behaviour* of our models shall resemble the *Communicating Sequential Agents* of [LRT92]. This means, a system consists of several distributed, autonomous agents, which mutually communicate. Each of the agents shall behave strictly sequentially, and non-deterministically.

Let Σ be a 1-safe net system, and t, t' two transitions of Σ. A marking M *concurrently* enables t and t' if M enables t, and $(M \setminus {}^\bullet t)$ enables t'. We call Σ *sequential* if no reachable marking concurrently enables two transitions.

Let $\{\Sigma_i = (P_i, T_i, F_i, M_i^0) \mid i \in Loc\}$ be a family of 1-safe, sequential net systems (called *agents*, or *components*) with pairwise disjoint sets P_i of places, indexed by a finite set Loc of *locations*. Note that the sets of transitions are not necessarily disjoint. In fact, we will interpret the execution of a transition that is common to several agents as a *synchronous communication* action of these agents, i.e., the communication capabilities are given by the common execution of joint transitions. Formally, a *distributed net system* $\Sigma_{Loc} = (N, M_0)$ is defined as the union of its components Σ_i:

$$P = \bigcup_{i \in Loc} P_i\ , \qquad T = \bigcup_{i \in Loc} T_i\ , \qquad F = \bigcup_{i \in Loc} F_i\ , \qquad M_0 = \bigcup_{i \in Loc} M_i^0\ .$$

Clearly, Σ_{Loc} is again 1-safe. The *location* $loc(x)$ of a node x is defined by $loc(x) := \{i \in Loc \mid x \in P_i \cup T_i\}$. A simple distributed net system consisting of two components is depicted in Fig. 1.

In [LRT92] also *asynchronous* communication (message passing) is considered. However, in general this yields systems with infinitely many states, making an algorithmic, state space based approach to model checking impossible. To model the asynchronous setting, we can assume some finite-state communication mechanism like e.g. bounded channels or buffers, which can easily be defined within the presented framework by considering a buffer as an agent of its own, (synchronously) communicating with both the agents that communicate (asynchronously) via this buffer.

Net unfoldings. As a partial order semantics of the behaviour of a distributed net system, we consider *net unfoldings*, also known as *branching processes*. They contain information about both concurrency and conflict.

Two nodes x, x' of a net (P, T, F) are *in conflict*, denoted $x \# x'$, if there exist two distinct transitions t, t' such that ${}^\bullet t \cap {}^\bullet t' \neq \emptyset$, and $(t, x), (t', x')$ belong to the reflexive, transitive closure of F. If $x \# x$, we say *x is in self-conflict*.

An *occurrence net* [NPW80] is a net $N' = (B, E, F)$ with the following properties: (1) for every $b \in B$, $|{}^\bullet b| \leq 1$, (2) the irreflexive transitive closure $<$ of F is well-founded and acyclic, i.e., for every node $x \in B \cup E$, the set $\{y \in B \cup E \mid y < x\}$ is finite and does not contain x, and (3) no element $e \in E$ is in self-conflict. The reflexive closure $\leq$ of $<$ is a partial order, called *causality relation*. In occurrence nets we speak of *conditions* and *events* instead of places and transitions, respectively. $Min(N')$ denotes the minimal elements of N' w.r.t. $\leq$.

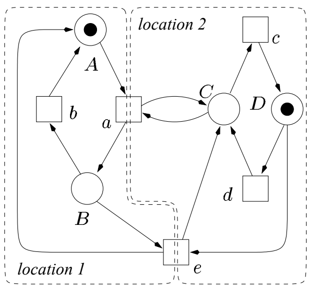

Fig. 1. Distributed net

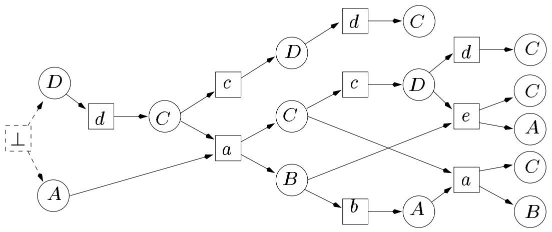

Fig. 2. Branching process

Given two nets N_1, N_2, the mapping $h : P_1 \cup T_1 \to P_2 \cup T_2$ is called a *homomorphism* if $h(P_1) \subseteq P_2, h(T_1) \subseteq T_2$, and for every $t \in T_1$ the restriction of h to ${}^\bullet t$, denoted $h|_{{}^\bullet t}$, is a bijection between ${}^\bullet t$ and ${}^\bullet h(t)$, and analogous for $h|_{t^\bullet}$.

A *branching process* [Eng91] of a net system $\Sigma = (N, M_0)$ is a pair $\beta = (N', \pi)$ where $N' = (B, E, F)$ is an occurrence net and $\pi : N' \to N$ is a homomorphism, such that the restriction of π to $Min(N')$ is a bijection between $Min(N')$ and M_0, and additionally for all $e_1, e_2 \in E$: if $\pi(e_1) = \pi(e_2)$ and ${}^\bullet e_1 = {}^\bullet e_2$ then $e_1 = e_2$. Loosely speaking, we unfold the net N to an occurrence net N', such that each node x of N' refers to node $\pi(x)$ of N. Two branching processes β_1, β_2 of Σ are *isomorphic* if there exists a bijective homomorphism $h : N_1 \to N_2$ such that the composition $\pi_2 \circ h$ equals π_1. In [Eng91] it is shown that each net system Σ has a unique maximal branching process up to isomorphism, which we call the *unfolding* of Σ, and denote by $Unf_\Sigma = (N', \pi)$.

In distributed net systems, the location $loc(x)$ of a node x of N' is given by $loc(x) = loc(\pi(x))$. By $E_i := \{e \in E \mid i \in loc(e)\}$, we denote the set of *i-events*.

Let $N'' = (B'', E'', F'')$ be a subnet of N', such that $e \in E''$ implies $e' \in E''$ for every $e' < e$, and $B'' = Min(N') \cup E''^\bullet$, and let π'' be the restriction of π onto the nodes of N''. We call $\beta'' = (N'', \pi'')$ a *prefix* of Unf_Σ. Fig. 2 shows a prefix of the infinite unfolding of the net system drawn in Fig. 1.

Configurations and Cuts. For the remainder of the section, let us fix the unfolding $Unf_\Sigma = (N', \pi)$ of the distributed net system Σ with $N' = (B, E, F)$.

A *configuration* $C \subseteq E$ is a causally downward-closed, conflict-free set of events, i.e., $\forall\, e \in C$: if $e' \le e$ then $e' \in C$, and $\forall\, e, e' \in C : \neg(e \# e')$. A finite configuration describes the initial part of a computation of the system. If we understand the *states* of the system as moments in time, then configurations represent the *past* (by exhibiting all the events that have occurred so far, and the causal structure among them), as well as the *present* and the *future*, as formalised in the following.

