Reasoning about layered message passing systems

B. Meenakshi and R. Ramanujam*
The Institute of Mathematical Sciences
CIT Campus, Tharamani
Chennai 600113, India.

Abstract

Lamport diagrams are partial orders which depict computations of message passing systems. It is natural to consider generalizations of linear time temporal logics over such diagrams. In [11], we presented a decidable temporal logic with local temporal modalities and a global 'previous' modality to talk of message receipts. It seems reasonable to extend the logic with a global 'next' modality as well, so that sending of messages may also be easily specified, but this (or other similar attempts) lead to undecidability. Hence we consider ways of restricting the models so as to obtain decidability, while retaining the expressiveness of global 'next' and global 'previous' modalities. For this, we consider Lamport diagrams presented as a sequence of *layers*. The layers themselves describe finite communication patterns and a diagram is obtained by sequential composition of such parallel processes. The logic is defined appropriately, with layer formulas describing communications within a layer, and temporal formulas describing the sequence of layers in the computation. When the number of events in layers is uniformly bounded and each layer is communication closed, we get decidability. Alternatively, a stronger uniform bound on what we term channel capacity also yields decidability. We present an example of system specification in the logic.

Keywords: Message passing, partial order models, layered diagrams, system specification, linear time temporal logic, decidability.

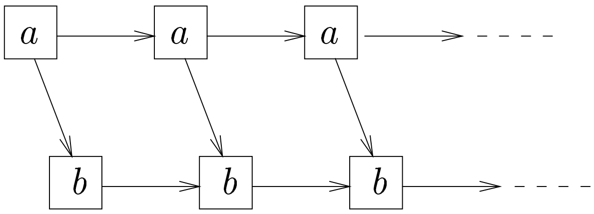
1 Summary

A distributed protocol consists of a fixed finite set of sequential programs called agents which exchange messages between each other. A computation of such a protocol can be depicted as a diagram which depicts the causal ordering of events that occur during the computation: while event occurrences associated with any particular agent are linearly ordered, the diagram itself constitutes a partial order on event occurrences. Causal dependence of one agent's event on another's may be interpreted as message passing, since the receipt of a message is causally dependent on its having been sent earlier. Thus we have diagrams where events of each agent in the computation are given in a line, and the sending and receipt of messages between agents are depicted by arrows between the lines. We refer to such partial orders as Lamport Diagrams (LDs)([8]).

What would be an appropriate linear time temporal logic to reason about systems whose computations are Lamport Diagrams ? One option is to consider the set of all sequentializations of all the computations, so that standard linear time temporal logics (LTL) may be employed. There are many drawbacks to such an approach. For one, it is very easy to come up with diagrams whose sequentializations constitute non-regular languages. Since LTL cannot express such languages, the diagrams specified would have to then exclude such behaviour. (For instance consider the producer — consumer example depicted in Figure 1; considering only finite behaviours, its sequentializations give us the language with all sequences over a's and b's, with every prefix having at least as many a's as b's, a context-free language.) The other drawback is that such a logic specifies how the global states of the system may evolve; we would like to specify the effect of message passing in the systems described, and this is more naturally done using local histories (events) of agents in the system rather than using the global histories of the system.

*Corresponding Author, Tel No: +91-44-22541856, Fax No: +91-44-22541586 E-mail ids: {bmeena,jam}@imsc.res.in

Figure 1: Lamport diagram of the producer — consumer problem



An alternative to linearizations is to interpret the logic directly over Lamport Diagrams themselves, in the spirit of locally based linear time logics ([16], [2], [14], [12]). In these logics, the formulas are interpreted on local event occurrences, and the modalities constrain the causal ordering appropriately. This paper follows such an approach, and the models of formulas are given by Lamport Diagrams, with modalities constraining the covering relation of the associated partial order to implicitly specify the effect of message passing. Coming to the technical issue of expressiveness, we then are faced with the question of whether formulas of such a logic describe regular sets of diagrams. This is the central issue addressed in this paper.

Thus, what we seek is a temporal logic where the modalities refer to successor (predecessor) and future (past) elements in the partial order; at an i-event, its j-successor may be constrained to satisfy a formula. Formulas of the logic constitute abstract (linear-time) agent-based specifications of the behaviour of a distributed system; they correspond to constraints on when an agent may send a message, and what the agent should do on receipt of a message. Further, a sender (receiver) may specify what information is conveyed through the message it sends (receives). It is easy to define a logic which is essentially an indexed version of LTL for each agent, with two additional modalities: $\mathbf{X}\alpha$, asserted by the sender of a message, that α holds on receipt, and $\mathbf{Y}\alpha$, asserted by the receiver of a message, that α held at the time of sending the message.

In [11] we studied m-LTL, a local temporal logic on Lamport diagrams. This logic is essentially like the one described above, except that it only has a modality analogous to the $\mathbf{Y}\alpha$ described above, and without the $\mathbf{X}\alpha$. It was shown that the logic was decidable. The omission of the $\mathbf{X}$ modality was justified on essentially philosophical grounds (that a sender of a message cannot hope to have any certain knowledge of the state of the receiver at the time of receipt). As it turns out, this was a happy choice, since the logic with both of these modalities is undecidable, as we show here. On the other hand, [12] considers a logic which (with some work) may be seen as a similar logic with the $\mathbf{X}$ modality and without the $\mathbf{Y}$ modality, and this logic is also decidable.

Given that the logic with both $\mathbf{X}$ and $\mathbf{Y}$ modalities is undecidable and that we may get decidability retaining only one of them, we may look for logics 'in between'. Note that the requirement that we be able to specify conditions on when an agent may send a message may be met by including special 'send' propositions in the logic, as the decision is local. Similarly, what the agent would (locally) do on receipt of a message, may be specified (less naturally) using special 'receive' propositions. So we may consider a combination of the $\mathbf{Y}$ modality with special send propositions, or the $\mathbf{X}$ modality with special receive propositions. Both these logics turn out to be undecidable as well. (We could consider a locally based logic with neither $\mathbf{X}$ nor $\mathbf{Y}$ modalities but with only the special send and receive propositions; this would specify "pure" signalling systems rather than message passing systems. This may be of independent interest, but not relevant to the discussion in this paper.)

Faced with such undecidability, we turn our attention to the models, and look for what constraints may be placed on the Lamport diagrams so that the restricted classes are decidable. The first observation is that the undecidability results hold for models based on *finite* (but unbounded) diagrams as well, and hence we need to somehow place bounds on the size of diagrams. We are therefore led to the consideration of Lamport diagrams that are obtained by concatenating bounded diagrams. Following this intuition, we define here a class of Layered Lamport Diagrams (LLDs), in such a way that every (countable) LD can be thought of as a (countable) concatenation of finite layers. Bounded LLDs are those where all layers are uniformly bounded.

Such a definition of layering allows very general ways of structuring distributed protocols in an *episodic* manner. The protocol itself is described using several possible episodes, each of which may be a pattern of communications with a bound on the number of messages used within the episode. The episode may be *communication closed*, or there may be *delays*, whereby messages sent during one episode may be received later, in another episode. Communication closed layering of systems is a concept that has been studied at length in the context of process algebra ([13]). On the other hand, it is interesting to consider LLDs where the layers need not be communication closed. In such diagrams, the receipt of a message maybe delayed unboundedly, and a natural condition is to insist that this be bounded as well. Diagrams with uniformly bounded delays correspond to systems with bounded channel capacities, a class commonly studied in distributed computing.

The layering of Lamport diagrams suggests a similar structuring of specifications as well. We have layer formulas that come from a standard modal logic on finite partial orders, and temporal formulas which are built from these using the usual linear time connectives. In such a logic we can describe many interesting properties of LLDs. Specifications in this logic describe a sequential composition of parallel processes, rather than the other way around (in the sense of [3]).

Now we can show that undecidability holds even when we restrict the size of layers to be uniformly bounded. However, imposing either of the restrictions, that the diagrams be communication closed, or that channel capacity be bounded along with a uniform bound on the size of the layers, leads to elementary decidability of the logic, though the complexity is high when layers are not communication closed. We also provide an example of system specification in the logic.

Lamport Diagrams bear a close resemblance to Message Sequence Charts (MSCs), a standard notation in the formal specification of communication protocols ([15], [17]). In fact, LDs can be seen as the partial orders generated by MSCs (in the obvious manner; for instance, see [1]). We conclude the paper with a discussion on this correspondence with MSCs as well as with other related work.

2 A logic on Lamport diagrams

Throughout the paper, we fix $n > 0$, and study distributed systems of n agents. We will follow the linguistic convention that the term 'system' will refer to composite distributed systems, and 'agent' to component processes in the system. The agents communicate with each other by exchanging messages across reliable and order-preserving channels. We refer to the agents by the indices i, $1 \leq i \leq n$ and use the notation $[n]$ to denote the set $\{1, 2, \ldots, n\}$. The set of natural numbers is denoted by $\mathbb{N}$ and the natural ordering on it by $\leq_{\mathbb{N}}$.

Lamport diagrams constitute simple representations of behaviours of such systems and a diagram represents a single (non-sequential) run of the system. They are partial orders with the underlying set of events partitioned into those of n agents in such a way that the events of every agent form a linear order. The ordering relation captures the causal dependence of events.

2.1 Lamport diagrams

Definition 2.1 *A* **Lamport diagram** *is a tuple* $D = (E, \leq, \phi)$ *where*

- E *is an at most countable set of* events.
- $\leq \subseteq (E \times E)$ *is a partial order called the* causality relation *such that for every* $e \in E$, ${\downarrow}e \stackrel{\text{def}}{=} \{e' \in E \mid e' \leq e\}$ *is finite.*
- $\phi : E \to [n]$ *is a* labelling function *which satisfies the following condition:* *Let* $E_i \stackrel{\text{def}}{=} \{e \in E \mid \phi(e) = i\}$ *and* $\leq_i \stackrel{\text{def}}{=} \leq \cap (E_i \times E_i)$*. Then, for every* $i \in [n]$, $\leq_i$ *is a total order on* E_i*.*

For example, the Lamport diagram corresponding to the producer—consumer problem is given in Figure 1 where the events of the agent representing the producer are labelled by a and the events of the agent representing the consumer are labelled by b.

Since for all $e \in E$, ${\downarrow}e$ is finite, $\leq$ must be discrete. Hence there exists $\lessdot \subset \leq$, the *immediate causality relation*, which generates the causality relation; that is: for all e, e', e'', if $e \lessdot e'$ and $e \leq e'' \leq e'$ then

$e'' \in \{e, e'\}$. We have: $\leq = (\lessdot)^*$. For $e, e' \in E$, define $e <_c e'$ iff $e \in E_i$, $e' \in E_j$, $i \neq j$ and $e \lessdot e'$. In this case, we interpret e as the sending of a message by agent i and e' as its corresponding receipt by j. Accordingly, if $e <_c e'$ then, e will be referred to as a send event and e' will be its corresponding receive event. (Note that references to send or receive events are merely conveniences for talking about the $<_c$ ordering.)

Given a Lamport diagram $D = (E, \leq, \phi)$ and $E' \subseteq E$, the Lamport diagram induced by E' is defined as $D_{E'} = (E', \leq \cap (E' \times E'), \phi \upharpoonright E')$ where $\phi \upharpoonright E'$ denotes the projection of ϕ on E'.

2.2 Logic

Fix countable sets of *propositional letters* $(P_1, P_2, \ldots P_n)$ where P_i consists of atomic local properties of agent i. Let $P \stackrel{\text{def}}{=} \cup_i P_i$. The set of i-local propositions P_i also includes a special *type proposition* τ_i which is true at all the events of agent i. Note that P_i and P_j need not be disjoint when $i \neq j$, since we can always use τ_i to disambiguate, if needed.

We first consider a modal logic LD_0, whose syntax is given below:

$$LD_0 ::= p \ \in \ P \mid \tau_i \mid \neg\, \alpha \mid \alpha_1 \ \vee \ \alpha_2 \mid \mathbf{X}\ \alpha \mid \mathbf{Y}\ \alpha \mid \mathbf{F}\ \alpha \mid \mathbf{P}\ \alpha$$

The logic is a standard propositional modal logic over any labelled partial order. The $\mathbf{X}$ modality refers to an immediate successor event and the $\mathbf{Y}$ modality refers to an immediate predecessor event. $\mathbf{F}$ stands for 'future' and $\mathbf{P}$ for 'past' respectively. We will use indexed modalities as abbreviations: for instance, $\mathbf{X}_i\alpha$ denotes the formula $\mathbf{X}(\tau_i \wedge \alpha)$; other abbreviations for $\mathbf{Y}_i$, $\mathbf{F}_i$ and $\mathbf{P}_i$ are defined similarly. The dual of $\mathbf{F}$ is denoted $\mathbf{G}$ and the dual of $\mathbf{P}$ is denoted $\mathbf{H}$. $\overline{\mathbf{X}}$ and $\overline{\mathbf{Y}}$ denote, respectively, the duals of $\mathbf{X}$ and $\mathbf{Y}$. (Note that the abbreviations denote implications for the dual modalities; for instance $\mathbf{G}_i\alpha$ denotes $\mathbf{G}(\tau_i \supset \alpha)$).

The formulas are interpreted on Lamport diagrams. We denote models by $M = (D, V_E)$ where $D = (E, \leq, \phi)$ is a Lamport diagram, $V_E : E \rightarrow 2^P$ is a valuation function such that $V_E(E_i) \subseteq P_i$ for all $i \in [n]$ and for all $e \in E$, $\tau_i \in V_E(e)$ iff $\phi(e) = i$.

Let $\alpha \in LD_0$ and $e \in E$. The notion that α holds at e in M is denoted $M, e \models \alpha$ and is defined inductively as follows:

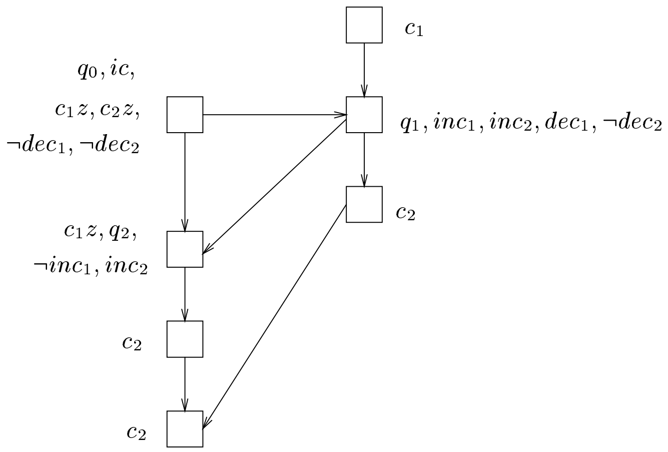
- $M, e \models p$ iff $p \in V_E(e)$.
- $M, e \models \neg\alpha$ iff $M, e \not\models \alpha$.
- $M, e \models \alpha \vee \beta$ iff $M, e \models \alpha$ or $M, e \models \beta$.
- $M, e \models \mathbf{X}\alpha$ iff there exists $e' \in E$ such that $e \lessdot e'$ and $M, e' \models \alpha$.
- $M, e \models \mathbf{F}\alpha$ iff there exists e' such that $e \leq e'$ and $M, e' \models \alpha$.
- $M, e \models \mathbf{Y}\alpha$ iff there exists $e' \in E$ such that $e' \lessdot e$ and $M, e' \models \alpha$.
- $M, e \models \mathbf{P}\alpha$ iff there exists e' such that $e' \leq e$ and $M, e' \models \alpha$.

We say that α is *satisfiable* iff there exists a model $M = (D, V_E)$, where $D = (E, \leq, \phi)$ and $e \in E$ such that $M, e \models \alpha$. When E is finite, then we strengthen the property and say α is *finitely satisfiable*. When $\mid E \mid \leq b$, we say α is *b-satisfiable*.

Except for the presence of special propositions τ_i, the logic is seen to be a standard tense logic on partial orders. The logic may further be strengthened with $\mathbf{U}$ and $\mathbf{S}$ modalities and the technical results which follow still hold for such a logic as well. We use the abbreviations True and False to denote the formulas $p_0 \vee \neg p_0$ and $p_0 \wedge \neg p_0$ respectively, where $p_0 \in P$. Note that the formula $\tau_1 \wedge \mathbf{X}_2\mathsf{True}$ may be seen as asserting the sending of a message by agent 1 to 2, and similarly $\tau_1 \wedge \mathbf{Y}_2\mathsf{True}$ asserts the receipt of a message by 1 from 2. The modalities $\mathbf{X}_i$ and $\mathbf{Y}_i$ act as *local* modalities when asserted at events of agent i and as *global* modalities when asserted at events of some agent j where $j \neq i$.

Theorem 2.2 *The (finite) satisfiability problem for LD_0 is undecidable.*

Figure 2: Lamport diagram corresponding to a run of M



The negative result here mainly stems from the fact that runs of non-deterministic 2-counter machines can be described using Lamport diagrams. A non-deterministic 2-counter machine is given by a tuple $M = (Q, \delta, q_0, q_f)$ where

- Q is a finite set of states.
- $q_0 \in Q$ is the initial state.
- $q_f \in Q$ is the final state.
- $\delta \subseteq T$ is the transition relation where $T = (Q \times \{0,1\}^2 \times Q \times \{-1,0,1\}^2)$

When $(q, x_1, x_2, q', y_1, y_2) \in \delta$ and the machine is in state q, the transition is enabled depending on whether the two counters are zero or non-zero as given by x_1 and x_2 and the machine changes state to q', and the counter values are decremented or incremented or left unchanged, depending on y_1 and y_2. We assume that if $(q, x_1, x_2, q', y_1, y_2) \in \delta$ then $y_i = -1$ implies $x_i = 1$ for $i = 1, 2$ (only a positive counter can be decremented) and $q \neq q_f$ (there are no transitions out of the final state). A *configuration* of M is a triple (q, n_1, n_2) where $q \in Q$ and $n_i \in \mathbb{N}$ are the values of the two counters. We say that $(q, n_1, n_2) \to (q', n_1 + y_1, n_2 + y_2)$ if $(q, x_1, x_2, q', y_1, y_2) \in \delta$ and $n_i = 0$ iff $x_i = 0$ for $i = 1, 2$. A *run* ρ of M is any sequence $(q_0, 0, 0) \to (q_1, n_1^1, n_2^1) \to (q_2, n_1^2, n_2^2) \to \ldots$. A configuration (q, n_1, n_2) is *reachable* if there exists a run of M from $(q_0, 0, 0)$ to (q, n_1, n_2). A configuration is *final* if $q = q_f$. Given a 2-counter machine, the problem of checking if a final configuration is reachable is undecidable.

Given a run ρ of M, we will define a 2-agent Lamport diagram that 'represents' ρ. For example, the Lamport diagram corresponding to the run $\rho = (q_0, 0, 0) \to (q_1, 1, 1) \to (q_2, 0, 2)$ is depicted in Figure 2. The propositions labelling an event are the ones that are satisfied at the event. The proposition ic represents the initial configuration. The propositions c_iz represent the fact that the value of counter $i = 0$. The (non-zero) values of counter i are coded up using propositions c_i for $i = 1, 2$ respectively which are repeated as many times as the value of the counter. A configuration (q, n_1, n_2) with $n_i \neq 0$ for $i = 1, 2$ is represented by a sequence of n_1 events labelled by c_1 followed by the proposition corresponding to q followed by a sequence of n_2 events labelled by c_2. If $n_i = 0$, then proposition c_iz is holds at the event labelled by a state proposition.

