Symbolically Bounding the Drift in Time-Constrained MSC Graphs*

S. Akshay[1], Blaise Genest[1,2], Loïc Hélouët[1], and Shaofa Yang[3]

[1] IRISA, INRIA Rennes - ENS Cachan Bretagne - CNRS, France
[2] CNRS, UMI IPAL joint with NUS and A*STAR/I2R, Singapore
[3] SIAT, Chinese Academy of Sciences, China
{akshay,bgenest}@irisa.fr, loic.helouet@inria.fr, sf.yang@siat.ac.cn

Abstract. Verifying systems involving both time and concurrency rapidly leads to undecidability, and requires restrictions to become effective. This paper addresses the emptiness problem for time-constrained MSC-Graphs (TC-MSC graphs for short), that is, checking whether there is a timed execution compatible with a TC-MSC graph specification. This problem is known to be undecidable in general [11], and decidable for some regular specifications [11]. We establish decidability of the emptiness problem under the condition that, for a given K, *no path* of the TC-MSC graph *forces* any node to take more than K time units to complete. We prove that this condition can be effectively checked. The proofs use a novel symbolic representation for runs, where time constraints are encoded as a system of inequalities. This allows us to handle *non-regular specifications* and improve efficiency w.r.t. using interleaved representations.

1 Introduction

In a distributed system, several processes interact to implement a protocol. One way to describe these interactions is through scenarios, formalized using Message Sequence Charts (MSCs) [13]. MSCs describe finite interactions among agents that communicate asynchronously. A protocol is described by allowing choices and repetition of these MSCs. To specify these main characteristics while abstracting away details of implementation, the formal methods community often considers *MSC graphs*, which are directed graphs whose nodes are labeled by MSCs. Protocol specifications also include timing requirements for messages as well as descriptions of how to recover from timeouts. To specify how time and concurrency influence each other, MSCs and MSC graphs have been generalized to *time-constrained MSCs* (TC-MSCs) and *time-constrained MSC graphs* (TC-MSC graphs) [2]. The timing information is captured by adding timing constraints between pairs of events, and transitions have additional timing constraints.

We consider decidability issues for TC-MSC graphs. This is a challenging task due to the presence of both time and concurrency. First, the set of executions of a TC-MSC graph is not regular in general. Even checking whether there exists a timed execution that is consistent with all the constraints of a model

* funded by the French Consulate at Guangzhou, ANR IMPRO, and the DST project.

is non-trivial. This question, called the *emptiness problem*, is undecidable for TC-MSC graphs in general [11]. However, it is decidable for (sequential) timed automata [4]. Extending decidability results to distributed systems has been done in two particular and limited settings. In the first setting [15, 10], clocks are local to a process, and so, one cannot specify time taken by a communication (message or synchronization). This limitation makes the specification formalism very weak. The second setting can relate clocks from different processes and specify how long a communication takes, but the specifications can only exhibit regular behaviors [2, 3, 7, 8, 18], which is a significant restriction in a concurrent setting where even the simple producer-consumer protocol is not regular. To obtain regularity (and hence decidability), these papers restrict the concurrency in a structural way, for instance considering only locally synchronized (see [16, 5, 12]) MSC graphs (in [2, 3]) or only safe Petri Nets (in [7, 8]). In [1], the language is restricted to being representable by a regular set, using both K-drift-boundedness — that we use in this paper and define below — *and* a restriction on Zeno behaviors. Decidability of checking K-drift-boundedness was however left open. Last, the procedures for TC-MSC graphs in [2, 3, 11, 1] construct an interleaved timed automaton, leading to a combinatorial explosion. This could be seen as going against the spirit of MSCs, which try to avoid interleavings. Further, the approaches in [2, 3, 11, 18, 1] add another blow-up in complexity through the use of zone construction [4].