Two nodes of N' are *concurrent* if they are neither in conflict nor causally related. A set $B' \subseteq B$ of conditions of N' is called a *cut* if B' is a maximal set of pairwise concurrent conditions. Every finite configuration C determines a cut $Cut(C) := (Min(N') \cup C^\bullet) \setminus {}^\bullet C$. The corresponding set $\pi(Cut(C)) \subseteq P$ of places is a reachable marking of Σ, denoted by $\mathcal{M}(C)$ and called *the state of* C. Notice that for every reachable marking M of Σ, there exists a (not necessarily unique) finite configuration with state M. We will often identify configurations with their state. Given a configuration C and a disjoint set E' of events, we call $C \oplus E'$ an *extension* of C if $C \cup E'$ is a configuration.

Let $\uparrow C := \{x \in (B \cup E) \mid \exists b \in \mathit{Cut}(C).\ b \leq x \quad \text{and} \quad \forall y \in C.\ \neg(x \# y)\}$. The *(branching) future* of a configuration C is given by the branching process $\beta(C) := (N'_C, \pi_C)$, where N'_C is the unique subnet of N' whose set of nodes is $\uparrow C$, and π_C is the restriction of π onto the nodes of N'_C. Let us call two configurations $\mathcal{M}$*-equivalent*, denoted $C \equiv_{\mathcal{M}} C'$, if $\mathcal{M}(C) = \mathcal{M}(C')$. It is easy to show that if $C \equiv_{\mathcal{M}} C'$ then there exists an isomorphism $I_C^{C'}$ from $\beta(C)$ to $\beta(C')$. It induces a mapping from the extensions of C onto the extensions of C', mapping $C \oplus E'$ onto $C' \oplus I_C^{C'}(E')$, which are again $\mathcal{M}$-equivalent.

Local states and views. The notion of *local state* arises by considering configurations that are determined by single events. For an event e, we call the set $\downarrow e := \{e' \in E \mid e' \leq e\}$ the *local configuration of* e. It is indeed a configuration, because no event is in self-conflict. If $e \in E_i$ is an i-event, we consider $\downarrow e$ to be an i*-local state*. It determines the local past of component i, as well as the local past of every component that communicated with i so far — directly, or indirectly via other components.

In distributed net systems, we define the i*-view* $\downarrow^i C$ of a configuration C as $\downarrow^i C := \{e \in C \mid \exists e_i \in (C \cap E_i).\, e \leq e_i\}$. Notice that the sequentiality of the components implies that for each $i \in \mathit{Loc}$, the i-events form a tree in *Unf*, i.e., in each configuration the i-events are totally ordered. Thus, the i-view of C is the local configuration of the unique, causally maximal i-event in C. Intuitively, $\downarrow^i C$ can be understood as the most recent i-local configuration that the whole system is aware of in the (global) configuration C. The i-view of a local configuration $\downarrow e$ is written as $\downarrow^i e$. Note that $\downarrow^i e = \downarrow e$ iff $i \in \mathit{loc}(e)$. We will interpret the empty configuration as the local configuration of a virtual event $\bot$, which can be seen as *initial* event with empty preset and $\mathit{Min}(N')$ as postset. We assume the set of events of Unf_Σ to contain this virtual event, $\bot \in E$, and set $\mathit{loc}(\bot) := \mathit{Loc}$.

Let $\mathcal{C}_{loc}(\mathit{Unf})$ denote the set of local configurations of *Unf* (abbreviated $\mathcal{C}_{loc}$ if *Unf* is clear), and let $\mathcal{C}^i_{loc} := \{\downarrow e \mid e \in E_i\}$ be the set of i-local configurations.

Correspondence of CSAs and unfoldings. Originally in [LRT92], the entire formalism relies on CSAs, a subclass of prime event structures. We note that net unfoldings as presented here, directly correspond to *rooted* CSAs. The differences are only technical. For details of this correspondence, cf. [HNW98a].

3 Temporal Logic for Communicating Sequential Agents

In [LRT92], Lodaya, Ramanujam, and Thiagarajan defined and axiomatised the temporal logic $\mathcal{L}_{CSA}$ that allows to express properties referring to the *latest gossip* of the agents in a distributed system. Let us give a brief idea of the logic, related to unfoldings of distributed net systems. For details, cf. [LRT92].

Basically, $\mathcal{L}_{CSA}$ consists of propositional logic. Additionally, it provides two temporal operators $\Diamond_i$, resp. $\Diamond\!\!\!\!-_i$, for each $i \in Loc$, referring to the *local* future, resp. local past, of agent i. All formulae are interpreted exclusively on the *local* configurations of a given unfolding.

Intuitively, $\Diamond\!\!\!\!-_i\, \varphi$ holds at $\downarrow\! e$ if some i-local configuration in the past of e satisfies φ. When e is a j-event, this can be read as "agent j has at its local state $\downarrow\! e$ enough gossip information to assert that φ was true in the past in agent i".

The local configuration $\downarrow\! e$ satisfies $\Diamond_i\, \varphi$ iff some i-local configuration in the i-local future of $\downarrow\! e$ satisfies φ, i.e., if there is some configuration $\downarrow\! e'$ $(e' \in E_i)$ such that $\downarrow\! e' \supseteq \downarrow^i\! e$ and $\downarrow\! e'$ satisfies φ. For $e \in E_j$, this can be read as "at the j-local state where e has just occurred, agent j has enough gossip information about agent i to assert that φ may hold eventually in i".

Typical specifications are properties like $\Diamond_i(x_i \to \bigwedge_{j \in Loc} \Diamond_j\, x_j)$: "whenever x_i holds in i, then agent i knows that x_j may hold eventually in all other agents j". For more detailed examples, cf. [LRT92].

A generalised syntax – $\mathcal{L}$. We now introduce a slightly extended language in which the temporal modalities $\Diamond, \Diamond\!\!\!\!-$ are separated from the gossip modality $@\, i\!:$. The separation yields a higher degree of modularity in the technical treatment and also saves redundant indices in nested formulae residing at a single location. The abstract syntax of $\mathcal{L}$ is

$$\varphi \;::=\; p \mid \neg\varphi \mid \varphi \vee \varphi \mid \Diamond\, \varphi \mid \Diamond\!\!\!\!-\, \varphi \mid @\, i\!: \varphi$$

where p ranges over a set of atomic propositions, and i over Loc. We require that every occurrence of a temporal modality lies within the scope of a gossip modality. The operators $\Diamond$ and $\Diamond\!\!\!\!-$ are now seen as temporal future and past modalities within a single location, which is determined by the next enclosing gossip modality $@\, i\!:$. For example, $\Diamond_i \Box_j \varphi$ will be written as $@\, i : \Diamond\, @\, j : \Box\varphi'$ in our syntax. Formally, the connection to the original $\mathcal{L}_{CSA}$ syntax is given in [HNW98a].

