Model checking distributed temporal logic

Francisco Dionísio
Dep. Matemática, Instituto Superior Técnico, Universidade de Lisboa, Portugal
Instituto de Telecomunicações, Portugal
francisco.dionisio@tecnico.ulisboa.pt

Jaime Ramos
Dep. Matemática, Instituto Superior Técnico, Universidade de Lisboa, Portugal
Instituto de Telecomunicações, Portugal
jaime.ramos@tecnico.ulisboa.pt

Fernando Subtil
Dep. Matemática, Instituto Superior Técnico, Universidade de Lisboa, Portugal
fmcsubtil@gmail.com

Luca Viganò
Department of Informatics, King's College London, London, UK
luca.vigano@kcl.ac.uk

March 14, 2024

Abstract

The distributed temporal logic DTL is a logic for reasoning about temporal properties of distributed systems from the local point of view of the system's agents, which are assumed to execute sequentially and to interact by means of synchronous event sharing. Different versions of DTL have been provided over the years for a number of different applications, reflecting different perspectives on how non-local information can be accessed by each agent. In this paper, we propose an automata-theoretic model checking algorithm for DTL. To this end, we propose a notion of distributed transition system that will be used to specify the system to be verified. The properties that the system should meet are specified in DTL. In order to capture the models of these properties, we propose the notions of generalized distributed Büchi automaton and of distributed Büchi automaton. With these concepts, we are able to adapt results from automata-theoretic approaches to model checking in LTL to the distributed case.

Keywords: Distributed Temporal Logic (DTL), Büchi automata, distributed systems, specification and verification, model checking.

1 Introduction

The *distributed temporal logic DTL* was introduced in [14, 15] as a logic for specifying and reasoning about distributed information systems. DTL allows one to reason about temporal properties of distributed systems from the local point of view of the system's agents, which are assumed to execute sequentially and to interact by means of synchronous event sharing.

In DTL, distribution is implicit and properties of entire systems are formulated in terms of the local properties of the agents and their interaction. The logic was shown to be decidable, as well as trace-consistent, which makes it suitable for model checking tasks.

Different versions of DTL have been given over the years for a number of different applications, reflecting different perspectives on how non-local information can be accessed by each agent. In particular, DTL has proved to be useful in the context of *security protocol analysis* in order to reason about the interplay between protocol models and security properties [9, 10, 5]. However, most of the results for security protocol analysis and for other case studies were obtained directly by semantic arguments.[1] To overcome this problem, a *labelled tableaux system* for DTL was proposed in [3, 4]. The main goal was to have a usable deductive system in which deductions followed closely semantic arguments, also thanks to the labelling of the formulas along with a labelling algebra capturing the different semantic properties.

The labelled tableaux system was proved to be sound and complete, but decidability was not considered in [3, 4] and the system included an infinite closure rule to capture eventualities that are always delayed. Hence, the labelled system proved to be quite hard to use in practice although several properties can still be proved using only the tableaux system. For instance, the correctness of the *two-phase commit protocol* is one of such examples where a decision procedure is not needed. The DTL specification for a simplified version of the protocol as well as a proof of correctness using labelled tableaux can be found in [4].

DTL was shown to be decidable via a translation to *linear temporal logic (LTL)*. However, when translating DTL specifications into LTL specifications, we lose one of the main advantages of DTL, namely the naturalness of the distributed essence of DTL, which allows for simpler and clearer specifications. Later, in [8], a *decidable tableaux system* was proposed for DTL. The tableaux system was built on top of a tableaux system for LTL as presented in [17]. Similar systems for LTL have also been proposed, e.g., [19]. In the case of DTL, the tableaux system integrated in a smooth way both the usual rules for the temporal operators and rules for tackling the specific communication features of DTL.

In this paper, we propose a model checking algorithm for DTL. This is a first step towards a tool for automated verification of distributed systems specified using DTL. Nowadays, IT systems are becoming more and more pervasive but at the same time more and more complex, which makes the task of verifying them considerably harder. Model checking stands out as an approach well suited for automatic verification, which has been successfully used in industry with several well documented cases, such as [22, 16, 2] to name just a few. Depending on the temporal logic considered [25, 12], the approach to model checking is different [11, 13, 32, 30]. The first successful approaches to LTL model checking where proposed in [18] and later developed in [31]. In the case of DTL, we adopt an approach closer to the usual automata-theoretic approaches to model checking in LTL [31, 32]. Our goal is to use Büchi automata to capture DTL models. To this end, we propose a distributed version of these automata and show how to use them for our purposes. We start by presenting the traditional notions of nondeterministic Büchi automaton and generalized nondeterministic Büchi automaton, and we use these notions to capture the local behaviour of the agents, given that each agent is linear. In this case, we follow the ideas discussed in [31, 1]. Then, we propose the notion of *distributed Büchi automaton* to capture the distributed nature of DTL. Similar constructions

[1] DTL is closely related to the family of temporal logics whose semantics are based on the models of true concurrency introduced and developed in [20, 21, 26]. In particular, the semantics of these logics are based on a conflict-free version of Winskel's event structures [33], enriched with information about sequential agents.

have been proposed, for instance, in [23, 29]. In [23], the notion of *direct product automata* requires that acceptance states of the product construction be products of acceptance states of the component automata. Additionally, the notion of *synchronized product automaton* is defined for transition systems (not for automata) and the acceptance states are added afterwards. In [29], the global initial states are external to the product construction and Büchi automata have finitary acceptance states as well as infinitary acceptance states. In our case, we do not consider finitary acceptance states, but, in general, the purpose of these constructions is similar. Note that, capitalizing on the translation of DTL to LTL, as observed above, we could use any model checker for LTL for model checking DTL. However, herein we propose a new model checking algorithm specifically tailored for DTL.

There are other temporal logics similar to DTL for reasoning about distributed systems. For instance, in [24] a hybrid logic $\mathcal{HL}^*$ is proposed for spatial reasoning. This logic is then *temporalized* into a logic called $\mathcal{LSTL}$ that has the ability to reason about temporal properties of distributed systems. However, in this case it is proved that this logic is sound and complete with respect to an axiomatization for reasoning about such systems but no model checking results are provided. In [7], HyLTL is proposed as logic for model checking hybrid systems. In this case, HyLTL is used for specifying complex properties of hybrid systems and model checking is addressed by translating a formula into an equivalent hybrid automaton. It is important to stress that the goal of this work is not to advocate the advantages or disadvantages of the logic DTL. This logic has been extensively used in a wide range of problems. Our goal here is to provide this logic with a model checking algorithm in order to facilitate the verification properties of complex distributed systems that may be specified in DTL.

We proceed as follows. In Section 2, we briefly introduce DTL, its syntax, semantics and some auxiliary notions that will be useful later. In Section 3, we present distributed Büchi automata for DTL and prove the correctness of the construction with respect to the semantics of DTL. In Section 4, we present the notion of distributed transition system and establish the relevant results for encompassing DTL with a model checking algorithm. In Section 5, we conclude and discuss future work.

2 The Distributed Temporal Logic DTL

In this section, we briefly introduce the syntax and semantics of DTL.

2.1 Syntax

The logic is defined over a *distributed signature* $\langle Id, \{Prop_i\}_{i\in Id}\rangle$, where Id is a finite non-empty set (of *agent identifiers*) and, for each agent $i \in Id$, $Prop_i$ is a set of *local state propositions*, which intuitively characterize the current local states of the agents. We assume that $Prop_i \cap Prop_j = \emptyset$, for $i \neq j$. In some cases, it will be useful to consider $Id = \{1, \ldots, n\}$.

Definition 2.1 *The* local language $\mathcal{L}_i$ *of each agent* $i \in Id$ *is defined by*

$$\mathcal{L}_i ::= Prop_i \mid \neg\, \mathcal{L}_i \mid \mathcal{L}_i \Rightarrow \mathcal{L}_i \mid \mathsf{X}\, \mathcal{L}_i \mid \mathsf{G}\, \mathcal{L}_i \mid \copyright_j[\mathcal{L}_j] \text{ with } j \in Id.$$

We will denote such *local formulas* by the letters φ and ψ. As the name suggests, local formulas hold locally for the different agents. For instance, locally for an agent i, the operators X and

G are the usual *next (tomorrow)* and *always in the future* temporal operators, whereas the *communication formula* $\copyright_j[\psi]$ means that agent i is communicating (synchronizing) with agent j, for whom ψ holds. For the sake of simplicity, the temporal operator until U is not considered here. Other logical connectives (conjunction $\wedge$, disjunction $\vee$, true $\top$, etc.) and temporal operators (sometime in the future F) can be defined as abbreviations as usual.

Definition 2.2 *The* global language $\mathcal{L}$ *is defined by*

$$\mathcal{L} ::= @_i[\mathcal{L}_i] \mid \neg\,\mathcal{L} \mid \mathcal{L} \Rightarrow \mathcal{L} \text{ with } i \in Id.$$

We will denote *global formulas* by the letters α and β. A *global formula* $@_i[\varphi]$ means that the local formula φ holds for agent i. As before, other global logical connectives can be defined as abbreviations in the usual way.

In the sequel, we will need some auxiliary notions. For each agent $i \in Id$, the set of *i-literals* is the set of all state propositions and their negations $Lit_i = Prop_i \cup \{\neg\, p \mid p \in Prop_i\}$. An *i-valuation* v_i is a set of i-literals such that for each $p \in Prop_i$, $p \in v_i$ iff $\neg\, p \notin v_i$. The *set of all i-valuations* is denoted by $\mathcal{V}_i$. Observe that $\mathcal{V}_i \subseteq 2^{Lit_i}$. We will sometimes confuse the notion of valuation as a set of literals with its counterpart notion of valuation as a map $v_i : Prop_i \to \{0, 1\}$ such that $p \in v_i$ iff $v_i(p) = 1$. Furthermore, given a set of i literals L_i, we write $@_i[L_i]$ to denote the set of global formulas $\{@_i[l] \mid l \in L_i\}$.

Definition 2.3 *Let $\alpha \in \mathcal{L}$. The set of all* subformulas *of α, denoted by subf(α), is inductively defined as follows:*

- $\alpha \in subf(\alpha)$,
- *if* $\neg\,\alpha_1 \in subf(\alpha)$ *then* $\alpha_1 \in subf(\alpha)$, *for* $\alpha_1 \in \mathcal{L}$,
- *if* $\alpha_1 \Rightarrow \alpha_2 \in subf(\alpha)$ *then* $\alpha_1, \alpha_2 \in subf(\alpha)$, *for* $\alpha_1, \alpha_2 \in \mathcal{L}$,
- *if* $@_i[\neg\,\varphi] \in subf(\alpha)$ *then* $@_i[\varphi] \in subf(\alpha)$, *for* $i \in Id$ *and* $\varphi \in \mathcal{L}_i$,
- *if* $@_i[\varphi_1 \Rightarrow \varphi_2] \in subf(\alpha)$ *then* $@_i[\varphi_1], @_i[\varphi_2] \in subf(\alpha)$, *for* $i \in Id$ *and* $\varphi_1, \varphi_2 \in \mathcal{L}_i$,
- *if* $@_i[\mathsf{X}\,\varphi] \in subf(\alpha)$ *then* $@_i[\varphi] \in subf(\alpha)$, *for* $i \in Id$ *and* $\varphi \in \mathcal{L}_i$,
- *if* $@_i[\mathsf{G}\,\varphi] \in subf(\alpha)$ *then* $@_i[\varphi] \in subf(\alpha)$, *for* $i \in Id$ *and* $\varphi \in \mathcal{L}_i$,
- *if* $@_i[\copyright_j[\varphi]] \in subf(\alpha)$ *then* $@_j[\varphi] \in subf(\alpha)$, *for* $i, j \in Id$ *and* $\varphi \in \mathcal{L}_j$.

For each $i \in Id$, the set of i-subformulas *of α is the set $subf_i(\alpha) = \{@_i[\varphi] \mid @_i[\varphi] \in subf(\alpha)\}$.*

Observe that, in general, formulas involving only propositional symbols, connectives and temporal operators can always be assigned to an agent because $Prop_i \cap Prop_j = \emptyset$. However, formulas involving communication cannot be directly assigned to an agent. Therefore, in order to know the agent where such a formula should be evaluated, that information must be known. For this reason, and for the sake of uniformity, we maintain all subformulas in the scope of an operator $@_i[\cdot]$ to identify the agent to which a formula corresponds.

We illustrate these concepts with a simple example. The set $subf(@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])])$ is

$$\{@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[p], @_i[\copyright_j[q_1 \Rightarrow q_2]], @_j[q_1 \Rightarrow q_2], @_j[q_1], @_j[q_2]\}$$

and the set $subf_i(@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])])$ is

$$\{@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[p], @_i[\copyright_j[q_1 \Rightarrow q_2]]\}.$$

Observe that a formula $@_i[\copyright_j[\varphi]] \in subf_i(\alpha)$ has no further structure from the point of view of agent i and it is treated as atomic. However, we will have $@_j[\varphi] \in subf_j(\alpha)$.

Definition 2.4 *The* local closure *for $i \in Id$ is the set of all i-subformulas and their local negations, i.e.,*

$$closure_i(\alpha) = subf_i(\alpha) \cup \{@_i[\neg\,\varphi] \mid @_i[\varphi] \in subf_i(\alpha)\}.$$

with the proviso that $@_i[\neg\,\neg\,\varphi]$ is identified with $@_i[\varphi]$.

The global closure *of α is the set of all local closures for all agents, together with its subformulas and their negations, i.e.,*

$$closure(\alpha) = \bigcup_{i \in Id} closure_i(\alpha) \cup subf(\alpha) \cup \{\neg\,\beta \mid \beta \in subf(\alpha)\}$$

with the proviso that $\neg\,\neg\,\beta$ is identified with β and $\neg\,@_i[\neg\,\varphi]$ is identified with $@_i[\varphi]$.

We illustrate these concepts with a simple example. The set $closure_i(@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])])$ is

$$\{@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[p], @_i[\copyright_j[q_1 \Rightarrow q_2]],$$
$$\{@_i[\neg\,\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[\neg(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2)]], @_i[\neg\,p], @_i[\copyright_j[\neg(q_1 \Rightarrow q_2)]]\}$$

and the set $closure(@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])])$ is

$$\{@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[p], @_i[\copyright_j[q_1 \Rightarrow q_2]],$$
$$@_i[\neg\,\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[\neg(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2)]], @_i[\neg\,p], @_i[\copyright_j[\neg(q_1 \Rightarrow q_2)]],$$
$$@_j[q_1 \Rightarrow q_2], @_j[q_1], @_j[q_2], @_j[\neg(q_1 \Rightarrow q_2)], @_j[\neg\,q_1], @_j[\neg\,q_2],$$
$$\neg @_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], \neg @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], \neg @_i[p], \neg\,@_i[\copyright_j[q_1 \Rightarrow q_2]],$$
$$\neg @_j[q_1 \Rightarrow q_2], \neg\,@_j[q_1], \neg\,@_j[q_2]\}.$$

Observe that $subf(\alpha) \subseteq \bigcup_{i \in Id} closure_i(\alpha)$ and so the definition of global closure can be simplified to

$$closure(\alpha) = \bigcup_{i \in Id} closure_i(\alpha) \cup \{\neg\,\beta \mid \beta \in subf(\alpha)\}.$$

Finally, given a set $B \subseteq closure(\alpha)$ and $i \in Id$, we write $B\!\downarrow_i$ to denote the subset of B containing the formulas $@_i[\varphi]$ of agent i in B, i.e., $B\!\downarrow_i = B \cap closure_i(\alpha)$.

2.2 Semantics

The semantics of DTL is inspired in the semantics of LTL. Indeed, each agent behaves *linearly*, as in LTL. Hence, an interpretation structure for an agent will be similar to an LTL interpretation structure. But, given the distributed nature of DTL, time is *internal* to each agent. There is no notion of a *global clock* given that are no global temporal operators. Furthermore, agents may synchronize at some time points, but these time points need not be the same for both agents, that is, the synchronization may happen at time t_1 of one agent and time t_2 of the other agent. For these reasons, we adopt as interpretation structures of DTL, *labelled distributed life-cycles* built upon a simplified form of Winskel's *event structures* [33].

Definition 2.5 *A* local life-cycle *of an agent $i \in Id$ is tuple $\lambda_i = \langle Ev_i, \rightarrow_i \rangle$, where Ev_i is a countable infinite set and $\rightarrow_i \subseteq Ev_i \times Ev_i$, the* immediate successor relation*, is such that its reflexive and transitive closure is a well-founded total order, which we denote by $\leq_i$.*

In the sequel, we may write $\lambda_i = \langle Ev_i, \leq_i \rangle$ for $\lambda_i = \langle Ev_i, \rightarrow_i \rangle$. The elements in Ev_i are called *local events* and $\leq_i$ the *local order of causality*. We denote by $<_i$ the irreflexive relation obtained from $\leq_i$, i.e., $e <_i e'$ if $e \leq_i e'$ and $e \neq e'$.

Definition 2.6 *A* distributed life-cycle *is a family $\lambda = \{\lambda_i\}_{i \in Id}$ of local life-cycles such that the relation $\leq = (\bigcup_{i \in Id} \leq_i)^*$ defines a partial order of global causality on the set of all events $Ev = \bigcup_{i \in Id} Ev_i$.*

Communication is modelled by event sharing, and thus for some event e we may have $e \in Ev_i \cap Ev_j$, with $i \neq j$. In that case, requiring $\leq$ to be a partial order amounts to requiring that the local orders are globally compatible, thus excluding the existence of another $e' \in Ev_i \cap Ev_j$ such that $e <_i e'$ but $e' <_j e$. We denote by $Ids(e)$ the set $\{i \in Id \mid e \in Ev_i\}$, for each $e \in Ev$.

We can check the progress of an agent by collecting all the local events that have occurred up to a given point. This yields the notion of the *local state* of agent i, which is a finite set $\xi_i \subseteq Ev_i$ down-closed for local causality, i.e., if $e \leq_i e'$ and $e' \in \xi_i$ then also $e \in \xi_i$.

Each non-empty local state ξ_i is reached by the occurrence of an event that we call $last(\xi_i)$ from the local state $\xi_i \setminus \{last(\xi_i)\}$. The set Ξ_i of all local states of an agent i is totally ordered by inclusion and has $\emptyset$ as the minimal element, as a consequence of the total order on local events. Since they are discrete and well-founded, we can enumerate them as follows:

- $\emptyset$ is the 0^{th} state;
- $\{e\}$, where e is the minimum of $\langle Ev_i, \leq_i \rangle$, is the 1^{st} state; and
- if ξ_i is the k^{th} state of agent i and $last(\xi_i) \rightarrow_i e$, then $\xi_i \cup \{e\}$ is agent i's $(k+1)^{\text{th}}$ state.