In this paper, we prove a novel decidability result for timed concurrent systems with global clocks having a possibly *non-regular* set of behaviors. We investigate the emptiness problem for TC-MSC graphs, and prove it to be decidable in the setting where a TC-MSC graph is prohibited from *forcing* any TC-MSC appearing along one of its paths to take an arbitrarily long amount of time to complete. More precisely, for a given integer K, for any path ρ of a TC-MSC graph, if there exists at least one execution of ρ, then we require that there exists one in which the occurrence times of any two events from the same TC-MSC differ by at most K. Such a TC-MSC graph is said to be *K-drift-bounded* [1]. We further show that given K, one can effectively test whether a TC-MSC graph G is K-drift-bounded. Both results are established without constructing an interleaved timed automaton or relying on the seminal result on decidability of emptiness of timed automata [4], avoiding both state space explosions. Instead, we translate the set of time constraints of a path into a *symbolic profile*, in the form of a system of inequalities. We show how to manipulate this system symbolically using Fourier-Motzkin elimination [9]. We approximate symbolic profiles by a bounded system of inequalities whose coefficients are integers in $[-K', K']$ for some integer K' depending on G and K. This does not hinder checking consistency of K-drift-bounded TC-MSC graphs. This forms the cornerstone of our decidability results, as finite state automata can keep track of bounded systems of inequalities.

The paper is organized as follows: Section 2 recalls basic definitions. Section 3 discusses drift-boundedness and its relevance. Section 4 shows how to check emptiness for K-drift-bounded TC-MSC graphs and Section 5 shows that checking K-drift-boundedness is decidable, for a given K. Omitted proofs are available in an extended version (`http://perso.crans.org/~genest/AGHY12full.pdf`).

2 Preliminaries

Let $\mathbb{R}_{\geq 0}$ denote the set of non-negative reals, $\mathbb{N}$ the set of integers and $\mathcal{I}$ the collection of open and closed intervals with end points in $\mathbb{N}$ as well as intervals of the form $[c, \infty), (c, \infty)$, where $c \in \mathbb{N}$. Throughout this paper, we fix a finite set $\mathcal{P}$ of processes and let p, q range over $\mathcal{P}$. Let $\Sigma = \{p!q, p?q \mid p, q \in \mathcal{P}, \ p \neq q\}$ be the *communication alphabet*. The letter $p!q$ represents p sending a message to q, while $p?q$ signifies p receiving a message sent by q. We define the map $loc : \Sigma \to \mathcal{P}$ via $loc(p!q) = p = loc(p?q)$, and call $loc(a)$ the *location* of a. We define Message Sequence Charts (MSCs) and time-constrained MSCs (TC-MSCs) as usual. We do not require FIFO ordering among messages.

Definition 1. *An* MSC *is a tuple* $(E, (<_p)_{p \in \mathcal{P}}, \mu, \lambda)$. *The set of events is* E *and* $\lambda : E \to \Sigma$ *labels events with letters. For each* p, $<_p$ *is a total order over* $E_p = \{e \in E \mid loc(\lambda(e)) = p\}$. *The message function* $\mu \subseteq E_S \times E_R$ *is a bijection, such that* $f = \mu(e)$ *implies* $\lambda(e) = p!q$, $\lambda(f) = q?p$ *for some* $p, q \in \mathcal{P}$, *with* $E_S = \{e \in E \mid \exists p, q \in \mathcal{P}, \lambda(e) = p!q\}$ *and* $E_R = \{f \in E \mid \exists p, q \in \mathcal{P}, \lambda(f) = q?p\}$. *We require that the transitive closure* $\leq$ *of* $\lessdot = \bigcup_{p \in \mathcal{P}} <_p \cup \mu$ *is a partial order.*

The relation $\leq$ reflects causal ordering of events. We will write $e < f$ when $e \leq f$ and $e \neq f$. Notice that E_p has a unique $<_p$-maximal event (respectively, minimal event), which we refer to as the last (respectively, first) event of E on p.