Like in $\mathcal{L}_{CSA}$, formulae are interpreted at local configurations only. The models of $\mathcal{L}$ are unfoldings of distributed net systems. The interpretation of the atomic propositions relies on the state function $\mathcal{M}$, i.e., we identify the atomic propositions with the set P of places of the system under consideration (without loosing expressive power by this convention), and evaluate a proposition at configuration $\downarrow\! e$ according to $\mathcal{M}(\downarrow\! e)$.

Formally, we define two satisfaction relations: a global relation $\models$, defined for the local configurations of *arbitrary* locations, and for each agent $i \in Loc$ a local relation $\models_i$, exclusively defined for the *i-local* configurations. These relations are inductively defined as follows:

$$\begin{array}{llll}
\downarrow\! e \models p & \text{iff } p \in \mathcal{M}(\downarrow\! e) & \downarrow\! e \models \varphi \vee \psi & \text{iff } \downarrow\! e \models \varphi \text{ or } \downarrow\! e \models \psi \\
\downarrow\! e \models \neg\varphi & \text{iff } \downarrow\! e \not\models \varphi & \downarrow\! e \models @\, i\!: \varphi & \text{iff } \downarrow^i\! e \models_i \varphi
\end{array}$$

$$\begin{array}{llll}
\downarrow\! e \models_i p & \text{iff } p \in \mathcal{M}(\downarrow\! e) & \downarrow\! e \models_i \varphi \vee \psi & \text{iff } \downarrow\! e \models_i \varphi \text{ or } \downarrow\! e \models_i \psi \\
\downarrow\! e \models_i \neg\varphi & \text{iff } \downarrow\! e \not\models_i \varphi & \downarrow\! e \models_i \Diamond\!\!\!\!-\, \varphi & \text{iff } \exists e' \in E_i \,.\, e' \leq e \text{ and } \downarrow\! e' \models_i \varphi \\
\downarrow\! e \models_i @\, j\!: \varphi & \text{iff } \downarrow^j\! e \models_j \varphi & \downarrow\! e \models_i \Diamond\varphi & \text{iff } \exists e' \in E_i \,.\, e' \geq e \text{ and } \downarrow\! e' \models_i \varphi
\end{array}$$

We say that the system Σ satisfies a formula φ if the empty configuration $\downarrow\perp$ of Unf_Σ satisfies φ, i.e., if $\downarrow\perp \models \varphi$. The *future fragment* $\mathcal{L}^+$ of $\mathcal{L}$ consists of all formulae without past-operator $\diamondsuit$.

4 Factorisation of the Unfolding

In general, the unfolding of a net system is infinite, even if the net is finite-state. Therefore, many model checking algorithms cannot directly be applied on a modal logic defined over the unfolding. One way to overcome this problem is to look for a factorisation of the unfolding by a decidable equivalence relation $\equiv$ that is finer than the distinguishing power of the formula to be evaluated, i.e., $C \equiv C'$ shall imply $C \models \varphi \Leftrightarrow C' \models \varphi$. The second requirement on $\equiv$ is that a set of representatives of its finitely many equivalence classes and a representation of the (transition) relations between the classes can be computed effectively. Then we can decide $C \models \varphi$ on *Unf* by transferring the question to the model checking problem $(C/\equiv) \models \varphi$ on $(Unf/\equiv, \longrightarrow)$.

The finite prefix. The first construction of an appropriate finite factorisation was given by McMillan [McM92]. He showed how to construct a finite prefix of the unfolding of a safe, i.e. finite-state, net system in which every reachable marking is represented by some cut. In terms of temporal logic, his approach means to consider formulae of the type $\Diamond\,\psi$ where $\Diamond$ is "global reachability" and ψ is a boolean combination of atomic propositions P. The key to the construction is that if the prefix contains several events with $\mathcal{M}$-equivalent local configurations, then their futures are isomorphic, i.e., they cannot be distinguished by the logic. Consequently, only one of them needs to be explored further, while the others become *cutoff* events. The *finite prefix Fin* is that initial part of the unfolding, that contains no causal successor of any cutoff, i.e., an event e' belongs to *Fin* iff no event $e < e'$ is a cutoff.

In general, the formal definition of a cutoff requires two crucial relations on configurations: An instance of the equivalence relation $\equiv$, and a partial order $\prec$. On the one hand, this partial order shall ensure that the expanded prefix contains a representative for each equivalence class. On the other hand, it shall guarantee that the prefix remains finite. The requirements for an *adequate* partial order $\prec$ (in conjunction with $\mathcal{M}$-equivalence) were examined very detailed in [ERV96]. They are as follows: it must be well-founded, it must respect set inclusion ($C \subset C'$ implies $C \prec C'$), and it must be preserved under *finite extensions*, i.e., if $C \equiv C'$ and $C \prec C'$ then $C \oplus E' \prec C' \oplus I_C^{C'}(E')$.

Such an adequate partial order is particularly useful, if it is *total*, such that for each two equivalent local configurations $\downarrow e \equiv \downarrow e'$ either e or e' can be discriminated as a cutoff. For 1-safe nets, a total order satisfying the above requirements was defined in [ERV96], yielding a minimal prefix.

In [McM92,ERV96] just $\mathcal{M}$-equivalence is considered. In conjunction with an adequate order $\prec$, the definition of *Fin* guarantees that each reachable marking is represented by the state of a configuration contained in *Fin*.

It was already observed in [HNW98b] that refining $\mathcal{M}$-equivalence yields an *extended prefix*, which – although being possibly larger than the prefix of [McM92,ERV96] – allows to apply a standard μ-calculus model checker for a location based modal logic called the *distributed μ-calculus*. We defined an equivalence $\equiv_{\mathcal{M}\text{-}loc}$ by $\downarrow e \equiv_{\mathcal{M}\text{-}loc} \downarrow e'$ iff $\downarrow e \equiv_{\mathcal{M}} \downarrow e'$ and $loc(e) = loc(e')$, and proved that $\equiv_{\mathcal{M}\text{-}loc}$-equivalence equals the distinguishing power of the distributed μ-calculus.