We will denote by ξ_i^k the k^{th} state of agent i, so $\xi_i^0 = \emptyset$ is the initial state and ξ_i^k is the state reached from the initial state after the occurrence of the first k events. In fact, ξ_i^k is the only state of agent i that contains k elements, i.e., where $|\xi_i^k| = k$. Given $e \in Ev_i$, $e{\downarrow_i} = \{e' \in Ev_i \mid e' \leq_i e\}$ is always a local state. Moreover, if ξ_i is non-empty, then $last(\xi_i){\downarrow_i} = \xi_i$.

We can also define the notion of *global state*. A *global state* is a finite set $\xi \subseteq Ev$ closed for global causality, i.e. if $e \leq e'$ and $e' \in \xi$ then also $e \in \xi$. The set Ξ of all global states constitutes a lattice under inclusion and has $\emptyset$ as the minimal element. Every global state ξ includes the local state $\xi|_i = \xi \cap Ev_i$ of each agent i. Given $e \in Ev$, $e{\downarrow} = \{e' \in Ev \mid e' \leq e\}$ is always a global state.

Figure 1 depicts a distributed life-cycle where each row comprises the local life-cycle of one agent. In particular, $Ev_i = \{e_1, e_4, e_5, e_8, \dots\}$ and $\rightarrow_i$ corresponds to the arrows in i's row. We can think of the occurrence of event e_1 as leading agent i from its initial state $\emptyset$ to the state $\{e_1\}$, and the occurrence of event e_4 as leading to state $\{e_1, e_4\}$, and so on. Shared events at communication points are highlighted by the dotted vertical lines. Note that the numbers annotating the events are there only for convenience since, in general, no global total order on events is imposed. Figure 2 shows the corresponding lattice of global states.

Definition 2.7 *An* global interpretation structure *$\mu = \langle \lambda, \vartheta \rangle$ consists of a distributed life-cycle λ and a family $\vartheta = \{\vartheta_i\}_{i \in Id}$ of local* labelling functions*, where, for each $i \in Id$, $\vartheta_i : \Xi_i \rightarrow \wp(Prop_i)$ associates a set of local state propositions to each local state. For each $i \in Id$, $\mu_i = \langle \lambda_i, \vartheta_i \rangle$ is the* local interpretation structure*.*

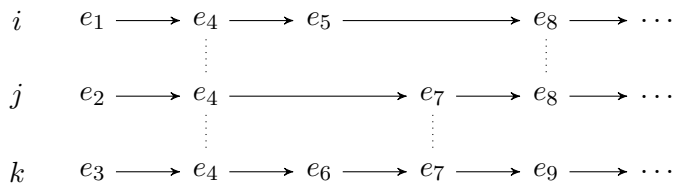

Figure 1: A distributed life-cycle for agents i, j and k.

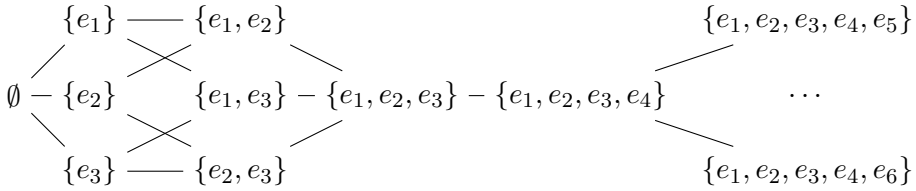

Figure 2: The lattice of global states.

We now define a *global* and *local satisfaction relation.*

Definition 2.8 *Let μ be a global interpretation structure and let ξ be a global state. The* global satisfaction relation*, denoted by $\Vdash$, is inductively defined as follows:*

- $\mu, \xi \Vdash @_i[\varphi]$ *if* $\mu_i, \xi|_i \Vdash_i \varphi$,
- $\mu, \xi \Vdash \neg\alpha$ *if* $\mu, \xi \nVdash \alpha$,
- $\mu, \xi \Vdash \alpha \Rightarrow \beta$ *if* $\mu, \xi \nVdash \alpha$ *or* $\mu, \xi \Vdash \beta$.

Additionally, let $i \in Id$ and let ξ_i be a local state of agent i. The local satisfaction relation *of agent i., denoted by $\Vdash_i$, is inductively defined as follows:*

- $\mu_i, \xi_i \Vdash_i p$ *if* $p \in \vartheta_i(\xi_i)$,
- $\mu_i, \xi_i \Vdash_i \neg\varphi$ *if* $\mu_i, \xi_i \nVdash_i \varphi$,
- $\mu_i, \xi_i \Vdash_i \varphi \Rightarrow \psi$ *if* $\mu_i, \xi_i \nVdash_i \varphi$ *or* $\mu_i, \xi_i \Vdash_i \psi$,
- $\mu_i, \xi_i \Vdash_i \mathsf{X}\varphi$ *if there is* $e \in Ev_i \setminus \xi_i$ *such that* $\xi_i \cup \{e\} \in \Xi_i$ *and* $\mu_i, \xi_i \cup \{e\} \Vdash_i \varphi$,
- $\mu_i, \xi_i \Vdash_i \mathsf{G}\varphi$ *if* $\mu_i, \xi_i' \Vdash_i \varphi$, *for every* $\xi_i' \in \Xi_i$ *such that* $\xi_i \subseteq \xi_i'$,
- $\mu_i, \xi_i \Vdash_i \textcircled{c}_j[\varphi]$ *if* $\xi_i \neq \emptyset$, $last(\xi_i) \in Ev_j$ *and* $\mu_j, last(\xi_i)\downarrow_j \Vdash_j \varphi$.

Herein, we adopt an *anchored* semantics (see [8]) instead of the traditional *floating* semantics of DTL. Anchored semantics is also the usual semantics used in LTL model checking. In this way, we say that μ *(globally) satisfies* α, or that μ is a *model* of α, written $\mu \Vdash \alpha$, whenever $\mu, \emptyset \Vdash \alpha$. As expected, α is said to be *satisfiable* whenever there is μ such that $\mu \Vdash \alpha$. We denote by $Mod(\alpha)$ the class of all models of α. We define a similar notion for the local languages. We say that μ_i *(locally) satisfies* φ, written $\mu_i \Vdash_i \varphi$ if $\mu_i, \emptyset \Vdash_i \varphi$.

The following results will be useful in the sequel. First, it holds trivially that $Mod(\neg\alpha) = \overline{Mod(\alpha)}$, where $\overline{Mod(\alpha)}$ is the complement of $Mod(\alpha)$. Moreover, it is true that $\mathsf{G}\varphi \Leftrightarrow (\varphi \wedge \mathsf{X}\mathsf{G}\varphi)$. This equivalence captures the traditional fixed-point characterization of the G temporal operator.

3 Distributed Büchi Automata

In this section, we propose Büchi automata for DTL. We start by presenting the traditional notions of nondeterministic Büchi automaton and generalized nondeterministic Büchi automaton. We use these notions to capture the local behaviour of the agents, given that each agent is linear. In this case, we follow the ideas discussed in [31, 1]. Then, we propose the notion of *distributed Büchi automaton* to capture the distributed nature of DTL.

Definition 3.1 *A nondeterministic Büchi automaton (NBA) is a tuple* $\mathcal{A} = \langle Q, \Sigma, \delta, Q_0, F\rangle$, *where:*

- Q *is a nonempty finite set of states,*
- Σ *is a finite set of alphabet symbols such that* $Q \cap \Sigma = \emptyset$,
- $\delta : Q \times \Sigma \to 2^Q$ *is the transition function,*
- $Q_0 \subseteq Q$ *is a set of initial states,*
- $F \subseteq Q$ *is a set of acceptance states (also called final states).*

We will write $q \xrightarrow{a} q'$ for $q' \in \delta(q, a)$. Let Σ^ω denote the set of all infinite words over Σ.

Definition 3.2 *A* run *for* $w = a_0a_1a_2\cdots \in \Sigma^\omega$ *in* $\mathcal{A}$ *is an infinite sequence* $q_0q_1q_2\ldots$ *of states in* $\mathcal{A}$ *such that* $q_0 \in Q_0$ *and* $q_k \xrightarrow{a_k} q_{k+1}$, *for* $k \in \mathbb{N}$:

$$q_0 \xrightarrow{a_0} q_1 \xrightarrow{a_1} q_2 \xrightarrow{a_2} \ldots$$

A run $q_0q_1q_2\ldots$ *is* accepting *if* $q_k \in F$ *for infinitely many indices* $k \in \mathbb{N}$. *The* accepted language *of* $\mathcal{A}$ *is*

$$L(\mathcal{A}) = \{w \in \Sigma^\omega \mid \textit{there exists an accepting run for } w \textit{ in } \mathcal{A}\}.$$

Next, we propose the notion generalized Büchi automaton as well as the analogous notions of run and accepted language. This construction will be useful for defining an automaton that accepts the models of a DTL formula.

Definition 3.3 *A* generalized nondeterministic Büchi automaton (GNBA) *is a tuple* $\mathcal{G} = \langle Q, \Sigma, \delta, Q_0, \mathcal{F}\rangle$, *where* Q, Σ, δ *and* Q_0 *are defined just as for NBA and* $\mathcal{F}$ *is a (possibly empty) subset of* 2^Q. *The elements of* $\mathcal{F}$ *are called* acceptance sets. *A* run *for* $w = a_0a_1a_2\cdots \in \Sigma^\omega$ *in* $\mathcal{G}$ *is defined as in the case of an NBA. A run* $q_0q_1q_2\ldots$ *is* accepting *if, for each acceptance set* $F \in \mathcal{F}$, *there are infinitely many indices* $k \in \mathbb{N}$ *such that* $q_k \in F$. *The* accepted language *for a GNBA is the set of words for which there is an accepting run.*

Observe that, for a GNBA $\mathcal{G}$, it may happen that $\mathcal{F} = \emptyset$. In this case, every run in $\mathcal{G}$ is accepting. This is equivalent to considering $\mathcal{F} = \{Q\}$. In the future, and for technical reasons, when $\mathcal{F} = \emptyset$ we will replace it with $\mathcal{F} = \{Q\}$. The classes of NBAs and GNBAs are *equivalent* in the sense that they accept exactly the same languages. Every NBA is a particular case of a GNBA. Furthermore, for each GNBA $\mathcal{G}$ there exists an NBA $\mathcal{A}_\mathcal{G}$ such that $L(\mathcal{A}_\mathcal{G}) = L(\mathcal{G})$. Details of this equivalence can be found, for instance, in [1].

Next, we propose the notion of distributed Büchi automata for DTL. From now on, we assume fixed a distributed signature $\langle Id, \{Prop\}_{i\in Id}\rangle$ and, for the sake of simplicity, we assume that $Id = \{1, \ldots, n\}$. For each $i \in Id$, let $\mathcal{A}_i = \langle Q_i, \Sigma_i, \delta_i, Q_{0_i}, F_i\rangle$ be an NBA. Given a tuple q, we denote by $q{\downarrow_i}$ the projection of q over component i, overloading the $\downarrow_i$ notation.

Definition 3.4 *A generalized distributed nondeterministic Büchi automaton (GDNBA) based on* $\{\mathcal{A}_i\}_{i\in Id}$ *is a tuple*

$$\mathcal{G} = \langle Q, \Sigma, \delta, Q_0, \mathcal{F}\rangle$$

such that:

- $Q = \prod_{i\in Id} Q_i$,
- $\Sigma = \{a \subseteq \bigcup_{i\in Id} \Sigma_i \mid a \neq \emptyset \text{ and } |a \cap \Sigma_i| \leq 1, \text{ for } i \in Id\}$,
- $\delta : Q \times \Sigma \to 2^Q$ *is defined as follows:* $\delta(q, a)$ *is the set of all states* q' *satisfying, for* $i \in Id$,
  - *if* $a \cap \Sigma_i = \emptyset$ *then* $q'{\downarrow_i} = q{\downarrow_i}$,
  - *if* $a \cap \Sigma_i \neq \emptyset$ *then* $q'{\downarrow_i} \in \delta_i(q{\downarrow_i}, a \cap \Sigma_i)$,
- $Q_0 = \prod_{i\in Id} Q_{0_i}$,
- $\mathcal{F} = \{\mathcal{F}_i \subseteq Q \mid \text{ for } i \in Id\}$ *such that* $\mathcal{F}_i = \{q \in Q \mid q{\downarrow_i} \in F_i\}$.

The states of the GDNBA are tuples of states from the local automata. Each symbol of the distributed alphabet is a nonempty set of symbols of the local automata with the proviso that in each global symbol there is at most one symbol from each agent. The transition from one state to the next at the global level is guided by the local behaviour of each component. If, for a particular global symbol a, agent i is not involved (in symbols, $a \cap \Sigma_i = \emptyset$), then for this transition the agent's local state will not change. If, on the other hand, the agent is involved in a (in symbols, $a \cap \Sigma_i \neq \emptyset$), then the agent's local state will change according to its local behaviour, which is dictated by δ_i. Note that, in this case, we are abusing notation. If $a \cap \Sigma_i \neq \emptyset$ then $a \cap \Sigma_i$ is a set, a singleton $\{a'\}$ with $a' \in \Sigma_i$, but nevertheless, a set. Hence, when we write $\delta_i(q{\downarrow_i}, a \cap \Sigma_i)$, we mean $\delta_i(q{\downarrow_i}, a')$.

The language accepted by the generalized distributed automaton $L(\mathcal{G})$ is based on the languages accepted by the local automata, that is, the automaton will accept all the local words of the local automata. However, we need one additional proviso: we only consider *fair* words, that is, words where each agent is involved infinitely often. This proviso guarantees that if we project a global word into one of its local counterparts we still obtain an infinite word for the local automaton. Without this condition, such projection could yield a finite word or even the empty word.

Definition 3.5 *Let* $\mathcal{G}$ *be GDNBA on* $\{\mathcal{A}_i\}_{i\in Id}$. *A global word* $a_0a_1a_2\ldots$ *is* fair *if, for every* $i \in Id$, $a_k \cap \Sigma_i \neq \emptyset$, *for infinitely many indices* $k \in \mathbb{N}$. *A* global run *for a fair word* w *in* $\mathcal{G}$ *is a sequence of states* $q_0q_1q_2\ldots$ *such that* $q_k \xrightarrow{a_k} q_{k+1}$. *A global run* $a_0a_1a_2\ldots$ *is* accepting *if, for each* $i \in Id$, $q_k \in \mathcal{F}_i$, *for infinitely many indices* $k \in \mathbb{N}$. *The* accepted language *of* $\mathcal{G}$ *is*

$$L(\mathcal{G}) = \{w \in \Sigma^\omega \mid w \text{ is fair and there is an accepting run for } w \text{ in } \mathcal{G}\}.$$

Let $w = a_0a_1a_2\cdots \in \Sigma^\omega$ be a fair global word. Then, we denote by $w{\downarrow_i}$ the local word obtained from w as follows:

- first, consider the projection $w' = (a_0 \cap \Sigma_i)(a_1 \cap \Sigma_i)(a_2 \cap \Sigma_i)\ldots$ over the alphabet Σ_i,

- then, let $w{\downarrow_i}$ be the local word obtained from w' by removing all the empty sets and replacing each nonempty set $\{a\}$ by its element a.

Recall that $|a_k \cap \Sigma_i| \leq 1$ hence $a_k \cap \Sigma_i$ is either a singleton or the empty set.

Similarly, let $\tau = q_0 q_1 q_2 \ldots$ be a global run for $w = a_0 a_1 a_2 \ldots$. Then, we denote by $\tau{\downarrow_{w,i}}$ the local run for $w{\downarrow_i}$ obtained from τ as follows:

- first, consider the projection $\tau' = q_0{\downarrow_i}\ q_1{\downarrow_i}\ q_2{\downarrow_i}\ \ldots$ over the states of $\mathcal{A}_i$,
- then, let $\tau{\downarrow_{w,i}}$ be the local run obtained from τ' by removing q_{k+1} if $(a_k \cap \Sigma_i) = \emptyset$, for $k \in \mathbb{N}$.

In this case, we project each global state on its local component for agent i and then remove all the states resulting from transitions where agent i was not involved. The following lemma proves that $w{\downarrow_i}$ is indeed a word in Σ_i^ω and that $\tau{\downarrow_{w,i}}$ is a local run for $w{\downarrow_i}$ in $\mathcal{A}_i$.

Lemma 3.6 *Let $\mathcal{G}$ be a GDNBA based on $\{\mathcal{A}_i\}_{i \in Id}$. If $\tau = q_0 q_1 q_2 \ldots$ is a global run for a fair global word $w = a_0 a_1 a_2 \cdots \in \Sigma^\omega$ then, for each $i \in Id$,*

1. *$w{\downarrow_i} \in \Sigma_i^\omega$;*
2. *$\tau{\downarrow_{w,i}}$ is a local run for $w{\downarrow_i}$ in $\mathcal{A}_i$.*

Furthermore, τ is accepting if and only if $\tau{\downarrow_{w,i}}$ is accepting, for every $i \in Id$.

Proof:

1. Follows from the definition of $w{\downarrow_i}$ and the fact that w is fair.
2. We briefly sketch the proof of this result. Consider the global run

$$\ldots \langle \ldots, q_k, \ldots \rangle \xrightarrow{a_k} \langle \ldots, q_{k+1}, \ldots \rangle \xrightarrow{a_{k+1}} \langle \ldots, q_{k+2}, \ldots \rangle \ldots$$

and assume that $a_k \cap \Sigma_i = \emptyset$ and $a_{k+1} \cap \Sigma_i \neq \emptyset$. Then, by construction, $q_{k+1}{\downarrow_i}$ is deleted in the local run $\tau{\downarrow_{w,i}}$ as follows:

$$\ldots q_k \xrightarrow{a_{k+1} \cap \Sigma_i} q_{k+2} \ldots$$

Moreover, by definition of δ, we have that

$$q_{k+1}{\downarrow_i} = q_k{\downarrow_i} \text{ and } q_{k+2}{\downarrow_i} \in \delta_i(q_{k+1}{\downarrow_i}, a_{k+1} \cap \Sigma_i).$$

Thus,

$$q_{k+2}{\downarrow_i} \in \delta_i(q_k{\downarrow_i}, a_{k+1} \cap \Sigma_i)\,,$$

which is the condition for a local run in $\mathcal{A}_i$. The other possible cases can be reduced to variations of this case. Furthermore, the fact that τ is accepting if and only if $\tau{\downarrow_{w,i}}$ is accepting, for every $i \in Id$, is an immediate consequence of the definition of global acceptance. □

The following example illustrates these constructions.

Figure 3: NBAs $\mathcal{A}_1$ (on the left) and $\mathcal{A}_2$ (on the right).

Example 3.7 Consider the NBAs $\mathcal{A}_1$ and $\mathcal{A}_2$ depicted in Figure 3, with $\Sigma_1 = \{0, 1\}$ and $\Sigma_2 = \{a, b\}$. $\mathcal{A}_1$ accepts all the infinite words over $\{0, 1\}$ with infinitely many 0s, and $\mathcal{A}_2$ accepts all the infinite words over $\{a, b\}$ with finitely many as.