Consider a configuration (q, n_1, n_2) represented in the above way say, as a sequence of events of agent 1. A transition of M from (q, n_1, n_2) to (q', n_1', n_2') is coded as follows. There is an immediate successor (to the event labelled by q) in agent 2 which is labelled by q'. The propositions inc_i and dec_i represent incrementing and decrementing counter i for $i = 1, 2$ respectively. The proposition inc_1 is satisfied at the q'-labelled event whenever counter 1 is incremented and is implemented by adding a new event in agent 2 labelled with c_1 just before the q'-labelled event. Similarly, proposition inc_2 is satisfied at the q'-labelled event whenever counter 2 is incremented and is implemented by adding a new event in agent 2 labelled with c_2 just after the q'-labelled event. For decrementing, the proposition dec_1 is satisfied at the q-labelled event whenever counter 1 is decremented and is implemented by not copying the last c_1-labelled event (just before the q-labelled event) from agent 1 to agent 2. Similarly, dec_2 holds at the q-labelled event whenever counter 2 is decremented and is implemented by not copying the first c_2-labelled event (just after the q-labelled event) from agent 1 to agent 2.

Each run of M is represented by a Lamport diagram in this way. We show that such runs can be described by a formula of the logic. That is, given a 2-counter machine M, we define a formula α_M such that α_M is satisfiable iff a final configuration of M is reachable.

It turns out that we can define α_M using a logic with a more restricted syntax than that of LD_0. Consequently, the undecidability of LD_0 follows from the undecidability of the logic LD_1 which is presented next.

It is worth remarking here that decidable logics over partial orders are generally hard to find. An early result of this kind was [9], where the decidability of a fragment of LD_0 without the immediate successor and predecessor modalities was shown. However, the proof was not automata theoretic, and it is instructive to note that the filtration techniques used there do not generalize when immediate predecessor and successor modalities are introduced.

2.3 Variations on the theme

Since the global $\mathbf{X}$ and $\mathbf{Y}$ modalities turn out to be the main source of undecidability, we consider ways of restricting their use. It turns out that the satisfiability problem becomes undecidable even if we consider a weaker logic where one of the $\mathbf{X}$ or $\mathbf{Y}$ modalities is global and the other is local along with special propositions to talk about receiving or sending messages respectively. In fact, we obtain undecidability without using the $\mathbf{P}$ modality and by using only a local $\mathbf{F}$ modality.

Consider the logic LD_1 defined as follows. Let r_i^j, $i,j \in [n]$, $i \neq j$ be special 'receive' propositions in P_i which code up the fact that a particular event is a receive event from agent j.

$$LD_1 ::= p \ \in \ P \mid \tau_i \mid r_i^j \mid \neg\, \alpha \mid \alpha_1 \ \vee \ \alpha_2 \mid \mathbf{X}_i \ \alpha \mid \mathbf{Y}_i \ \alpha \mid \mathbf{F}_i \ \alpha$$

where the semantics is defined inductively as done for LD_0. The only interesting cases are:

- $M, e \models r_i^j$ iff $e \in E_i$ and there exists $e' \in E_j$ such that $e' \lessdot e$.
- $M, e \models \mathbf{X}_i\alpha$ iff there exists $e' \in E_i$ such that $e \lessdot e'$ and $M, e' \models \alpha$.
- $M, e \models \mathbf{Y}_i\alpha$ iff $e \in E_i$ and there exists $e' \in E_i$ such that $e' \lessdot e$ and $M, e' \models \alpha$.
- $M, e \models \mathbf{F}_i\alpha$ iff $e \in E_i$ and there exists $e' \in E_i$ such that $e \leq e'$ and $M, e' \models \alpha$.

The modality $\mathbf{X}_i$ above is a global modality as it can be satisfied at any immediate successor event. The modalities $\mathbf{Y}_i$ and $\mathbf{F}_i$ interpreted at events of agent i are local to i.

We can also define the logic LD_2, a symmetric version of LD_1 with a global $\mathbf{Y}$ modality and a local $\mathbf{X}$ modality and special send propositions. Let s_i^j, $i,j \in [n]$, $i \neq j$ be special 'send' propositions in P_i which code up the fact that a particular event is a send event to agent j.

$$LD_2 ::= p \ \in \ P \mid \tau_i \mid s_i^j \mid \neg\, \alpha \mid \alpha_1 \ \vee \ \alpha_2 \mid \mathbf{X}_i \ \alpha \mid \mathbf{Y}_i \ \alpha \mid \mathbf{F}_i \ \alpha$$

The semantics is again defined inductively as done for LD_0. We note the changes below.

- $M, e \models s_i^j$ iff $e \in E_i$ and there exists $e' \in E_j$ such that $e \lessdot e'$.
- $M, e \models \mathbf{X}_i\alpha$ iff $e \in E_i$ and there exists $e' \in E_i$ such that $e \lessdot e'$ and $M, e' \models \alpha$.
- $M, e \models \mathbf{Y}_i\alpha$ iff there exists $e' \in E_i$ such that $e' \lessdot e$ and $M, e' \models \alpha$.

We show that the satisfiability problems for the logics LD_1 and LD_2 are undecidable.

Theorem 2.3 *The (finite) satisfiability problem for LD_1 is undecidable.*

Proof: As explained earlier, given a 2-counter machine M, we construct a formula α_M such that α_M is satisfiable iff a final configuration of M is reachable.

The set of propositions P_i of agent i include the set $\{c_k z, c_k, inc_k, dec_k, ic\} \cup Q$ for $k = 1, 2$ in addition to special propositions τ_i and r_i^j, $(j \neq i)$ for $i = 1, 2$. In the definition of the formula α_M and in the proofs, we will use the following notation. Given $i \in \{1, 2\}$, $i^{co} = 1$ if $i = 2$ and $i^{co} = 2$ if $i = 1$.

The formula α_M is given by $\alpha_M \stackrel{\text{def}}{=} \mathsf{init} \wedge \mathsf{inv} \wedge \mathsf{fin}$ where init, inv and fin are defined as follows.

- The formula
  $\mathsf{init} \stackrel{\text{def}}{=} \tau_1 \wedge q_0 \wedge c_1 z \wedge c_2 z \wedge \overline{\mathbf{Y}}_1 \mathsf{False} \wedge \neg \mathsf{r}_1^2$
  codes up the initial configuration of $(q_0, 0, 0)$ of M.

- The formula $\mathsf{fin} \stackrel{\text{def}}{=} \mathbf{F}_1 \mathsf{fin}_1 \vee \mathbf{X}_2 \mathbf{F}_2 \mathsf{fin}_2$ where $\mathsf{fin}_1 \stackrel{\text{def}}{=} q_f \wedge \mathbf{G}_1 \overline{\mathbf{X}}_2 \mathsf{False}$ and $\mathsf{fin}_2 \stackrel{\text{def}}{=} q_f \wedge \mathbf{G}_2 \overline{\mathbf{X}}_1 \mathsf{False}$ asserts that a final configuration is reachable and there are no moves out of it.

- The transitions of M and other facts about the two counter values encoded by c_j are defined by the following formulas. $\mathsf{inv} \stackrel{\text{def}}{=} \mathbf{G}_1 \mathsf{inv}_1 \wedge \overline{\mathbf{X}}_2 \mathbf{G}_2 \mathsf{inv}_2$ where $\mathsf{inv}_i \stackrel{\text{def}}{=} (\mathsf{state} \vee \mathsf{ctr}_1 \vee \mathsf{ctr}_2) \wedge \mathsf{trans}_i \wedge \mathsf{consis}_i$ for $i = 1, 2$ and these formulas are defined below.

  – $\mathsf{state} \stackrel{\text{def}}{=} \bigvee_{q \in Q} (q \wedge \bigwedge_{q' \in Q, q \neq q'} \neg q') \wedge \neg c_1 \wedge \neg c_2.$

  – $\mathsf{ctr}_1 \stackrel{\text{def}}{=} c_1 \wedge (\bigwedge_{q \in Q} \neg q) \wedge \neg c_2 \wedge \neg c_1 z.$

  – $\mathsf{ctr}_2 \stackrel{\text{def}}{=} c_2 \wedge (\bigwedge_{q \in Q} \neg q) \wedge \neg c_1 \wedge \neg c_2 z.$

  – The transitions of M are coded up by trans_i as follows.
  For a tuple $t = (q, x, y, q', x', y') \in T$ and $q \neq q_f$, we first define a formula θ_t which codes up the move defined by t as follows.

  $$\theta_t \stackrel{\text{def}}{=} q \wedge (\gamma_x^1 \wedge \gamma_y^2 \wedge \mathbf{X}_{i^{co}} q' \wedge \xi_{x',i}^1 \wedge \xi_{y',i}^2)$$

  where the various sub-formulas are given by

  $\gamma_0^j = c_j z$ $\qquad \gamma_1^j = \neg c_j z$ $\qquad \xi_{0,i}^j = \neg dec_j \wedge \mathbf{X}_{i^{co}} \neg inc_j$
  $\xi_{1,i}^j = \neg dec_j \wedge \mathbf{X}_{i^{co}} inc_j$ $\qquad \xi_{-1,i}^j = dec_j \wedge \mathbf{X}_{i^{co}} \neg inc_j$

  Notice that the $\mathbf{X}_{i^{co}}$ is used as a global modality (to specify what is true at the other agent). Also note that the 'decrement j' decision of a transition causes dec_j to hold at the same event where q holds, whereas the 'increment j' causes inc_j to hold at the event satisfying the successor q' in the other agent i^{co}.
  Now, the transitions of M are coded up as
  $\mathsf{trans}_i \stackrel{\text{def}}{=} (\bigvee_{q \in Q, q \neq q_f} q \supset \bigvee_{t = (q, x, y, q', x', y') \in \delta} \theta_t) \wedge \bigwedge_{t' \notin \delta} \neg \theta_{t'}.$

  – $\mathsf{consis}_i \stackrel{\text{def}}{=} \mathsf{local\text{-}consis}_i \wedge \mathsf{across\text{-}consis}_i$ where
  $\mathsf{local\text{-}consis}_i \stackrel{\text{def}}{=} \mathsf{seq}_i \wedge \mathsf{z\text{-}consis}_i \wedge \mathsf{inc\text{-}consis}_i \wedge \mathsf{dec\text{-}consis}_i \wedge \mathsf{z\text{-}dec\text{-}consis}_i \wedge \mathsf{state\text{-}inc\text{-}dec\text{-}consis}_i$
  where

    * $\mathsf{seq}_i \stackrel{\text{def}}{=} (c_1 \supset \overline{\mathbf{X}}_i (c_1 \vee \mathsf{state})) \wedge (c_2 \supset \overline{\mathbf{X}}_i (c_2 \vee c_1 \vee (\mathsf{state} \wedge c_1 z))) \wedge$
    $(\mathsf{state} \supset \overline{\mathbf{X}}_i (c_2 \vee c_1 \vee (\mathsf{state} \wedge c_1 z)))$
    ensures that in a configuration, the propositions c_1 coding the value of the first counter occur before the state and the propositions c_2 encoding the value of the second counter occur after the state.

* $\mathsf{z\text{-}consis}_i \stackrel{\text{def}}{=} (\mathsf{state} \supset (c_1z \equiv \overline{\mathbf{Y}}_i\neg c_1) \wedge (c_2z \equiv \overline{\mathbf{X}}_i\neg c_2))$
ensures that the value of counter 1 (2) is zero iff there are no events labelled with c_1 (c_2) in the past (future) of an event labelled by a state.
* $\mathsf{inc\text{-}consis}_i \stackrel{\text{def}}{=} (((\mathsf{state} \wedge inc_1) \supset \mathbf{Y}_i(c_1 \wedge \neg r_i^{i^{co}})) \wedge ((\mathsf{state} \wedge inc_2) \supset \mathbf{X}_i(c_2 \wedge \neg r_i^{i^{co}})))$
ensures that a new event labelled by c_j (which is not a copy of some event labelled by c_j) precedes/succeeds the event corresponding to a state whenever the value of counter j is incremented for $j = 1, 2$.
* $\mathsf{dec\text{-}consis}_i \stackrel{\text{def}}{=} (((\mathsf{state} \wedge dec_1) \supset \mathbf{Y}_i(c_1 \wedge \overline{\mathbf{X}}_{i^{co}}\mathsf{False})) \wedge ((\mathsf{state} \wedge dec_2) \supset \mathbf{X}_i(c_2 \wedge \overline{\mathbf{X}}_{i^{co}}\mathsf{False})))$
ensures that the counter proposition c_j at an event of agent i is not copied to agent i^{co} whenever counter j is decremented for $j = 1, 2$.
* $\mathsf{z\text{-}dec\text{-}consis}_i \stackrel{\text{def}}{=} (c_1z \supset \neg dec_1) \wedge (c_2z \supset \neg dec_2)$ ensures that decrement is not possible if the counter value is zero.
* $\mathsf{state\text{-}inc\text{-}dec\text{-}consis}_i \stackrel{\text{def}}{=} \mathsf{state} \supset (\mathsf{dec\text{-}options} \wedge (q_f \vee \mathbf{X}_{i^{co}}\mathsf{inc\text{-}options}))$ where
$\mathsf{dec\text{-}options} \stackrel{\text{def}}{=} (dec_1 \wedge dec_2) \vee (dec_1 \wedge \neg dec_2) \vee (\neg dec_1 \wedge dec_2) \vee (\neg dec_1 \wedge \neg dec_2)$ and
$\mathsf{inc\text{-}options} \stackrel{\text{def}}{=} (inc_1 \wedge inc_2) \vee (inc_1 \wedge \neg inc_2) \vee (\neg inc_1 \wedge inc_2) \vee (\neg inc_1 \wedge \neg inc_2)$
ensures that the counters are either decremented or incremented or left unchanged at every transition.

– $\mathsf{across\text{-}consis}_i \stackrel{\text{def}}{=} \mathsf{state} \supset ((q_f \vee \mathbf{X}_{i^{co}}\mathsf{state}) \wedge (ic \vee r_i^{i^{co}})) \wedge$
$c_1 \supset ((\mathbf{X}_{i^{co}}c_1 \vee \mathbf{X}_i(\mathsf{state} \wedge dec_1)) \wedge (r_i^{i^{co}} \vee \mathbf{X}_i(\mathsf{state} \wedge inc_1))) \wedge$
$c_2 \supset ((\mathbf{X}_{i^{co}}c_2 \vee \mathbf{Y}_i(\mathsf{state} \wedge dec_2)) \wedge (r_i^{i^{co}} \vee \mathbf{Y}_i(\mathsf{state} \wedge inc_2)))$
ensures that the counter values and the states are updated appropriately from one agent to the other, implementing a transition of M.

We can show that α_M is satisfiable iff a final configuration of M is reachable, which implies that (finite) satisfiability of LD_1 is undecidable. The details of the proof are given in the Appendix. □

The proof of Theorem 2.2 follows from the above proof as the formulas of LD_1 are also formulas of LD_0 with the occurrences of the formula $\neg r_j^i$ replaced by the formula $\tau_j \wedge \overline{\mathbf{Y}}_i\mathsf{False}$ and the formula r_j^i by $\tau_j \wedge \mathbf{Y}_i\mathsf{True}$ for $i, j \in \{1, 2\}$.

Theorem 2.4 *The (finite) satisfiability problem for LD_2 is undecidable.*

Proof: The proof proceeds similarly as above, defining a formula α_M of LD_2 for a given 2-counter machine M. We have to re-write some of the formulas used in the proof of Theorem 2.3 using the global $\mathbf{Y}$ modality and special send propositions instead of the global $\mathbf{X}$ modality. In addition, we create a new event preceding the event corresponding to the initial configuration. This helps us to re-write some of formulas in a way that is symmetric to the corresponding ones in LD_1. The formulas which are re-written are listed below.

- $\mathsf{init} \stackrel{\text{def}}{=} \tau_1 \wedge ic \wedge q_0 \wedge c_1z \wedge c_2z \wedge \overline{\mathbf{Y}}_1\mathsf{False} \wedge \overline{\mathbf{X}}_1\mathbf{G}_1\neg ic \wedge \mathbf{Y}_2((q_0 \wedge c_1z \wedge c_2z) \wedge \overline{\mathbf{Y}}_2\mathsf{False} \wedge \overline{\mathbf{X}}_2\mathbf{G}_2\neg ic)$.
- $\mathsf{fin} \stackrel{\text{def}}{=} \mathbf{F}_1\mathsf{fin}_1 \vee \mathbf{Y}_2\mathbf{F}_2\mathsf{fin}_2$ where $\mathsf{fin}_1 \stackrel{\text{def}}{=} q_f \wedge \mathbf{G}_1\neg s_1^2$ and $\mathsf{fin}_2 \stackrel{\text{def}}{=} q_f \wedge \mathbf{G}_2\neg s_2^1$.
- $\mathsf{inv} \stackrel{\text{def}}{=} \mathbf{G}_1\mathsf{inv}_1 \wedge \overline{\mathbf{Y}}_2\mathbf{G}_2\mathsf{inv}_2$.
- For $t = (q, x, y, q', x', y') \in T$ and $q \neq q_f$, we define

$$\theta_t \stackrel{\text{def}}{=} (q' \wedge \neg ic) \wedge (\mathbf{Y}_{i^{co}}(\gamma_x^1 \wedge \gamma_y^2 \wedge q) \wedge \xi_{x',i}^1 \wedge \xi_{y',i}^2))$$

where

$$\gamma_0^j = c_jz \qquad \gamma_1^j = \neg c_jz \qquad \xi_{0,i}^j = \neg inc_j \wedge \mathbf{Y}_{i^{co}}\neg dec_j$$
$$\xi_{1,i}^j = inc_j \wedge \mathbf{Y}_{i^{co}}\neg dec_j \qquad \xi_{-1,i}^j = \neg inc_j \wedge \mathbf{Y}_{i^{co}}dec_j$$

Now, $\mathsf{trans}_i \stackrel{\mathrm{def}}{=} (q' \wedge \neg ic) \supset \bigvee_{t=(q,x,y,q',x',y')\in\delta} \theta_t \wedge \bigwedge_{t\notin\delta} \neg\theta_t$.