Generalised cutoffs. Now we look for more general conditions on equivalence relations that ensure that all equivalence classes can be computed by a prefix construction. Let us call a decidable equivalence relation $\equiv$ on configurations of *Unf* to be *adequate* if it refines $\mathcal{M}$-equivalence and has finite index. I.e., $C \equiv C'$ implies $C \equiv_{\mathcal{M}} C'$ and $\equiv$ has only finitely many equivalence classes on *Unf*. We give a generalised definition of a *cutoff* event by

$$e \in E \text{ is a } \textit{cutoff} \quad \text{iff} \quad \exists e' \in E, \text{ such that } \downarrow e' \equiv \downarrow e \text{ and } \downarrow e' \prec \downarrow e$$

where $\equiv$ is an adequate equivalence relation and $\prec$ is an adequate partial order. The *finite prefix Fin* constructed for $\equiv$ is given by the condition: e' belongs to *Fin* iff no event $e < e'$ is a cutoff. It is obvious from the cutoff definition that *Fin* constructed for $\equiv$ contains a representative for each $\equiv$-class of *Unf*.

Proposition 1. *The prefix Fin constructed for an adequate $\equiv$ is finite.*

An adequate equivalence finer than $\mathcal{L}$. In difference to S4 as used in [Esp94] and the distributed μ-calculus in [HNW98b], an equivalence finer than the distinguishing power of $\mathcal{L}$ has infinite index. However, by each finite set of $\mathcal{L}$-formulae we can only distinguish finitely many classes of configurations. Thus we can hope for a model checking procedure following the outline from the beginning of the section, if we find an equivalence which is at least as discriminating as the Fisher-Ladner-closure of a $\mathcal{L}$-formula φ, because this is the set of formulae relevant for model checking φ on *Unf*. First, we need some technical definitions.

Let us denote the *gossip-past-depth* of a formula $\varphi \in \mathcal{L}$ by $gpd(\varphi)$. It shall count how often in the evaluation of φ we have to change the local view or to go back into the local past. The inductive definition is

$$\begin{array}{ll} gpd(p) = 1 & gpd(\neg\varphi) = gpd(\varphi) \\ gpd(\varphi \vee \psi) = \max\{gpd(\varphi), gpd(\psi)\} & gpd(\Diamond\varphi) = gpd(\varphi) \\ gpd(@i\colon \varphi) = gpd(\varphi) + 1 & gpd(\Diamond\!\!\!\!-\, \varphi) = gpd(\varphi) + 1 \end{array}$$

Now we are ready to define the crucial equivalence relation $\equiv_i^n$, which is the basis for model checking $\mathcal{L}$. It is parameterised by a natural number n (which will be the gossip-past-depth of a given formula) and by a location i (at which the formula is interpreted). Formally, we define $\equiv_i^n \subseteq \mathcal{C}_{loc}^i \times \mathcal{C}_{loc}^i$ to be the coarsest equivalence relation satisfying:

$$\begin{array}{lll} \downarrow e \equiv_i^0 \downarrow f & \text{implies} & \forall p \in P_i \,.\; p \in \mathcal{M}(\downarrow e) \;\Leftrightarrow\; p \in \mathcal{M}(\downarrow f) \\ \downarrow e \equiv_i^1 \downarrow f & \text{implies} & \forall j,k \in Loc \,.\; \downarrow^j e \subseteq \downarrow^k e \;\Leftrightarrow\; \downarrow^j f \subseteq \downarrow^k f \end{array}$$

and for all $n \geq 0$ moreover

$$\begin{array}{rl} {\downarrow} e \equiv_i^{n+1} {\downarrow} f & \text{implies } \forall j \in Loc \,.\, {\downarrow}^j e \;\equiv_j^n\; {\downarrow}^j f \\ & \textit{(*) and } \forall e' \in ({\downarrow} e \cap E_i) \,.\, \exists f' \in ({\downarrow} f \cap E_i) \,.\, {\downarrow} e' \equiv_i^n {\downarrow} f' \\ & \quad \textit{and } \forall f' \in ({\downarrow} f \cap E_i) \,.\, \exists e' \in ({\downarrow} e \cap E_i) \,.\, {\downarrow} e' \equiv_i^n {\downarrow} f' \end{array}$$

The first condition is an i-localised version of $\mathcal{M}$-equivalence. The second one refers to the *latest information* concerning agents other than i, and the third condition inductively lifts the equivalence with respect to the levels of the gossip-past-depth. Let us briefly collect some important facts about the equivalence.

Observation 2. *The equivalence relation $\equiv_i^n$ is decidable and of finite index for every $n \geq 0$. Furtheron, $\equiv_i^{n+1}$ is refining $\equiv_i^n$, i.e., $\equiv_i^{n+1} \subseteq \equiv_i^n$ for all n. Finally, it respects $\mathcal{M}$-equivalence, i.e., ${\downarrow} e \equiv_i^n {\downarrow} f$ implies $\mathcal{M}({\downarrow} e) = \mathcal{M}({\downarrow} f)$ for all $n > 0$.*

Remark 3. Note that the last two lines of the third condition after (*) can be omitted if we restrict ourselves to the (still very useful) sublanguage $\mathcal{L}^+$, yielding considerable savings: With this condition, the number of equivalence classes of $\equiv_i^n$ may grow non-elementarily with n, forbidding any consideration of practicability, whereas without this condition the number of equivalence classes grows exponentially with n.

The most important property of the equivalence used in the proof of the main result is that it is preserved by local successors, as stated in Lemma 4.

Lemma 4. *Let $e \leq e'$, and $f \leq f'$ be i-events, such that ${\downarrow} e \;\equiv_i^n\; {\downarrow} f$, and let I be the isomorphism from $\beta({\downarrow} e)$ onto $\beta({\downarrow} f)$. If $f' = I(e')$ then also ${\downarrow} f' \equiv_i^n {\downarrow} e'$.*

Proof. This the most involved proof, and a main result of the paper. Please note that (for reasons of readability) the proof given here only deals with the pure future fragment $\mathcal{L}^+$ of the logic $\mathcal{L}$, i.e. the third condition of the definition of the equivalence relation $\equiv_i^n$ has to be read without the last two lines after the (*). For the (even more involved) proof for the full logic $\mathcal{L}$, i.e., inclusive the condition (*) of the $\equiv_i^n$ definition, see [HNW98a].

Let us define some notions and notations: Since we will often talk about a number of view changes in sequence, we introduce "paths" through the locations of the system: Let $\sigma = l_1 l_2 \ldots l_n$ be a sequence of locations (called *location path*), i.e., $l_j \in Loc$ for all $1 \leq j \leq n$. Given any configuration C, we define ${\downarrow}^\sigma C := {\downarrow}^{l_1}({\downarrow}^{l_2}(\ldots({\downarrow}^{l_n} C)\ldots))$. We set ${\downarrow}^\varepsilon e := {\downarrow} e$, where ε is the (empty) sequence of length 0. Note that a location path may include repetitions, i.e., $l_i = l_j$ for $i \neq j$ is allowed. Given an event g and some location path σ, we denote by g_σ the event that determines the σ-view of ${\downarrow} g$, i.e., ${\downarrow}^\sigma g = {\downarrow} g_\sigma$.