Now let us consider the GDNBA $\mathcal{G}$ based on $\{\mathcal{A}_1, \mathcal{A}_2\}$. The alphabet has as elements sets with one symbol from one agent or from both agents:

$$\Sigma = \{\{0\}, \{1\}, \{a\}, \{b\}, \{0, a\}, \{0, b\}, \{1, a\}, \{1, b\}\}.$$

The set of states is

$$Q = \{\langle q_0, p_0\rangle, \langle q_0, p_1\rangle, \langle q_1, p_0\rangle, \langle q_1, p_1\rangle\}.$$

Of these, only $\langle q_0, p_0\rangle$ is initial, and

$$F_1 = \{\langle q_1, p_0\rangle, \langle q_1, p_1\rangle\} \text{ and } F_2 = \{\langle q_0, p_1\rangle, \langle q_1, p_1\rangle\}.$$

The transition function δ is depicted is Figure 4.

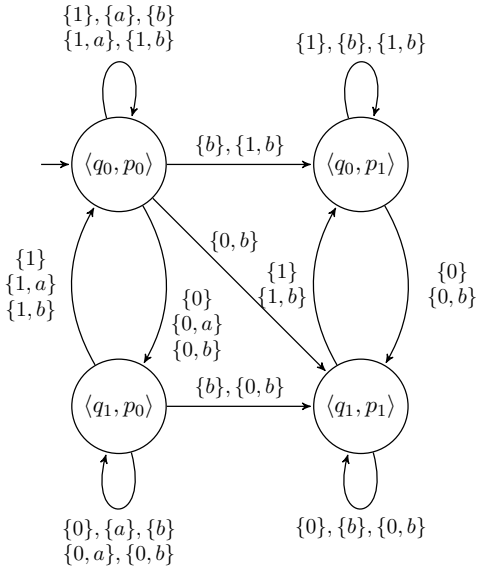

Figure 4: GDNBA $\mathcal{G}$ based on $\{\mathcal{A}_1, \mathcal{A}_2\}$.

In this case, the accepted words are those with an infinite number of 0's and a finite number of a's. For instance, the word

$$\{1\}\{a\}\{1, b\}\{0, b\}\{0\}\{b\}\{0, b\}\{0\}\{b\}\ldots$$

is accepted given that state $\langle q_1, p_1\rangle$ is visited infinitely often, i.e., a final state from $\mathcal{A}_1$ and a final state from $\mathcal{A}_2$ are visited infinitely often. However, if this was the only requirement for acceptance, then the word

$$\{1\}\{a\}\{1,b\}\{0,b\}\{0\}\{0\}\{0\}\{0\}\ldots$$

would also be accepted given that state $\langle q_1, p_1\rangle$, in this case, is also visited infinitely often. We do not want this word to be accepted because its projection on $\mathcal{A}_2$ yields the finite word *abb*, which is not part of the language of $\mathcal{A}_2$. However, this global word is not a fair word, so it will not be accepted by $\mathcal{D}$. □

In general, LTL model checking techniques use automata to capture the models of the envisaged formula ([1]). Herein, we aim at adopting a similar technique with the proviso that instead of capturing linear interpretation structures as in the case of LTL, we need to capture distributed interpretation structures that are models of the envisaged formula. Therefore, our goal is to define a GDNBA $\mathcal{G}^\alpha$ for a given formula $\alpha \in \mathcal{L}$ that accepts all the models of α and only those. Before we do so, we present some auxiliary notions.

Definition 3.8 *Let α be a global formula α. A set $B \subseteq closure(\alpha)$ is*

- α-consistent *with respect to propositional logic if:*
  - *$\alpha_1 \Rightarrow \alpha_2 \in B$ iff $\neg\,\alpha_1 \in B$ or $\alpha_2 \in B$, for $\alpha_1 \Rightarrow \alpha_2 \in closure(\alpha)$,*
  - *$@_i[\varphi_1 \Rightarrow \varphi_2] \in B$ iff $@_i[\neg\,\varphi_1] \in B$ or $@_i[\varphi_2] \in B$, for $@_i[\varphi_1 \Rightarrow \varphi_2] \in closure(\alpha)$,*
  - *$\neg\,@_i[\varphi] \in B$ iff $@_i[\neg\,\varphi] \in B$,*
  - *if $\alpha_1 \in B$ then $\neg\,\alpha_1 \notin B$;*
- α-locally consistent *with respect to the temporal operator G if:*
  - *if $@_i[\mathsf{G}\,\varphi] \in B$ then $@_i[\varphi] \in B$, for every $@_i[\mathsf{G}\,\varphi] \in closure(\alpha)$;*
- α-maximal *if for all $\alpha_1 \in closure(\alpha)$:*
  - *if $\alpha_1 \notin B$ then $\neg\,\alpha_1 \in B$;*
- α-elementary *if it is α-consistent with respect to propositional logic, α-locally consistent with respect to the temporal operator G, and α-maximal.*

When no confusion arises, we may drop the reference to α. Elementary sets capture all the properties that can be asserted at a given instant. Recall the formula $@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])]$ with the set of subformulas

$$\{@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[p], @_i[\copyright_j[q_1 \Rightarrow q_2]], \\ @_j[q_1 \Rightarrow q_2], @_j[q_1], @_j[q_2]\}.$$

The set

$$\begin{aligned} B_1 \;=\; & \{@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[p], \\ & @_i[\copyright_j[q_1 \Rightarrow q_2]], @_j[q_1 \Rightarrow q_2], @_j[\neg\,q_1], @_j[\neg\,q_2], \neg\,@_j[q_1], \neg\,@_j[q_2]\} \end{aligned}$$

is an example of an elementary set. However, the set

$$B_2 = \{@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p], @_i[\neg \copyright_j[q_1 \Rightarrow q_2]] \dots\}$$

is not elementary because it is not consistent with propositional logic. In total, there are 28 elementary sets with respect to this formula, as we illustrate in Example 3.21.

We have all we need to define the envisaged GDNBA for a global formula α. Given α, we start by defining local GNBAs $\mathcal{G}_i^\alpha$ for each agent $i \in Id$ (the construction is similar to the one presented in [1]). From these, we can then obtain equivalent NBAs $\mathcal{A}_i^\alpha$ that will be used to define the GDNBA $\mathcal{G}^\alpha$.

Definition 3.9 *Given a global formula α, GNBA $\mathcal{G}_i^\alpha = \langle Q_i^\alpha, \mathcal{V}_i^\alpha, \delta_i^\alpha, Q_{0_i}^\alpha, \mathcal{F}_i^\alpha \rangle$ is as follows:*

- $Q_i^\alpha = \{B{\downarrow_i} \mid B \subseteq closure(\alpha)$ *and* B *is* α*-elementary*$\}$,
- $\mathcal{V}_i^\alpha = \mathcal{V}_i$ *where* $\mathcal{V}_i$ *is the set of all* i*-valuations,*
- $Q_{0_i}^\alpha = \{B{\downarrow_i} \mid B{\downarrow_i} \in Q_i^\alpha, @_i[\copyright_j[\varphi]] \notin B{\downarrow_i}$ *and* $\alpha \in B\}$,
- $\mathcal{F}_i^\alpha = \{F_{@_i[\mathsf{G}\,\varphi]} \mid @_i[\mathsf{G}\,\varphi] \in closure(\alpha)\}$ *where*

  $$F_{@_i[\mathsf{G}\,\varphi]} = \{q \in Q_i^\alpha \mid @_i[\mathsf{G}\,\varphi] \in q \text{ or } @_i[\varphi] \notin q\},$$

- $\delta_i^\alpha : Q_i^\alpha \times \mathcal{V}_i^\alpha \to 2^{Q_i^\alpha}$ *is such that, for* $q \in Q_i^\alpha$ *and* $v \in \mathcal{V}_i^\alpha$*:*
  - *if* $@_i[v] \neq q \cap @_i[Lit_i]$ *then* $\delta_i^\alpha(q, v) = \emptyset$,
  - *if* $@_i[v] = q \cap @_i[Lit_i]$ *then* $\delta_i^\alpha(q, v)$ *is the set of all states* $q' \in Q_i^\alpha$ *such that:*
    (1) $@_i[\mathsf{X}\,\varphi] \in q$ *iff* $@_i[\varphi] \in q'$*, for every* $@_i[\mathsf{X}\,\varphi] \in closure(\alpha)$,
    (2) $@_i[\mathsf{G}\,\varphi] \in q$ *iff* ($@_i[\varphi] \in q$ *and* $@_i[\mathsf{G}\,\varphi] \in q'$*), for every* $@_i[\mathsf{G}\,\varphi] \in closure(\alpha)$.

Recall that we use $@_i[L]$ to denote the set $\{@_i[l] \mid l \in L\}$, that $\mathcal{V}_i$ is the set of all i-valuations and that a valuation v is a set of literals such that, for each propositional symbol either the symbol is in the valuation or its negation is. $\mathcal{V}_i$ is the alphabet of the automaton. The states of the automaton are all the projections of α-elementary sets on i. Each state contains all the formulas of the form $@_i[\varphi]$, i.e., formulas of agent i that are intended to hold at that state. Furthermore, initial states characterize the initial set-up conditions. As imposed by the semantics of DTL, there can only be synchronizations after the first event occurs. Hence, there can be no communication formulas in any initial state. Moreover, initial states are obtained from α-elementary sets that contain α, representing the fact that α should hold in the initial state. Regarding the transition function, given an alphabet symbol v and a state q, the transition will only be enabled if the valuation v *agrees* with the information in q with respect to propositional symbols of agent i. Additionally, conditions (1) and (2) reflect the semantics of temporal operators. In particular, condition (2) is based on the fixed-point semantics of the G operator. However this condition is not sufficient to fully capture the semantics of this operator. Indeed, it allows runs where, from a certain point on, a formula $@_i[\varphi]$ is always true, i.e., it is present in all the states, but $@_i[\mathsf{G}\,\varphi]$ is not. Consider, for instance, the following run:

$$q_0 \xrightarrow{v_0} \dots \xrightarrow{v_{k-1}} \{\dots, @_i[\varphi], \dots\} \xrightarrow{v_k} \{\dots, @_i[\varphi], \dots\} \xrightarrow{v_{k+1}} \dots$$

The set $F_{@_i[\mathsf{G}\,\varphi]}$ of final states is defined in order to exclude such runs from being accepting runs. In fact, if $@_i[\varphi]$ is true from a certain point k on then, in order for the run to be accepting, $F_{@_i[\mathsf{G}\,\varphi]}$ must be visited infinitely often. In other words, since $@_i[\varphi] \in q_n$, for $n \geq k$, then $@_i[\mathsf{G}\,\varphi]$ must occurs in infinitely many of these states. And so, by condition (2) of the transition function, $@_i[\mathsf{G}\,\varphi]$ must be in all of these states, as intended.

We have defined a local GNBA $\mathcal{G}_i^\alpha$, for each $i \in Id$. Let $\mathcal{A}_i^\alpha$ be an NBA equivalent to $\mathcal{G}_i^\alpha$ (see [1]). We are ready to define the envisaged GDNBA $\mathcal{G}^\alpha$.

Definition 3.10 *For each $i \in Id$, let $\mathcal{A}_i^\alpha$ be an NBA equivalent to $\mathcal{G}_i^\alpha$. The GDNBA $\mathcal{G}^\alpha = \langle Q^\alpha, \Sigma^\alpha, \delta^\alpha, Q_0^\alpha, \mathcal{F}^\alpha\rangle$ is the GDNBA based on $\{\mathcal{A}_i^\alpha\}_{i\in Id}$ meeting the following constraints: for every $q, q' \in Q^\alpha$, $a \in \Sigma^\alpha$ such that $q' \in \delta^\alpha(q, a)$ and $i, j \in Id$,*

(ST) $\bigcup_{i\in Id} q{\downarrow_i} \subseteq B$ *for some α-elementary set B,*

(SC1) *if $@_i[©_j[\varphi]] \in q'{\downarrow_i}$ and $a \cap Lit_i \in \mathcal{V}_i$ then $@_j[\varphi] \in q'{\downarrow_j}$ and $a \cap Lit_j \in \mathcal{V}_j$,*

(SC2) *if $a \cap Lit_i \in \mathcal{V}_i$ and $a \cap Lit_j \in \mathcal{V}_j$ and $@_j[\varphi] \in q'{\downarrow_j}$ then $@_i[©_j[\varphi]] \in q'{\downarrow_i}$, for each $@_i[©_j[\varphi]] \in$ closure(α).*

Condition (ST) guarantees that local states from different agents must agree on the same global information. Observe that if $\bigcup_{i\in Id} q{\downarrow_i}$ can be extended to an α-elementary set, then this set is unique, as proved in the following lemma.

Lemma 3.11 *Let $q \in Q^\alpha$ and B_1 be an α-elementary set such that $\bigcup_{i\in Id} q{\downarrow_i} \subseteq B_1$. If $\bigcup_{i\in Id} q{\downarrow_i} \subseteq B_2$ for some α-elementary set B_2 then $B_1 = B_2$.*

Proof: We prove that for any global formula β, $\beta \in B_1$ if and only if $\beta \in B_2$, by induction in the structure of β.

Basis: β is $@_i[\varphi]$ for some local formula φ and agent $i \in Id$. Assume that $@_i[\varphi] \in B_1$. Then, by definition of elementary set, $@_i[\varphi] \in q{\downarrow_i}$. Indeed, if $@_i[\varphi] \notin q{\downarrow_i}$ then, by definition of state of $\mathcal{G}_i^\alpha$, as $q{\downarrow_i} = B{\downarrow_i}$ for some α-elementary set B, it follows that $@_i[\varphi] \notin B{\downarrow_i}$ and, consequently, $@_i[\varphi] \notin B$. Since B is maximal, it follows that $\neg\, @_i[\varphi] \in B$ and, as B is consistent with respect to propositional logic, $@_i[\neg\,\varphi] \in B$. Thus, $@_i[\neg\,\varphi] \in q{\downarrow_i}$ and also $@_i[\neg\,\varphi] \in B_1$, contradicting the consistency of B_1. Thus, $@_i[\varphi] \in q{\downarrow_i}$ implies that $@_i[\varphi] \in B_2$. The proof for the converse is similar.

Induction step: this is an immediate consequence of the induction hypothesis. □

Condition (SC1) states that in every state q' reached by a transition where i was an active participant (expressed by $a \cap Lit_i \in \mathcal{V}_i$), if $©_j[\varphi]$ holds for i, then i and j must have just synchronized, and so j must also have been an active participant in a (expressed by $a \cap Lit_j \in \mathcal{V}_j$) and φ must hold for j. Finally, condition (SC2) states that if i and j were both active in the last transition and if φ holds for agent j then, in the event that i wants to communicate with j, agent i will be able to infer that φ holds for j, i.e., $©_j[\varphi]$ holds for i, provided that $©_j[\varphi]$ is a relevant formula.

We now proceed to show the correctness of this construction. We aim at proving that any word accepted by the automaton is captured by a DTL model of α and that any DTL model of α is represented by a word accepted by the automaton.

Definition 3.12 *Let $w = a_0a_1a_2\ldots\cdots \in L(\mathcal{G}^\alpha)$ and $\tau = q_0q_1q_2\ldots$ be an accepting run for w. The interpretation structure induced by w and τ is $\mu^{w,\tau} = \langle\lambda, \vartheta\rangle$ defined as follows:*

- $Ev = \{e_k \mid k \in \mathbb{N}^+\}$, *and for each* $i \in Id$*:*
  - $Ev_i = \{e_k \in Ev \mid a_{k-1} \cap Lit_i \in \mathcal{V}_i\}$,
  - $\lambda_i = \langle Ev_i, \leq_i \rangle$ *is the local life-cycle such that* $e_{k_1} \leq_i e_{k_2}$ *if* $k_1 \leq k_2$,
  - $\vartheta_i : \Xi_i \to 2^{Prop_i}$ *is such that, for every* $p \in Prop_i$ *and* $e_k \in Ev_i$*:*
    - $\vartheta_i(\emptyset) = \{p \in Prop_i \mid @_i[p] \in q_0{\downarrow_i}\}$,
    - $\vartheta_i(e_k{\downarrow_i}) = \{p \in Prop_i \mid @_i[p] \in q_k{\downarrow_i}\}$,
- $\lambda = \{\lambda_i\}_{i \in Id}$,
- $\vartheta = \{\vartheta_i\}_{i \in Id}$.

The underlying idea for this interpretation structure is that we have a denumerable set of events and each event e_k is assigned to agent i if a_{k-1} involves literals from that agent, that is, agent $i \in Id$ was involved moving from q_{k-1} to q_k. Local valuations are defined in each state by the propositional symbols that appear in the corresponding state of the automaton. Note that each ϑ_i is well defined because the components of q_k are subsets of the same α-elementary set. In the sequel, we will consider the following enumeration of global states:

- $\xi^0 = \emptyset$,
- $\xi^k = \{e_1, \ldots, e_k\}$, for $k \geq 1$.

Before we establish the main theorem, we prove an auxiliary result.

Lemma 3.13 *Let* ξ^k *be a state of* $\mu^{w,\tau}$*. If* $\xi^k|_i = \emptyset$ *then* $q_k{\downarrow_i} = \cdots = q_0{\downarrow_i}$*, whereas if* $\xi^k|_i \neq \emptyset$ *then* $q_k{\downarrow_i} = \cdots = q_{k'}{\downarrow_i}$ *for* $e_{k'} = last(\xi^k|_i)$*.*

Proof: The proof follows by induction on k. If $k = 0$ the result is immediate. If $k > 0$ and if $e_k \in Ev_i$ then $last(\xi^k|_i) = e_k$ and once again the result follows. If $e_k \notin Ev_i$ then $a_{k-1} \cap \mathcal{V}_i = \emptyset$ and, by definition of GDNBA, we have that $q_k{\downarrow_i} = q_{k-1}{\downarrow_i}$. If $\xi^{k-1}|_i = \emptyset$ then $q_{k-1}{\downarrow_i} = \cdots = q_0{\downarrow_i}$, using the induction hypothesis, and so $q_k{\downarrow_i} = \cdots = q_0{\downarrow_i}$. If $\xi^{k-1}|_i \neq \emptyset$ then, using again the induction hypothesis, $q_{k-1}{\downarrow_i} = \cdots = q_{k'}{\downarrow_i}$ for $e_{k'} = last(\xi^{k-1}|i)$. So, $q_k{\downarrow_i} = \cdots = q_{k'}{\downarrow_i}$ for $e_{k'} = last(\xi^{k-1}|i) = last(\xi^k|i)$, given that $e_k \notin Ev_i$. □

Theorem 3.14 *If* $w \in L(\mathcal{G}^\alpha)$*, with accepting run* τ*, then* $\mu^{w,\tau} \in Mod(\alpha)$*.*

Proof: Let $\tau = q_0 q_1 q_2 \ldots$. Our goal is to prove that $\mu^{w,\tau} \Vdash \alpha$. We start by establishing a preliminary result for the local level: $\mu_i^{w,\tau}, \xi^k|_i \Vdash_i \psi$ iff $@_i[\psi] \in q_k{\downarrow_i}$, for every $@_i[\psi] \in closure(\alpha)$. The proof follows by induction on the structure of ψ, simultaneously for all agents.