- $\mathsf{dec\text{-}consis}_i \stackrel{\mathrm{def}}{=} (((\mathsf{state} \wedge dec_1) \supset \mathbf{Y}_i(c_1 \wedge \neg s_i^{i^{co}})) \wedge ((\mathsf{state} \wedge dec_2) \supset \mathbf{X}_i(c_2 \wedge \neg s_i^{i^{co}})))$.
- $\mathsf{inc\text{-}consis}_i \stackrel{\mathrm{def}}{=} (((\mathsf{state} \wedge inc_1) \supset \mathbf{Y}_i(c_1 \wedge \overline{\mathbf{Y}}_{i^{co}}\mathsf{False})) \wedge ((\mathsf{state} \wedge inc_2) \supset \mathbf{X}_i(c_2 \wedge \overline{\mathbf{Y}}_{i^{co}}\mathsf{False})))$
- $\mathsf{across\text{-}consis}_i \stackrel{\mathrm{def}}{=} (\mathsf{state} \supset (ic \vee \mathbf{Y}_{i^{co}}\mathsf{state}) \wedge (q_f \vee s_i^{i^{co}})) \wedge$
  $c_1 \supset ((s_i^{i^{co}} \vee \mathbf{X}_i(\mathsf{state} \wedge dec_1)) \wedge (\mathbf{Y}_{i^{co}}c_1 \vee \mathbf{X}_i(\mathsf{state} \wedge inc_1))) \wedge$
  $c_2 \supset ((s_i^{i^{co}} \vee \mathbf{Y}_i(\mathsf{state} \wedge dec_2)) \wedge (\mathbf{Y}_{i^{co}}c_2 \vee \mathbf{Y}_i(\mathsf{state} \wedge inc_2)))$.
- $\mathsf{state\text{-}inc\text{-}dec\text{-}consis}_i \stackrel{\mathrm{def}}{=} \mathsf{state} \supset ((ic \vee \mathbf{Y}_{i^{co}}\mathsf{dec\text{-}options}) \wedge \mathsf{inc\text{-}options})$ where dec-options and inc-options are as defined for LD_1.

It is easy to see that relating models of α_M and halting runs of M can be carried out as in the undecidability proof for LD_1; the details are provided in the appendix. □

In [11], we had conjectured that m-LTL enriched with special send propositions is decidable; this theorem shows that the conjecture is false. Though the semantics of m-LTL is slightly different, the argument is very similar.

2.4 Bounded models

On the other hand, it is easy to see that these logics are decidable over bounded-size models.

Theorem 2.5 *The satisfiability of a formula of length m in LD_0 over b-bounded diagrams is decidable in time $m \cdot 2^{O(b^2)}$.*

Fix $b \in \mathbb{N}$ and consider the set of Lamport diagrams whose number of events is at most b. This is a finite set, call it $\mathcal{L}_b$. Given a formula $\alpha \in LD_0$, the problem is to check if there exists a model for α among the diagrams in $\mathcal{L}_b$.

Consider a model $M = (D, V_E)$ where $D \in \mathcal{L}_b$ and a formula $\alpha \in LD_0$. To check whether $M, e \models \alpha$ for some event e of M, we can use a simple labelling algorithm which labels each event with a set of sub-formulas of α such that it terminates, and on termination, for every sub-formula β and every event e' in M, we have $M, e' \models \beta$ iff $\beta \in \mathsf{label}(e')$. The algorithm enumerates sub-formulas of α as $\{\alpha_1, \ldots \alpha_m\}$ such that if α_i is a sub-formula of α_j then, $i \leq j$. Thus, when the algorithm considers a formula $Z\alpha$ at an event e where Z is a modality ($\mathbf{X}$, $\mathbf{Y}$, $\mathbf{F}$ or $\mathbf{P}$), it has already decided whether $\alpha \in \mathsf{label}(e')$ or not, for every e'. It is easy to see that such an algorithm needs time at most $O(k^2 \cdot m)$, where k is the number of events in M, and the length of α is m.

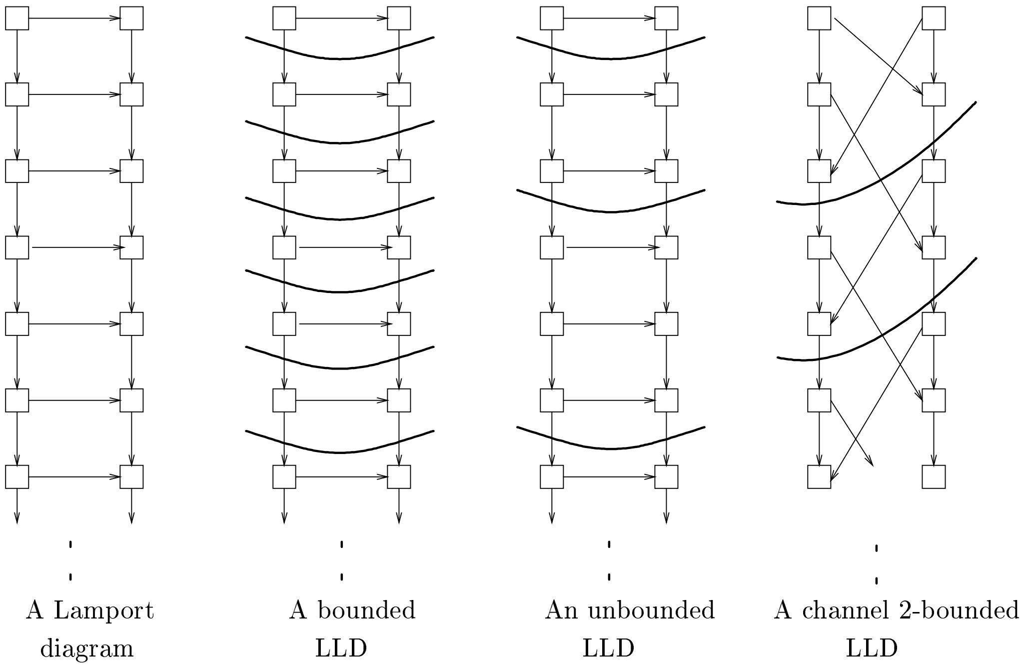
Thus, for checking satisfiability of layer formulas against the finite set $\mathcal{L}_b$ of Lamport diagrams, we can consider the diagrams in $\mathcal{L}_b$ one at a time and check if α holds at an event. From the argument above, we know that the time taken to check the satisfiability of a formula α of length m against a model of size $\leq b$ is at most $O(m \cdot b^2)$. $\mathcal{L}_b$ consists of all Lamport diagrams of size $\leq b$, and the number of Lamport diagrams of size k is at most 2^{k^2}. Therefore, $|\mathcal{L}_b| \leq 2^{n^2} + 2^{(n+1)^2} + \ldots + 2^{b^2}$ (since $b \geq n$) $\leq b \cdot 2^{b^2}$. Hence the total time taken to check the satisfiability of a formula α against all the models in $\mathcal{L}_b$ is $\leq O(m \cdot 2^{b^2})$.

Since LD_1 and LD_2 are translatable into LD_0, the corresponding theorem for them results as a corollary of the theorem above (though checking that a specific model satisfies a given formula of these logics can be done more efficiently than indicated above).

3 Layered Lamport diagrams

As explained in the introduction, we now turn our attention to restricted classes of models. For this, we define a notion of layering on Lamport diagrams.

Figure 3: A layered Lamport diagram



Definition 3.1 *A* **layered Lamport diagram** *is a tuple $D = (E, \leq, \phi, \lambda)$ where $(E, \leq, \phi)$ is a Lamport diagram and $\lambda : E \to \mathbb{N}$ is a* layering function *which satisfies the following conditions:*

- *for all $e \in E$, if $\lambda(e) = k$ then, for all $i \in [n]$, there exists $e' \in E_i$ such that $\lambda(e') = k$.*
- *for $e, e' \in E$, $e \leq e'$ implies $\lambda(e) \leq_{\mathbb{N}} \lambda(e')$.*
- *for every k, $\lambda^{-1}(k)$ is finite.*

Thus a layer is a finite set of events that includes at least one event of each agent, and the layering respects the causality relation.

Given a layered Lamport diagram $D = (E, \leq, \phi, \lambda)$, $\lambda(E)$ is an infinite set and can be denoted by an increasing (infinite) sequence of natural numbers. More precisely, let ν_D denote the sequence of natural numbers $k_1, k_2, \ldots$ such that $\lambda(E) = \{k_1, k_2, \ldots\}$ and $k_1 <_{\mathbb{N}} k_2 <_{\mathbb{N}} \ldots$. For $k \in \lambda(E)$, $\lambda^{-1}(k)$ induces a (finite) Lamport diagram which we call a layer of D and denote by D_k. Notice that a Lamport diagram which occurs as a layer has the additional property of having at least one event of each agent.

We now consider natural conditions that can be imposed on layers of an LLD.

Definition 3.2 *Consider a layered Lamport diagram $D = (E, \leq, \phi, \lambda)$.*

1. *D is said to be* **communication closed** *if for every $e, e' \in E$ such that $e <_c e'$, $\lambda(e) = \lambda(e')$.*
2. *Let $b > 0$. D is said to be b-***bounded***, if for all $k \in \lambda(E)$, $|\lambda^{-1}(k)| \leq b$. D is said to be* **bounded** *if there exists $b \in \mathbb{N}$ such that D is b-bounded.*

Thus, a communication closed LLD is one in which a send event and its corresponding receive event belong to the same layer. When an LLD is not communication closed, we call the layering *interleaved.* Figure 3 shows a Lamport diagram and LLDs with bounded, unbounded and interleaved layerings. Note that when D is b-bounded, $b \geq n$, since every layer is assumed to contain at least one event per agent. The layering in the second diagram in the figure is bounded as every layer has exactly 2 events but, the third diagram has two events in the first layer, four events in the second layer, six events in the third layer and so on and hence, is an unbounded layering. The fourth diagram has an interleaved layering, where a message sent in one layer is received in the next. Such diagrams are of importance, since there are many interesting message passing systems (the alternating bit protocol, for instance) which cannot be presented as a language of communication closed LLDs.

In the context of LLDs with interleaved layering, there is a different source of unboundedness, even if there is a uniform bound on every layer. Let $e_1, e \in E_i$ such that $e_1 \leq e$. Call e_1 a pending send to j at e if there exists $e_2 \in E_j$ such that $e_1 <_c e_2$ and $\lambda(e) < \lambda(e_2)$. Now, for $e \in E_i$, and $j \neq i$, there need not be any uniform bound on the number of events $e' \leq e$ such that e' is a pending send to j. We can consider these pending sends to constitute the state of the communication *channel* from i to j when the agent i is in the local state when e has just occurred. It is then reasonable to consider systems where the capacity of every such channel is uniformly bounded.

Definition 3.3 *Consider a layered Lamport diagram $D = (E, \leq, \phi, \lambda)$. Let ν' be a finite prefix of ν_D ending in k_m; let $F_{\nu'} = \cup_{k_l \leq k_m} \lambda^{-1}(k_l)$; for $i, j \in [n]$, $i \neq j$, define $\delta_{\nu'}(i,j) = |\{e \in F_{\nu'} \cap E_i \mid$ for some $e' \in (E_j - F_{\nu'}), e <_c e'\}|$. We say that D is* **channel b-bounded**, *if D is b-bounded and for all prefixes ν' of ν_D and $i, j \in [n]$, $\delta_{\nu'}(i,j) \leq b$.*

Thus, in a channel b-bounded LLD, for every pair of agents i, j such that $i \neq j$, at any layer, the number of send-events from i to j until this layer, for which corresponding receive-events by j have not been included yet, can be at most b.

A collection $\mathcal{L}$ of layered Lamport diagrams is said to be communication closed if every Lamport diagram in $\mathcal{L}$ is communication closed. $\mathcal{L}$ is said to be channel bounded if there exists $b \in \mathbb{N}$ such that for every $D \in \mathcal{L}$, D is channel b-bounded.

Let $\mathcal{B}$ denote the class of all bounded LLDs, $\mathcal{S}_b$ the class of channel-b-bounded LLDs and $\mathcal{C}_b$ the class of LLDs which are both b-bounded and communication closed. These classes will be of importance when we study decidability issues below.

Reconstructing LLDs

When D is communication closed, the sequence of layers $D_{k_1}, D_{k_2}, \ldots$, where $\nu_D = k_1, k_2 \ldots$, completely specifies the diagram D, in the sense that D can be reconstructed as a countable concatenation of this sequence of diagrams. This can be important for system modelling: typical communication patterns are modelled as layers, and constitute a finite alphabet of patterns. A diagram then is simply an infinite word over this finite alphabet. Just as we speak of the sequence of finite diagrams associated with D, we can conversely speak of the diagram D associated with a given countable sequence of finite diagrams. We make this precise below.

Consider Lamport diagrams $D_1 = (E_1, \leq_1, \phi_1)$ and $D_2 = (E_2, \leq_2, \phi_2)$ with $E_1 \cap E_2 = \emptyset$. The **concatenation** of D_1 and D_2 is a Lamport diagram D denoted by $D = D_1 \bullet D_2$ which is defined as follows:

$D = (E, \leq, \phi)$ where

- $E = E_1 \cup E_2$.
- $\phi = \begin{cases} \phi_1(e) & e \in E_1 \\ \phi_2(e) & e \in E_2 \end{cases}$
- $\leq = (\leq_1 \cup \leq_2 \cup \cup_{i=1}^{n} \{(e_1, e_2) \mid e_1 \in E_1 \cap E_i \text{ and } e_2 \in E_2 \cap E_i\})^*$.

Note that the concatenation operation $\bullet$ is associative. Now consider a communication closed LLD $D = (E, \leq, \phi, \lambda)$ with $\nu_D = k_1, k_2, \ldots$. Then D can be written as $D = D_{k_1} \bullet D_{k_2} \bullet \ldots$ where the concatenation is as defined above and the layering function λ is given by $\lambda(e) = k_i$ for all events e of D_{k_i}.

However, in the case of diagrams which are not communication closed, a layer D_k is typically *incomplete*. 'Communication edges' across layers in the Hasse diagram of D are missing in the diagram given by ν_D. Reconstructing D from ν_D for general diagrams requires a good deal of work. The layers are now *fragments*, which need to be tiled together, matching sends and receives appropriately to make up the original diagram. Moreover, the same event may be a pending send for several agents, and so also, a delayed receive may be from several agents. Therefore, the fragment must carry this information. However, it is easy to see that if e is a pending send for i to j in a layer, every later send event for i to j in that layer must be a pending send. Similarly, when e is a delayed receive for i from j in a layer, every preceding receive for i from j in that layer must be a delayed receive. These observations lead to the following way of reconstructing LLDs which have interleaved layering.

Define a **labelled fragment** to be a tuple $F = (E, \leq, \phi, A, \eta)$ where $(E, \leq, \phi)$ is a Lamport diagram, A is some (abstract) alphabet of labels, and η is a map that demarcates "hanging" sends and receives as follows. Let $T = (\{r, s\} \times [n] \times A)$. Elements of T represent 'tags'; for instance, the tag (s, i, a) denotes a hanging send to i labelled a, and similarly (r, j, a') is a hanging receive from j labelled a'. $\eta : E \longrightarrow (A \times 2^T)$ associates with each event, a set of tags, subject to the following condition: if $\eta(e) = (a, X)$, then $(s, i, a') \in X$ implies that $e \notin E_i$ and for every $e' \in E$ such that $\phi(e) = \phi(e')$ and $e \leq e'$, there does not exist $e'' \in E_i$ such that $e' <_c e''$, and $(r, j, a') \in X$ implies that $e \notin E_j$ and for every $e' \in E$ such that $\phi(e) = \phi(e')$ and $e' \leq e$, there does not exist $e'' \in E_j$ such that $e'' <_c e$. Also, $(s, i, a_1) \in X$ and $(s, i, a_2) \in X$ implies $a_1 = a_2$ and similarly, $(r, j, a_1) \in X$ and $(r, j, a_2) \in X$ implies $a_1 = a_2$.

With this preparation, we can see the relationship between D and ν_D. Given D, there exists an alphabet A such that the sequence $\nu_D = k_1, k_2 \ldots$ associates a sequence of fragments $F_{k_1}, F_{k_2}, \ldots$, such that: for any k_j, if there exists $e \in F_{k_j} \cap E_j$ such that $\eta(e) = (a, X)$ and $(s, i, a') \in X$, then there exists $e' \in F_{k_\ell} \cap E_i$ for some $k_\ell > k_j$ such that $e <_c e'$ and $\eta(e') = (a', Y)$ and $(r, j, a) \in Y$. A similar condition holds the other way as well: for k_j, e as above, if $(r, i, a') \in X$, then there exists $e' \in F_{k_\ell} \cap E_i$ for some $k_\ell < k_j$ such that $e' <_c e$ and $\eta(e') = (a', Y)$ and $(s, j, a) \in Y$. (Strictly speaking, labels are not necessary for describing this correspondence, but we need them later on.) In the case of b-bounded diagrams, a finite alphabet suffices above. We say that the sequence of fragments $F_{k_1}, F_{k_2}, \ldots$ is *matched in D*.

Clearly, every countable sequence of finite labelled fragments does not give rise to an LLD, since pending sends and delayed receives must be matched up appropriately.

In the definition of layers, we have assumed that every layer includes at least one event of every agent. We can define *weak layering* maps which do not satisfy this condition. Weakly layered and communication closed diagrams were introduced by [5] and extensively developed by [13] and others. For ease of proofs, we have preferred to study layered diagrams rather than weakly layered ones; for instance, a layering with exactly one event in every layer is a weak layering. Similarly, one where every layer is either an internal event or a pair of events representing a communication, is a layering that is weak and communication closed. As we will see, the technical results obtained do not depend crucially on whether we use weak layerings or not.

Example

In this section we present an example of a distributed system corresponding to an Automatic Teller Machine (ATM). The system has three agents—**User, Bank** and the **ATM**. The ATM provides options for the user to check the balance in his/her account and to withdraw cash after validating the balance. For example, Lamport diagrams representing the possible computations for withdrawing cash are depicted in Figure 4. These diagrams focus on the sequence of messages exchanged and not on the local actions of a particular agent. Note that the message labels (Withdraw, Req-amt etc.) are actually propositions labelling the corresponding send and receive events. For example, the proposition Withdraw holds at the initial event of the agent **User** (a send event) and at the initial event of the agent **ATM** (the corresponding receive event). The propositions are presented as labels of messages for the sake of clarity.

The computations given in Figure 4 can be layered as shown—the initial layer containing the scenarios corresponding to validating the bank balance is repeated in both the computations. The layering presented in the figure helps us to come up with a concise representation of the various computations of the ATM. These are presented in Figure 5. The computations are layered in such a way that the collection of all the computations forms a bounded and communication closed collection of LLDs. We obtain the various computations by concatenating the Lamport diagrams labelling the nodes of various paths starting from the node labelled by "Get Option".