Now let $e \leq e'$ and $f \leq f'$ be events of E_i, and $n \geq 1$, as in the assumptions of the Lemma. First of all, we note that the required isomorphism I exists because $\equiv_i^n$-equivalence implies $\mathcal{M}$-equivalence.

We have to show ${\downarrow} f' \equiv_i^n {\downarrow} e'$. A key observation is the following: for every location path σ, it holds that if $e'_\sigma \not\leq e$ then $I(e'_\sigma) = f'_\sigma \not\leq f$.

This is the basis for the induction on $m \leq n$: for each sequence σ of length $n-m$ with $e'_\sigma \not\leq e$ (and also $f'_\sigma \not\leq f$), it holds that $\downarrow e'_\sigma \equiv^m_j \downarrow f'_\sigma$, where j is either the first location occurring in the sequence σ (if $n > m$), or $j := i$ (if $n = m$ and the empty sequence ε is the only sequence of length $n - m$). In the latter case, $\downarrow^i e' = \downarrow e'$ (because $e' \in E_i$), and $\downarrow^i f' = \downarrow f'$, we thus obtain $\downarrow e' \equiv^n_i \downarrow f'$ as required. The induction relies on a case analysis according to the following cases: $m = 0$, $n = m = 1$, $n = m > 1$, $n > m = 1$, and finally $n > m > 1$.

• For $m = 0$ we have to show that $\downarrow e'_\sigma \equiv^0_j \downarrow f'_\sigma$. This is clear, because $I(e'_\sigma) = f'_\sigma \in E_j$ and thus the j-local part of the markings of $\downarrow e'_\sigma$ and $\downarrow f'_\sigma$ coincide, because $\pi(e'_\sigma)^\bullet = \pi(f'_\sigma)^\bullet$.

• For $n = m = 1$ we have to show that $\downarrow e \equiv^1_i \downarrow f$ implies $\downarrow e' \equiv^1_i \downarrow f'$, i.e., (1) for all $j \in Loc$: $\downarrow^j e' \equiv^0_j \downarrow^j f'$, and (2) for all $j, k \in Loc$: $e'_j \leq e'_k$ iff $f'_j \leq f'_k$.
If $e'_j \leq e$ then $\downarrow e' \setminus \downarrow e$ contains no j-event, which means that $e'_j = e_j$ and similarly $f'_j = f_j$, so (1) follows easily. If $e'_j \not\leq e$ then also $f'_j \not\leq f$, in which case $\downarrow^j e' \equiv^0_j \downarrow^j f'$ follows by induction.
So consider (2). Let $j, k \in Loc$. We show that $e'_j \leq e'_k$ iff $f'_j \leq f'_k$, using a similar case analysis. If $e'_j, e'_k \not\leq e$, then the isomorphism preserves the order. If $e'_j, e'_k \leq e$, then $e'_j = e_j$ and $e'_k = e_k$, (and similarly $f'_j = f_j$, $f'_k = f_k$), and so the order is inherited from the corresponding local views of $\downarrow e$ and $\downarrow f$, which by assumption match. The third case is $e'_j \leq e$, but $e'_k \not\leq e$, and thus similarly $f'_j \leq f$, but $f'_k \not\leq f$. Since this is the most sophisticated argument and used also in the other cases, the situation is illustrated in Figure 3. $e'_j \leq e$ implies $e'_j = e_j$. Now we choose an $l \in Loc$, such that $e_j \leq e_l \leq e'_k$, and moreover e_l is (causally)

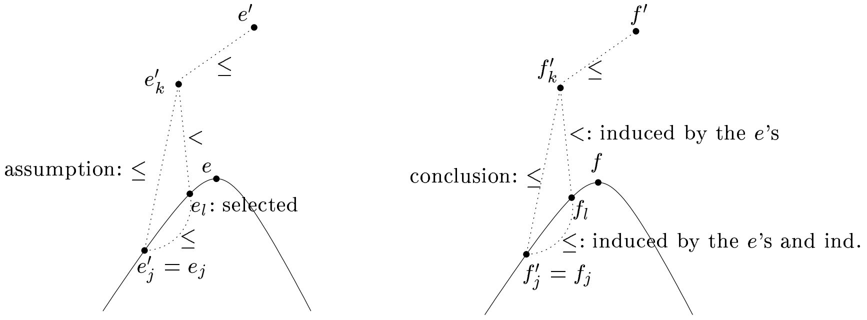

Fig. 3. Situation: $e'_k \not\leq e$ and $e'_j \leq e$

maximal with this respect. For at least one of the possible choices of l, there exists an event $e'' \in E_l$, such that $e'' \in (\downarrow^k e' \setminus \downarrow e)$. By the isomorphism, we have that $I(e'') = f'' \in (\downarrow^k f' \setminus \downarrow f)$. By assumption on the equivalence of e and f we can conclude $f'_j = f_j \leq f_l \leq f'_l \leq f'_k$, i.e., $\downarrow^j f' \subseteq \downarrow^k f'$ as desired.

• For $n = m > 1$ the reasoning is similar to the case $n = m = 1$, except that the argument for the gossip aspect of the equivalence is not needed.

• For $n > m = 1$, let $\sigma = (i'\sigma')$ be a sequence of length $n - 1$ with $e'_\sigma \not\leq e$.

Again, we have to show $\downarrow^\sigma e' \equiv^1_{i'} \downarrow^\sigma f'$. Let $j \in Loc$. For the case of $e'_{j\sigma} \not\leq e$ the $\equiv^0_j$-equivalence is a consequence of $I(e'_{j\sigma}) = f'_{j\sigma}$. For $e'_{j\sigma} \leq e$ there exists again an $l \in Loc$ with $e'_{j\sigma} \leq e_l \leq e'_\sigma$, so that e_l is maximal in this respect, and as above we also obtain $f'_{j\sigma} \leq f_l \leq f'_\sigma$. Moreover, in this case it holds that $e_{jl} = e'_{j\sigma}$ and similarly $f_{jl} = f'_{j\sigma}$. By assumption, we have $\downarrow(e_{jl}) \equiv^{n-2}_j \downarrow(f_{jl})$, and because of $n \geq 2$, in particular $\downarrow(e_{jl}) \equiv^0_j \downarrow(f_{jl})$, as desired.
The argument concerning the relative orders of j-views and k-views of e_σ and e'_σ is the same as for the case of $n = m = 1$.
• For $n > m > 1$ let σ be of length $n - m$, such that σ has j as first element, and such that $e'_\sigma \not\leq e$, and similarly $f'_\sigma \not\leq f$. We have to show that for each $k \in Loc$ it holds that $\downarrow^k e'_\sigma \equiv^{m-1}_k \downarrow^k f'_\sigma$. For $e'_{k\sigma} \not\leq e$ and similarly $f'_{k\sigma} \not\leq f$ this follows from the induction hypothesis. For $e'_{k\sigma} \leq e$ there exists (again) a location l, such that $e'_{k\sigma} \leq e_l \leq e'_\sigma$ and e_l is causally maximal in this respect. Then $\downarrow^k e'_\sigma = \downarrow^k e_l \equiv^{n-2}_k \downarrow^k f_l = \downarrow^k f'_\sigma$, where $n - 2 \geq m - 1$, so that the desired claim follows from the observation $\equiv^{\tilde{n}+\tilde{m}}_k \subseteq \equiv^{\tilde{n}}_k$. □