Basis: $\psi \in Prop_i$. Then, if $k = 0$ then $\xi^0|_i = \xi^0 = \emptyset$ and $@_i[p] \in q_0{\downarrow_i}$ iff $p \in \vartheta_i(\emptyset)$ iff $\mu_i^{w,\tau}, \xi^0|_i \Vdash_i p$. If $k > 0$ and $\xi^k|_i = \emptyset$ then, by Lemma 3.13, $q_k{\downarrow_i} = q_0{\downarrow_i}$ and the result follows as for $k = 0$. If $\xi^k|_i \neq \emptyset$, let $e_{k'} = last(\xi^k|_i)$. Then $\xi^k|_i = \xi^{k'}|_i$. Furthermore, by Lemma 3.13, $q_k{\downarrow_i} = q_{k'}{\downarrow_i}$. Hence, $@_i[p] \in q_k{\downarrow_i}$ iff $@_i[p] \in q_{k'}{\downarrow_i}$ iff $p \in \vartheta_i(\xi^{k'}{\downarrow_i})$ iff $\mu_i^{w,\tau}, \xi^{k'}{\downarrow_i} \Vdash_i p$ iff $\mu_i^{w,\tau}, \xi^k|_i \Vdash_i p$.

Induction step: The case of propositional formulas is an immediate consequence of the definition of elementary set and we omit the details.

Assume that $\psi = \mathsf{X}\,\psi_1$ and let $k_1 > k$ be such that $\xi^k{\downarrow_i} \cup \{e_{k_1}\} \in \Xi_i$ (k_1 is the index of the first event after e_k where agent i is active). Note that $\xi^{k_1}|_i = \xi^k|_i \cup \{e_{k_1}\}$ and $\xi^k|_i = \xi^{k_1-1}|_i$. By Lemma 3.13, it follows that $q_{k_1-1}{\downarrow_i} = \cdots = q_k{\downarrow_i}$. Finally, given that $e_{k_1} \in Ev_i$, we have that $q_{k_1}{\downarrow_i} \in \delta_i^\alpha(q_{k_1-1}{\downarrow_i}, a_{k_1-1} \cap Lit_i)$, which implies that $@_i[\mathsf{X}\,\psi_1] \in q_{k_1-1}{\downarrow_i}$ iff $@_i[\psi_1] \in q_{k_1}{\downarrow_i}$, by condition (1) in the definition of δ_i^α. Then, $\mu_i^{w,\tau}, \xi^k|_i \Vdash_i \mathsf{X}\,\psi_1$ iff $\mu_i^{w,\tau}, \xi^{k_1}|_i \Vdash_i \psi_1$ iff (using the induction hypothesis) $@_i[\psi_1] \in q_{k_1}{\downarrow_i}$ iff $@_i[\mathsf{X}\,\psi_1] \in q_{k_1-1}{\downarrow_i}$ iff $@_i[\mathsf{X}\,\psi_1] \in q_k{\downarrow_i}$.

Assume now that ψ is $\mathsf{G}\,\psi_1$. We start by observing that for any run $B_0B_1B_2\ldots$ in $\mathcal{G}_i$ if $@_i[\mathsf{G}\,\psi_1] \in B_k$ then $@_i[\mathsf{G}\,\psi_1] \in B_{k'}$ for every $k' \geq k$. This is an immediate consequence of condition (2) in the definition of δ_i^α and can easily be established by induction. Assume first that $@_i[\mathsf{G}\,\psi] \in q_k{\downarrow_i}$. By the previous claim and Lemma 3.6, it follows that $@_i[\mathsf{G}\,\psi_i] \in q_{k'}{\downarrow_i}$ for every $k' \geq k$. As each set $q_{k'}{\downarrow_i}$ is elementary then it is locally consistent with respect to the temporal operator G and so $@_i[\psi_1] \in q_{k'}{\downarrow_i}$, for every $k' \geq k$. Using the induction hypothesis, it follows that $\mu_i^{w,\tau}, \xi^{k'}|_i \Vdash_i \psi_1$, for every $k' \geq k$. This last condition implies that $\mu_i^{w,\tau}, \xi^k|_i \Vdash_i \mathsf{G}\,\psi_1$. For the proof of the opposite direction now assume that $\mu_i^{w,\tau}, \xi^k|_i \Vdash_i \mathsf{G}\,\psi_1$. Then, $\mu_i^{w,\tau}, \xi_i' \Vdash_i \psi_1$, for every $\xi_i' \supseteq \xi^k|_i$. A simple inductive argument allows us to conclude that $\mu_i^{w,\tau}, \xi^{k'}|_i \Vdash_i \psi_1$, for every $k' \geq k$. Note that if $e_{k'} \in Ev_i$ then $\xi^{k'}|_i = \xi_i' \cup \{e_{k'}\}$, which is in Ξ_i and satisfies $\xi_i' \cup \{e_{k'}\} \supseteq \xi^k|_i$. If $e_{k'} \notin Ev_i$ then $\xi^{k'}|_i = \xi^{k'-1}|_i$ and again $\xi^{k'}|_i \supseteq \xi^k|_i$. Hence, by the induction hypothesis, it follows that $@_i[\psi_1] \in q_{k'}{\downarrow_i}$, for every $k' \geq k$. Since the run is accepting, infinitely many of these states must be in $F_{\mathsf{G}\,\psi_1}$. Let $q_{k_1}{\downarrow_i}$, with $k_1 \geq k$, be the first of such states. Hence, it must be the case that $k_1 = k$. In fact, if $k_1 > k$, given that $@_i[\psi] \in q_{k_1}{\downarrow_i}$ then, by condition (2) in the definition of δ_i^α this would imply that $@_i[\mathsf{G}\,\psi_1] \in q_{k_1-1}{\downarrow_i}$ forcing $q_{k_1-1}{\downarrow_i}$ to also be in $F_{\mathsf{G}\,\psi_1}$ and thus contradicting the fact the k_1 was the first final state after k. Hence, $@_i[\mathsf{G}\,\psi_1] \in q_k{\downarrow_i}$.

Finally, assume that $\psi = \copyright_j[\psi_1]$. If $\xi^k|_i = \emptyset$ then, by Lemma 3.13, $q_k{\downarrow_i} = q_0{\downarrow_i}$ and by definition of initial state, it follows that $@_i[\copyright_j[\psi_1]] \notin q_k{\downarrow_i}$. Furthermore, by definition of satisfaction, $\mu_i^{w,\tau}, \xi^k|_i \nVdash_i \copyright_j[\psi_1]$ and the result follows. Assume now that $\xi^k|_i \neq \emptyset$ and let $e_{k'} = last(\xi^k|_i)$. Note that, $\xi^k|_i = \xi^{k'}|_i$ and, by Lemma 3.13, $q_k{\downarrow_i} = q_{k'}{\downarrow_i}$. Assume first that $@_i[\copyright_j[\psi_1]] \in q_k{\downarrow_i}$, i.e., $@_i[\copyright_j[\psi_1]] \in q_{k'}{\downarrow_i}$. As $e_{k'} \in Ev_i$ then $a_{k'-1} \cap Lit_i \in \mathcal{V}_i$ and, by condition (SC1), it follows that $@_j[\psi_1] \in q_{k'}{\downarrow_j}$ and $a_{k'-1} \cap Lit_j \in \mathcal{V}_j$. By the induction hypothesis, $\mu_j^{w,\tau}, \xi^{k'}|_j \Vdash_j \psi_1$ Furthermore, $e_{k'} \in Ev_j$ which implies that $\xi^{k'}|_j = e_{k'}{\downarrow_j} = last(\xi^{k'}|_i){\downarrow_j}$. By definition of satisfaction, it follows that $\mu_i^{w,\tau}, \xi^{k'}|_i \Vdash_i \copyright_j[\psi_1]$, i.e., $\mu_i^{w,\tau}, \xi^k|_i \Vdash_i \copyright_j[\psi_1]$, given that $\xi^k|_i = \xi^{k'}|_i$. For the proof of the opposite direction now assume that $\mu_i^{w,\tau}, \xi^k|_i \Vdash_i \copyright_j[\psi_1]$. As $last(\xi^k|_i) = e_{k'}$ it follows that $e_{k'} \in Ev_j$ (and thus $a_{k'-1} \cap Lit_j \in \mathcal{V}_j$) and $\mu_j^{w,\tau}, e_{k'}{\downarrow_j} \Vdash_j \psi_1$. By the induction hypothesis, $@_j[\psi_1] \in q_{k'}{\downarrow_j}$, given that $e_{k'}{\downarrow_j} = \xi^{k'}|_j$. Since $e_{k'} \in Ev_i$, we have that $a_{k'-1} \cap Lit_i \in \mathcal{V}_i$ and, by condition (SC2), it follows that $@_i[\copyright_j[\psi_1]] \in q_{k'}{\downarrow_i}$, i.e., $@_i[\copyright_j[\psi_1]] \in q_k{\downarrow_i}$.

Next, we prove a similar result for the global level, that is, we prove that $\mu^{w,\tau}, \xi^k \Vdash \alpha_1$ iff $\alpha_1 \in B_k$, for every $\alpha_1 \in \mathit{closure}(\alpha)$, where, for each q_k, B_k is the α-elementary set such that $\bigcup_{i \in Id} q_k{\downarrow_i} \subseteq B_k$, which exists by condition (ST) and is unique by Lemma 3.11. Again, the proof follows by induction on the structure of α_1. The cases of propositional connectives are an immediate consequence of the properties of elementary sets. So, let $\alpha_1 = @_i[\varphi]$. Then, $\mu^{w,\tau}, \xi^k \Vdash @_i[\varphi]$ iff $\mu_i^{w,\tau}, \xi^k|_i \Vdash_i \varphi$ iff $@_i[\varphi] \in q_k{\downarrow_i}$, by the previous result, iff $@_i[\varphi] \in B_k$.

Finally, observe that, when $k = 0$ then $\bigcup_{i \in Id} q_0{\downarrow_i} \subseteq B_0$. By definition of initial states in $Q_{0_i}^\alpha$ it must be the case that $q_0{\downarrow_i} = B{\downarrow_i}$ for some α-elementary set B such that $\alpha \in B$.

But, by Lemma 3.11, B_0 is the unique α-elementary set such that $\bigcup_{i\in Id} q_0{\downarrow_i} \subseteq B_0$. Hence, it follows that $\alpha \in B_0$ and so, it must be the case that $\mu^{w,\tau}, \xi_0 \Vdash \alpha$, that is, $\mu^{w,\tau} \Vdash \alpha$. $\square$

We now prove the converse, i.e., we prove that any DTL model of α can be captured by $\mathcal{G}^\alpha$. Let $\mu = \langle \lambda, \vartheta \rangle$ be an interpretation structure and let $\langle Ev, \leq_{Ev} \rangle$ be the underlying global order on events. It is always possible to linearize $\langle Ev, \leq_{Ev} \rangle$, i.e., it is always possible to define a bijection $\ell : \mathbb{N}_1 \to Ev$ such that if $k_1 <_{\mathbb{N}} k_2$ then $\ell(k_1) <_{Ev} \ell(k_2)$, where $<_{\mathbb{N}}$ is the usual ordering on the natural numbers (see, for instance, [6]). In this conditions, we will refer to ℓ as a linearization of the interpretation structure μ.

Definition 3.15 *Let ℓ be a linearization of $\mu = \langle \langle Ev, \leq \rangle, \vartheta \rangle$. Then, ℓ induces an enumeration of the global states, defined as follows:*

- $\xi^0 = \emptyset$,
- $\xi^k = \xi^{k-1} \cup \{\ell(k)\}$, *for each* $k \geq 1$.

The word induced by μ and ℓ is $w^{\mu,\ell} = a_0 a_1 a_2 \ldots$, defined as follows:

$$a_k = \bigcup_{i \in Ids(\ell(k+1))} \{l \in Lit_i \mid \mu_i, \xi^k|_i \Vdash_i l\}.$$

This word represents one possible *evolution* of the system represented by μ, where each a_k is the set of literals that are satisfied by μ at ξ^k. Our goal is to show that, if $\mu \in Mod(\alpha)$ then $w^{\mu,\ell}$ is captured by $\mathcal{G}^\alpha$, i.e., to show that $w^{\mu,\ell} \in L(\mathcal{G}^\alpha)$.

Theorem 3.16 *If $\mu \in Mod(\alpha)$, then $w^{\mu,\ell} \in L(\mathcal{G}^\alpha)$, for any linearization ℓ of μ.*

Proof: To show that $w^{\mu,\ell} \in L(\mathcal{G}^\alpha)$ holds, we need to present an accepting run for it. For each $k \in \mathbb{N}$, consider the sets of formulas induced by μ:

- $y_k^i = \{@_i[\varphi] \in closure_i(\alpha) \mid \mu_i, \xi^k|_i \Vdash_i \varphi\}$,
- $q_k = \prod_{i \in Id}(y_k^i)$.

Each y_k^i has all information about the local formulas of agent i at instant k. We start by establishing a structural result on the local component of the states. If, for $i \in Id$, $\ell(k+1) \notin Ev_i$ then $y_{k+1}^i = y_k^i$. This is an immediate consequence of the fact that if $\ell(k+1) \notin Ev_i$ then $\xi^{k+1}|_i = \xi^k|_i$ and so $y_{k+1}^i = y_k^i$.

Having established this result, we prove that each q_k is a state in $\mathcal{G}^\alpha$. Let $B_k = \{\beta \in closure(\alpha) \mid \mu, \xi^k \Vdash \beta\}$, for $k \in \mathbb{N}$. Each B_k is α-elementary. All conditions are a consequence of the definition of the satisfaction relation. Moreover, for each $k \in \mathbb{N}$ and each $i \in Id$, $y_k^i = B_k{\downarrow_i}$ and, consequently, y_k^i is a state of the local automaton $\mathcal{G}_i^\alpha$. Furthermore, condition (ST) is satisfied because $\bigcup_{i \in Id} y_k^i \subseteq B_k$.

Next, we prove that $q_0 q_1 q_2 \ldots$ is a run for $w^{\mu,\ell}$ in $\mathcal{G}^\alpha$, i.e., we prove that $q_0 \in Q_0^\alpha$ and $q_k \xrightarrow{a_k} q_{k+1}$, for every $k \in \mathbb{N}$. The fact that $q_0 \in Q_0^\alpha$ is straightforward. Observe that $B_0 = \{\beta \in closure(\alpha) \mid \mu, \emptyset \Vdash \beta\}$. Therefore, $\alpha \in B_0$ because $\mu \in Mod(\alpha)$. Furthermore, B_0 has no communication formulas because $\xi^0 = \emptyset$. This implies that each $y_0^i \in Q_{0_i}^\alpha$ and thus $q_0 = \prod_{i \in Id}(y_0^i) \in Q_0^\alpha$.

To prove that $q_k \xrightarrow{a_k} q_{k+1}$ we consider two cases, imposed by the definition GDNBA: (i) $a_k \cap \mathcal{V}_i = \emptyset$; and (ii) $a_k \cap \mathcal{V}_i \neq \emptyset$, for $i \in Id$.

(i) If $a_k \cap \mathcal{V}_i = \emptyset$ then $i \notin Ids(\ell(k+1))$, which implies that $\xi^{k+1}|_i = \xi^k|i$ and so $y^i_{k+1} = y^i_k$. Hence, $q_{k+1}{\downarrow_i} = q_k{\downarrow_i}$.

(ii) If $a_k \cap \mathcal{V}_i \neq \emptyset$ then $\ell(k+1) \in Ev_i$ and $\xi^{k+1}|_i = \xi^k|_i \cup \{\ell(k+1)\}$. We prove that, in this case, $q_{k+1}{\downarrow_i} \in \delta^\alpha_i(q_k{\downarrow_i}, a_k \cap \mathcal{V}_i)$, i.e., $y^i_{k+1} \in \delta^\alpha_i(y^i_k, a_k \cap \mathcal{V}_i)$:

- $@_i[a_k \cap \mathcal{V}_i] = y^i_k \cap @_i[\mathcal{V}_i]$: follows by construction of a_k and y^i_k;
- $@_i[\mathsf{X}\psi] \in y^i_k$ iff $\mu_i, \xi^k|_i \Vdash_i \mathsf{X}\psi$ iff $\mu_i, \xi^k|_i \cup \{\ell(k+1)\} \Vdash_i \psi$ iff $\mu_i, \xi^{k+1}|_i \Vdash_i \psi$ iff $@_i[\psi] \in y^i_{k+1}$;
- $@_i[\mathsf{G}\psi] \in y^i_k$ iff $\mu_i, \xi^k|_i \Vdash_i \mathsf{G}\psi$ iff $\mu_i, \xi' \Vdash_i \psi$, for $\xi' \supseteq \xi^k|_i$, iff $\mu_i, \xi^k|_i \Vdash_i \psi$ and $\mu_i, \xi' \Vdash_i \psi$, for $\xi' \supseteq \xi^k|_i \cup \{\ell(k+1)\}$, iff $\mu_i, \xi^k|_i \Vdash_i \psi$ and $\mu_i, \xi^k|_i \cup \{\ell(k+1)\} \Vdash_i \mathsf{G}\psi$ iff $\mu_i, \xi^k|_i \Vdash_i \psi$ and $\mu_i, \xi^{k+1}|_i \Vdash_i \mathsf{G}\psi$ iff $@_i[\psi] \in y^i_k$ and $@_i[\mathsf{G}\psi] \in y^i_{k+1}$.

Hence, from (i) and (ii), it follows that $q_{k+1} \in \delta(q_k, a_k)$, as desired.

We also need to establish that the run is accepting. We start by showing that each local run is accepting. Let $@_i[\mathsf{G}\varphi] \in \mathit{closure}(\alpha)$. We need to show that there are infinitely many indices k such that $q_k{\downarrow_i} \in F_{@_i[\mathsf{G}\varphi]}$. Assume that this is not the case, i.e., assume that there are only finitely many indices k such that $q_k{\downarrow_i} \in F_{@_i[\mathsf{G}\varphi]}$. Then, there is $n \in \mathbb{N}$ such that $q_k{\downarrow_i} \notin F_{@_i[\mathsf{G}\varphi]}$, for every $k \geq n$. But, if $q_k{\downarrow_i} \notin F_{@_i[\mathsf{G}\varphi]}$ then $@_i[\mathsf{G}\varphi] \notin q_k{\downarrow_i}$ and $@_i[\varphi] \in q_k{\downarrow_i}$, for $k \geq n$. If $@_i[\mathsf{G}\varphi] \notin q_k{\downarrow_i} = y^i_k$ then $\mu_i, \xi^k|_i \not\Vdash \mathsf{G}\varphi$. This implies that there is $\xi' \supseteq \xi^k|_i$ such that $\mu_i, \xi' \not\Vdash \varphi$. It is not very difficult to see that there is an $m \geq k$ such that $\xi' = \xi^m|_i$. Hence $\mu_i, \xi^m|_i \not\Vdash \varphi$ which implies that $@_i[\varphi] \notin y^i_m = q_m{\downarrow_i}$. But, since $m \geq k \geq n$ then $@_i[\varphi] \in q_m{\downarrow_i}$ and we reach a contradiction. This means that each set $F_{@_i[\mathsf{G}\varphi]}$ is visited infinitely often and, thus, the local run is accepting. The fact that the global word $w^{\mu,\ell}$ is fair is a consequence of the definition of linearization. Then, by Lemma 3.6, we can conclude that the global run is also accepting.