Figure 4: Lamport diagrams representing a computation of the ATM

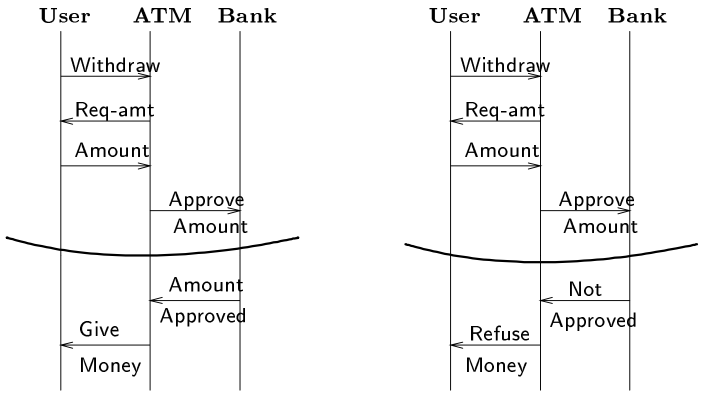
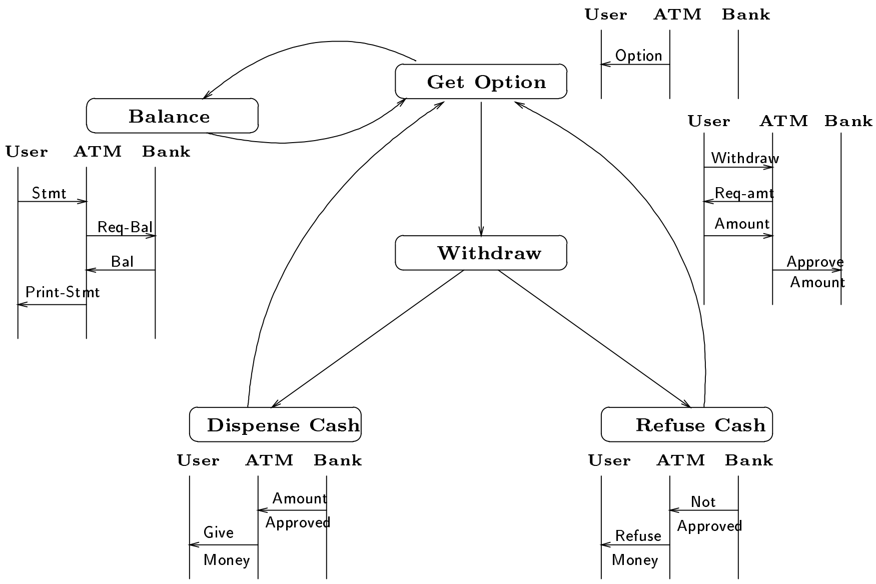


Figure 5: A system representing an ATM scenario



4 The temporal Logic

The advantage of layering Lamport diagrams is that every such diagram can then be given as a sequence of layers, each layer itself being given by a finite diagram. This offers a style of layered specification for message passing systems. If we conceive of a system as working with a fixed finite set of communication patterns between agents, then each such pattern would be given by a finite diagram, and the system itself would be described by choosing patterns dynamically and composing them. This suggests that the specification of a layer itself may be as in LD_0, but sequences of layers be given by a linear time temporal logic that works with layer names. We present such a logic below.

We fix a finite set of *layer names*, Γ. Formulas of our logic, which we call λ-**LTL**, are defined using a two-level syntax. We first define *layer formulas* which will be interpreted at a particular layer of an LLD . We use *temporal formulas* to talk about sequences of layers that make up an LLD.

The syntax of layer formulas is given as in LD_0.

$$\Phi_l ::= p \ \in \ P \mid \tau_i \mid \neg\, \alpha \mid \alpha_1 \ \vee \ \alpha_2 \mid \mathbf{X}_j \ \alpha \mid \mathbf{Y}_j \ \alpha \mid \mathbf{F} \ \alpha \mid \mathbf{P} \ \alpha$$

where $i, j \in [n]$. Note that the $\mathbf{X}$ and $\mathbf{Y}$ modalities are global indexed modalities. The formula $\mathbf{X}_j\alpha$ interpreted at an event e of agent i asserts that there exists an immediate successor of e in agent j at which α holds. Similarly, the formula $\mathbf{Y}_j\alpha$ interpreted at an event e of agent i asserts that there exists an immediate predecessor of e in agent j at which α holds.

Temporal formulas are given by the following syntax:

$$\Psi ::= \alpha @ i, \ \alpha \ \in \ \Phi_l \ , \ i \ \in \ [n] \mid a, \ a \ \in \ \Gamma \mid \neg\, \varphi \mid \varphi_1 \ \vee \ \varphi_2 \mid \bigcirc\, \varphi \mid \varphi_1 \ \mathbf{U} \ \varphi_2$$

This is the same as the standard propositional temporal logic of linear time, but built up from layer formulas and layer propositions. The propositional connectives $(\wedge, \supset, \equiv)$ and the derived temporal modalities $(\Box, \Diamond)$ are defined as usual.

The formulas are interpreted on *countable* layered Lamport diagrams. Formally, models are layered Lamport diagrams equipped with two valuation functions; one maps the events to a subset of propositions and the other maps a particular layer to some layer name from Γ. We will denote models by $M = (D, V_E, V_\lambda)$ where $D = (E, \leq, \phi, \lambda)$ is a countable layered Lamport diagram, $V_E : E \to 2^P$ is such that for all $e \in E$ and $i \in [n]$, $\tau_i \in V_E(e)$ iff $\phi(e) = i$ and $V_\lambda : \lambda(E) \to \Gamma$.

We will first define the semantics of layer formulas, over the layers of a given model. For this, it matters crucially whether the layer is communication closed or not, since the $\mathbf{X}_j$ and $\mathbf{Y}_j$ modalities are constrained to be satisfied within the layer or not, appropriately. For technical convenience, we always interpret $\mathbf{F}$ and $\mathbf{P}$ modalities within layers, and locally for every agent. As we will see later, this suffices for many interesting specifications of message behaviours. Also, it turns out that the classes of LLDs over which this logic is decidable have bounded layers (apart from other properties) and hence $\mathbf{F}$ and $\mathbf{P}$ modalities can be written using a "sequence" of local $\mathbf{X}_j$ and $\mathbf{Y}_j$ modalities respectively.

Let $\alpha \in \Phi_l$ and $e \in E$. We define two notions below: the first notion that α holds at e in M is denoted $M, e \models_l \alpha$ and is defined inductively as done before. The base case for propositions and the case when α is a boolean combination of formulas are clear. The semantics of the various modalities are given below.

- $M, e \models_l \mathbf{X}_j\alpha$ iff there exists $e' \in E_j$ such that $e \lessdot e'$ and $M, e' \models_l \alpha$.
- $M, e \models_l \mathbf{F}\alpha$ iff there exists e' such that $e \leq e'$, $\phi(e) = \phi(e')$, $\lambda(e) = \lambda(e')$ and $M, e' \models_l \alpha$.
- $M, e \models_l \mathbf{Y}_j\alpha$ iff there exists $e' \in E_j$ such that $e' \lessdot e$ and $M, e' \models_l \alpha$.
- $M, e \models_l \mathbf{P}\alpha$ iff there exists e' such that $e' \leq e$, $\phi(e) = \phi(e')$, $\lambda(e) = \lambda(e')$ and $M, e' \models_l \alpha$.

The second notion is defined on communication closed layers. Let M be a model based on a communication closed LLD. The notion that α holds at e in M is denoted $M, e \models_{cl} \alpha$ and is defined inductively as above. The only changes are:

- $M, e \models_{cl} \mathbf{X}_j\alpha$ iff there exists $e' \in E_j$ such that $e \lessdot e'$, $\lambda(e) = \lambda(e')$ and $M, e' \models_{cl} \alpha$.

- $M, e \models_{cl} \mathbf{Y}_j\alpha$ iff there exists $e' \in E_j$ such that $e' \lessdot e$, $\lambda(e) = \lambda(e')$ and $M, e' \models_{cl} \alpha$.

Temporal formulas are interpreted at layers of a layered Lamport diagram. Given a model $M = (D, V_E, V_\lambda)$ and $\varphi \in \Psi$, the notion that φ holds in the layer k of D is denoted $M, k \models \varphi$ and is defined inductively as follows:

- $M, k \models \alpha@i$ iff $M, e \models_l \alpha$ where e is the i-minimum event of D_k.
- $M, k \models a,\ a \in \Gamma$ iff $V_\lambda(k) = a$.
- $M, k \models \neg\varphi$ iff $M, k \not\models \varphi$.
- $M, k \models \varphi \vee \psi$ iff $M, k \models \varphi$ or $M, k \models \psi$.
- $M, k \models \bigcirc\varphi$ iff $M, k' \models \varphi$ where k' is the successor of k in ν_D.
- $M, k \models \varphi\mathbf{U}\psi$ iff there exists $k' \in l(E)$: $k \leq_{\mathbb{N}} k', M, k' \models \psi$ and for all $k'' \in l(E) : k \leq_{\mathbb{N}} k'' <_{\mathbb{N}} k'$: $M, k'' \models \varphi$.

When M is a model based on a communication closed LLD, we define $M, k \models_c \varphi$ exactly as above, except for the base case: $M, k \models_c \alpha@i$ iff $M, e \models_{cl} \alpha$ where e is the i-minimum event of D_k.

For a model M and $\varphi \in \Psi$, we say that $M \models \varphi$ iff $M, k_{min} \models \varphi$ where k_{min} denotes the first element in the sequence ν_D. The notion $M \models_c \varphi$ is defined similarly. φ is *satisfiable* iff there exists a model M such that $M \models \varphi$. φ being C-satisfiable is defined similarly.

We will be interested in other restricted satisfiability notions as well. Say φ is *B-satisfiable* if there exists a model M based on a bounded LLD such that $M \models \varphi$. The notion of S_b-satisfiability is defined similarly using a model based on a channel b-bounded LLD, and C_b-satisfiability using a model based on an LLD that is both communication closed and b-bounded.

Example

We consider the example of the distributed system corresponding to an ATM presented in Section 3. Formulas in λ-**LTL** can specify various properties of the system. The agents are referred to by their names, which are **User, ATM** and **Bank**. The set of propositions is given by { Option, Withdraw, Req-amt, Amount, Approve-Amount, Amount-Approved, Give-Money, Not-Approved, Refuse-Money, Stmt, Req-Bal, Bal, Print-Stmt }.

The following are some properties specified in λ-LTL.

- ATM accepts options to either check balance or to withdraw cash.

  $\Box(((\mathsf{Option} \wedge \tau_{\mathbf{ATM}})@\mathbf{ATM}) \supset$
  $\bigcirc((\tau_{\mathbf{User}} \wedge \mathsf{stmt})@\mathbf{User} \vee (\tau_{\mathbf{User}} \wedge \mathsf{Withdraw})@\mathbf{User}))$

- Every request for withdrawal from the user is approved or rejected by the ATM depending on whether the bank approves or rejects the amount.

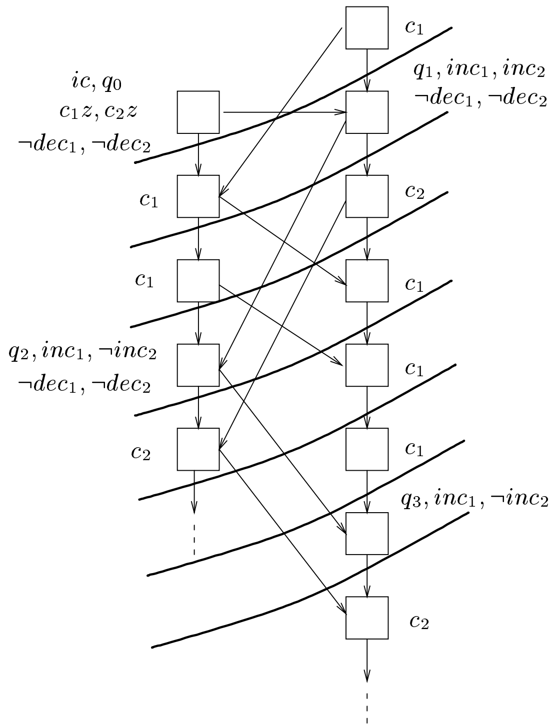
  $\Box[(\tau_{\mathbf{User}} \wedge \mathsf{Withdraw} \wedge \mathbf{X}_{\mathbf{ATM}}(\tau_{\mathbf{ATM}} \wedge \mathsf{Withdraw})@\mathbf{User}) \supset$
  $\bigcirc((\tau_{\mathbf{Bank}} \wedge \mathsf{Amount} - \mathsf{Approved} \wedge \mathbf{X}_{\mathbf{ATM}}(\tau_{\mathbf{ATM}} \wedge \mathsf{Amount} - \mathsf{Approved} \wedge \mathbf{X}_{\mathbf{ATM}}\mathsf{Give} - \mathsf{Money})@\mathsf{Bank}) \vee$
  $(\tau_{\mathbf{Bank}} \wedge \mathsf{Not} - \mathsf{Approved} \wedge \mathbf{X}_{\mathbf{ATM}}(\tau_{\mathbf{ATM}} \wedge \mathsf{Not} - \mathsf{Approved} \wedge \mathbf{X}_{\mathbf{ATM}}\mathsf{Refuse} - \mathsf{Money})@\mathsf{Bank}))]$

Undecidable problems

We now show that B-satisfiability and C-satisfiability are undecidable.

Theorem 4.1 *B-satisfiability is undecidable.*

Figure 6: LLD representing a run of M



The result is proved in the same way as we did in the previous section. Of course, the diagram constructed there was a finite diagram, whereas we need countable diagrams as models. This is easily achieved by appending an infinite sequence of dummy local events to each process.

To show that B-satisfiability is undecidable, consider the layering which picks exactly one event from each agent for a layer. Clearly this is a bounded layering and it is not communication closed. Also, such a layering is not always channel bounded. For example, consider the Lamport diagram representing a run of the 2-counter machine M (as defined in Section 2) where counter 1 is incremented at every transition. This implies that a new event labelled by c_1 is added immediately before every event labelling the resulting state of a transition. For example, a prefix of the LLD corresponding to the run $\rho = (q_0, 0, 0) \rightarrow (q_1, 1, 1) \rightarrow (q_2, 2, 1) \rightarrow (q_3, 3, 1) \ldots$ is depicted in Figure 6. Consider the prefixes of this LLD ending at events labelled by states q_0 and q_2. There is only one pending send (from agent 1 to agent 2) at the prefix ending at q_0-labelled event whereas there are two pending sends (from agent 1 to agent 2) at the prefix ending at q_2-labelled event. Using the fact that we consider layers having exactly one event per agent, we can argue that the number of pending sends from agent 1 to agent 2 will grow unboundedly as we consider successive layers ending at state-labelled events. That is, such an LLD is not channel b-bounded for any $b \in \mathbb{N}$.

We can now show that such a layering as well as the diagram itself can be described by a formula φ_M of the logic, proceeding along the same lines as before. The various formulas comprising φ_M are basically the same as the one used in the proof of Theorem 2.3. We replace the $\mathbf{G}$ and $\mathbf{F}$ modalities by $\Box$ and $\Diamond$ appropriately. In addition, we add two formulas to describe the layering and to add dummy events at the end. The details are given in the Appendix.

Note: Since we get 2-agent Lamport diagrams in the coding and the layering is such that it includes exactly one event per agent, the above result actually shows that the problem of checking if a given formula has a model based on an n-bounded LLD (where n is the number of agents) is undecidable.

Theorem 4.2 *C-satisfiability is undecidable.*

The result is again proved in the same way as we did in the previous section. Dummy local events are appended to the finite diagram which represents a run of M that ends in a final configuration. Now consider the part of the diagram representing a run of M that ends in a final configuration as one layer, and each subsequent layer as containing exactly one event (a local event, without any communications) per agent. This is a communication closed (but unbounded) layering, and this can be specified using a formula as well. Hence C-satisfiability is also undecidable. The details are given in the Appendix.

5 Decidability

The decision procedures are automata theoretic in the sense that they rely on associating a nondeterministic Büchi automaton with every formula such that the language accepted by the automaton is exactly the set of models of the formula. The construction of such an automaton is more straightforward for C_b-satisfiability than for S_b-satisfiability.

5.1 Diagram automata

Since models are b-bounded layered Lamport diagrams for C_b-satisfiability, the automata defined here run on such diagrams, with transitions labelled by b-bounded layers. Let $\mathcal{LC}_b$ denote the set of b-bounded communication closed layers; it is clearly a finite set upto isomorphism.

A **Diagram automaton** is given by a tuple $\mathcal{A} = (Q, \mathcal{D}, \rightarrow, I, G)$ where Q is a finite set of states, $\mathcal{D} \subseteq \mathcal{LC}_b$, $I \subseteq Q$ is the set of initial states, $G \subseteq Q$ is the set of good states and $\rightarrow \subseteq (Q \times \mathcal{D} \times Q)$ is the transition relation. That is, a diagram automaton is just a Büchi automaton whose alphabet is a finite set of b-bounded Lamport diagrams which may occur as layers of a communication-closed LLD.

Consider a sequence of states of $\mathcal{A}$, $\rho = q_0, q_1, \ldots$ such that $q_0 \in I$ and $q_i \xrightarrow{D_i} q_{i+1}$ for all $i \geq 0$. Let D be the b-bounded communication closed LLD such that $D_0, D_1, \ldots$ is the sequence of communication closed diagrams defined by ν_D. We say that ρ is a run of $\mathcal{A}$ on D. ρ is said to be accepting if $inf(\rho) \cap G \neq \emptyset$, where $inf(\rho) = \{q \mid q = q_i$ for infinitely many i in $\rho\}$. The language of b-bounded communication closed LLDs accepted by $\mathcal{A}$ is denoted by $L^b_C(\mathcal{A})$ and is defined as $L^b_C(\mathcal{A}) = \{D \mid$ there exists an accepting run of $\mathcal{A}$ on $D\}$.

Lemma 5.1 *Given a diagram automaton $\mathcal{A} = (Q, \mathcal{D}, \rightarrow, I, G)$, the problem of checking if $L^b_C(\mathcal{A}) \neq \emptyset$ is decidable in time $O(|Q|^2)$.*

Proof: We show that $L^b_C(\mathcal{A}) \neq \emptyset$ iff there exists a strongly connected component in the graph of $\mathcal{A}$ such that the following conditions are satisfied:

- The strongly connected component contains a good state of $\mathcal{A}$.
- It is reachable from one of the initial states of $\mathcal{A}$.

Thus, checking if $L^b_C(\mathcal{A}) \neq \emptyset$ would amount to checking for the existence of a strongly connected component with the above properties and this can be done in time $O(|Q|^2)$.

Suppose $L^b_C(\mathcal{A}) \neq \emptyset$. Let $D \in L^b_C(\mathcal{A})$ and ρ be an accepting run of $\mathcal{A}$ on D. Then using the sequence of states in ρ, we can easily define a strongly connected component in the graph of $\mathcal{A}$ containing a good state such that it is reachable from an initial state.

Conversely, suppose the graph of $\mathcal{A}$ has a strongly connected component with the required properties. Considering the path of the strongly connected component from the initial state and unwinding the strongly connected component, we get an infinite sequence $D_0, D_1, \ldots$ of b-bounded Lamport diagrams and a sequence of states $\rho = q_0, q_1, \ldots$ such that $q_0 \in I$ and $q_i \xrightarrow{D_i} q_{i+1}$ for all $i \geq 0$.

Define a layered Lamport diagram $D = D_0 \bullet D_1 \bullet \ldots$ as in Section 3 with $\lambda(e) = i$ for every event e of D_i. Then, the sequence ρ as above is an accepting run of $\mathcal{A}$ on D and hence $D \in L^b_C(\mathcal{A})$. $\square$

5.2 C_b-Satisfiability

In this section we show that C_b-satisfiability is decidable by associating a diagram automaton with every formula.