Theorem 5. *Let φ be an $\mathcal{L}$-formula of gossip-past-depth n, and let $e, f \in E_i$ with $\downarrow e \equiv^n_i \downarrow f$. Then $\downarrow e \models_i \varphi$ iff $\downarrow f \models_i \varphi$.*

Proof. By structural induction on φ: For atomic propositions, note that $\downarrow e \equiv^1_i \downarrow f$ implies $\downarrow e \equiv_{\mathcal{M}} \downarrow f$ (cf. Observation 2), and hence $\downarrow e \models_i p$ iff $\downarrow f \models_i p$. The induction for boolean connectives is obvious.

For $gpd(\Diamond\varphi) = gpd(\varphi) = n$ let $\downarrow e \models_i \Diamond\varphi$ and $\downarrow e \equiv^n_i \downarrow f$. We have to show that also $\downarrow f \models_i \Diamond\varphi$ (all other cases follow by symmetry). By definition, there exists $e' \geq e$ with $e' \in E_i$ and $\downarrow e' \models_i \varphi$. By Lemma 4 the event $f' = I(e') \in E_i$ obtained from the isomorphism I due to the $\mathcal{M}$-equivalence of $\downarrow e$ and $\downarrow f$ satisfies $f \leq f'$ and $\downarrow e' \equiv^n_i \downarrow f'$. By induction, $\downarrow f' \models_i \varphi$ and finally $\downarrow f \models_i \Diamond\varphi$.

Now let $\varphi = @j : \psi$ with $gpd(\varphi) = gpd(\psi) + 1 = n$. If $\downarrow e \models_i \varphi$ then $\downarrow^j e \models_j \psi$, and by definition $\downarrow^j e \equiv^{n-1}_j \downarrow^j f$. Thus, by induction, $\downarrow^j f \models_j \psi$, and finally $\downarrow f \models_i \varphi$.

Finally, let $\varphi = \diamondminus \psi$, with $gpd(\psi) = n-1$, and $\downarrow e \models_i \diamondminus \psi$, i.e., there exists an event $e' \in E_i$, s.t. $e' \leq e$ and $\downarrow e' \models_i \psi$. Due to the third condition (*), there exists an $f' \in E_i$, s.t. $f' \leq f$ and $\downarrow f' \equiv^{n-1}_i \downarrow e'$. Hence, by induction, also $\downarrow f' \models_i \psi$, and thus $\downarrow f \models_i \varphi$. □

Based on the local equivalences, we define an adequate equivalence relation for the construction of a finite prefix by $\downarrow e \equiv^n \downarrow f$ iff $loc(e) = loc(f)$ and $\downarrow e \equiv^n_i \downarrow f$ for all $i \in loc(e)$. The next and last step to transfer the $\mathcal{L}$ model checking problem from the unfolding to an equivalent model checking problem over a finite structure is the definition of the transitions between the $\equiv^n$-equivalence classes of *Unf*. This is done in the next section.

5 Model checking

In this section we propose a verification technique for $\mathcal{L}$. Following the lines of [HNW98b], we will sketch a reduction of a given instance of the problem to a suitable input for well investigated model checkers like e.g. [CES86].

Let us consider a distributed net system Σ and an $\mathcal{L}$-formula φ of gossip-past-depth n. We have shown so far how to construct a finite prefix *Fin* of the unfolding Unf_Σ that contains representatives for all $\equiv_i^n$ equivalence classes. Now we want to compute a finite, multi modal Kripke structure on the representatives that is equivalent to Unf_Σ with respect to the evaluation of φ. What is missing are the transitions between the representatives.

Computing a finite Kripke structure. Let $n \in \mathbb{N}$, and $\mathit{Unf}_\Sigma = (N', \pi)$ with $N' = (B, E, F)$ be fixed, and let $\equiv^n$ be the equivalence relation used for the construction of *Fin*. The state space $\mathcal{S}_n$ of the desired Kripke structure consists of one representative of each $\equiv^n$ equivalence class. Note that by using the adequate *total* partial order $\prec$ of [ERV96], these representatives are unique, and so the state space is given by $\mathcal{S}_n := \{{\downarrow}e \mid e \in \mathit{Fin} \text{ and } e \text{ is not a cutoff}\}$. If the used order $\prec$ is not total, we fix one non-cutoff (resp. its local configuration) of the prefix as the representative of each $\equiv^n$ equivalence class. For every local configuration ${\downarrow}e$ of Unf_Σ, let $\mathit{rep}({\downarrow}e) \in \mathcal{S}_n$ denote the unique representative.

Now let us consider the transitions of the Kripke structure. We introduce a transition relation for each of the modalities of the logic. Let ${\downarrow}e, {\downarrow}f \in \mathcal{S}_n$.

$$\begin{aligned} {\downarrow}e \xrightarrow{\Diamond i}_n {\downarrow}f &\text{ iff } e, f \in E_i \text{ and } \exists f' \in E_i \,.\, f' \geq e \wedge \mathit{rep}({\downarrow}f') = {\downarrow}f \\ {\downarrow}e \xrightarrow{@j} {\downarrow}f &\text{ iff } e \in E_i,\, f \in E_j \wedge {\downarrow}^j e = {\downarrow}f \\ {\downarrow}e \xrightarrow{\diamondminus i} {\downarrow}f &\text{ iff } e, f \in E_i \wedge f \leq e \end{aligned}$$

Note that the definitions of $\xrightarrow{@j}$ and $\xrightarrow{\diamondminus i}$ rely on the fact that the set of configurations in *Fin* (and thus also in $\mathcal{S}_n$) is downward closed, i.e., the j-view of any element of $\mathcal{S}_n$ is again in $\mathcal{S}_n$ for every j, and of course past configurations as well. On the whole, we obtain the multi modal Kripke structure $\mathcal{T}_n = (\mathcal{S}_n, \{\xrightarrow{\Diamond i}_n, \xrightarrow{@i}, \xrightarrow{\diamondminus i} \mid i \in \mathit{Loc}\}, {\downarrow}\bot)$ with root ${\downarrow}\bot$.