Finally, we need to prove that the run is such that it respects conditions (ST), (SC1) and (SC2) of the automaton, and thus can be accepted. Condition (ST) was already established. For condition (SC1), let $q_{k+1} \in \delta(q_k, a_k)$, $@_i[\copyright_j[\varphi]] \in q_{k+1}{\downarrow_i}$ and $a_k \cap \mathit{Lit}_i \in \mathcal{V}_i$. From $a_k \cap \mathit{Lit}_i \in \mathcal{V}_i$ it follows that $\ell(k+1) \in Ev_i$ and $\mathit{last}(\xi^{k+1}|_i) = \ell(k+1)$. If $@_i[\copyright_j[\varphi]] \in q_{k+1}{\downarrow_i}$ then $@_i[\copyright_j[\varphi]] \in y^i_{k+1}$, which implies that $\mu_i, \xi^{k+1}|_i \Vdash_i \copyright_j[\varphi]$. Thus, $\mathit{last}(\xi^{k+1}|_i) = \ell(k+1) \in Ev_j$ and $\mu_j, \ell(k+1){\downarrow_j} \Vdash_j \varphi$. Hence, $a_k \cap \mathit{Lit}_j \in \mathcal{V}_j$ and $@_j[\varphi] \in y^j_{k+1}$, i.e., $@_j[\varphi] \in q_{k+1}{\downarrow_j}$. The proof for condition (SC2) is similar. □

We now study the relationship between μ and $\mu^{w,\tau}$, where τ is an accepting run for $w^{\mu,\ell}$, and between w and $w^{\mu^{w,\tau},\ell}$. If we start with a model μ, then construct $w^{\mu,\ell}$ and finally define $\mu^{w^{\mu,\ell}}$, then we will conclude that these interpretation structures are isomorphic. Conversely, if we start with a word w (with accepting run τ) and build the model $\mu^{w,\tau}$, then we can show that we can find a linearization ℓ such that $w^{\mu^{w,\tau},\ell}$ is w.

Lemma 3.17 *Let $w \in L(\mathcal{G}^\alpha)$ be a word and τ an accepting run for w. Then, there is a linearization ℓ for the underlying global order on events of μ such that $w = w^{\mu^{w,\tau},\ell}$.*

Proof: Let $w \in L(\mathcal{G}^\alpha)$ with accepting run $\tau = q_0 q_1 \ldots$ and consider $\mu^{w,\tau}$ defined in Definition 3.12. Consider the linearization $\ell : \mathbb{N}^+ \to Ev$ such that $\ell(k) = e_k$. First, we observe that

$\xi^0 = \emptyset$ and $\xi^k = \{\ell(1), \ldots, \ell(k)\} = \{e_1, \ldots, e_k\}$. Let $l \in Lit_i$. Then, for $k \in \mathbb{N}$

$$\begin{array}{rlll} l \in w_k^{\mu^{w,\tau},\ell} & \text{iff} & \mu_i^{w,\tau}, \xi^k|_i \Vdash_i l & \text{(by definition of } w^{\mu,\ell}\text{)} \\ & \text{iff} & @_i[l] \in q_k{\downarrow_i} & \text{(by definition of } \mu^{w,\tau}\text{)} \\ & \text{iff} & l \in w_k & \text{(by definition of } \delta_i^\alpha\text{).} \end{array}$$

Hence, $w_k^{\mu^{w,\tau},\ell} = w_k$, for every $k \in \mathbb{N}$, and so $w^{\mu^{w,\tau},\ell} = w$. □

We can state a similar result for μ and $\mu^{w^{\mu,\ell}}$. We say that two distributed life-cycles $\lambda_1 = \{\langle Ev_i^1, \leq_i^1\rangle_i\}_{i \in Id}$ and $\lambda_2 = \{\langle Ev_i^2, \leq_i^2\rangle_i\}_{i \in Id}$ are *isomorphic*, written $\lambda^1 \cong_f \lambda^2$, if there is a bijection $f : Ev^1 \to Ev^2$ such that $e \in Ev_i^1$ if and only if $f(e) \in Ev_i^2$ and $e \leq_i^1 e'$ if and only if $f(e) \leq_i^2 f(e')$, for every $e, e' \in Ev_i^1$ and $i \in Id$. In this case, the bijection f is called an isomorphism from λ_1 to λ_2. It follows that f preserves global order on events. We say that two interpretation structures $\mu_1 = \langle \lambda_1, \vartheta_1\rangle$ and $\mu_2 = \langle \lambda_2, \vartheta_2\rangle$ are *isomorphic*, written $\mu_1 \cong_f \mu_2$, if $\lambda_1 \cong_f \lambda_2$ and $\vartheta_i^1(\xi_i) = \vartheta_i^2(f(\xi_i))$, for every state $\xi_i \in \Xi_i^1$. Once again, the bijection f is called an isomorphism from μ_1 to μ_2. We may drop the reference to f and simply write $\lambda_1 \cong \lambda_2$. We start by proving that f establishes a bijection between states of λ_1 and λ_2, as would be expected.

Lemma 3.18 *Let f be an isomorphism from λ_1 to λ_2. Then,*

1. *$\{e_1, \ldots, e_k\} \in \Xi_i^1$ if and only if $\{f(e_1), \ldots, f(e_k)\} \in \Xi_i^2$ for each $i \in Id$ and $k \in \mathbb{N}$.*
2. *The set of local states Ξ_i^1 of λ_1 is isomorphic to the set of local states Ξ_i^2 of λ_2.*
3. *The set of global states Ξ^1 of λ_1 is isomorphic to the set of global states Ξ^2 of λ_2.*

Proof: Assume that $\{e_1, \ldots, e_k\} \in \Xi_i^1$ and that $\{f(e_1), \ldots, f(e_k)\} \notin \Xi_i^2$. Then, there are $e, e' \in Ev_i^2$ such that $e \leq_i^2 e'$, $e' \in \{f(e_1), \ldots, f(e_k)\}$ and additionally, $e \notin \{f(e_1), \ldots, f(e_k)\}$. As f is a bijection it has an inverse f^{-1} and so it follows, by definition of isomorphism, that $f^{-1}(e) \leq_i^1 f^{-1}(e')$, $f^{-1}(e') \in \{e_1, \ldots, e_k\}$ and $f^{-1}(e) \notin \{e_1, \ldots, e_k\}$, contradicting the hypothesis that $\{e_1, \ldots, e_k\}$ is a local state of λ_i^1. The proof of the converse is similar. Conditions 2 and 3 follow from 1. □

Next, we prove that isomorphic interpretation structures satisfy exactly the same formulas.

Lemma 3.19 *Let μ^1 and μ^2 be isomorphic interpretation structures and let α_1 be a global formula. Then $\mu^1 \Vdash \alpha_1$ if and only if $\mu^2 \Vdash \alpha_1$.*

Proof: If μ^1 and μ^2 are isomorphic then there is a bijection f. We start by proving that for local formula $\varphi \in \mathcal{L}_i$ and local state $\xi \in \Xi_i^1$, with $i \in Id$, $\mu_i^1, \xi \Vdash_i \varphi$ if and only if $\mu_i^2, f(\xi) \Vdash_i \varphi$. The proof follows by induction in the structure of φ.

Basis: $\varphi \in Prop_i$. In this case, $\mu_i^1, \xi \Vdash_i \varphi$ if and only if $\varphi \in \vartheta_i^1(\xi)$ if and only if $\varphi \in \vartheta_i^2(f(\xi))$ if and only if $\mu_i^2, f(\xi) \Vdash_i \varphi$.

Induction step: The case for propositional formulas is an immediate consequence of the induction hypothesis and we omit the details.

φ is $\mathsf{X}\psi$. If $\mu_i^1, \xi \Vdash_i \mathsf{X}\psi$ then there is $e \in Ev_i^1$ such that $\xi \cup \{e\} \in \Xi_i^1$ and $\mu_i^1, \xi \cup \{e\} \Vdash_i \psi$. By Lemma 3.18, $f(\xi \cup \{e\}) = f(\xi) \cup \{f(e)\} \in \Xi_i^2$. Hence, using the induction hypothesis,

we have that $\mu_i^2, f(\xi) \cup \{f(e)\} \Vdash_i \psi$ and so $\mu_i^2, f(\xi) \Vdash_i \mathsf{X}\psi$. The proof of the converse uses a similar argument.

φ is $\mathsf{G}\psi$. The proof is similar to the previous one.

φ is $\copyright_j[\psi]$. Assume that $\mu_i^1, \xi \Vdash_i \copyright_j[\psi]$. Then, it follows that $\xi \neq \emptyset$, $last(\xi) \in Ev_j^1$ and $\mu_j^1, last(\xi)\downarrow_j \Vdash_j \psi$. By the induction hypothesis, it follows that $\mu_j^2, f(last(\xi)\downarrow_j) \Vdash_j \psi$. Using again Lemma 3.18, we have that $f(last(\xi)\downarrow_j) = f(last(\xi))\downarrow_j$ and that $f(last(\xi)) = last(f(\xi))$, and we also know that $f(last(\xi)) \in Ev_j^2$. Hence, we can state that $\mu_j^2, last(f(\xi))\downarrow_j \Vdash_j \psi$ and $last(f(\xi)) \in Ev_j^2$ which implies that $\mu_i^2, f(\xi) \Vdash_i \copyright_j[\psi]$. The proof of the converse is similar.

We consider now the global case. We prove that $\mu^1, \xi \Vdash \alpha_1$ if and only if $\mu^2, f(\xi) \Vdash \alpha_1$, for any global formula α_1 and global state ξ of μ^1, using again an inductive argument in the structure of α_1.

Basis: α_1 is $@_i[\varphi]$. In this case, $\mu^1, \xi \Vdash @_i[\varphi]$ if and only if $\mu_i^1, \xi|_i \Vdash_i \varphi$ if and only if $\mu_i^2, f(\xi|_i) \Vdash_i \varphi$ if and only if, using Lemma 3.18, $\mu_i^2, f(\xi)|_i \Vdash_i \varphi$ if and only if $\mu^2, f(\xi) \Vdash @_i[\varphi]$.

Induction step: the propositional cases are an immediate consequence of the induction hypothesis.

Hence, using the fact that global states of μ^1 and global states of μ^2 are isomorphic, as stated in Lemma 3.18, we can conclude that $\mu^1 \Vdash \alpha_1$ if and only if $\mu^2 \Vdash \alpha_1$. □

In the future, given a set of interpretation structures I we denote by $I^\bullet$ the closure of I with respect to isomorphism, that is, $I^\bullet$ contains all interpretations in I and all interpretation isomorphic to an interpretation in I. We can now state the converse result of Lemma 3.17.

Lemma 3.20 *Let $\mu \in Mod(\alpha)$ be an interpretation structure. Then, $\mu \cong \mu^{w,\tau}$, where τ is a run for $w^{\mu,\ell}$ in $\mathcal{G}^\alpha$, for a given linearization ℓ of μ. Furthermore, $\mu^{w,\tau} = \langle \lambda^{w,\tau}, \vartheta^{w,\tau} \rangle \in Mod(\alpha)$.*

Proof: We start by defining a bijection between Ev and $Ev^{w,\tau}$, where $Ev^{w,\tau}$ denotes the set of events of $\mu^{w,\tau}$. Let $f : Ev \to Ev^{w,\tau}$ be such that $f(e) = e_k$ where k is such that $\ell(k) = e$. This k exists because ℓ is a linearization of $\langle Ev, \leq \rangle$. Furthermore, let $e, e' \in Ev$ such that $\ell(k) = e$ and $\ell(k') = e'$, for some $k, k' \in \mathbb{N}_1$. Then $e \leq e'$ iff $\ell(k) \leq \ell(k')$ iff $k \leq k'$ iff $e_k \leq^{w,\tau} e_{k'}$ iff $f(e) \leq^{w,\tau} f(e')$. Hence, $\lambda \cong \lambda^{w,\tau}$. Recall that $f(\emptyset) = \emptyset$ and $f(\xi^k) = f(\{\ell(1), \ldots, \ell(k)\}) = \{e_1, \ldots, e_k\}$. Next, we prove that $\mu \cong \mu^{w,\tau}$. Let $p \in Prop_i$. Then, for $k \in \mathbb{N}$,

$$
\begin{array}{lll}
p \in \vartheta_i^{w,\tau}(f(\xi^k)|_i) & \text{iff} \quad p \in \vartheta_i^{w,\tau}(\{e_1, \ldots, e_k\}|_i) & \text{(as observed above)} \\
& \text{iff} \quad @_i[p] \in q_k\downarrow_i & \text{(by definition of } \mu^{w,\tau}) \\
& \text{iff} \quad p \in w_k^{\mu^{w,\tau},\ell} & \text{(by definition of } \delta_i^\alpha) \\
& \text{iff} \quad \mu_i^{w,\tau}, \xi^k|_i \Vdash_i p & \text{(by definition of } w^{\mu,\ell}) \\
& \text{iff} \quad p \in \vartheta_i(\xi^k|_i) & \text{(by definition of satisfaction).}
\end{array}
$$

Hence, we have established the condition on ϑ and $\vartheta^{w,\tau}$ and so we can conclude that $\mu \cong \mu^{w,\tau}$. We now prove that $\mu^{w,\tau} \in Mod(\alpha)$. But this is an immediate consequence of Lemma 3.19, given that $\mu \in Mod(\alpha)$. □

Lemma 3.17 and Lemma 3.20 allow us to conclude that $Mod(\alpha)$ and $L(\mathcal{G}^\alpha)$ have essentially the same information.

We now illustrate all these constructions with an example.

Example 3.21 Consider the signature $\langle\{i,j\},\{Prop_i, Prop_j\}\rangle$ with $Prop_i = \{p\}$ and $Prop_j = \{q_1, q_2\}$ and let α be the formula $@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])]$. We start by presenting the α-elementary sets B_i for $i = 1, \ldots, 28$. For the sake of simplicity we omit references to global negations, given that in any elementary set B, $\neg\, @_i[\varphi] \in B$ if and only if $@_i[\neg\, \varphi] \in B$.

1. $\{@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[p], @_i[\copyright_j[q_1 \Rightarrow q_2]], @_j[q_1 \Rightarrow q_2], @_j[q_1], @_j[q_2]\}$
2. $\{@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[p], @_i[\copyright_j[q_1 \Rightarrow q_2]], @_j[q_1 \Rightarrow q_2], @_j[\neg\, q_1], @_j[q_2]\}$
3. $\{@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[p], @_i[\copyright_j[q_1 \Rightarrow q_2]], @_j[q_1 \Rightarrow q_2], @_j[\neg\, q_1], @_j[\neg\, q_2]\}$
4. $\{@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[p], @_i[\copyright_j[q_1 \Rightarrow q_2]], @_j[\neg(q_1 \Rightarrow q_2)], @_j[q_1], @_j[\neg\, q_2]\}$
5. $\{@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[\neg\, p], @_i[\copyright_j[q_1 \Rightarrow q_2]], @_j[q_1 \Rightarrow q_2], @_j[q_1], @_j[q_2]\}$
6. $\{@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[\neg\, p], @_i[\copyright_j[q_1 \Rightarrow q_2]], @_j[q_1 \Rightarrow q_2], @_j[\neg\, q_1], @_j[q_2]\}$
7. $\{@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[\neg\, p], @_i[\copyright_j[q_1 \Rightarrow q_2]], @_j[q_1 \Rightarrow q_2], @_j[\neg\, q_1], @_j[\neg\, q_2]\}$
8. $\{@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[\neg\, p], @_i[\copyright_j[q_1 \Rightarrow q_2]], @_j[\neg(q_1 \Rightarrow q_2)], @_j[q_1], @_j[\neg\, q_2]\}$
9. $\{@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[\neg\, p], @_i[\neg\, \copyright_j[q_1 \Rightarrow q_2]], @_j[q_1 \Rightarrow q_2], @_j[q_1], @_j[q_2]\}$
10. $\{@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[\neg\, p], @_i[\neg\, \copyright_j[q_1 \Rightarrow q_2]], @_j[q_1 \Rightarrow q_2], @_j[\neg\, q_1], @_j[q_2]\}$
11. $\{@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[\neg\, p], @_i[\neg\, \copyright_j[q_1 \Rightarrow q_2]], @_j[q_1 \Rightarrow q_2], @_j[\neg\, q_1], @_j[\neg\, q_2]\}$
12. $\{@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[\neg\, p], @_i[\neg\, \copyright_j[q_1 \Rightarrow q_2]], @_j[\neg(q_1 \Rightarrow q_2)], @_j[q_1], @_j[\neg\, q_2]\}$
13. $\{@_i[\neg\, \mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[p], @_i[\copyright_j[q_1 \Rightarrow q_2]], @_j[q_1 \Rightarrow q_2], @_j[q_1], @_j[q_2]\}$
14. $\{@_i[\neg\, \mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[p], @_i[\copyright_j[q_1 \Rightarrow q_2]], @_j[q_1 \Rightarrow q_2], @_j[\neg\, q_1], @_j[q_2]\}$
15. $\{@_i[\neg\, \mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[p], @_i[\copyright_j[q_1 \Rightarrow q_2]], @_j[q_1 \Rightarrow q_2], @_j[\neg\, q_1], @_j[\neg\, q_2]\}$
16. $\{@_i[\neg\, \mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[p], @_i[\copyright_j[q_1 \Rightarrow q_2]], @_j[\neg(q_1 \Rightarrow q_2)], @_j[q_1], @_j[\neg\, q_2]\}$
17. $\{@_i[\neg\, \mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[\neg\, p], @_i[\copyright_j[q_1 \Rightarrow q_2]], @_j[q_1 \Rightarrow q_2], @_j[q_1], @_j[q_2]\}$
18. $\{@_i[\neg\, \mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[\neg\, p], @_i[\copyright_j[q_1 \Rightarrow q_2]], @_j[q_1 \Rightarrow q_2], @_j[\neg\, q_1], @_j[q_2]\}$
19. $\{@_i[\neg\, \mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[\neg\, p], @_i[\copyright_j[q_1 \Rightarrow q_2]], @_j[q_1 \Rightarrow q_2], @_j[\neg\, q_1], @_j[\neg\, q_2]\}$
20. $\{@_i[\neg\, \mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[\neg\, p], @_i[\copyright_j[q_1 \Rightarrow q_2]], @_j[\neg(q_1 \Rightarrow q_2)], @_j[q_1], @_j[\neg\, q_2]\}$
21. $\{@_i[\neg\, \mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[\neg\, p], @_i[\neg\, \copyright_j[q_1 \Rightarrow q_2]], @_j[q_1 \Rightarrow q_2], @_j[q_1], @_j[q_2]\}$
22. $\{@_i[\neg\, \mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[\neg\, p], @_i[\neg\, \copyright_j[q_1 \Rightarrow q_2]], @_j[q_1 \Rightarrow q_2], @_j[\neg\, q_1], @_j[q_2]\}$
23. $\{@_i[\neg\, \mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[\neg\, p], @_i[\neg\, \copyright_j[q_1 \Rightarrow q_2]], @_j[q_1 \Rightarrow q_2], @_j[\neg\, q_1], @_j[\neg\, q_2]\}$
24. $\{@_i[\neg\, \mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[\neg\, p], @_i[\neg\, \copyright_j[q_1 \Rightarrow q_2]], @_j[\neg(q_1 \Rightarrow q_2)], @_j[q_1], @_j[\neg\, q_2]\}$
25. $\{@_i[\neg\, \mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[\neg(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p], @_i[\neg\, \copyright_j[q_1 \Rightarrow q_2]], @_j[q_1 \Rightarrow q_2], @_j[q_1], @_j[q_2]\}$
26. $\{@_i[\neg\, \mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[\neg(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p], @_i[\neg\, \copyright_j[q_1 \Rightarrow q_2]], @_j[q_1 \Rightarrow q_2], @_j[\neg\, q_1], @_j[q_2]\}$
27. $\{@_i[\neg\, \mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[\neg(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p], @_i[\neg\, \copyright_j[q_1 \Rightarrow q_2]], @_j[q_1 \Rightarrow q_2], @_j[\neg\, q_1], @_j[\neg\, q_2]\}$
28. $\{@_i[\neg\, \mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[\neg(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p], @_i[\neg\, \copyright_j[q_1 \Rightarrow q_2]], @_j[\neg(q_1 \Rightarrow q_2)], @_j[q_1], @_j[\neg\, q_2]\}$