Theorem 5.2 *Given a temporal formula φ_0 of length m, the satisfiability of φ_0 over b-bounded communication closed diagrams can be checked in time $2^{O(m+b^2)}$.*

This theorem is proved by associating a diagram automaton $\mathcal{A}_0$ (over the alphabet of models based on b-bounded layers) with φ_0 such that $L_C^b(\mathcal{A}_0) = \{M \mid M$ is a model of φ_0 based on a b-bounded communication closed LLD $\}$. Hence, φ_0 is satisfiable iff $L_C^b(\mathcal{A}_0) \neq \emptyset$. Since the emptiness of the language accepted by a diagram automaton is decidable, it follows that checking if φ_0 is satisfiable is also decidable. Note that the states and transitions are the same as what we would use for LTL; we only need to decide the diagram alphabet carefully, and this relies on decidability of the logic LD_0 over b-bounded diagrams (Theorem 2.5).

We start by defining the sub-formula closure $CL(\varphi_0)$ of φ_0. Define $CL'(\varphi_0)$ to be the least set of formulas such that the following conditions are satisfied:

- $\varphi_0 \in CL'(\varphi_0)$.
- $\neg\psi \in CL'(\varphi_0)$ implies $\psi \in CL'(\varphi_0)$.
- $\psi_1 \vee \psi_2 \in CL'(\varphi_0)$ implies $\psi_1, \psi_2 \in CL'(\varphi_0)$.
- $\bigcirc\psi \in CL'(\varphi_0)$ implies $\psi \in CL'(\varphi_0)$.
- $\psi_1 \mathbf{U} \psi_2 \in CL'(\varphi_0)$ implies $\psi_1, \psi_2, \bigcirc(\psi_1 \mathbf{U} \psi_2) \in CL'(\varphi_0)$.

Now define $CL(\varphi_0) \stackrel{\text{def}}{=} CL'(\varphi_0) \cup \{\neg\psi \mid \psi \in CL'(\varphi_0)\} \cup \{a_0, \neg a_0\}$ with $\neg\neg$ treated as identity and $a_0 \in \Gamma$. The presence of a_0 is to ensure that at least one layer name is included, even if none is mentioned in φ_0.

A set $A \subseteq CL(\varphi_0)$ is said to be an atom iff it satisfies the following conditions:

- for every formula $\neg\psi \in CL(\varphi_0)$, $\neg\psi \in A$ iff $\psi \notin A$.
- for every formula $\psi_1 \vee \psi_2 \in CL(\varphi_0)$, $\psi_1 \vee \psi_2 \in A$ iff $\psi_1 \in A$ or $\psi_2 \in A$.
- for every formula $\psi_1 \mathbf{U} \psi_2 \in CL(\varphi_0)$, $\psi_1 \mathbf{U} \psi_2 \in A$ iff either $\psi_2 \in A$ or $\psi_1, \bigcirc(\psi_1 \mathbf{U} \psi_2) \in A$.
- $\mid A \cap \Gamma \mid = 1$.

Let $AT(\varphi_0)$ denote the set of atoms of φ_0. Let P_0 denote the set of propositions that appear in φ_0 and $\Gamma_0 = \Gamma \cap CL(\varphi_0)$. The alphabet of the automaton $\mathcal{A}_0$ is the set $\mathcal{LC}_b^{P_0} = \{M = (D, V_E, a) \mid D = (E, \leq, \phi) \in \mathcal{LC}_b$ and $V_E : E \to 2^{P_0}$ and $a \in \Gamma_0\}$. That is, the alphabet of the diagram automaton is the set of all b-bounded Lamport diagrams along with their valuations. Notice that given a model $M = (D, V_E, V_\lambda)$ where D is a communication closed and b-bounded LLD with $\nu_D = 0, 1, \ldots$ (without loss of generality), we can view M as an infinite word over $\mathcal{LC}_b^{P_0}$ given by $M = M_0 \bullet M_1, \bullet \ldots$ where $M_i = (D_i, V_E \restriction D_i, a_i)$ for all i, D_i is the Lamport diagram corresponding to layer i of D and $V_\lambda(i) = a_i$.

Define a diagram automaton $\mathcal{A}_0$ over $\mathcal{LC}_b^{P_0}$ as follows:

$\mathcal{A}_0 = (Q, \mathcal{LC}_b^{P_0}, \rightarrow, I, F)$ where

1. $Q = AT(\varphi_0) \times U$ where $U = \{\psi_1 \mathbf{U} \psi_2 \mid \psi_1 \mathbf{U} \psi_2 \in CL(\varphi_0)\}$.
2. $I = \{(A, \emptyset) \mid A \in AT(\varphi_0)$ such that $\varphi_0 \in A\}$.
3. $F = \{(A, \emptyset) \mid A \in AT(\varphi_0)\}$, and
4. $\rightarrow$ is given by $(A, u) \xrightarrow{M} (B, v)$ for $A, B \in Q$ and $M = (D, V_E, a)$ iff

   (a) $a \in A$,

(b) $\alpha@i \in A$ iff $M, e \models_{cl} \alpha$ where e is the i-minimum event in D,

(c) for every $\bigcirc\psi \in CL(\varphi_0)$, $\bigcirc\psi \in A$ iff $\psi \in B$, and

(d) if $u \neq \emptyset$ then, $v = \{\psi_1 \mathbf{U} \psi_2 \mid \psi_1 \mathbf{U} \psi_2 \in u$ and $\psi_2 \notin B\}$. Otherwise, $v = \{\psi_1 \mathbf{U} \psi_2 \mid \psi_1 \mathbf{U} \psi_2 \in B$ and $\psi_2 \notin B\}$.

In order to construct the set Q of states, we need to check for the satisfiability of layer formulas present in atoms over b-bounded diagrams. The proof of Theorem 2.5 gives us an algorithm to check this. Since the complexity of checking the satisfiability of the layer formulas against b-bounded diagrams is $m \cdot O(2^{b^2})$, it follows that Q is constructible in time $2^{O(m+b^2)}$.

Lemma 5.3 $L^b_C(\mathcal{A}_0) = \{M \mid M$ *is a model of* φ_0 *based on a* b*-bounded communication closed LLD* $\}$.

Proof: Let $M = M_0 \bullet M_1 \bullet \ldots \in L^b_C(\mathcal{A}_0)$. We show that $M, k_0 \models_c \varphi_0$. Let $\rho = (A_0, u_0), (A_1, u_1), \ldots$ be an accepting run of $\mathcal{A}_0$ on M. We first prove the following claim.

Claim: For all $\psi \in CL(\varphi_0)$, for all $i \in \mathbb{N}$, $M, i \models_c \psi$ iff $\psi \in A_i$.

The claim is proved by induction on the structure of ψ and since the proof follows the standard argument for linear time temporal logic, it is omitted here. From the claim it follows that $M, k_0 \models_c \varphi_0$ as required.

Conversely, to show that every model of φ_0 is in $L^b_C(\mathcal{A}_0)$, let $M = M_0 \bullet M_1 \bullet \ldots$ be such that $M, 0 \models_c \varphi_0$. We have to exhibit an accepting run of $\mathcal{A}_0$ on M.

For $i \geq 0$, define $A_i = \{\psi \in CL(\varphi_0) \mid M, i \models_c \psi\}$. We can show that each A_i as defined above is an atom.

Now, define a sequence $\rho = (A_0, u_0), (A_1, u_1), \ldots$ where the A_i's are as given before. We define u_i by induction on i. $u_0 = \emptyset$ and for $i > 0$, define $u_{i+1} = \{\psi_1 \mathbf{U} \psi_2 \mid \psi_1 \mathbf{U} \psi_2 \in A_{i+1},\ \psi_2 \notin A_{i+1}\}$ if $u_i = \emptyset$ and $\{\psi_1 \mathbf{U} \psi_2 \mid \psi_1 \mathbf{U} \psi_2 \in u_i,\ \psi_2 \notin A_{i+1}\}$ otherwise. We can show that ρ is an accepting run of $\mathcal{A}_0$ on M and hence $M \in L^b_C(\mathcal{A}_0)$. $\square$

5.3 Fragment Automata

While considering S_b-satisfiability, we need to define automata with fragments, rather than diagrams as input. Fix a finite alphabet A and let $\mathcal{LF}_b(A)$ denote the set of b-bounded A-labelled fragments (which occur as layers of a b-bounded LLD); it is again a finite set.

A **Fragment automaton** is given by a tuple $\mathcal{B} = (Q, A, \mathcal{F}, \rightarrow, I, G)$ where A is a finite alphabet, $\mathcal{F} \subseteq \mathcal{LF}_b(A)$, Q is a finite set of states, $\rightarrow \subseteq (Q \times \mathcal{F} \times Q)$, $I \subseteq Q$ is the set of initial states and $G \subseteq Q$ is the set of good states. A run of $\mathcal{B}$ on a channel b-bounded LLD D is a sequence $\rho = q_0, q_1, \ldots$ such that $q_0 \in I$ and $q_i \xrightarrow{F_i} q_{i+1}$ for all $i \geq 0$ where the sequence $F_0, F_1, \ldots$ of fragments is *matched* in D. ρ is said to be accepting if $inf(\rho) \cap G \neq \emptyset$, where $inf(\rho) = \{q \mid q = q_i$ for infinitely many i in $\rho\}$. The language of channel b-bounded LLDs accepted by $\mathcal{B}$ is denoted by $L^b_S(\mathcal{B})$ and is defined as $L^b_S(\mathcal{B}) = \{D \mid$ there exists an accepting run of $\mathcal{B}$ on $D\}$.

Lemma 5.4 *Given a fragment automaton* $\mathcal{B} = (Q, A, \mathcal{F}, \rightarrow, I, G)$*, the problem of checking if* $L^b_S(\mathcal{B}) \neq \emptyset$ *is decidable in time* $O(|Q|^2 \times k^{b^2})$*, where* $k = |A|$.

Emptiness checking is no longer a simple search for a connected component containing a good state. We need to check in addition that every pending 'send' is matched with a 'hanging receive' with labels appropriately matched.

We first build a larger automaton $\mathcal{C}$ based on $\mathcal{B}$ and show that $L^b_S(\mathcal{B}) \neq \emptyset$ iff there exists a strongly connected component in $\mathcal{C}$ containing good states of $\mathcal{C}$ reachable from one of the initial states of $\mathcal{C}$. For this, we first define a queue data structure that will be useful. Let $\overline{A} = \{x \in A^* \mid |x| \leq b\}$. We will be considering maps $\chi : ([n] \times [n]) \rightarrow (\overline{A} \times \overline{A})$. Let Ξ denote the set of all such maps. Let $\widetilde{Q} = (Q \times \Xi)$.

For any fragment $F = (E, \leq, \phi, A, \eta)$ (which corresponds to a layer of some channel b-bounded LLD) and $i, j \in [n]$, define two sequences F^s_{ij} and F^r_{ji} in $\overline{A}$ as follows:

- We know that E_i is a finite set linearly ordered by $\leq$; denote this sequence of events by $e_1 \cdots e_m$. Now $F^s_{ij} = a_{k_1} \cdots a_{k_\ell}$, where for all $p \in \{k_1, \ldots, k_\ell\}$, (s, j, a_p) is a tag in $\eta(e_p)$ and for all $p \in (\{1, \ldots, m\} \setminus \{k_1, \ldots, k_\ell\})$, there is no $a' \in A$ such that (s, j, a') is a tag in $\eta(e_p)$.

  The ordered sequence $e_{k_1}, e_{k_2}, \ldots e_{k_\ell}$ of events of E_i is called as the event sequence corresponding to F^s_{ij} and is denoted by $\mathsf{Ev}(F^s_{ij})$.

- Again, we know that E_j is a finite set linearly ordered by $\leq$; denote this sequence of events by $f_1 \cdots f_{m'}$. Now $F^r_{ji} = b_{k'_1} \cdots b_{k'_{\ell'}}$, where for all $p \in \{k'_1, \ldots, k'_{\ell'}\}$, $\eta(f_p) = (b_p, X)$ with $(r, i, b') \in X$ for some $b' \in A$ and for all $p \in (\{1, \ldots, m'\} \setminus \{k'_1, \ldots, k'_{\ell'}\})$, there is no $b' \in A$ such that (r, i, b') is a tag in $\eta(f_p)$.

  The ordered sequence $f_{k'_1}, f_{k'_2}, \ldots f_{k'_{\ell'}}$ of events of E_j is called as the event sequence corresponding to F^r_{ji} and is denoted by $\mathsf{Ev}(F^r_{ji})$.

Note that both are b-bounded sequences. Below, we will use the following notation: let $x, y \in \overline{A}$; we write $y \sqsubseteq x$ to denote that y is a prefix of x and when $y \sqsubseteq x$, so that $x = y \cdot z$ for some $z \in \overline{A}$, we write $x - y$ to denote the sequence z.

Now define $\mathcal{C} = (\widetilde{Q}, A, \mathcal{F}, \Rightarrow, \widetilde{I}, \widetilde{G})$, where $\widetilde{Q}$ is defined as above, $\widetilde{I} = \{(q, \chi_0) \in \widetilde{Q} \mid q \in I, \chi_0(i, j) = (\epsilon, \epsilon)$ for all $i, j \in [n]\}$, and $\widetilde{G} =< G_{ij} \mid i, j \in [n] >$, where $G_{ij} = \{(q, \chi) \in \widetilde{Q} \mid q \in G, \chi(i, j) = (x, \epsilon), x \in \overline{A}\}$. $\Rightarrow$ is defined as follows.

$(q, \chi) \overset{F}{\Rightarrow} (q', \chi')$ iff:

- $q \xrightarrow{F} q'$ in $\mathcal{B}$.
- Let $\chi(i, j) = (x, y)$ and $\chi'(i, j) = (x', y')$ where $i, j \in [n]$ such that $i \neq j$. Then
  - $F^r_{ji} \sqsubseteq x, x' = (x - F^r_{ji}) \cdot F^s_{ij}$.
  - if $y \neq \epsilon$ then ($F^r_{ji} \sqsubseteq y$ and $y' = y - F^r_{ji}$); otherwise, $y' = x'$.

A transition of $\mathcal{C}$ from (q, χ) to (q', χ') on F first checks if $q \xrightarrow{F} q'$ in the underlying fragment automaton. In addition, for every pair of agents $i, j \in [n]$ such that $i \neq j$, the second set of conditions on x, y and x', y' are used to identify pending sends from i to j with their corresponding hanging receives. The condition that F^r_{ji} is a prefix of x ensures that all the hanging receives of agent j from agent i in F have corresponding sends (so that they can be matched up appropriately). The transition also appends pending sends of F from i to j to the resulting state (so that they may be matched later). Note that y is used to remember older pending sends, and when this becomes null, it is a signal that these have been matched up.

Call a connected component in $\mathcal{C}$ good only if it contains some state in G_{ij} for each pair $i, j \in [n]$ and is reachable from $\widetilde{I}$. We can now show that $L^b_S(\mathcal{B}) \neq \emptyset$ iff $\mathcal{C}$ has a good connected component.

Suppose $L^b_S(\mathcal{B}) \neq \emptyset$. Let D be a channel b-bounded diagram accepted by $\mathcal{B}$. Suppose $F_1, F_2, \ldots$ is the sequence of fragments of D as given by ν_D. Then, there is an accepting run, say, $\rho = q_0 \xrightarrow{F_1} q_1 \xrightarrow{F_2} \ldots$ of $\mathcal{B}$ on D. We show that $\mathcal{C}$ has a good component by inductively constructing a sequence of states of $\mathcal{C}$ below.

Fix $i, j \in [n]$ such that $i \neq j$. Let $\Pi(i, j, k) = \{e \in E_i \mid \lambda(e) \leq k$ and e is a pending send to j at $k\}$ for $k \geq 1$. (Precisely speaking, by pending send at k, we mean a pending send at e' where e' is the i-maximum event such that $\lambda(e') = k$.) Note that $\Pi(i, j, k)$ is linearly ordered by $\leq$, $\mid \Pi(i, j, k) \mid \leq b$. Let this sequence of events be given by $e_1 e_2 \ldots e_m$. For $1 \leq j \leq m$, $\eta_\ell(e_j) = (a'_j, T_j)$ where $e_j \in F_\ell$, $\ell \leq k$ such that the tag $(s, j, a_j) \in T_j$. Denote the sequence $a_1 a_2 \ldots a_m$ by $\Pi(i, j, k) \lceil A$.

The sequences $x_0, x_1 \ldots$ and $y_0, y_1 \ldots$ are defined as follows. Define $x_0 = \epsilon$ and for $k > 0$, $x_k = \Pi(i, j, k) \lceil A$. The sequence y_k is defined inductively: $y_0 = \epsilon$; for $k > 0$, if $y_{k-1} = \epsilon$ then $y_k = x_k$, otherwise let $\ell \leq k$ such that for all ℓ' such that $\ell \leq \ell' \leq k$, $y_{\ell'} \neq \epsilon$; define $y_k = (\Pi(i, j, \ell) \cap \Pi(i, j, k)) \lceil A$.

Define $\rho' = (q_0, \chi_0), (q_1, \chi_1), \ldots$ where $q_0, q_1, \ldots$ is as in ρ and $\chi_k(i, j) = (x_k, y_k)$ is defined using x_k and y_k as above.

We now claim that the sequence $\rho' = (q_0, \chi_0), (q_1, \chi_1), \ldots$ defines a good connected component of $\mathcal{C}$. Clearly, by definition of ρ', $(q_0, \chi_0) \in \widetilde{I}$. To show that $(q_k, \chi_k) \overset{F_{k+1}}{\Rightarrow} (q_{k+1}, \chi_{k+1})$ for all $k \geq 0$, we only need to observe that for all $k > 0$, $i, j \in [n]$ such that $i \neq j$, $\Pi(i, j, k) \lceil A = (\Pi(i, j, k-1) \lceil A - F^r_{ji}(k)) \cdot F^s_{ij}(k)$. To show that there exist infinitely many $k \geq 0$ such that $\chi(i,j) = (x, \epsilon)$ for $i, j \in [n]$ such that $i \neq j$, observe that for every $k \geq 0$, there exists $\ell \geq k$ such that $\Pi(i,j,k) \cap \Pi(i,j,\ell) = \emptyset$, and this is guaranteed by the fact that the sequence of fragments $F_1, F_2, \ldots$ is matched in D.

Conversely, suppose there exists a good connected component of $\mathcal{C}$. We have to show that $L_S^b(\mathcal{B}) \neq \emptyset$. Let $\rho = (q_0, \chi_0) \overset{F_1}{\Rightarrow} (q_1, \chi_1) \overset{F_2}{\Rightarrow} \ldots$ be the sequence obtained by unwinding the good connected component of $\mathcal{C}$. That is $(q_0, \chi_0) \in \widetilde{I}$ and for each pair $i, j \in [n]$ such that $i \neq j$, there exists infinitely many $k \geq 0$ such that $q_k \in G$ and $\chi_k(i,j) = (x, \epsilon)$ for some $x \in \overline{A}$.

We inductively define a Lamport diagram $D = (E, \leq, \phi, \lambda)$ and show that

- D is channel b-bounded and the sequence $F_1, F_2, \ldots$ corresponds to D.
- The run $\rho' = q_0 \overset{F_1}{\to} q_1 \overset{F_2}{\to} \ldots$ (obtained by projecting ρ to the states in Q) is an accepting run of $\mathcal{B}$ on D.