As a corollary to Theorem 5 we obtain the following characterisation of the semantics of $\mathcal{L}$ formulae over $\mathcal{T}_n$:

Corollary 6. *Let $\varphi \in \mathcal{L}$ be a formula of gossip-past-depth $m \leq n$, and let ${\downarrow}e \in \mathcal{S}_n$ be an i-local configuration, i.e., $e \in E_i$.*

1. *If $\varphi = \Diamond\psi$ then ${\downarrow}e \models_i \varphi$ iff $\exists\, {\downarrow}f \in \mathcal{S}_n$ with ${\downarrow}e \xrightarrow{\Diamond i}_n {\downarrow}f$ and ${\downarrow}f \models_i \psi$.*
2. *If $\varphi = @j{:}\psi$ then ${\downarrow}e \models_i \varphi$ iff $\exists\, {\downarrow}f \in \mathcal{S}_n$ with ${\downarrow}e \xrightarrow{@j} {\downarrow}f$ and ${\downarrow}f \models_j \psi$.*
3. *If $\varphi = \diamondminus\,\psi$ then ${\downarrow}e \models_i \varphi$ iff $\exists\, {\downarrow}f \in \mathcal{S}_n$ with ${\downarrow}e \xrightarrow{\diamondminus i} {\downarrow}f$ and ${\downarrow}f \models_i \psi$.*

Proof. (1) follows from the semantics of $\Diamond$ and the fact that by construction of $\mathcal{T}_n$ for any pair of states ${\downarrow}f'$ and ${\downarrow}f = \mathit{rep}({\downarrow}f')$, we have that ${\downarrow}f \models_i \varphi$ iff ${\downarrow}f' \models_i \varphi$ for any formula φ with $\mathit{gpd}(\varphi) = m \leq n$. (2) and (3) are trivial. $\square$

Thus, if we are able to actually compute (the transitions of) $\mathcal{T}_n$ then we can immediately reduce the model checking problem of $\mathcal{L}$ to a standard model checking problem over finite transition systems, applying e.g. [CES86].

Computing the transitions ${\downarrow}e \xrightarrow{@j} {\downarrow}f$ in $\mathcal{T}_n$ is trivial: ${\downarrow}f = {\downarrow}^j e$. Similarly computing the $\xrightarrow{\diamondminus i}$ *successors* of ${\downarrow}e$ is very easy. It is more difficult to compute the transitions ${\downarrow}e \xrightarrow{\Diamond i}_n {\downarrow}f$, if only *Fin* is given. To achieve this, we use a modified version of the algorithm proposed in [HNW98b].

An algorithm to compute the $\xrightarrow{\diamond i}_n$ transitions. We assume in the following that the algorithm for constructing the prefix *Fin* uses a total, adequate order $\prec$. The construction of *Fin* provides some useful structural information: each cutoff e has a *corresponding* event e^0, such that $\downarrow e^0 \equiv^n \downarrow e$, and $\downarrow e^0 \prec \downarrow e$. Clearly, we choose $rep(\downarrow e) := \downarrow e^0$ for each cutoff e, and for non-cutoffs f, we set $rep(\downarrow f) := \downarrow f$. For technical reasons, we extend the definition of $\xrightarrow{\diamond i}_n$: we define $C \xrightarrow{\diamond i}_n \downarrow e$ for any local *or* global configuration $C \subseteq \downarrow e'$, with $rep(\downarrow e') = \downarrow e$ and $e, e' \in E_i$. The construction of *Fin* also provides a function *shift**, which maps any configuration $C = C_1$ of Unf_Σ containing some cutoff, onto a configuration *shift*$^*(C) = C_m$ not containing a cutoff, hence being present in *Fin*. This function works by repeatedly applying $C_{k+1} := \downarrow e_k^0 \oplus I_{\downarrow e_k}^{\downarrow e_k^0}(C_k \setminus \downarrow e_k)$ with $e_k \in C_k$ being a cutoff of *Fin*, and e_k^0 being its corresponding, equivalent event. This iterative application terminates, because the sequence $C_1, C_2, ..$ decreases in the underlying (well-founded) order $\prec$. Obviously, this function implies the existence of an isomorphism I between $\beta(C)$ and $\beta(shift^*(C))$, which is the composition of the isomorphisms $I_{\downarrow e_k}^{\downarrow e_k^0}$ induced by the chosen cutoff events. Moreover, $shift^*(\downarrow e) \prec \downarrow e$ for any $e \in \beta(C)$, and hence for any e for which $C \xrightarrow{\diamond i}_n \downarrow e$.

The most important part of the algorithm (cf. Fig. 4) is the recursive procedure *successors* which, when called *from the top level* with a pair $(\downarrow e, i)$, returns the $\xrightarrow{\diamond i}_n$ -successors of $\downarrow e$ in the finite structure. More generally, *successors* performs a depth first search through pairs (C, i), where C is an arbitrary, not necessarily local configuration not containing a cutoff and i is a location. It determines the subset of local configurations in $\mathcal{S}_n$ that represent the $\xrightarrow{\diamond i}_n$ -successors of C. Formally, $\downarrow e \in successors(C, i)$ iff there exists $\downarrow e'$ in *Unf*, which is $\equiv^n$-equivalent to $\downarrow e$, and $C \xrightarrow{\diamond i}_n \downarrow e'$.

Proposition 7. *Compute_Multi_Modal_Kripke_Structure computes the* $\xrightarrow{\diamond i}_n$ *-,* $\xrightarrow{\diamond i}$ *-, and* $\xrightarrow{@j}$ *-transitions.*

The proof can be found in [HNW98a]. Note that at top level, *successors* is always called with a *local* configuration $\downarrow e$ as parameter, but the extension of $\downarrow e$ with cutoffs requires that we can also handle global configurations. In this paper, we focus on decidability but not on efficiency. For heuristics on efficiency improvements we refer the reader to [HNW98b].

6 Conclusion

We have shown the decidability of the model checking problem for $\mathcal{L}$, a location based branching-time temporal logic including temporal and gossip modalities. The method is based on a translation of the modalities over net unfoldings (or prime event structures) into transitions of a sequential transition system, for which established model checkers for sequential logics can be applied.