We now define the GNBAs $\mathcal{G}_i^\alpha$ and $\mathcal{G}_j^\alpha$. Automaton $\mathcal{G}_i^\alpha = \langle Q_i^\alpha, \mathcal{V}_i^\alpha, \delta_i^\alpha, Q_{0_i}^\alpha, \mathcal{F}_i^\alpha\rangle$ is as follows:

- $Q_i^\alpha = \{q_1^i, q_2^i, q_3^i, q_4^i, q_5^i, q_6^i, q_7^i\}$, where
  - $q_1^i = \{@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[p], @_i[\copyright_j[q_1 \Rightarrow q_2]]\}$
  - $q_2^i = \{@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[\neg p], @_i[\copyright_j[q_1 \Rightarrow q_2]]\}$
  - $q_3^i = \{@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[\neg p], @_i[\neg \copyright_j[q_1 \Rightarrow q_2]]\}$
  - $q_4^i = \{@_i[\neg \mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[p], @_i[\copyright_j[q_1 \Rightarrow q_2]]\}$
  - $q_5^i = \{@_i[\neg \mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[\neg p], @_i[\copyright_j[q_1 \Rightarrow q_2]]\}$
  - $q_6^i = \{@_i[\neg \mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]], @_i[\neg p], @_i[\neg \copyright_j[q_1 \Rightarrow q_2]]\}$
  - $q_7^i = \{@_i[\neg \mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[\neg(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])], @_i[p], @_i[\neg \copyright_j[q_1 \Rightarrow q_2]]\}$
- $\mathcal{V}_i^\alpha = \{\{@_i[\neg p]\}, \{@_i[p]\}\}$,
- δ_i^α is defined as follows

	$\{@_i[\neg p]\}$	$\{@_i[p]\}$
q_1^i	$\{\}$	$\{q_1^i, q_2^i, q_3^i\}$
q_2^i	$\{q_1^i, q_2^i, q_3^i\}$	$\{\}$
q_3^i	$\{q_1^i, q_2^i, q_3^i\}$	$\{\}$

- $Q_{0_i}^\alpha = \{q_3^i\}$,
- $\mathcal{F}_i^\alpha = \{F_\alpha\}$ where $F_\alpha = \{q_1^i, q_2^i, q_3^i, q_7^i\}$.

Observe that, for instance, q_1^i is $B_1{\downarrow_i} = B_2{\downarrow_i} = B_3{\downarrow_i} = B_4{\downarrow_i}$ and q_2^i is $B_5{\downarrow_i} = B_6{\downarrow_i} = B_7{\downarrow_i} = B_8{\downarrow_i}$. In what concerns initial states, we have that $\alpha \in q_1^i$, $\alpha \in q_2^i$ and $\alpha \in q_3^i$ but we also have $@_i[\copyright_j[q_1 \Rightarrow q_2]] \in q_1^i$ and $@_i[\copyright_j[q_1 \Rightarrow q_2]] \in q_2^2$. So $q_1^i, q_2^i \notin Q_{0_i}$ and q_3^i is the only initial state of $\mathcal{G}_i^\alpha$. We only present the transition function for the reachable states. In this case, observe that states $q_4^i, q_5^i, q_6^i, q_7^i$ are not reachable. Finally, we only have one set of acceptance states F_α as α is the only formula involving one temporal operator G. States q_4^i, q_5^i, q_6^i are not accepting because $@_i[p \Rightarrow \copyright_j[q_1 \Rightarrow q_2]]$ occurs in those states but $@_i[\mathsf{G}(p \Rightarrow \copyright_j[q_1 \Rightarrow q_2])]$ does not. The diagram for the reachable part of $\mathcal{G}_i^\alpha$ is presented in Figure 5, where we omit the reference to global formulas in the literals, that is, we write p instead of $@_i[p]$ and $\neg p$ instead of $@_i[\neg p]$.

Automaton $\mathcal{G}_j^\alpha = \langle Q_j^\alpha, \mathcal{V}_j^\alpha, \delta_j^\alpha, Q_{0_j}^\alpha, \mathcal{F}_j^\alpha \rangle$ is as follows:

- $Q_j^\alpha = \{q_1^j, q_2^j, q_3^j, q_4^j\}$, where:
  1. $q_1^j = \{@_j[q_1 \Rightarrow q_2], @_j[q_1], @_j[q_2]\}$
  2. $q_2^j = \{@_j[q_1 \Rightarrow q_2], @_j[\neg q_1], @_j[q_2]\}$
  3. $q_3^j = \{@_j[q_1 \Rightarrow q_2], @_j[\neg q_1], @_j[\neg q_2]\}$
  4. $q_4^j = \{@_j[\neg(q_1 \Rightarrow q_2)], @_j[q_1], @_j[\neg q_2]\}$
- $\mathcal{V}_j^\alpha = \{\{@_j[\neg q_1], @_j[\neg q_2]\}, \{@_j[\neg q_1], @_j[q_2]\}, \{@_j[q_1], @_j[\neg q_2]\}, \{@_j[q_1], @_j[q_2]\}\}$,
- δ_j^α is defined as follows

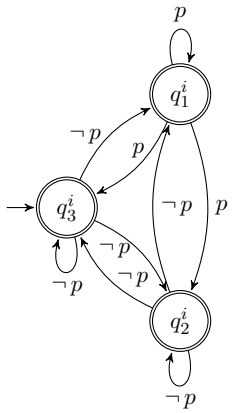

Figure 5: Automaton $\mathcal{G}_i^\alpha$.

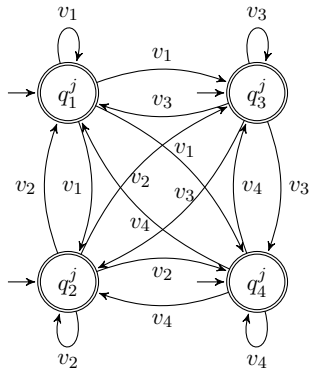

Figure 6: Automaton $\mathcal{G}_j^\alpha$.

	$\{@_j[\neg q_1], @_j[\neg q_2]\}$	$\{@_j[\neg q_1], @_j[q_2]\}$	$\{@_j[q_1], @_j[\neg q_2]\}$	$\{@_j[q_1], @_j[q_2]\}$
q_1^j	$\{\}$	$\{\}$	$\{\}$	$\{q_1^j, q_2^j, q_3^j, q_4^j\}$
q_2^j	$\{\}$	$\{q_1^j, q_2^j, q_3^j, q_4^j\}$	$\{\}$	$\{\}$
q_3^j	$\{q_1^j, q_2^j, q_3^j, q_4^j\}$	$\{\}$	$\{\}$	$\{\}$
q_4^j	$\{\}$	$\{\}$	$\{q_1^j, q_2^j, q_3^j, q_4^j\}$	$\{\}$

- $Q_{0_j}^\alpha = Q_j^\alpha$,
- $\mathcal{F}_j^\alpha = \{\}$.

A consequence of $\mathcal{F}_j^\alpha = \{\}$ is that every run in $\mathcal{G}_j^\alpha$ is accepting. As noted above (after the definition of GNBA), we can replace $\mathcal{F}_j^\alpha = \{\}$ by $\mathcal{F}_j^\alpha = \{Q_j^\alpha\}$. The diagram for $\mathcal{G}_j^\alpha$ is presented in Figure 6, where $v_1 = \{@_j[q_1], @_j[q_2]\}$, $v_2 = \{@_j[\neg q_1], @_j[q_2]\}$, $v_3 = \{@_j[\neg q_1], @_j[\neg q_2]\}$, and $v_4 = \{@_j[q_1], @_j[\neg q_2]\}$.

Observe that both $\mathcal{G}_i^\alpha$ and $\mathcal{G}_j^\alpha$ are also NBAs because the set of acceptance sets is singular and, in this case, a GNBA can be regarded as an NBA. Finally, we consider the GDNBA $\mathcal{G}^\alpha$ based on $\mathcal{G}_i^\alpha, \mathcal{G}_j^\alpha$. To keep the construction simple, when defining this automaton, we consider only the reachable part of $\mathcal{G}_i^\alpha$.

- $Q^\alpha = Q_i^\alpha \times Q_j^\alpha = \{\langle q_1^i, q_1^j\rangle, \langle q_1^i, q_2^j\rangle, \langle q_1^i, q_3^j\rangle, \langle q_1^i, q_4^j\rangle, \langle q_2^i, q_1^j\rangle, \langle q_2^i, q_2^j\rangle, \langle q_2^i, q_3^j\rangle, \langle q_2^i, q_4^j\rangle,$

$$\langle q_3^i, q_1^j\rangle, \langle q_3^i, q_2^j\rangle, \langle q_3^i, q_3^j\rangle, \langle q_3^i, q_4^j\rangle\}$$

- $\Sigma^\alpha = \{\{@_i[\neg\, p]\}, \{@_i[p]\},$
  $\{@_j[\neg\, q_1], @_j[\neg\, q_2]\}, \{@_j[\neg\, q_1], @_j[q_2]\}, \{@_j[q_1], @_j[\neg\, q_2]\}, \{@_j[q_1], @_j[q_2]\}\},$
  $\{@_i[\neg\, p], @_j[\neg\, q_1], @_j[\neg\, q_2]\}, \{@_i[\neg\, p], @_j[\neg\, q_1], @_j[q_2]\},$
  $\{@_i[\neg\, p], @_j[q_1], @_j[\neg\, q_2]\}, \{@_i[\neg\, p], @_j[q_1], @_j[q_2]\}\},$
  $\{@_i[p], @_j[\neg\, q_1], @_j[\neg\, q_2]\}, \{@_i[p], @_j[\neg\, q_1], @_j[q_2]\},$
  $\{@_i[p], @_j[q_1], @_j[\neg\, q_2]\}, \{@_i[p], @_j[q_1], @_j[q_2]\}\}$
- δ^α is defined as follows, where we refrain from presenting the whole transition function but present only some examples:
  - $\delta^\alpha(\langle q_1^i, q_1^j\rangle, \{@_i[p]\}) = \{\langle q_3^i, q_1^j\rangle\}$
  - $\delta^\alpha(\langle q_1^i, q_1^j\rangle, \{@_j[q_1], @_j[q_2]\}) = \{\langle q_1^i, q_1^j\rangle, \langle q_1^i, q_2^j\rangle, \langle q_1^i, q_3^j\rangle, \langle q_1^i, q_4^j\rangle\}$
  - $\delta^\alpha(\langle q_1^i, q_1^j\rangle, \{@_i[p], @_j[q_1], @_j[q_2]\}) =$
    $\{\langle q_1^i, q_1^j\rangle, \langle q_1^i, q_2^j\rangle, \langle q_1^i, q_3^j\rangle, \langle q_2^i, q_1^j\rangle, \langle q_2^i, q_2^j\rangle, \langle q_2^i, q_3^j\rangle, \langle q_3^i, q_4^j\rangle\}$
- $Q_0^\alpha = Q_{0_i}^\alpha \times Q_{0_j}^\alpha = \{\langle q_3^i, q_1^j\rangle, \langle q_3^i, q_2^j\rangle, \langle q_3^i, q_3^j\rangle, \langle q_3^i, q_4^j\rangle\}$,
- $\mathcal{F}^\alpha = \{\mathcal{F}_i^\alpha, \mathcal{F}_j^\alpha\}$ where $\mathcal{F}_i^\alpha = Q_i^\alpha$ and $\mathcal{F}_j^\alpha = Q_j^\alpha$.

Observe that in the case of $\delta^\alpha(\langle q_1^i, q_1^j\rangle, \{@_i[p]\})$, only $\mathcal{G}_i^\alpha$ is advancing so, by (SC1), the target states cannot contain communication formulas. Hence $\langle q_1^i, q_1^j\rangle$ and $\langle q_2^i, q_1^j\rangle$ cannot be considered. In the case of $\delta^\alpha(\langle q_1^i, q_1^j\rangle, \{@_j[q_1], @_j[q_2]\})$ there are no restrictions because there are no communication formulas involved. In the case of $\delta^\alpha(\langle q_1^i, q_1^j\rangle, \{@_i[p], @_j[q_1], @_j[q_2]\})$ as both automata advance, (SC1) and (SC2) state that $@_i[©_i[\varphi]] \in q'{\downarrow_i}$ if and only if $@_j[\varphi] \in q'{\downarrow_j}$ for any target state q'. □

As it can be perceived by the previous example, in general, the size of $\mathcal{G}^\alpha$ is $O(2^{|\alpha|})$, even though some optimizations can be carried out in order to reduce its size. For the purpose of model checking, it is convenient to consider a particular case of GDNBA with a single set of acceptance states, which we call *distributed nondeterministic Büchi automaton (DNBA)*. It is possible to transform a GDNBA into an equivalent DNBA, i.e., a DNBA that accepts the same language as the original GDNBA. This transformation is analogous to a transformation from generalized nondeterministic Büchi automata to nondeterministic Büchi automata (cf. [1]). In the sequel, we will consider the DNBA $\mathcal{D}^\alpha$ equivalent to $\mathcal{G}^\alpha$ such that $L(\mathcal{D}^\alpha)$ captures $Mod(\alpha)$.

4 Model checking in DTL

In this section, we establish the relevant results to propose an automata-theoretic model checking algorithm for DTL. We follow an approach similar to the usual automata-theoretic approach to model checking in LTL [31, 30]. We start by introducing the notions of transition system and distributed transition system. When specifying a multi-agent system, we usually define a transition system for each agent and then compose the systems in one single transition system that captures the whole multi-agent system. Herein, we try to keep the components isolated and, to this end, use the notion of a distributed transition system.

Definition 4.1 *A* transition system *is a tuple* $\mathcal{T} = \langle S, A, \longrightarrow, I, P, L\rangle$ *such that*

- S *is a nonempty finite set of* states*;*
- A *is a finite set of* action symbols*;*
- $\longrightarrow : S \times A \to S$ *is the* transition function*;*
- $I \subseteq S$ *is a set of* initial states*;*
- P *is a finite set of* propositional symbols*;*
- $L : S \to 2^P$ *is a* labelling function*.*

As before, we will write $s \xrightarrow{a} s'$ whenever $s' = \longrightarrow(a, s)$. A *path* in $\mathcal{T}$ is an infinite sequence $s_0 s_1 s_2 \ldots$ of states such that $s_0 \in I$ and $s_k \xrightarrow{a_{k+1}} s_{k+1}$, for every $a_k \in A$ and $k \in \mathbb{N}$. We denote by $\mathit{Paths}(\mathcal{T})$ the set of all paths in $\mathcal{T}$. The *trace* of a path $\pi = s_0 s_1 s_2 \ldots$ is the sequence $L(s_0)L(s_1)L(s_2)\ldots$ and we denote it by $\mathit{trace}(\pi)$. Since each set of propositional symbols $L(s)$ defines a valuation, a trace can be seen as an infinite sequence of valuations. In the sequel, we will write $\overline{L(s)}$ to denote the valuation $\overline{L(s)} = L(s) \cup \{\neg p \mid p \notin L(s)\}$. Additionally, we will write $\mathit{Traces}(\mathcal{T})$ to denote the set of all traces of $\mathcal{T}$.

Definition 4.2 *For each* $i \in Id$*, let* $\mathcal{T}_i = \langle S_i, A_i, \longrightarrow_i, I_i, P_i, L_i\rangle$ *be a transition system such that for distinct* $i, j \in Id$*,* $P_i \cap P_j = \emptyset$*. A* distributed transition system (DTS) *for* $\{\mathcal{T}_i\}_{i\in Id}$ *is the transition system* $\mathcal{T} = \langle S, A, \longrightarrow, I, P, L\rangle$ *such that:*

- $S = \prod_{i\in Id} S_i$*;*
- $A = \bigcup_{i\in Id} A_i$*;*
- $\longrightarrow : S \times A \to S$ *is such that* $s \xrightarrow{a} s'$ *when:*
  - $s{\downarrow_i} \xrightarrow{a}_i s'{\downarrow_i}$ *if* $a \in A_i$*;*
  - $s{\downarrow_i} = s'{\downarrow_i}$ *if* $a \notin A_i$*;*
- $I = \prod_{i\in Id} I_i$*;*
- $P = \bigcup_{i\in Id} P_i$*;*
- $L : S \to 2^P$ *is such that* $L(s) = \bigcup_{i\in Id} L_i(s{\downarrow_i})$*.*

A distributed transition system captures the parallel composition of transition systems, where synchronization is achieved by action synchronization. If two or more agents share the same action symbol, then a transition label by that symbol will occur simultaneously for all agents sharing that symbol while other agents remain idle. In the sequel, it will be useful to be able to identify the agents that share an action symbol. To this end, we define $\mathit{act}(a) = \{i \in Id \mid a \in A_i\}$. Furthermore, in order to relate transition systems and DTL formulas, we will assume that $P_i = \mathit{Prop}_i$, for every $i \in Id$.