D will be defined by concatenating the fragments F_i. We first define the operation of concatenation below.

Let $F_1 = (E_1, \leq_1, \phi_1, A, \eta_1)$ and $F_2 = (E_2, \leq_2, \phi_2, A, \eta_2)$ be two A-labelled (b-bounded) fragments. The **concatenation** of F_1 and F_2 is a fragment F denoted by $F = F_1 \bullet F_2$ and is defined iff the following conditions are satisfied.

For all $i, j \in [n]$ such that $i \neq j$, we have

- $F^r_{1_{(ji)}} = \epsilon$ and
- $F^r_{2_{(ji)}}$ is a prefix of $F^s_{1_{(ij)}}$.

Then, $F = F_1 \bullet F_2$ is defined as $F = (E, \leq, \phi, A, \eta)$ where

- $E = E_1 \cup E_2$.
- $\phi = \begin{cases} \phi_1(e) & e \in E_1 \\ \phi_2(e) & e \in E_2 \end{cases}$
- $\leq = (\leq_1 \cup \leq_2 \cup \cup_{i=1}^{n} \{(e_1, e_2) \mid e_1 \in E_1 \cap E_i \text{ and } e_2 \in E_2 \cap E_i\}$ $\cup \{(e_1, f_1), \ldots, (e_m, f_m) \mid \mathsf{Ev}(F^r_{2_{(ji)}}) = f_1, \ldots, f_m \text{ and } \mathsf{Ev}(F^s_{1_{(ij)}}) = e_1, \ldots, e_l\})^*$. (Since $F^r_{2_{(ji)}}$ is a prefix of $F^s_{1_{(ij)}}$, we know that $m \leq_{\mathbb{N}} l$).
- For $e \in E_k$, $k = 1, 2$, if $\eta_k(e) = (a, T)$ then $\eta(e) = (a, T')$, where $T' = \emptyset$ if $T = \emptyset$, and $T' = T - (\{t \mid t$ is a receive tag$\} \cup \{t \mid t = (s, j, a)$ where $e \in E_1$, there exists $e' \in E_2$ such that $\phi_2(e') = j \neq \phi_1(e)$, $e \leq e'\})$.

Observe that F is a fragment as well, and hence the operation of concatenation is well-defined. It is not associative, so we follow the convention that $F_1 \bullet F_2 \bullet F_3$ denotes the fragment $((F_1 \bullet F_2) \bullet F_3)$.

Now define $D = ((F_1 \bullet F_2) \bullet \ldots)$ with the layering function λ defined by $\lambda(e) = i$ for every event e of F_i. We claim that $D = (E, \leq, \phi, \lambda)$ is a channel b-bounded layered Lamport diagram. We can easily show that D is a layered Lamport diagram. We now argue that D is channel b-bounded. D is b-bounded as each F_i is b-bounded. Consider a finite prefix ν' of ν_D, say, $\nu' = 1, \ldots, k$. We have to show that $\delta_{\nu'}(i,j) \leq b$ for every $i, j \in [n]$ such that $i \neq j$ (where $\delta_{\nu'}(i,j)$ is as defined in Definition 3.3). Since ρ is a run of $\mathcal{C}$ on $F_1, \ldots F_k$, it follows that the state (q_k, χ_k) is such that $\chi_k(i,j) = (x_k, y_k)$ where $|x_k(i,j)| \leq b$ for all $i, j \in [n]$ such that $i \neq j$. First observe that $\Pi(i,j,k) = (\Pi(i,j,k-1) \setminus \mathsf{Ev}(F^r_{ji}(k)) \cup \mathsf{Ev}(F^s_{ij}(k))$, for $k > 0$ (which follows from the definition of $\bullet$). From this, we can show by induction on k that for $k > 0$, $x_k(i,j) = \Pi(i,j,k)\lceil A$. Since $\delta_{\nu'}(i,j) = \mid \Pi(i,j,k) \mid$, we get $\delta_{\nu'}(i,j) = |x_k(i,j)|$. Hence $\delta_{\nu'}(i,j) \leq b$.

We need to show that $D \in L_S^b(\mathcal{B})$. Since whenever $(q_k, \chi_k) \overset{F_{k+1}}{\Rightarrow} (q_{k+1}, \chi_{k+1})$ in $\mathcal{C}$, we also have $q_k \overset{F_{k+1}}{\to} q_{k+1}$ in $\mathcal{B}$, it suffices to show that the sequence $F_0, F_1, \ldots$ of fragments is matching in D. For this, consider k and $e \in (F_k \cap E_i)$ such that $(s, j, a) \in \eta_k(e)$. We need to show that there exists $\ell > k$ and $e' \in (F_\ell \cap E_j)$ such that $e \lessdot e'$ and $\eta_\ell(e') = (a, T)$. By construction, we can check that $e \in \Pi(i,j,k)$, and since we have already established that D is channel b-bounded, there exists ℓ such that $k < \ell < k + b$ such that $e \notin \Pi(i,j,\ell)$. However, for all $k > 0$, $(\Pi(i,j,k-1) \setminus \Pi(i,j,k))\lceil A = F^r_{ji}(k)$ and hence the required ℓ and e' can be found.

5.4 S_b-satisfiability

We now show that S_b-satisfiability is decidable by associating a fragment automaton with every formula such that every model of the formula corresponds with an accepting run of the associated automaton, and vice versa. Note that this proof cannot simply mimic the proof of C_b-satisfiability, since labelling events of fragments with layer atoms does not give models per se; we get models only when the fragments are 'tiled' suitably, yielding a layered Lamport diagram.

Theorem 5.5 *Given a temporal formula φ_0 of length m, the satisfiability of φ_0 over channel b-bounded diagrams can be checked in time $2^{O(m \cdot b^2)}$.*

We prove this theorem by associating a fragment automaton $\mathcal{A}_0$ (over an alphabet of fragments which occur as layers of channel b-bounded LLDs) with every formula φ_0 such that $L_S^b(\mathcal{A}_0) = \{M \mid M$ is a model of φ_0 based on a channel b-bounded LLD $\}$. Hence, φ_0 is satisfiable iff $L_S^b(\mathcal{A}_0) \neq \emptyset$. Since the emptiness of the language accepted by a fragment automaton is decidable, it follows that checking if φ_0 is satisfiable is also decidable.

The sub-formula closure $CL(\varphi_0)$ and the set of temporal atoms $AT(\varphi_0)$ are defined as in the proof of Theorem 5.2. For every formula of the form $\alpha@i \in CL(\varphi_0)$, the set $SL(\alpha)$ of layer sub-formulas of α is defined below. Let SF_l denote the set of layer sub-formulas associated with φ_0. That is, $SF_l = \cup_{\alpha@i \in CL(\varphi_0)} SL(\alpha)$. The definition of when $A \subseteq SF_l$ is called a layer atom is also given below. We use the notation AT_l to denote the set of layer atoms associated with φ_0.

The sub-formula closure $SL(\alpha_0)$ of a layer formula $\alpha_0 \in \Phi_l$ is defined as follows. Define $SL'(\alpha_0)$ to be the least set of formulas such that the following conditions are satisfied:

- $\alpha_0 \in SL'(\alpha_0)$.
- $\alpha_1 \vee \alpha_2 \in SL'(\alpha_0)$ implies $\alpha_1, \alpha_2 \in SL'(\alpha_0)$.
- if $C\alpha \in SL'(\alpha_0)$ then $\alpha \in SL'(\alpha_0)$, where $C \in \{\neg, \overline{\mathbf{X}}_i, \overline{\mathbf{Y}}_i, \mathbf{F}, \mathbf{G}, \mathbf{P}, \mathbf{H}\}$, $i \in [n]$.
- if $\mathbf{X}_i\alpha \in SL'(\alpha_0)$ then $\overline{\mathbf{X}}_i\alpha \in SL'(\alpha_0)$, $i \in [n]$.
- if $\mathbf{Y}_i\alpha \in SL'(\alpha_0)$ then $\overline{\mathbf{Y}}_i\alpha \in SL'(\alpha_0)$, $i \in [n]$.
- $\mathbf{X}_i\mathsf{True} \in SL'(\alpha_0)$ for all $i \in [n]$, and $\mathbf{Y}_i\mathsf{True} \in SL'(\alpha_0)$ for all $i \in [n]$.
- for all $i \in [n]$, $\tau_i \in SL'(\alpha_0)$.

Now define $SL(\alpha_0) \stackrel{\text{def}}{=} SL'(\alpha_0) \cup \{\neg\alpha \mid \alpha \in SL'(\alpha_0)\}$ with $\neg\neg$ treated as identity.
A set $A \subseteq SF_l$ is said to be a layer atom iff it satisfies the following conditions:

- for every formula $\neg\alpha \in SL(\alpha_0)$, $\neg\alpha \in A$ iff $\alpha \notin A$.
- for every formula $\alpha_1 \vee \alpha_2 \in SL(\alpha_0)$, $\alpha_1 \vee \alpha_2 \in A$ iff $\alpha_1 \in A$ or $\alpha_2 \in A$.
- if $\mathbf{X}_i\alpha \in A$ then $\mathbf{X}_i\mathsf{True} \in A$ and $\overline{\mathbf{X}}_i\alpha \in A$.
- if $\mathbf{Y}_i\alpha \in A$ then $\mathbf{Y}_i\mathsf{True} \in A$ and $\overline{\mathbf{Y}}_i\alpha \in A$.
- if $\mathbf{G}\alpha \in A$ then $\alpha \in A$.
- if $\mathbf{H}\alpha \in A$ then $\alpha \in A$.
- $\mid \{\tau_i \mid i \in [n], \tau_i \in A\} \mid = 1$.

Define a relation $\preceq$ on layer atoms as follows: Let A and B be layer atoms such that $\tau_i \in A$ and $\tau_j \in B$. Then $A \preceq B$ iff:

- if $\overline{\mathbf{X}}_j\alpha \in A$ then $\alpha \in B$.

- if $\overline{\mathbf{Y}}_i\alpha \in B$ then $\alpha \in A$.

Call an n-tuple of layer atoms $(A_1, \ldots, A_n)$ coherent if, for all $i \in [n]$, $\tau_i \in A_i$. Given two coherent n-tuples of layer atoms $(A_1, \ldots, A_n)$ and $(B_1, \ldots, B_n)$, we say $(A_1, \ldots, A_n) \preceq (B_1, \ldots, B_n)$ if for all $i \in [n]$, $A_i \preceq B_i$. Let $\widetilde{AT_l}$ denote the set of coherent n-tuples of layer atoms. Below, we will also need the set $A_{tup} = \widetilde{AT_l} \cup \{(\bot_1, \ldots, \bot_n)\}$, where the symbol $\bot_i$ will be used to denote that no layer atom is associated with i. By convention, set $(\bot_1, \ldots, \bot_n) \preceq X$, for every $X \in \widetilde{AT_l}$.

We now move on to computing the alphabet of the fragment automaton $\mathcal{A}_0$ to be associated with φ_0. For this first define the set $\mathcal{LF}_b(AT_l)$ of b-bounded fragments which are labelled by layer atoms. Define $LF = (F, a)$ where $F = (E, \leq, \phi, AT_l, \eta)$ and $a \in \Gamma_0$ ($\Gamma_0 = \Gamma \cap CL(\varphi_0)$). Define $\nu : E \to AT_l$ by: $\nu(e) = A$, where $\eta(e) = (A, T)$. We require that ν satisfies the following *consistency* conditions:

- for all $e \in E$, $\phi(e) = i$ iff $\tau_i \in \nu(e)$.
- for all $e, e' \in E$, if $e \lessdot e'$ then $\nu(e) \preceq \nu(e')$.
- for all $e \in E_i$, for all $\mathbf{F}\alpha \in SF_l$, $\mathbf{F}\alpha \in \nu(e)$ iff there exists $e' \in E_i$ such that $e \leq e'$ and $\alpha \in \nu(e')$.
- for all $e \in E_i$, for all $\mathbf{P}\alpha \in SF_l$, $\mathbf{P}\alpha \in \nu(e)$ iff there exists $e' \in E_i$ such that $e' \leq e$ and $\alpha \in \nu(e')$.
- for all $e \in E_i$, if $(s, j, A) \in \eta(e)$ then $\mathbf{X}_j\mathsf{True} \in \nu(e)$ and $\nu(e) \preceq A$.
- for all $e \in E_i$, if $(r, j, A) \in \eta(e)$ then $\mathbf{Y}_j\mathsf{True} \in \nu(e)$ and $A \preceq \nu(e)$.

By $\nu_{min}(F)$, we denote the (coherent) n-tuple of layer atoms $(\nu(e_1), \ldots, \nu(e_n))$, where e_i is the i-minimum event in F. Similarly, by $\nu_{max}(F)$, we denote the (coherent) n-tuple of layer atoms $(\nu(f_1), \ldots, \nu(f_n))$, where f_i is the i-maximum event in F.

We are now ready to define the required fragment automaton working on the alphabet $\mathcal{LF}_b(AT_l)$ of fragments. The automaton construction follows the one used for C_b-satisfiability, but states are now enriched with coherent tuples of layer atoms (except initial states, where we resort to the use of $\bot_i$). Such a tuple corresponds to the layer atoms labelling the maximal events that the automaton has seen thus far. Hence the transition relation checks that this tuple is $\preceq$-related to the minimal ones of the fragment labelling the transition, thus ensuring that the fragment may be joined to the diagram being constructed by the accepting run of the automaton.

Formally, the automaton $\mathcal{A}_0$ associated with φ_0 is defined as follows:

$$\mathcal{A}_0 = (Q, \mathcal{LF}_b(AT_l), \rightarrow, I, F)$$

where

- $Q = AT(\varphi_0) \times U \times A_{tup}$
  where $U = \{\psi_1\mathbf{U}\psi_2 \mid \psi_1\mathbf{U}\psi_2 \in CL(\varphi_0)\}$.
- $I = \{(T, \emptyset, (\bot_1, \ldots, \bot_n) \mid T \in AT(\varphi_0)$ such that $\varphi_0 \in T\}$.
- $F = \{(T, \emptyset, X) \mid T \in AT(\varphi_0), X \in \widetilde{AT_l}\}$.
- $\rightarrow$ is given by
  $(T, u, (A_1, \ldots, A_n)) \xrightarrow{LF} (T', u', (A'_1, \ldots, A'_n))$ (where $LF = (F, a)$, iff
  1. $a \in T$.
  2. For every $\bigcirc\psi \in CL(\varphi_0)$, $\bigcirc\psi \in T$ iff $\psi \in T'$.
  3. If $u \neq \emptyset$ then, $u' = \{\psi_1\mathbf{U}\psi_2 \mid \psi_1\mathbf{U}\psi_2 \in u$ and $\psi_2 \notin T'\}$. Otherwise, $u' = \{\psi_1\mathbf{U}\psi_2 \mid \psi_1\mathbf{U}\psi_2 \in T'$ and $\psi_2 \notin T'\}$.
  4. $\nu_{max}(F) = (A'_1, \ldots, A'_n)$.

5. $(A_1, \ldots, A_n) \preceq \nu_{min}(F)$.
6. for $i \in [n]$, $\alpha @ i \in T$ iff $\alpha \in B_i$, where $\nu_{min}(F) = (B_1, \ldots, B_n)$.

Lemma 5.6 $L_S^b(\mathcal{A}_0) = \{M \mid M$ *is a model of* φ_0 *based on a channel b-bounded LLD* $\}$.

Proof:
Let $D = ((LF_0 \bullet LF_1) \bullet \ldots) \in L_S^b(\mathcal{A}_0)$, where $LF_k = (F_k, a_k)$. Suppose $D = (E, \leq, \phi, \lambda)$. Note that, from the way concatenation of fragments has been defined, we can define the map $\nu : E \rightarrow AT_l$ (uniquely), based on the map ν_k. Now define $M = (D, V_E, V_\lambda)$ by: for all $e \in E$, $V_E(e) = \nu(e) \cap P$ and $V_\lambda(k) = a_k$.

Note that D has a crucial property: for all $e \in E_i$, if e comes from $F_k, k \geq 0$ and $(s, j, A) \in \eta_k(e)$, then there exists $\ell > k$ and $e' \in E_j$ such that e' comes from F_ℓ, $(r, i, B) \in \eta_\ell(e')$, $e \lessdot e'$, $\nu(e') = A$ and $\nu(e) = B$. A similar assertion holds the other way as well: for e, k as above, if $(r, j, A) \in \eta_k(e)$, then there exists $\ell < k$ and $e' \in E_j$ such that e' comes from F_ℓ, $(s, i, B) \in \eta_\ell(e')$, $e' \lessdot e$, $\nu(e') = A$ and $\nu(e) = B$. In addition, if e_1 is the i-maximum event of F_k and e_2 the i-minimum event of F_{k+1}, then by definition of the transition relation $\rightarrow$, we have that $\nu(e_1) \preceq \nu(e_2)$. From these, and the property of ν_k for each fragment F_k, we have the following, which we call the *successor labelling property*:

- for all $e, e' \in E$, if $e \lessdot e'$, then $\nu(e) \preceq \nu(e')$.
- for all $e \in E$, if $\mathbf{X}_i\mathsf{True} \in \nu(\mathsf{e})$, then there exists $e' \in E_i$ such that $e \lessdot e'$.
- if $\mathbf{Y}_i\mathsf{True} \in \nu(\mathsf{e})$, then there exists $e' \in E_i$ such that $e' \lessdot e$.

We first prove the following claim.
Claim: For all $e \in E$, $\alpha \in SF_l$, $M, e \models_l \alpha$ iff $\alpha \in \nu(e)$.
Proof of Claim: The proof proceeds by induction on the structure of α. The base case, when $\alpha \in P$, is seen to hold by the way V_E was defined. The cases when α is of the form $\neg\beta$ or of the form $\beta_1 \vee \beta_2$ routinely follow by application of the induction hypothesis.

Now suppose $\alpha = \mathbf{X}_i\beta$ and let $e \in E$. If $M, e \models_l \mathbf{X}_i\beta$, then there exists $e' \in E_i$ such that $e \lessdot e'$ and $M, e' \models_l \beta$. We have, by successor labelling property, $\nu(e) \preceq \nu(e')$. By induction hypothesis, $\beta \in \nu(e')$. Then, by definition of $\preceq$, $\mathbf{X}_i\beta \in \nu(e)$, as required.

On the other hand, if $\mathbf{X}_i\beta \in \nu(e)$, then by definition of layer atoms, $\mathbf{X}_i\mathsf{True} \in \nu(\mathsf{e})$. By successor labelling property, there exists $e' \in E_i$ such that $e \lessdot e'$ and hence $\nu(e) \preceq \nu(e')$. Since $\nu(e)$ is an atom, $\overline{\mathbf{X}}_i\beta \in \nu(e)$ as well, and hence, by definition of $\preceq$, $\beta \in \nu(e')$. But then, by induction hypothesis, $M, e' \models_l \beta$ and hence we get: $M, e \models_l \mathbf{X}_i\beta$, as required.