While the method as presented is non elementary for the full logic $\mathcal{L}$, the restriction to the future fragment $\mathcal{L}^+$ has "only" exponential complexity but still allows to express interesting properties.

```
type Vertex = {$C$: Configuration; $i$: Location; pathmark: bool; (* for dfs *) }

$prefix\_successors(C,i) = \{rep(\downarrow e) \mid \downarrow e \in \mathcal{S}_n \;\wedge\; C \xrightarrow{\diamond i}_n \downarrow e\}$
$compatible\_cutoffs(C) = \{e \mid e$ is cutoff and $\downarrow e \cup C$ is a configuration in $Fin\}$

proc $successors(C,i)$: ConfigurationSet;
{   var result: ConfigurationSet;             (* result accumulator for current vertex *)
    Vertex v := $findvertex(C,i)$;            (* lookup in hash table, if not found then *)
                                              (* create new vertex with pathmark = false *)
    if  v.pathmark then return $\emptyset$; fi     (* we have closed a cycle *)
    result := $prefix\_successors(C,i)$;      (* directly accessible successors *)
    v.pathmark:=true;                         (* put vertex on path *)
    for $e_c \in compatible\_cutoffs(C)$ do   (* find successors outside Fin behind $e_c$ *)
        result := result $\cup$ $successors(shift^*(C \cup \downarrow e_c), i)$;
    od ;
    v.pathmark:=false;                        (* take vertex from path *)
    return result;
}

proc $Compute\_Multi\_Modal\_Kripke\_Structure$;
{   InitializeTransitionSystem($\mathcal{T}_n, Fin$);   (* extract state space from Fin *)
    for $\downarrow e \in \mathcal{S}_n, i \in Loc$ do
        add transition $\downarrow e \xrightarrow{@i} \downarrow^i e$;
    for $i \in Loc, \downarrow e, \downarrow f \in \mathcal{S}_n \cap \mathcal{C}^i_{loc}, \downarrow f \subseteq \downarrow e$ do
        add transition $\downarrow e \xrightarrow{\ominus i} \downarrow f$;
        for $\downarrow e' \in successors(\downarrow e, i)$ do
            add transition $\downarrow e \xrightarrow{\diamond i}_n \downarrow e'$;
        od
    od
}
```

Fig. 4. The conceptual algorithm to compute the transitions of $\mathcal{T}_n$.

We also hope that the presented results can be used as a methodological approach to model checking temporal logics of *causal knowledge* [Pen98].

The main difficulty, the solution of which is also the major contribution of the paper, was to find an adequate equivalence relation on local states that allows to construct a finite transition system containing a representative for each class of equivalent local states. If the method really is to be applied, then refinements of the equivalence bring it closer to the logical equivalence and thus leading to a smaller index will be crucial. We believe that the potential for such improvements is high at the price of much less understandable definitions.

For the treatment of past an alternative and potentially more efficient approach in the line of [LS95] – elimination of past modalities in CTL – might come to mind, but the techniques used there can at least not directly be transferred to $\mathcal{L}_{CSA}$ because of the intricate interaction between past and gossip modalities.

References

[CEP95] A. Cheng, J. Esparza, and J. Palsberg, *Complexity results for 1-safe nets*, Theoretical Computer Science (1995), no. 147, 117–136.

[CES86] E.M. Clarke, E.A. Emerson, and A.P. Sistla, *Automatic verification of finite-state concurrent systems using temporal logic specifications*, ACM Transactions on Programming Languages and Systems **8** (1986), no. 2, 244–263.

[Eng91] J. Engelfriet, *Branching processes of Petri nets*, Acta Informatica **28** (1991), 575–591.

[ERV96] J. Esparza, S. Römer, and W. Vogler, *An Improvement of McMillan's Unfolding Algorithm*, Proc. of TACAS '96, LNCS, vol. 1055, Springer, 1996, 87–106.

[Esp94] J. Esparza, *Model checking using net unfoldings*, Science of Computer Programming **23** (1994), 151–195.

[HNW98a] M. Huhn, P. Niebert, and F. Wallner, *Model checking gossip modalities*, Technical Report 21/98, Univ. Karlsruhe, Fakultät für Informatik, Aug. 1998.

[HNW98b] M. Huhn, P. Niebert, and F. Wallner, *Verification based on local states*, Proc. of TACAS '98, LNCS, vol. 1384, Springer, 1998, 36–51.

[LRT92] K. Lodaya, R. Ramanujam, and P.S. Thiagarajan, *Temporal logics for communicating sequential agents: I*, Int. Journal of Foundations of Computer Science **3** (1992), no. 2, 117–159.

[LS95] F. Laroussinie and P. Schnoebelen, *A hierarchy of temporal logics with past*, Theoretical Computer Science **148** (1995), 303–324.

[LT87] K. Lodaya and P.S. Thiagarajan, *A modal logic for a subclass of event structures*, Automata, Languages and Programming, LNCS, vol. 267, Springer, 1987, 290–303.

[McM92] K.L. McMillan, *Using unfoldings to avoid the state explosion problem in the verification of asynchronous circuits*, Proc. of CAV '92, LNCS, vol. 663, Springer, 1992, 164–174.

[Nie98] Peter Niebert, *A temporal logic for the specification and verification of distributed behaviour*, PhD thesis, Institut für Informatik, Universität Hildesheim, March 1998.

[NPW80] M. Nielsen, G. Plotkin, and G. Winskel, *Petri nets, event structures and domains*, Theoretical Computer Science **13** (1980), no. 1, 85–108.

[NRT90] M. Nielsen, G. Rozenberg, and P.S. Thiagarajan, *Behavioural notions for elementary net systems*, Distributed Computing **4** (1990), no. 1, 45–57.

[Pen97] W. Penczek, *Model-Checking for a Subclass of Event Structures*, Proc. of TACAS '97, LNCS, vol. 1217, 1997.

[Pen98] W. Penczek, *Temporal logic of causal knowledge*, Proc. of WoLLiC '98, 1998.

[Ram95] R. Ramanujam, *A Local Presentation of Synchronizing Systems*, Structures in Concurrency Theory, Workshops in Computing, 1995, 264–279.

[Thi94] P.S. Thiagarajan, *A Trace Based Extension of PTL*, Proc. of 9th LICS, 1994.

[Thi95] P.S. Thiagarajan, *A Trace Consistent Subset of PTL*, Proc. of CONCUR '95, LNCS, vol. 962, Springer, 1995, 438–452.

[Wal98] F. Wallner, *Model checking LTL using net unfoldings*, Proc. of CAV '98, LNCS, vol. 1427, Springer, 1998.

[Win87] Glynn Winskel, *Event structures*, Advances in Petri Nets, LNCS, vol. 255, Springer, 1987.