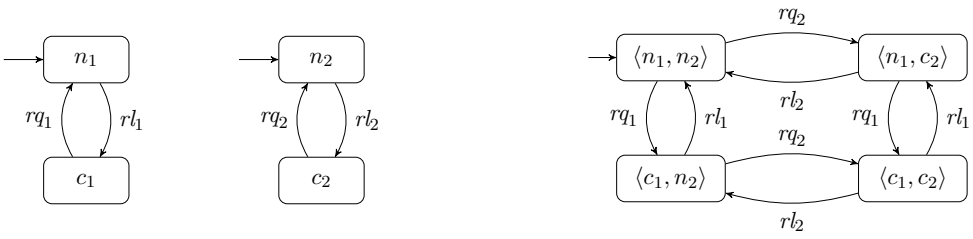

Figure 7: On the left, $\mathcal{T}_1$ and $\mathcal{T}_2$. On the right, the DTS $\mathcal{T}^{\#}$.

Example 4.3 Consider the system composed of two agents trying to access a critical section. The transition system for these two agents, $\mathcal{T}_1$ and $\mathcal{T}_2$, is depicted in Figure 7, on the left. The DTS $\mathcal{T}^{\#}$ for $\{\mathcal{T}_1, \mathcal{T}_2\}$ is depicted in Figure 7, on the right. The initial state n_i is depicted by an incoming arrow, and the labelling functions are defined by $L_1(n_1) = L_2(n_2) = \emptyset$, $L_1(c_1) = \{c_1\}$ and $L_2(c_2) = \{c_2\}$.

By construction, it follows that the initial state of $\mathcal{T}^{\#}$ is $\langle n_1, n_2\rangle$ and the labelling function is $L^{\#}(\langle n_1, n_2\rangle) = \emptyset$, $L^{\#}(\langle c_1, n_2\rangle) = \{c_1\}$, $L^{\#}(\langle n_1, c_2\rangle) = \{c_2\}$ and $L^{\#}(\langle c_1, c_2\rangle) = \{c_1, c_2\}$. This specification is clearly wrong, as both agents can visit the critical section simultaneously. To solve the problem, we add a third agent, a semaphore, to control the access to the critical section. The corresponding transition system $\mathcal{T}_3$ is depicted in Figure 8, on the left, where the label function is defined by $L_3(f) = \emptyset$ and $L_3(b) = \{b\}$. This agent shares actions rq_1, rl_1 with agent 1 and rq_2, rl_2 with agent 2. The DTS $\mathcal{T}^{\bullet}$ for $\{\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3\}$ is depicted in Figure 8, on the right, displaying only the reachable states.

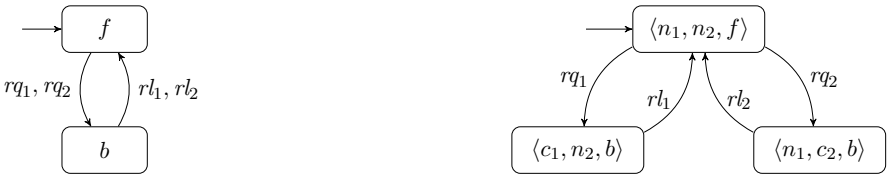

Figure 8: On the left, $\mathcal{T}_3$. On the right, the DTS $\mathcal{T}^{\bullet}$.

□

It is possible to extract a DTL interpretation structure from a path of a distributed transition system.

Definition 4.4 *Let $\pi = s_0 s_1 s_2 \cdots \in Paths(\mathcal{T})$. Define $Ev^{\pi} = \{a \in A | s_k \xrightarrow{a} s_{k+1}$, for some $k \in \mathbb{N}\}$, where we denote the action symbol labelling the transition from s_k to s_{k+i} by a_{k+1}, and let $\mu^{\pi} = \langle \lambda^{\pi}, \vartheta^{\pi}\rangle$ be the interpretation structure induced by π, defined as follows:*

- *$\lambda^{\pi} = \{\lambda_i^{\pi}\}_{i \in Id}$ is such that $\lambda_i^{\pi} = \langle Ev_i^{\pi}, \leq_i^{\pi}\rangle$ and*
  - *$Ev_i^{\pi} = Ev^{\pi} \cap A_i$*
  - *$a_k \leq_i^{\pi} a_{k'}$ for $a_k, a_{k'} \in Ev_i^{\pi}$ such that $k \leq k'$.*
- *$\vartheta^{\pi} = \{\vartheta_i^{\pi}\}_{i \in Id}$ is such that:*
  - *$\vartheta_i^{\pi}(\emptyset) = L_i(s_0{\downarrow_i})$;*

– $\vartheta_i^\pi(\xi_i) = L_i(s_k{\downarrow_i})$, *for* $\xi_i \in \Xi_i$ *such that* $last(\xi_i) = a_k$ *for some* $k \in \mathbb{N}$.

It is straightforward to conclude that, given a DTS $\mathcal{T}$ for $\{\mathcal{T}_i\}_{i\in Id}$ and $\pi \in Traces(\mathcal{T})$, μ^π is an interpretation structure. Moreover, π induces a particular linearization of μ^π. We say that a path π satisfies a formula α, written $\pi \Vdash \alpha$, if $\mu^\pi \Vdash \alpha$. Furthermore, we say that a DTS $\mathcal{T}$ satisfies a formula α, written $\mathcal{T} \vDash \alpha$, if $\pi \Vdash \alpha$ for every path $\pi \in Paths(\mathcal{T})$.

We proceed now to construct the product of a distributed transition system and a distributed Büchi automaton.

Definition 4.5 *Let* $\mathcal{T} = \langle S, A, \longrightarrow, I, P, L\rangle$ *be a distributed transition system for* $\{\mathcal{T}_i\}_{i\in Id}$, *with* $\mathcal{T}_i = \langle S_i, A_i, \longrightarrow_i, I_i, Prop_i, L_i\rangle$, *and* $\mathcal{D}_\alpha = \langle Q^\alpha, \Sigma^\alpha, \delta^\alpha, Q_0^\alpha, F^\alpha\rangle$ *be a DNBA based on* $\{\mathcal{A}_i^\alpha\}_{i\in Id}$, *with* $\mathcal{A}_i^\alpha = \langle Q_i^\alpha, \mathcal{V}_i^\alpha, \delta_i^\alpha, Q_{0_i}^\alpha, F_i^\alpha\rangle$. *Then, the product of* $\mathcal{T}$ *and* $\mathcal{D}^\alpha$ *is the distributed transition system* $\mathcal{T} \otimes \mathcal{D} = \langle S^\otimes, A^\otimes, \longrightarrow^\otimes, I^\otimes, P^\otimes, L^\otimes\rangle$:

- $S^\otimes = S \times Q^\alpha$;
- $A^\otimes = A$;
- $\langle s, q\rangle \overset{a}{\longrightarrow}^\otimes \langle s', q'\rangle$ *if*
  - $s \overset{a}{\longrightarrow} s'$;
  - $q' \in \delta^\alpha(q, \cup_{i\in Id}\overline{L_i(s{\downarrow_i})})$;
- $I^\otimes = \{\langle s_0, q_0\rangle \in S^\otimes : s_0 \in I, q_0 \in Q_0^\alpha\}$;
- $P^\otimes = Q^\alpha$;
- $L^\otimes(\langle s, q\rangle) = \{q\}$.

The idea of this product construction is to recognise traces of paths of the DTS $\mathcal{T}$ as words of the automaton $\mathcal{D}^\alpha$. This is captured by the following diagram, where $v_k = L(s_k)$:

$$s_0 \xrightarrow{a_1} s_1 \xrightarrow{a_2} s_2 \xrightarrow{a_3} \dots$$

$$q_0 \xrightarrow{v_0} q_1 \xrightarrow{v_1} s_2 \xrightarrow{v_2} \dots$$

Hence, in order to have a transition in the distributed transition system $\mathcal{T} \otimes \mathcal{D}^\alpha$ from $\langle s_0, q_0\rangle$ to $\langle s_1, q_1\rangle$ by a_1 it must be the case that $s_0 \xrightarrow{a_1} s_1$ in $\mathcal{T}$ and that at the same time $\mathcal{D}^\alpha$ recognizes the label of s_0, i.e., $q_1 \in \delta(q_0, v_0)$, where v_0 is the union of the labels of the states in s_0 (that is a tuple) of the local transition systems that share the action symbol a_1.

Note that if we define the product of the transition system $\mathcal{T}_i$ by the NBA $\mathcal{A}_i^\alpha$ for each $i \in Id$, which is still a transition system, and then define $\mathcal{T} \otimes \mathcal{D}^\alpha$ as the distributed transition system for $\{\mathcal{T}_i \otimes \mathcal{A}_i^\alpha\}_{i\in Id}$, then the resulting construction yields a transition system isomorphic to the one we defined.

Consider now a set of propositional symbols Pr and a propositional formula γ over Pr. A *persistence property* induced by γ is a set $Per_\gamma \subseteq (2^{Pr})^\omega$ such that

$$Per_\gamma = \{v_0 v_1 v_2 \cdots \mid \text{there is } k \geq 0 \text{ such that for all } k' \geq k,\ v_{k'} \Vdash \gamma\}.$$

This means that γ eventually holds forever. This will be useful to capture the non acceptance condition of a DNBA. Given a DNBA $\mathcal{D}^\alpha$ with set of acceptance states F, we define the

persistence property $Per_{\mathcal{D}^\alpha}$ as the persistence property induced by $\neg F^\alpha$, where $\neg F^\alpha$ denotes the formula

$$\bigwedge_{q \in F^\alpha} \neg q.$$

The problem of verifying if a DTS satisfies a persistence property amounts to finding a loop in the transition system, a well known result that we state without proof [1].

Theorem 4.6 *Let $\mathcal{T}$ be a distributed transition system and γ a propositional formula. Then,*

$$Traces(\mathcal{T}) \not\subseteq Per_\gamma$$

iff

there exists a reachable state s in a cycle such that $L(s) \not\Vdash \gamma$.

The following theorem is at the heart of the model checking algorithm. It states that in order to verify if a DTS $\mathcal{T}$ satisfies a DTL formula α, it is enough to verify if the product $\mathcal{T} \otimes \mathcal{D}^{\neg\alpha}$ satisfies the persistence property $\neg F^\alpha$. This last condition can be checked using the previous result.

Theorem 4.7 *Let $\mathcal{T}$ be a DTS and $\mathcal{D}^{\neg\alpha}$ be a DNBA for $\neg\alpha$. Then, the following statements are equivalent:*

1. $\mathcal{T} \models \alpha$
2. $\{\mu^\pi \mid \pi \in Paths(\mathcal{T})\} \cap \{\mu^{w,\tau} \mid w \in L(\mathcal{D}^{\neg\alpha}), \tau \text{ is accepting run for } w\}^\bullet = \emptyset$
3. $Traces(\mathcal{T} \otimes \mathcal{D}^{\neg\alpha}) \subseteq Per_{\mathcal{D}^{\neg\alpha}}$.

Proof: $(1 \Rightarrow 2)$: Assume that there is $\mu \in \{\mu^\pi \mid \pi \in Paths(T)\} \cap \{\mu^{w,\tau} \mid w \in L(\mathcal{D}^{\neg\alpha}), \tau$ is accepting run for $w\}^\bullet$, i.e., there is a path $\pi \in Paths(T)$ such that $\mu = \mu^\pi$ and there is a word $w \in L(\mathcal{D}^{\neg\alpha})$ with accepting run τ such that $\mu \cong \mu^{w,\tau}$. Then, by Theorem 3.14, we have that $\mu^{w,\tau} \in Mod(\neg\alpha)$ and by Lemma 3.19, we have that $\mu \in Mod(\neg\alpha)$, that is, $\mu^\pi \in Mod(\neg\alpha)$. Since $Mod(\neg\alpha) = \overline{Mod(\alpha)}$, we have $\mu^\pi \not\Vdash \alpha$. Hence $\pi \not\Vdash \alpha$, which implies that $\mathcal{T} \not\models \alpha$.

$(2 \Rightarrow 1)$: Assume now that $\mathcal{T} \not\models \alpha$. Then, there is $\pi \in Paths(\mathcal{T})$ such that $\pi \not\Vdash \alpha$. This means that $\mu^\pi \not\Vdash \alpha$, i.e., $\mu^\pi \notin Mod(\alpha)$. Thus, $\mu^\pi \in Mod(\neg\alpha)$. By Lemma 3.20, there is a run τ for $w^{\mu^\pi,\ell}$ in $\mathcal{D}^{\neg\alpha}$ such that $\mu^\pi \cong \mu^{w,\tau}$. Hence, $\mu^\pi \in \{\mu^{w,\tau} \mid w \in L(\mathcal{D}^{\neg\alpha}), \tau$ is accepting run for $w\}^\bullet$.

$(2 \Rightarrow 3)$: Assume that there is $\sigma \in Traces(\mathcal{T} \otimes \mathcal{D}^{\neg\alpha})$ such that $\sigma \notin Per_{\mathcal{D}^{\neg\alpha}}$. Hence, there is a path $\pi' = \langle s_0, q_0\rangle\langle s_1, q_1\rangle\langle s_2, q_2\rangle \ldots$ such that $trace(\pi') = \sigma$, i.e, $\sigma = \{q_0\}\{q_1\}\{q_2\}\ldots$. Furthermore, as $\sigma \notin Per_{\mathcal{D}^{\neg\alpha}}$, there are infinitely many acceptance states in σ. Let π and τ be the projections of π' on $\mathcal{T}$ and $\mathcal{D}^{\neg\alpha}$, respectively, i.e., $\pi = s_0 s_1 s_2 \ldots$ and $\tau = q_0 q_1 q_2 \ldots$. Note that π is a path in $\mathcal{T}$ and τ is a run for $trace(\pi) = L(s_0)L(s_1)L(s_2)\ldots$ in $\mathcal{D}^{\neg\alpha}$, by construction of $\mathcal{T} \otimes \mathcal{D}^{\neg\alpha}$, and we know that this run is accepting. Let $\mu = \mu^\pi$. We observe that, by construction of μ^π, π induces a linearization ℓ of μ. Now, consider the word $w^{\mu,\ell}$. Observe that $w^{\mu,\ell} = trace(\pi)$. Hence, by Lemma 3.20, $\mu^{w,\tau} \cong \mu$ and so $\mu \in \{\mu^{w,\tau} \mid w \in L(\mathcal{D}^{\neg\alpha}), \tau$ is accepting run for $w\}^\bullet$.

$(3 \Rightarrow 2)$: Assume, as before, that there is $\mu \in \{\mu^\pi \mid \pi \in Paths(T)\} \cap \{\mu^{w,\tau} \mid w \in L(\mathcal{D}^{\neg\alpha}), \tau$ is accepting run for $w\}^\bullet$, i.e., there a path $\pi = s_0 s_1 \cdots \in Paths(T)$ such that $\mu = \mu^\pi$ and

there is a word $w \in L(\mathcal{D}^{\neg\alpha})$ with accepting run τ such that $\mu \cong_f \mu^{w,\tau}$, for some bijection f. As before, we observe that π induces a linearization ℓ of μ and this linearization is such that $\vartheta_i(\ell(k)\downarrow_i) = L_i(s_k\downarrow_i)$. Consider the trace $w^\pi = \mathit{trace}(\pi) = L(s_0)L(s_1)\ldots$ and the word $w^{\mu^{w,\tau},\ell}$. We have that $w^\pi = w^{\mu^{w,\tau},\ell}$. Indeed, let $p \in \mathit{Prop}_i$. Then, $p \in w_k^{\mu^{w,\tau},\ell}$ if and only if $\mu^{w,\tau}, f(\ell(k))\downarrow_i \Vdash_i p$ if and only if $p \in \vartheta_i^{\mu,\tau}(f(\ell(k))\downarrow_i)$ if and only if, by Lemma 3.18, $p \in \vartheta_i(\ell(k))\downarrow_i)$ if and only if, as observed above, $p \in L_i(s_k\downarrow_i)$ if and only if $p \in w_k^\pi$. By Theorem 3.14, we know that $\mu^{w,\tau} \mathit{Mod}(\neg\alpha)$ and by Theorem 3.16, it follows $w^{\mu^{w,\tau},\ell} \in L(\mathcal{D}^{\neg\alpha})$. Hence, $w^\pi \in L(\mathcal{D}^{\neg\alpha})$. Let $\tau = q_0q_1q_2\ldots$ be an accepting run for w^π in $\mathcal{D}^{\neg\alpha}$, i.e.,

$$q_0 \xrightarrow{L(s_0)} q_1 \xrightarrow{L(s_1)} q_2 \xrightarrow{L(s_2)} \ldots$$

As the run is accepting, we know that $q_i \in F^{\neg\alpha}$ for infinitely many indices. We now combine path π with run τ in order to obtain a path in $\mathcal{T} \otimes \mathcal{D}^{\neg\alpha}$:

$$\pi' = \langle s_0, q_0\rangle\langle s_1, q_1\rangle\langle s_2, q_2\rangle\ldots$$

Note that, by construction, π' is indeed a path in $\mathcal{T} \otimes \mathcal{D}^{\neg\alpha}$. Furthermore, its trace is $\sigma = \mathit{trace}(\pi') = \{q_0\}\{q_1\}\{q_2\}\ldots$ and thus $\sigma \notin \mathit{Per}_{\mathcal{D}^{\neg\alpha}}$, given that there are infinitely many acceptance states in σ. □

We can summarise the previous results in the *abstract model checking algorithm* presented in Figure 9. We illustrate this algorithm with some examples.

Input: DTS $\mathcal{T}$ and DTL formula α
Output: True if $\mathcal{T} \vDash \alpha$, False otherwise

---

1. *Construct* $\mathcal{D}^{\neg\alpha}$
2. *Construct* $\mathcal{T} \otimes \mathcal{D}^{\neg\alpha}$
3. **if** *there is a reachable state* s *in* $\mathcal{T} \otimes \mathcal{D}^{\neg\alpha}$ *appearing in a cycle such that* $L(s) \nVdash \neg F^{\neg\alpha}$
   then
      return False
   else
      return True

Figure 9: Abstract model checking algorithm for DTL.

Example 4.8 In this example, we present a simplified version of a two-phase commit protocol from [28] used to commit a transaction in a distributed system. In this protocol, one process acts as the coordinator and works with multiple subordinates. We denote the coordinator by C and assume that there are two subordinates, A and B. The behaviour of these three agents is depicted in Figure 10 by the transitions systems T_C, T_A and T_B.