The case when $\alpha = \mathbf{Y}_i\beta$ is similar. The cases when $\alpha = \mathbf{F}\beta$ or $\alpha = \mathbf{P}\beta$ are proved easily using the consistency condition on ν_k where $k = \lambda(e)$ and the induction hypothesis. This completes the induction, proving the Claim.

We now show that $M, 0 \models \varphi_0$. Let $\rho = (T_0, u_0, X_0), (T_1, u_1, X_1), \ldots$ be an accepting run of $\mathcal{A}_0$ on D. For this, it suffices to show that for every temporal formula $\psi \in CL(\varphi_0)$ and every layer k, $M, k \models \psi$ iff $\psi \in T_k$. The proof of this assertion, by induction on the structure of ψ, is exactly as that of the corresponding one in Lemma 5.3 except for the base case when $\psi = \alpha @ i$.

For the base case, observe that $M, k \models \alpha @ i$ iff $M, e \models_l \alpha$ where e is the i-minimum event of the fragment F_k. But, from the claim above we know that $M, e \models_l \alpha$ iff $\alpha \in \nu(e)$. By definition of $\rightarrow$, $\alpha \in \nu(e)$ iff $\alpha @ i \in T_k$, when e is the i-minimum event of F_k. The other base case, when $\psi \in \Gamma$, is also easily seen to follow from the definition of $\rightarrow$.

Conversely, to show that every model of φ_0 is in $L_S^b(\mathcal{A}_0)$, let $M = (D, V_E, V_\lambda)$ be such that such that $M, 0 \models \varphi_0$. Let $F_0, F_1, \ldots$ be the sequence of fragments associated with D. We show that $L_S^b(\mathcal{A}_0) \neq \emptyset$ by exhibiting an accepting run of $\mathcal{A}_0$.

For an event e of M, define $A_e = \{\alpha \in SF_l \mid M, e \models_l \alpha\}$. It can be easily checked that A_e is a layer atom. Now let $k \geq 0$. Define $T_k = \{\psi \in CL(\varphi_0) \mid M, k \models \psi\}$. We can show that T_k is a temporal atom. For $j \in [n]$, define A_j^k as follows: $A_j^0 = \perp_j$; $A_j^{k+1} = A_{e_k^j}$ where e_k^j is the j-maximum event of F_k.

Let $LF_k = (F_k, a)$ where for all e in the fragment F_k, $\nu_k(e) = A_e$, and $a \in \Gamma$ such that $M, k \models a$. We now construct a sequence $\rho = (T_0, u_0, (A_1^0, \ldots A_n^0)), (T_1, u_1, (A_1^1, \ldots A_n^1)), \ldots$ of states of $\mathcal{A}_0$ and show that ρ is a accepting run of $\mathcal{A}_0$ on $LF = LF_0, LF_1, \ldots$. The sets u_k are defined by induction on k. Define $u_0 = \emptyset$

and suppose u_k has been defined. Define $u_{k+1} = \{\psi_1 \mathbf{U} \psi_2 \mid \psi_1 \mathbf{U} \psi_2 \in T_{k+1}, \psi_2 \notin T_{k+1}\}$ if $u_k = \emptyset$ and $u_{k+1} = \{\psi_1 \mathbf{U} \psi_2 \mid \psi_1 \mathbf{U} \psi_2 \in u_k, \psi_2 \notin T_{k+1}\}$ otherwise.

We can then easily show that ρ is an accepting run of $\mathcal{A}_0$ on LF. □

6 Discussion

We have discussed decidability of linear time temporal logics defined on Lamport Diagrams. We have suggested that layered diagrams as well as layered specifications are of interest.

Lamport Diagrams are depictions of computations of distributed systems rather than descriptions of systems themselves. In this paper, we have not addressed the important question of *system models* whose behaviour is given in terms of LDs. The diagram automata and fragment automata presented in the paper can be seen as possible candidates for finite state system models. For such models, the theorems presented here, yield as corollaries, model checking procedures for specifications in the logic. This is because the class of diagrams accepted by these automata are easily seen to be closed under intersection. For instance, given a diagram automaton $\mathcal{A}$ and a formula ϕ_0, we can check whether $L^b_C(\mathcal{A}) \cap L^b_C(\mathcal{A}_\prime)$ is empty, where $\mathcal{A}_\prime$ is the diagram automaton associated with the formula $\neg\phi_0$. However, it is hardly clear whether theoretical constructs like diagram or fragment automata can be considered as satisfactory system models for message passing. One reason for this is that Lamport diagrams depict causality rather than describe system behaviour. The reading of causal ordering in terms of communication is only an interpretation, and the terminology of send / receive events used here is indicative rather than literal.

To see this, consider events $e_1 \lessdot e_2$ of agent 1, $e_3 \lessdot e_4$ of agent 2 and $e_2 \lessdot e_3$. Clearly, by transitivity, $e_1 \leq e_4$, and though this is not in the covering relation of the partial order, this could well be a send / receive pair. If this is seen as message overtaking, and hence perhaps not important for specifications, consider now $e_5 \lessdot e_6$ of agent 3 with $e_4 \lessdot e_5$. Once again $e_1 \leq e_6$, but if this is a send / receive pair, it is hard to dismiss such a possibility as message overtaking, since e_1 was meant for agent 3 and e_2 was meant for agent 2. Thus some relevant detail may be hidden away in these descriptions.

Better candidate classes for system models are those based on Message Sequence Charts (MSCs) ([15], [17]), such as Message Sequence Graphs ([3]) and Message Passing Automata ([7]). In fact, LDs can be seen as the partial orders generated by MSCs (in the obvious manner; for instance, see [1]). With such an understanding, layered diagrams with bounded and communication closed layers may be seen to correspond to the class of Message Sequence Graphs (MSGs) which are finite state automata with their nodes labelled by MSCs. Bounded LLDs which are not communication closed, but where channel capacity is bounded, correspond to the class of Compositional Message Sequence Graphs (CMSGs) introduced in [6]. We can show that Lamport diagrams corresponding to MSG-generated MSCs form a subclass of communication closed and bounded LLDs, and a similar remark applies to CMSGs.

In [3], the authors study model checking problems for MSCs and MSGs. They consider the problem of deciding whether the linearizations of events of all the MSCs defined by an MSG satisfy some linear-time property (presented as an automaton) and show that the problem is undecidable. They then identify a restricted class of MSGs (called locally synchronized or bounded MSGs) for which the problem becomes decidable. Regular MSC languages were studied in [7] where an MSC language is a collection of all linearizations of the MSC. As we have already remarked, many protocols (for example, the producer — consumer protocol) can be described using bounded communication closed diagrams, but, the associated set of linearizations of such a diagram need not be a regular MSC language. In fact, this protocol cannot be described by a locally synchronized MSG either.

We remark that the logics studied here are structural (in the sense of [10]) rather than one defined on linearizations. Thus it is closer to the logic TLC^- defined by Peled in [12]. Formulas of TLC^- are interpreted on the partially ordered structure of an MSC. Peled considers the model checking problem for this logic with the system being modelled as an MSG and the decidability proof is done by considering linearizations. Bollig and Leucker ([4]) study generalizations of Lamport diagrams and MSCs using a related logic which is called $LTML^-$. Again, they work with linearizations of the corresponding diagrams and given a formula and a (fixed) finite channel, they construct an alternating Büchi automaton which accepts those linearizations satisfying the formula and corresponding to the channel.

[10] addresses model checking problems for MSCs against specifications given in monadic second order logic (MSO). The interest here is that model checking MSO specifications are decidable (in this context) even though satisfiability is undecidable. We can study MSO formulas interpreted over LDs and LLDs and show that the theorems presented here generalize to MSO specifications as well. In the presence of uniform bounds (on size of layers and channel capacity), the subclasses studied are restricted enough to yield decidability of satisfiability as well.

Acknowledgements: We thank the referees for comments which helped to improve the presentation.

Appendix

Proof of Theorem 2.3:

Here we give details of the proof of Theorem 2.3, by showing that models for formulas of the logic LD_1 correspond to halting computations of 2-counter machines.

Assume that we are given a non-deterministic 2-counter machine $M = (Q, \delta, q_0, q_f)$ where $\delta \subseteq (Q \times \{0,1\}^2 \times Q \times \{-1,0,1\}^2)$ is the transition relation of M. Consider the formula α_M associated with this machine, as defined in Section 2. We now show that α_M is satisfiable iff a final configuration of M is reachable. This implies that finite satisfiability of LD_1 is undecidable.

Suppose there exists a run of M in which a final configuration is reachable, say $\rho = (q^0, n_1^0, n_2^0) \to \ldots (q^k, n_1^k, n_2^k) \to \ldots \to (q^m, n_1^m, n_2^m)$ where $q^0 = q_0$, $n_1^0 = n_2^0 = 0$ and $q^m = q_f$. We will show that α_M is satisfiable by defining a Lamport diagram D_M and a valuation function V_M and then by proving that $M' = (D_M, V_M)$ is a model of α_M.

Define $D_M = (E, \leq, \phi)$ as follows:

- $E = \cup_{0 \leq k \leq m} E_k$ where $E_k = \{c_1e(j,k) \mid 1 \leq_{\mathbb{N}} j \leq_{\mathbb{N}} n_1^k\} \cup \{se_k\} \cup \{c_2e(j,k) \mid 1 \leq_{\mathbb{N}} j \leq_{\mathbb{N}} n_2^k\}$.

  For $0 \leq k \leq m$, define

  $emax_k = \begin{cases} se_k & \text{if } n_2^k = 0 \\ c_2e(n_2^k, k) & \text{otherwise} \end{cases}$

  and

  $emin_k = \begin{cases} se_k & \text{if } n_1^k = 0 \\ c_1e(1, k) & \text{otherwise} \end{cases}$

- $\phi(e) = \begin{cases} 1 & \text{if } e \in E_{2l},\ 0 \leq_{\mathbb{N}} 2l \leq_{\mathbb{N}} m \\ 2 & \text{otherwise} \end{cases}$

- $\leq \ \stackrel{\text{def}}{=} \ (\mathsf{local} \cup \mathsf{comm})^*$ where

  $\mathsf{local} = \{(emax_k, emin_{k+2}) \mid 0 \leq_{\mathbb{N}} k <_{\mathbb{N}} (m-1)\}$
  $\cup_{0 <_{\mathbb{N}} k \leq_{\mathbb{N}} m} (\cup\{(c_1e(j,k), c_1e(j+1,k)) \mid 1 \leq_{\mathbb{N}} j <_{\mathbb{N}} n_1^k\}$
  $\cup \{(c_1e(n_1^k, k), se_k) \mid n_1^k >_{\mathbb{N}} 0\}$
  $\cup \{(se_k, c_2e(1,k)) \mid n_2^k >_{\mathbb{N}} 0\}$
  $\cup \{(c_2e(j,k), c_2e(j+1,k)) \mid 1 \leq_{\mathbb{N}} j <_{\mathbb{N}} n_2^k\})$

  and

  $\mathsf{comm} = \{(se_k, se_{k+1}) \mid 0 \leq_{\mathbb{N}} k <_{\mathbb{N}} m\}$
  $\cup \{(c_1e(j,k), c_1e(j,k+1)) \mid 1 \leq_{\mathbb{N}} j \leq_{\mathbb{N}} min\{n_1^k, n_1^{k+1}\}, 0 \leq_{\mathbb{N}} k <_{\mathbb{N}} m\}$
  $\cup \{(c_2e(l_1(j),k), c_2e(l_2(j),k+1)) \mid 1 \leq_{\mathbb{N}} j \leq_{\mathbb{N}} min\{n_2^k, n_2^{k+1}\}, 0 \leq_{\mathbb{N}} k <_{\mathbb{N}} m\}$ where

  - $l_1(j) = l_2(j) = j$ if $n_2^k = n_2^{k+1}$,
  - $l_1(j) = j+1, l_2(j) = j$ if $n_2^k > n_2^{k+1}$ and
  - $l_1(j) = j, l_2(j) = j+1$ if $n_2^k < n_2^{k+1}$.

The valuation function V_M is defined as $V_M(e) = V_k(e)$ for all $e \in E_k$ where V_k is defined as follows:

- For $l = 1, 2$,
$$V_k(c_l e(j,k)) = \begin{cases} \{c_l, \tau_1\} & \text{if } k \bmod 2 = 0 \\ \{c_l, \tau_2\} & \text{otherwise} \end{cases}$$

- $V_k(se_k) = \{q^k, \tau_{f(k)}\} \cup \{ic \mid k = 0\} \cup \cup_{l \in \{1,2\}}(\{c_l z \mid n_l^k = 0\} \cup \{inc_l \mid n_l^k >_{\mathbb{N}} n_l^{k-1}, 0 <_{\mathbb{N}} k \leq_{\mathbb{N}} m\})$
$\cup \{dec_l \mid n_l^k >_{\mathbb{N}} n_l^{k+1}, 0 <_{\mathbb{N}} n_l^k, 0 \leq_{\mathbb{N}} k <_{\mathbb{N}} m\}$ where
$$f(k) = \begin{cases} 1 & \text{if } k \bmod 2 = 0 \\ 2 & \text{otherwise} \end{cases}$$

We now show that M' is a model of α_M.

- We can easily show that M' satisfies the formulas $\mathsf{init}, \mathsf{fin}, \mathsf{state}, \mathsf{ctr}_1$ and ctr_2 by using the definition of V_M.

- We now show that M' satisfies trans_i. To show that M' satisfies $\bigvee_{t \in \delta} \theta_t$, consider an event $se_k \in E_k$ such that $V_k(se_k) = q^k$ where $q^k \neq q_f$. Then, by the definition of D_M, we know that the event $se_{k+1} \in E_{k+1}$ is such that $q^{k+1} \in V_{k+1}(se_{k+1})$ where $(q^k, n_1^k, n_2^k) \to (q^{k+1}, n_1^{k+1}, n_2^{k+1})$ in ρ (say, through the transition $t' \in \delta$). Using the definition of V_{k+1}, we can show that $M', se_k \models \theta_{t'}$. Let $t' = (q^k, x_1, y_1, q', x_1', y_1')$. Suppose $M', se_k \models \theta_{t''}$ for some $t'' \in T \setminus \delta$. Let $t'' = (q^k, x_2, y_2, q', x_2', y_2')$. By the definition of θ_t and from the fact that se_{k+1} is the unique i^{co}-successor to se_k (where se_k is an event of agent i), we know that $M', se_{k+1} \models q'$. But, $M', se_{k+1} \models q^{k+1}$ as well, and since $M', se_{k+1} \models \mathsf{state}$, we get $q' = q^{k+1}$. Similarly, using $\mathsf{z\text{-}consis}_i$, we can show that $x_1 = x_2$ and $y_1 = y_2$, and using $\mathsf{state\text{-}inc\text{-}dec\text{-}consis}_i$, we can show that $x_1' = x_2'$ and $y_1' = y_2'$. But then $t'' = t'$ contradicting the assumption that $t'' \notin \delta$.

- We now prove that M' satisfies consis_i. From the first definition on the local ordering of events in E_k and from the definition of V_k it follows that seq_i is satisfied by M'.

- To show that M' satisfies $\mathsf{z\text{-}consis}_i$, we know by the definition of V_M that $c_j z$ is true at an event se_k only if state is true. In addition, we also know that $c_j z$ is true at e_k iff $n_j^k = 0$ for $j = 1, 2$. If $n_1^k = 0$, then, from the definition of D_M, we see that there is no event of the form $c_1 e(j,k)$ immediately preceding e_k. Since c_1 holds only at events of the form $c_1 e(j,k)$ we can conclude that $\overline{\mathbf{Y}}_i \neg c_1$ holds at se_k. Similarly, we can show that $c_2 z$ holds iff $\overline{\mathbf{X}}_i \neg c_2$ holds at se_k.

- We can show that M' satisfies $\mathsf{inc\text{-}consis}_i$, $\mathsf{dec\text{-}consis}_i$, $\mathsf{z\text{-}dec\text{-}consis}_i$ and $\mathsf{state\text{-}inc\text{-}dec\text{-}consis}_i$ by using the definition of V_M.

- Finally, to show that M' satisfies $\mathsf{across\text{-}consis}_i$, consider an event se_k such that $M', se_k \models \mathsf{state}$. If $k = m$, we know that $q_f \in V_m(se_k)$ and so $M', se_k \models q_f$. Also, if $k = 0$, we know that $ic \in V_k(se_k)$ and so $M', se_k \models ic$. For $0 < k < m$, by the definition of D_M, we know that $se_{k+1} \in E_{k+1}$ is such that $se_k \lessdot se_{k+1}$ and $q^{k+1} \in V_{k+1}(se_{k+1})$. Also, $\phi(se_{k+1}) = i^{co}$ if $\phi(se_k) = i$. This implies $M', se_k \models \mathbf{X}_{i^{co}}\mathsf{state}$. Similarly, $se_{k-1} \in E_{k-1}$ is such that $se_{k-1} \lessdot se_k$ and $\phi(se_{k-1}) = i^{co}$ if $\phi(se_k) = i$. This implies $M', se_k \models r_i^{i^{co}}$.

  Now, consider an event e such that $M', e \models c_1$ where e is of the form $c_1 e(j,k)$. If $j < n_1^k$ then, we know from the definition of M' that $c_1 e(j, k+1)$ is such that $c_1 e(j,k) \lessdot c_1 e(j,k+1)$, $M', c_1 e(j,k+1) \models c_1$ and $\phi(c_1 e(j,k+1)) = i^{co}$ if $\phi(c_1 e(j,k)) = i$. Hence $M', c_1 e(j,k) \models \mathbf{X}_{i^{co}} c_1$. Suppose $n_1^{k+1} = n_1^k - 1$. Then, we know from the definition of V_M that dec_1 holds at se_k and hence $M', c_1 e(n_1^k, k) \models \mathbf{X}_i(\mathsf{state} \wedge dec_1)$.

  Since $k > 0$, for $j < n_1^k$, the event $c_1 e(j, k-1)$ is such that $c_1 e(j,k-1) \lessdot c_1 e(j,k)$ and $\phi(c_1 e(j,k-1)) = i^{co}$ if $\phi(c_1 e(j,k)) = i$. Hence $M', c_1 e(j,k) \models r_i^{i^{co}}$. When $n_1^{k+1} = n_1^k + 1$, we can show that $\mathbf{X}_i(\mathsf{state} \wedge inc_1)$ holds at $c_1 e(n_1^k, k)$.

  We can argue as above using the definition of $\leq$ in D_M and the definition of the valuation function to show that the sub-formula involving c_2 also is satisfied.

Conversely, suppose α_M is satisfiable. We have to show that a final configuration of M is reachable. We will prove the existence of a run ρ of M such that ρ terminates at a final configuration. Let $M' = (D, V)$ where $D = (E, \leq, \phi)$ be a model of α_M. Since M' satisfies init, we know that there exists $e_0 \in E$ such that $M', e_0 \models \mathsf{init}$ and since M' satisfies fin, we know that there exists an event $e_m \in E$ such that $e_0 \leq e_m$ and $M', e_m \models q_f$. Let $X = \{e \in E \mid e_0 \leq e \leq e_m \text{ and } M', e \models \vee_{q \in Q} q\}$.