The commit protocol begins when the coordinator informs her subordinates that she is starting the protocol and that they should prepare to commit. She does this by executing an action (denoted by *prep*) that is synchronized with the actions of the subordinates with the same names. When a subordinate receives the commit request, he checks if he is ready

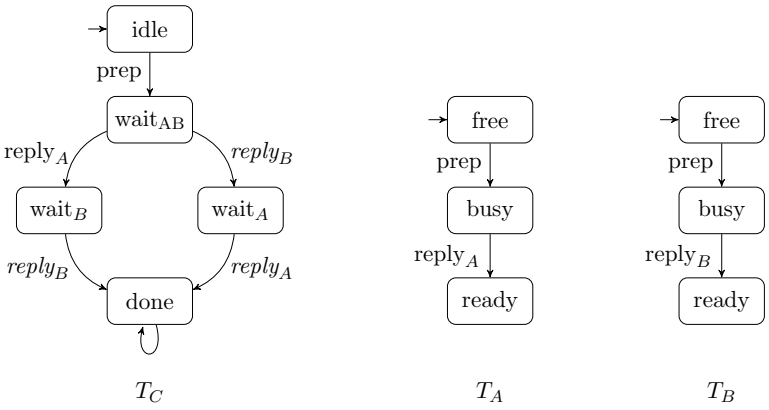

Figure 10: Transition systems for coordinator C and subordinates A and B.

to commit. When he is ready, he informs the coordinator by executing the action $reply_A$ or reply_B. The protocol ends when the coordinator receives the replies from both subordinates.

We want to verify that the system satisfies the property that whenever the coordinator starts a protocol it will eventually end. This is expressed by the formula $\textcircled{C}_c[\mathsf{G}(\text{idle} \Rightarrow \mathsf{F}\,\text{done})]$ that we designate by α_1.

We start by defining the composed system T for $\{T_C, T_A, T_B\}$ that we present in Figure 11, and we want to verify that $T \vDash \alpha_1$. For that purpose, we will use the abstract model checking algorithm, and to do so, we need an automaton for $\neg\,\alpha_1$. For the sake of simplicity, and in order to illustrate the model checking algorithm, we consider a simpler automaton $\mathcal{D}_1$ with only 3 states that accepts $Mod(\neg\,\alpha_1)$, instead of using the general construction for DNBA described above. The automaton $\mathcal{D}_1$ for $Mod(\neg\,\alpha_1)$ is presented in Figure 12 and the composed system $T \otimes \mathcal{D}_1$ is presented in Figure 13.

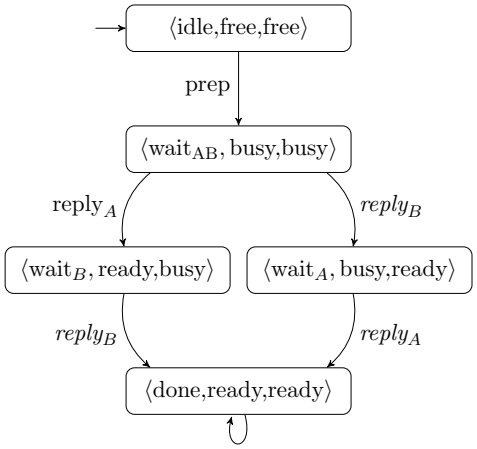

Figure 11: Composed transition system T for $\{T_C, T_A, T_B\}$.

Finally, we observe that the acceptance state q_1 is only visited a finite number of times in $T \otimes \mathcal{D}_1$. Therefore, even though q_1 is reachable, it does not appear in a cycle and so the algorithm returns *True*. Indeed, in this case, by Theorem 4.6, $Traces(T \otimes \mathcal{D}_1) \subseteq Per_{\mathcal{D}_1}$, and by Theorem 4.7, we can conclude that $T \vDash \alpha_1$. □

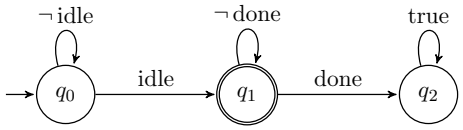

Figure 12: Automaton $\mathcal{D}_1$ for $\neg \textcircled{\scriptsize C}_c[\mathsf{G}(\text{idle} \Rightarrow \mathsf{F}\,\text{done})]$.

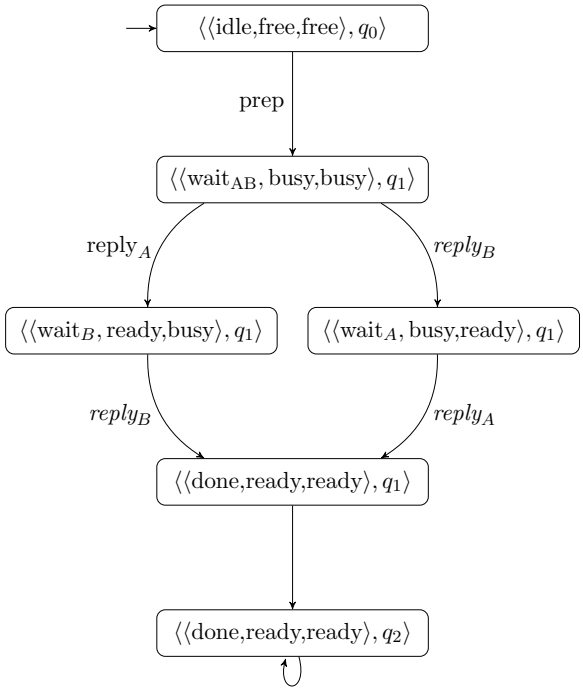

Figure 13: Product construction $T \otimes \mathcal{D}_1$.

Example 4.9 Recall Example 4.3, and consider the following distributed signature $\langle\{1,2,3\}, \{Prop_1, Prop_2, Prop_3\}\rangle$, with $Prop_1 = \{c_1\}$, $Prop_2 = \{c_2\}$ and $Prop_3 = \{b\}$. Over this signature, the formula $\alpha_2 = @_1[\mathsf{G}(\neg(c_1 \wedge \textcircled{\scriptsize C}_2[c_2]))]$ expresses a mutual exclusion property, from the point of view of agent 1. We aim to prove the following assertions:

1. $\mathcal{T}^\bullet \models \alpha_2$,
2. $\mathcal{T}^\# \not\models \alpha_2$.

For that purpose, we will again use the model checking algorithm. As before, we consider a simpler automaton $\mathcal{D}_2$ for $Mod(\neg\,\alpha_2)$ instead of the one obtained by the general construction that we present in Figure 14. The label $\neg(c_1 \wedge c_2)$ is meant to denote all valuations that satisfy

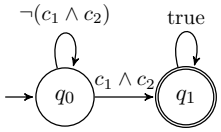

Figure 14: Automaton $\mathcal{D}_2$ for $\neg\, @_1[\mathsf{G}(\neg(c_1 \wedge \textcircled{\scriptsize C}_2[c_2]))]$.

this formula. For instance, $\{\{\neg n_1, c_1\}, \{n_2, \neg c_2\}\}$ satisfies such formula. On the other hand, the valuation $\{\{\neg n_1, c_1\}, \{\neg n_2, c_2\}\}$ does not satisfy this formula but satisfies the formula $c_1 \wedge c_2$.

We now define $\mathcal{T}^\bullet \otimes \mathcal{D}_2$. For the sake of simplicity, only the reachable fragment of the transition system is presented in Figure 15.

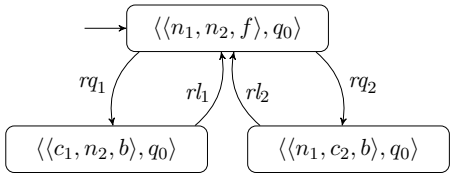

Figure 15: Product construction $\mathcal{T}^\bullet \otimes \mathcal{D}_2$.

Note that transition

$$\langle n_1, n_2, f\rangle \xrightarrow{rq_1} \langle c_1, n_2, b\rangle$$

in $\mathcal{T}^\bullet$ induces the transition

$$\langle\langle n_1, n_2, f\rangle, q_0\rangle \xrightarrow{rq_1} \langle\langle c_1, n_2, b\rangle, q_0\rangle$$

because

$$\bigcup_{i \in Id} \overline{L_i(\langle n_1, n_2, f\rangle)} = \{\{n_1, \neg c_1\}, \{n_2, \neg c_2\}, \{f, \neg b\}\}$$

and

$$\delta(q_0, \{\{n_1, \neg c_1\}, \{n_2, \neg c_2\}, \{f, \neg b\}\}) = \{q_0\}.$$

In the case of the transition system $\mathcal{T}^\bullet$ there is no state where c_1 and c_2 occur simultaneously. Hence, there is no transition in $\mathcal{T}^\bullet \otimes \mathcal{D}_2$ to a state with the label q_1. Therefore, there is no reachable state that satisfies q_1 and so the algorithm returns *True*. Once again, by Theorem 4.6, $\mathit{Traces}(\mathcal{T}^\bullet \otimes \mathcal{D}_2) \subseteq \mathit{Per}_{\mathcal{D}_2}$ and by Theorem 4.7, we can conclude that $\mathcal{T}^\bullet \vDash \alpha_2$.

In Figure 16, we present the reachable fragment of $\mathcal{T}^\# \otimes \mathcal{D}_2$. In this case, there are reachable states with label q_1 that appear within a cycle. Consider, for instance, the path $\langle\langle n_1, n_2\rangle, q_0\rangle\langle\langle n_1, c_2\rangle, q_0\rangle\langle\langle c_1, c_2\rangle, q_0\rangle(\langle\langle n_1, c_2\rangle, q_1\rangle\langle\langle n_1, n_2\rangle, q_1\rangle)^\omega$. In fact, any state with label q_1 appears in a cycle. Therefore, the algorithm returns *False*. Theorem 4.6 implies that $\mathit{Traces}(\mathcal{T}^\# \otimes \mathcal{D}_2) \not\subseteq \mathit{Per}_{\mathcal{D}_2}$ and so, by Theorem 4.7, we have that $\mathcal{T}^\# \nvDash \alpha_2$.

A similar analysis could be made for other properties. For instance, the property that if an agent i, for $i = 1, 2$, is in the critical section, then the semaphore is busy can be expressed by the formula $@_i[\mathsf{G}(c_i \Rightarrow \text{©}_3[b])]$. Conversely, the formula $@_3[\mathsf{G}(b \Rightarrow (\text{©}_1[c_1] \vee \text{©}_2[c_2])])$ expresses that if the semaphore is busy, then one of the agents is in the critical section. □

Note that these examples are distributed in nature, involving several concurrent agents. In this setting, the specification of the systems and their properties can be written in a way that preserves their distributed nature, without the need to resort to translations to sequential systems. This is one of the main advantages of using DTL for distributed system verification.

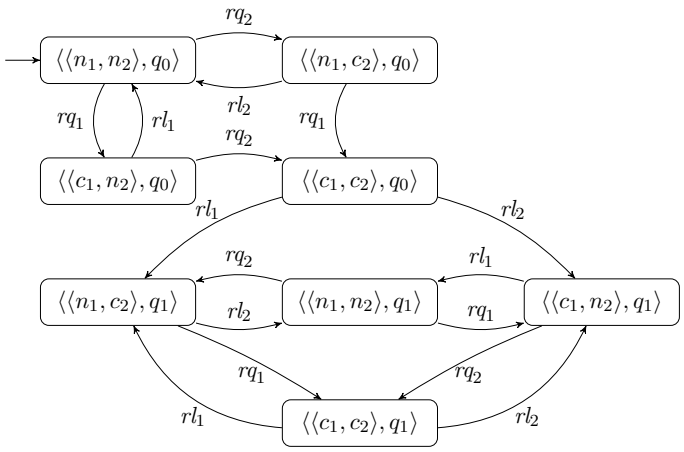

Figure 16: Product construction $\mathcal{T}^{\#} \otimes \mathcal{D}_2$.

5 Concluding remarks

DTL has been widely used for specification and verification of distributed systems and, in particular, for the specification and verification of security protocols [5, 9, 10]. In this context, it seems reasonable to aim at having an automated verification tool. In this paper, we have addressed this issue by endowing DTL with a model checking algorithm. To this end, we have proposed a notion of *distributed Büchi automaton*. We have then endowed DTL with an operational semantics based on DNBAs, where for the local components (that have a linear behaviour) an approach similar to the one followed in [31, 32, 1] was adopted. The construction was proved correct and complete with respect to DTL semantics. Then we have proposed the notion of distributed transition system as a specification tool for distributed systems. With these concepts, we were able to adapt results from automata-theoretic approaches to model checking in LTL to the case of DTL.

As future work, we believe that it would be interesting to extend our approach to other temporal operators, like the until operator and past operators. No surprises are expected as these have been widely studied for LTL and the local agents of DTL have a linear time behaviour.

The work presented in this paper proposes an algorithm towards endowing DTL with a model checking tool. It is also our goal to use this tool to verify some of the problems to which DTL has successfully been applied, but in an automated way [5, 9, 10]. Further along the way, we also intend to study the complexity of our approach and compare it with existing tools.

We aim at adapting existing symbolic model checking techniques for LTL to the case of DTL, along the lines of [22, 27]. Real-time constraints and probabilistic DTL (pDTL) and model checking pDTL are also on the horizon.

Funding

This work was supported in part by Fundação para a Ciência e a Tecnologia, Instituto de Telecomunicações Research Unit under Grant UIDB/50008/2020.

Acknowledgements

The authors would like to thank the anonymous referees whose valuable comments and suggestions greatly improved this work.

References

[1] F. Baier and J.-P. Katoen. *Principles of Model Checking*. The MIT Press, 2008.

[2] E. Bartocci and C. R. Ramakrishnan. Preface of the special issue on model checking of software. *International Journal on Software Tools for Technology Transfer*, 18(4):355–357, 2016.

[3] D. Basin, C. Caleiro, J. Ramos, and L. Viganò. A labeled tableaux for the distributed temporal logic DTL. In *Proceedings of the 15th Int. Symp. on Temporal Representation and Reasoning (TIME 2008)*, pages 101–109. IEEE Computer Society Press, 2008.

[4] D. Basin, C. Caleiro, J. Ramos, and L. Viganò. Labelled tableaux for distributed temporal logic. *Journal of Logic and Computation*, 19:1245–1279, 2009.

[5] D. Basin, C. Caleiro, J. Ramos, and L. Viganò. Distributed temporal logic for the analysis of security protocol models. *Theoretical Computer Science*, 412(31):4007–4043, 2011.

[6] E. Best and C. Fernández. *Nonsequential processes – A Petri net view*. Springer-Verlag, 1988.

[7] D. Bresolin. HyLTL: a temporal logic for model checking hybrid systems. *Electronic Proceedings in Theoretical Computer Science*, 124:73–84, 2013.

[8] C. Caleiro, P. Gouveia, J. Ramos, and L. Viganò. A tableaux-based decision procedure for distributed temporal logic. In C. Caleiro, F. Dionísio, P. Gouveia, P. Mateus, and J. Rasga, editors, *Essays in Honour of Amilcar Sernadas*, Logic and Computation, pages 73–124. College Publications, London, 2017.

[9] C. Caleiro, L. Viganò, and D. Basin. Metareasoning about security protocols using distributed temporal logic. *Electronic Notes in Theoretical Computer Science*, 125(1):67–89, 2005. Preliminary version presented at IJCAR'04 ARSPA Workshop.

[10] C. Caleiro, L. Viganò, and D. Basin. Relating strand spaces and distributed temporal logic for security protocol analysis. *Logic Journal of the IGPL*, 13(6):637–664, 2005.

[11] E. A. Clarke and E. M. Edmund. Characterizing correctness properties of parallel programs using fixpoints. In J. de Bakker and J. van Leeuwen, editors, *Automata, Languages and Programming. ICALP 1980*, Lecture Notes in Computer Science. Springer, Berlin, 1980.

[12] E. M. Clarke and E. A. Emerson. Design and synthesis of synchronization skeletons using branching time temporal logic. In D. Kozen, editor, *Logics of Programs*, pages 52–71. Springer Berlin Heidelberg, 1982.

[13] E. M. Clarke, E. A. Emerson, and A. P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Trans. Program. Lang. Syst.*, 8(2):244–263, 1986.

[14] H.-D. Ehrich and C. Caleiro. Specifying communication in distributed information systems. *Acta Informatica*, 36:591–616, 2000.

[15] H.-D. Ehrich, C. Caleiro, A. Sernadas, and G. Denker. Logics for specifying concurrent information systems. In J. Chomicki and G. Saake, editors, *Logic for Databases and Information Systems*, pages 167–198. Kluwer Academic Publishers, 1998.

[16] G. Holzmann. *The SPIN Model Checker: Primer and Reference Manual*. Addison-Wesley Professional, 2004.

[17] F. Kröger and S. Merz. *Temporal logic and state systems*. Springer, 2008.

[18] O. Lichtenstein and A. Pnueli. Checking that finite state concurrent programs satisfy their linear specification. In *POPL, Proc. 12th ACM Symp.*, pages 97–107, 1985.

[19] O. Lichtenstein and A. Pnueli. Propositional Temporal Logic: Decidability and Completeness. *Logic Journal of the IGPL*, 8(1):55–85, 2000.

[20] K. Lodaya, R. Ramanujam, and P. Thiagarajan. Temporal logics for communicating sequential agents: I. *International Journal of Foundations of Computer Science*, 3(1):117–159, 1992.

[21] K. Lodaya and P. Thiagarajan. A modal logic for a subclass of event structures. In *Proceedings of 14th ICALP*, LNCS 267, pages 290–303. Springer, 1987.

[22] K. L. McMillan. *Symbolic Model Checking*. Kluwer Academic Publishers, Norwell, MA, USA, 1993.

[23] M. Mukund. Automata on distributed alphabets. In D. D'Souza and P. Shankar, editors, *Modern Applications of Automata Theory*, pages 257–288. World Scientific, 2012.

[24] D. Pattinson and B. Reus. A complete temporal and spatial logic for distributed systems. In B. Gramlich, editor, *Frontiers of Combining Systems*, LNCS 3717, pages 122–137. Springer, 2005.

[25] A. Pnueli. The temporal logic of programs. In *FOCS*, pages 46–57, 1977.

[26] R. Ramanujam. Locally linear time temporal logic. In *Proceeding of 11th LICS*, pages 118–127. IEEE Computer Society Press, 1996.

[27] K.Y. Rozier. Linear temporal logic symbolic model checking. *Computer Science Review*, 5(2):163–203, 2011.

[28] A. Tanenbaum and M. van Steen. *Distributed Systems: Principles and Paradigms (2nd Edition)*. Prentice-Hall, Inc., 2006.

[29] P.S. Thiagarajan and J. G. Henriksen. Distributed versions of linear time temporal logic: A trace perspective. In Wolfgang Reisig and Grzegorz Rozenberg, editors, *Lectures on Petri Nets I: Basic Models: Advances in Petri Nets*, pages 643–681. Springer Berlin Heidelberg, 1998.

[30] M. Vardi. From Church and Prior to PSL. In *Proceedings of Workshop on 25 Years of Model Checking*, pages 150–171, 2008.

[31] M. Vardi and P. Wolper. An automata-theoretic approach to automatic program verification. In *Proc. of 1st LICS*, pages 332–344. Cambridge, 1986.

[32] M.Y. Vardi and P. Wolper. Reasoning about infinite computations. *Information and Computation*, 115(1):1 – 37, 1994.

[33] G. Winskel. Event structures. In W. Brauer, W. Reisig, and G. Rozenberg, editors, *Petri Nets: Applications and Relationships to Other Models of Concurrency*, LNCS 255, pages 325–392. Springer-Verlag, 1987.