We claim that events in the set X can be ordered as $e_0 = f_0 \lessdot f_1 \lessdot \ldots \lessdot e_m = f_m$, where $\phi(f_0) = 1$ and if $\phi(f_k) = i$ then $\phi(f_{k+1}) = i^{co}$ for $0 \leq k < m$.

We prove the claim by induction on k. Initially, set $f_0 = e_0$. $\phi(e_0) = 1$ since $M', e_0 \models \mathsf{init}$. If $q_0 = q_f$, then $e_0 = e_m$ and we are done.

Otherwise, inductively suppose $f_0 \lessdot f_1 \lessdot \ldots \lessdot f_k$ $(k \geq 0)$ has been defined such that

1. $f_0, f_1, \ldots f_k$ satisfy the above properties,

2. For $k \geq 2$, there exists no event $e \in E$ such that $f_j < e < f_{j+2}$, $0 \leq_{\mathbb{N}} j <_{\mathbb{N}} (k-1)$ and $M', e \models \mathsf{state}$.

We define f_{k+1} as follows. Suppose $\phi(f_k) = i$ for $i \in \{1, 2\}$. Since $M', f_k \models \mathsf{state}$ we know that $M', f_k \models \mathsf{across\text{-}consis}_i$ and so there exists $e' \in E$ such that $f_k \lessdot e'$, $\phi(e') = i^{co}$ and $M', e' \models \mathsf{state}$. If we show that $e' \leq e_m$, then, $e' \in X$ and we set $f_{k+1} = e'$. Suppose $e' \not\leq e_m$. Then, $M', e' \models \mathsf{across\text{-}consis}_{i^{co}}$ implies there exists $e'' \in E$ such that $e' \lessdot e''$, $\phi(e'') = i$ and $M', e'' \models \mathsf{state}$. If $e'' \leq e_m$, then we get $e' \leq e'' \leq e_m$, contradicting the assumption that $e' \not\leq e_m$. If not, we can repeat the above argument for e'' and its successor events (resulting from the fact that they all satisfy $\mathsf{across\text{-}consis}_i$) to reach e_m to conclude that $e_m \notin X$, a contradiction.

We now show that there exists no event $e_1 \in E$ such that $f_{k-1} < e_1 < f_{k+1}$ $(k \geq 1)$ such that $M', e_1 \models \mathsf{state}$. Suppose there exists such an event e_1. Then, $M', e_1 \models r^i_{i^{co}}$ (since $\mathsf{across\text{-}consis}_{i^{co}}$ holds at e_1) and so there exists an event $e_2 \in E$ such that $e_2 \lessdot e_1$ and $\phi(e_2) = i$. Now $f_k \not\leq e_2$ as it would contradict the fact that $f_k \lessdot f_{k+1}$. Hence $e_2 \leq f_k$. We can show that $M', e_2 \models \mathsf{state}$ using the fact that state holds at e_1 and that e_2 is the unique i-predecessor of e_1. This contradicts the inductive assumption for the events f_{k-2} and f_k.

Hence induction on k is complete and we have: $X = \{f_0, f_1, \ldots f_m\}$.

We now define a sequence $\rho = (q_0, n_1^0, n_2^0), (q_1, n_1^1, n_2^1), \ldots, (q_m, n_1^m, n_2^m)$ where $q_m = q_f$, q_k is such that $M', f_k \models q_k$ and n_1^k, n_2^k are defined as follows. Initially, $n_1^0 = n_2^0 = 0$. To define n_1^k and n_2^k for $k > 0$, consider the event f_k. If $M', f_k \models c_i z$, we define $n_i^k = 0$ for $i = 1, 2$. Otherwise, we know that there exists at least one event immediately before f_k satisfying c_1. Let $e_1, e_2, \ldots, e_m$ be the maximum sequence of events in E such that $e_m \lessdot \ldots \lessdot e_1 \lessdot f_k$, $\phi(f_k) = \phi(e_j)$ and $V(e_j) = c_1$ for all j, $1 \leq j \leq m$. Define $n_1^k = m$. Similarly, we define $n_2^k = m'$ where $e'_1, e'_2, \ldots e'_{m'}$ is the maximum sequence of events in E such that $f_k \lessdot e'_1 \lessdot \ldots \lessdot e'_{m'}$, $\phi(f_k) = \phi(e'_j)$ and $V(e'_j) = c_2$ for all j, $1 \leq j \leq m'$.

We now show that $(q_k, n_1^k, n_2^k) \to (q_{k+1}, n_1^{k+1}, n_2^{k+1})$ for all k, $0 \leq k \leq m-1$ by using the fact that $M' \models \mathsf{trans}_i$ and that $M' \models \mathsf{consis}_i$ for $i = 1, 2$. Consider the event f_k which satisfies q_k. Suppose $\phi(f_k) = i$. We know that f_{k+1} is such that $\phi(f_{k+1}) = i^{co}$ and $M', f_{k+1} \models q_{k+1}$. Since $M', f_k \models \mathsf{trans}_i$ and f_{k+1} is the unique immediate successor of f_k in i^{co}, it follows that there exists a transition $t = (q_k, x, y, q_{k+1}, x', y') \in \delta$ such that $M', f_k \models \theta_t$. Let $\theta_t = r_x^1 \wedge r_x^2 \wedge \mathbf{X}_{i^{co}} q_{k+1} \wedge \xi^1_{x',i} \wedge \xi^2_{y',i}$. We have to show that $n_1^{k+1} = n_1^k + x'$ and $n_2^{k+1} = n_2^k + y'$. Suppose $x' = -1$. Then, since $M', f_k \models \xi^1_{-1,i}$, we know that $M', f_k \models dec_1$ and there exists an event e' such that $f_k \lessdot e'$, $\phi(e') = i^{co}$ and $M', e' \models \neg inc_1$. Now using the fact that M' satisfies $\mathsf{dec\text{-}consis}_i$, we know that there exists an event e'' satisfying c_1 which is immediately before f_k in i and it does not have an immediate successor in i^{co}. Also, $\mathsf{across\text{-}consis}_i$ holds and so all the events in the past of e'' satisfying the proposition c_1 (representing n_1^k in agent i) have successors in agent i^{co}. So, all the events satisfying c_1 representing the counter value n_1^k get copied to agent i^{co} except for the event e''. Hence we can conclude that $n_1^{k+1} = n_1^k - 1$. We can similarly argue for the other values of x' and y' using the fact that M' satisfies $\mathsf{inc\text{-}consis}_i \wedge \mathsf{z\text{-}dec\text{-}consis}_i$ (when x' or y' is 1) to show that $n_1^{k+1} = n_1^k + x'$ and $n_2^{k+1} = n_2^k + y'$. Hence ρ as defined above is a run of M and since $q_m = q_f$ it follows that ρ is a run of M ending in a final configuration.

This completes our demonstration that the reduction of the halting problem for 2-counter machines to finite satisfiability of LD_1 is correct.

Proof of Theorem 2.4:

The proof is very similar to that of Theorem 2.3. Given a run ρ of M ending in a final configuration, we define a model of α_M as done in the above proof. In addition, we add the set E_0' to E and extend the valuation function suitably to take care of the extra initial event satisfying ic.

That is, define a model $M' = (D', V')$ where $D' = (E', \leq', \phi')$ is given by

- $E' = E_0' \cup E$ where $E_0' = \{e_0'\}$ and E is as defined in the above proof.
- $\phi'(e_0') = 2$ and $\phi'(e) = \phi(e)$ for every $e \in E$.
- $\leq' = (\leq' \cup \{(e_0', e_0') \cup (e_0', se_0)\})^*$.

Finally, the valuation function V' is given by $V'(e_0') = \{\tau_1, q_0, c_1z, c_2z, ic\}$ and $V'(e) = V(e)$ for all $e \in E$.

We now show that M' is a model of α_M by arguing that M' satisfies all the formulas which were re-defined for α_M in this theorem.

To start with, $M', se_0 \models \mathsf{init}$ as $V_0 = \{q_0, c_1z, c_2z, ic\}$ and $e_0' <_c se_0$ is such that $\phi'(e_0') = 2$ and $M', e_0' \models (q_0 \wedge c_1z \wedge c_2z \wedge ic)$. We can show that the formulas fin, trans_i, $\mathsf{inc\text{-}consis}_i$ and $\mathsf{dec\text{-}consis}_i$ are all satisfied by M' as done in the proof of Theorem 2.3 above. To show that M' satisfies $\mathsf{across\text{-}consis}_i$, we first note that $\mathbf{X}_{i^{co}}\beta \supset s_i^{i^{co}}$ for any formula β which is not False and the truth of sub-formulas involving the $\mathbf{Y}_{i^{co}}$ modality in $\mathsf{across\text{-}consis}_i$ can be proved in a way similar to the proof of Theorem 2.3.

Conversely, suppose α_M is satisfiable. We will construct a run ρ of M such that ρ terminates at a final configuration as done in the proof of Theorem 2.3 above. The set X of events satisfying states is defined in the same way and we prove that events in X can be arranged in a sequence by working backwards from the event e_m which satisfies q_f. The sequence ρ is defined exactly as done in the above proof and we can again show that ρ defines a run of M to a configuration containing q_f.

Proof of Theorem 4.1:

Here we give details to show that B-satisfiability is undecidable.

We define the sets P_1 and P_2 of propositions of agent 1 and 2 as done in the proof of Theorem 2.3. In addition, we add a proposition d to represent dummy events and define the set of layer names $\Gamma = \{a\}$.

The formula φ_M is given by $\varphi_M \stackrel{\text{def}}{=} \varphi_b \wedge \varphi_d \wedge \varphi_{\mathsf{init}} \wedge \varphi_{\mathsf{fin}} \wedge \varphi_{\mathsf{inv}}$ and these formulas are defined below.

- The formula $\varphi_b \stackrel{\text{def}}{=} \Box(a \equiv (\tau_1@1 \wedge \tau_2@2))$ ensures that every layer includes exactly one event of each agent.
- The formula $\varphi_d \stackrel{\text{def}}{=} \Box(\vee_{i=1,2}(q_f@i) \supset (c_2@i\,\mathbf{U}\,d@i)) \wedge \bigcirc\Box(((d \wedge \tau_i)@i) \wedge (d \wedge \tau_{i^{co}})@i^{co})$ appends dummy events at the end of the finite diagram representing a run of M.
- The formula $\varphi_{\mathsf{init}} \stackrel{\text{def}}{=} \mathsf{init}@1$ where init is as defined in the proof of Theorem 2.3 specifies the initial configuration.
- The formula $\varphi_{\mathsf{fin}} \stackrel{\text{def}}{=} \Diamond(q_f@1 \vee q_f@2)$ asserts that a final configuration is reachable.
- The formula $\varphi_{\mathsf{inv}} \stackrel{\text{def}}{=} \Box(\wedge_{i=1,2}(\tau_i \supset \mathsf{inv}_i)@i)$ where inv_i are defined below.

  $\mathsf{inv}_1 \stackrel{\text{def}}{=} (\mathsf{state} \vee \mathsf{ctr}_1 \vee \mathsf{ctr}_2 \vee \mathsf{dummy}) \wedge \mathsf{trans}_1 \wedge \mathsf{consis}_1$ and $\mathsf{inv}_2 \stackrel{\text{def}}{=} (\mathsf{state} \vee \mathsf{ctr}_1 \vee \mathsf{ctr}_2 \vee \mathsf{dummy}) \wedge \mathsf{trans}_2 \wedge \mathsf{consis}_2$ where trans_i and consis_i are as defined in the proof of Theorem 2.3 and the other sub-formulas are defined below:

  - $\mathsf{state} \stackrel{\text{def}}{=} \bigvee_{q \in Q} (q \wedge \bigwedge_{q' \in Q, q \neq q'} \neg q') \wedge \neg c_1 \wedge \neg c_2 \wedge \neg d.$
  - $\mathsf{ctr}_1 \stackrel{\text{def}}{=} c_1 \wedge (\bigwedge_{q \in Q} \neg q) \wedge \neg c_2 \wedge \neg c_1 z \wedge \neg d.$
  - $\mathsf{ctr}_2 \stackrel{\text{def}}{=} c_2 \wedge (\bigwedge_{q \in Q} \neg q) \wedge \neg c_1 \wedge \neg c_2 z \wedge \neg d.$

– dummy $\stackrel{\text{def}}{=}$ $d \wedge (\bigwedge_{q \in Q} \neg q) \wedge \neg c_1 \wedge \neg c_2 \wedge \neg c_1 z \wedge \neg c_2 z$.

– The formulas trans_i and consis_i are the same as the one used in the proof of Theorem 2.3 where we replace the special proposition r_i^j by $\tau_i \wedge \mathbf{Y}_j\mathsf{True}$ and $\neg r_i^j$ by $\tau_i \wedge \overline{\mathbf{Y}}_j\mathsf{False}$.

We show that φ_M is 2-satisfiable iff a final configuration of M is reachable, thus establishing the undecidability of B-satisfiability. The proof is very similar to that of Theorem 2.3.

Suppose there exists a run ρ of M in which a final configuration is reachable. We will show that φ_M is satisfiable by inductively defining a model M' of φ_M. Let $\rho = (q^0, n_1^0, n_2^0) \to \dots (q^k, n_1^k, n_2^k) \to \dots (q^m, n_1^m, n_2^m)$ where $q^0 = q_0$, $n_1^0 = n_2^0 = 0$ and $q^m = q_f$. We define the Lamport diagram D_M and valuation function V_M exactly as done in the proof of Theorem 2.3. Now define $M' = (D, V)$ where $D = (E, \leq, \phi, \lambda)$ is given by

- $E = E_M \cup \{e_i^1, e_i^2 \mid i \in \mathbb{N}\}$ where $e_i^j \notin E_M$ for all $i \in \mathbb{N}$ and $j = 1, 2$.
- $\phi(e) = \phi_M(e)$ for $e \in E_k$ and $\phi(e_i^j) = j$ for all $i \in \mathbb{N}$.
- $\leq = (\leq_M \cup \{(e_i^1, e_{i+1}^1) \mid i \in \mathbb{N}\} \cup \{(e_i^2, e_{i+1}^2) \mid i \in \mathbb{N}\})^*$.
- It is easy to see that $\leq$ is a total order on the events of agent j of E. Let $e_1, e_2, \dots$ be an enumeration of the events of agent 1 (with respect to $\leq_1$) and let $f_1, f_2, \dots$ be an enumeration of events of agent 2 (with respect to $\leq_2$). Define $\lambda(e_i) = \lambda(f_i) = i$ for all i.

and $V(e) = V_M(e)$ for all $e \in E_M$ and $V(e_i^j) = \{\tau_j, d\}$ for all $i \in \mathbb{N}$ and $j = 1, 2$.

It is easy to see that D is a layered Lamport diagram. The rest of the proof goes along the lines of the proof of Theorem 2.3.

Proof of Theorem 4.2:

The set of propositions are as defined for the undecidability of B-satisfiability. We use the formula α_M as defined for the proof of undecidability of LD_1. In addition, we need to define a layering with the finite Lamport diagram representing a run of M ending in a final configuration as the first layer and each subsequent layer as containing exactly one event per agent. The following formulas capture this layering.

- The formula $\alpha_M@1$ where α_M is the formula defined in the proof of Theorem 2.3 specifies that the first layer is the diagram representing a run of M ending in a final configuration.
- The formulas $\Box(\bigcirc(\bigwedge_{p_1 \in P_1 \setminus \{d, \tau_1\}} (\neg p_1 \wedge d \wedge \tau_1)@1))$ and $\Box(\bigcirc(\bigwedge_{p_2 \in P_2 \setminus \{d, \tau_2\}} (\neg p_2 \wedge d \wedge \tau_2)@2))$ specify that the events of agent i from the second layer onwards are labelled only by the proposition d.
- The formula $\Box(\bigcirc(\bigwedge_i (\tau_i \wedge \overline{\mathbf{X}}_i \neg\mathsf{False})@\mathrm{i}))$ ensures that every layer (from the second layer onwards) includes exactly one event of each agent.

Let φ_M denote the conjunction of all the above formulas. If M has a halting run, then we inductively define a model $M' = (D, V)$ where $D = (E, \leq, \phi, \lambda)$ and E, $\leq$, ϕ and V are as defined in the proof of Theorem 4.1 and λ is given by $\lambda(e) = 1$ for all $e \in E_M$ and $\lambda(e_i^j) = i + 1$ for all $i \in \mathbb{N}$. The rest of the proof follows that of Theorem 2.3.

References

[1] Alur, R., Holzmann, G. J. and Peled, D., "An analyzer for message sequence charts", *Software Concepts and Tools*, 17(2), 1996, 70-77.

[2] Alur, R., Peled, D. and Penczek, W., "Model checking of causality properties", *Proc. LICS*, 1995, 90-100.

[3] Alur, R. and Yannakakis, M., "Model checking of message sequence charts", *Proc. CONCUR 1999*, Springer-Verlag *LNCS 1664*, 1999, 114-129.

[4] Bollig, B. and Leucker, M., "Modelling, specifying and verifying message passing systems", *Proc. TIME 2001*, 2001, 240-247.

[5] Elrad, T. and Francez, N., "Decomposition of distributed programs into communication closed layers", *Science of Computer Programming, 2*, 1982, 155-173.

[6] Gunter, E., Muscholl, A. and Peled, D., "Compositional message sequence charts", *Proc. TACAS 2001*, Springer-Verlag *LNCS 2031*, 2001, 496-511.

[7] Henriksen, J.G., Mukund, M., Narayan Kumar and Thiagarajan, P.S., "Regular collections of message sequence charts", *Proc. MFCS 2000*, Springer-Verlag *LNCS 1893*, 2000, 405-414.

[8] Lamport, L., "Time, clocks and ordering of events in a distributed system", *Communications of ACM 21(7)*, 1978, 558-565.

[9] Lodaya, K., Ramanujam, R. and Thiagarajan, P. S., "Decidability of a partial order based temporal logic", *Proc. ICALP 1993*, Springer-Verlag *LNCS 700*, 1993, 582-592.

[10] Madhusudan, P., "Reasoning about sequential and branching behaviours of message sequence graphs", *Proc. ICALP 2001*, Springer-Verlag *LNCS 2076*, 2001, 396-407.

[11] Meenakshi, B. and Ramanujam, R., "Reasoning about message passing in finite state environments", *Proc. ICALP 2000*, *LNCS 1853*, 2000, 487-498.

[12] Peled, D., "Specification and verification of message sequence charts", *In Proc. IFIP FORTE/PSTV*, 2000, 139-154.

[13] Poel, M. and Zwiers, J., "Layering techniques for development of parallel systems", *Proc. CAV*, 1992, 16-29.

[14] Ramanujam, R., "Locally linear time temporal logic", *Proc. LICS*, 1996, 118-127.

[15] Rudolph, E., Graubmann, P. and Grabowski, J., "Tutorial on message sequence charts", *Computer Networks and ISDN Systems—SDL and MSC, Vol. 28*, 1996, 1629-1641.

[16] Thiagarajan, P.S., "A trace based extension of propositional linear time temporal logic", *Proc. LICS*, 1994, 438-447.

[17] *ITU-TS Recommendation Z.120—Message sequence chart*, ITU-TS, 1997.