Definition 2. *A* TC-MSC *is a tuple* $(E, (<_p)_{p \in \mathcal{P}}, \mu, \lambda, \delta)$ *where* $(E, (<_p)_{p \in \mathcal{P}}, \mu, \lambda)$ *is an MSC and* δ *is a function associating an interval* $\delta(e, e') \in \mathcal{I}$ *to each* $e \lessdot e'$.

For each pair of events $e \lessdot e'$, the interval $\delta(e, e')$ constrains the range in which the difference between the occurrence time of e' and the occurrence time of e can lie. For clarity, we shall refer to occurrence times as *dates*. A TC-MSC T defines a collection of MSCs with dates such that the relative differences of dates fulfill the constraints asserted in T.

Definition 3. *Let* $T = (E, (<_p)_{p \in \mathcal{P}}, \mu, \lambda, \delta)$ *be a TC-MSC. A* dated MSC *generated by* T *is a tuple* $(E, (<_p)_{p \in \mathcal{P}}, \mu, \lambda, d)$ *where* $d : E \to \mathbb{R}^+$ *is such that for each* $e \lessdot e'$, $d(e') - d(e)$ *is in the interval* $\delta(e, e')$.

We denote by $\mathcal{L}(T)$ the set of dated MSCs generated by T. To capture infinite collections of TC-MSCs, we define TC-MSC graphs as in [2, 11], which are finite graphs whose nodes are labeled by TC-MSCs. Each path ρ of a TC-MSC graph G induces a TC-MSC by concatenating TC-MSCs labeling nodes of ρ. Transitions of G are labeled by interval constraints, one for each process, that act as constraints on the timing between the last and first event of each process in consecutive nodes of ρ.

Definition 4. *A* TC-MSC graph *is a structure* $G = (N, \mathcal{T}, \Lambda, n_{in}, N_{fi}, \longrightarrow, \Delta)$ *where* N *is a finite non-empty set of nodes,* $\mathcal{T}$ *a finite set of TC-MSCs,* $\Lambda : N \to \mathcal{T}$ *labels each node with a TC-MSC,* n_{in} *is the initial node,* N_{fi} *the set of final nodes,* $\longrightarrow \subseteq N \times N$ *is the transition relation, and* Δ *is a labeling function which associates an interval* $\Delta_p(n \to n') \in \mathcal{I}$ *to each transition* $n \to n'$ *and each process* p, *such that* $\Delta_p(n \to n') = [0, \infty)$ *if* $\Lambda(n)$ *or* $\Lambda(n')$ *has no event on process* p.

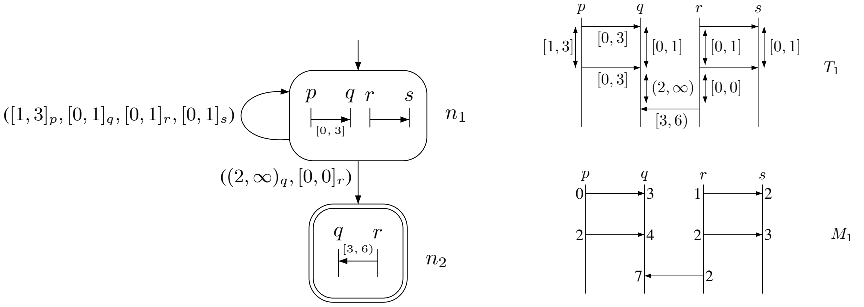


Fig. 1. A TC-MSC graph G_1, a TC-MSC T_1 and a dated MSC $M_1 \in \mathcal{L}(G_1)$.

A *path* ρ of the TC-MSC graph G is a sequence $n_0 n_1 \dots n_\ell$ such that $n_0 = n_{in}$ and $n_i \rightarrow n_{i+1}$ for $i = 0, \dots, \ell - 1$. The path ρ is said to be *final* if $n_\ell \in N_{fi}$. For each $n \rightarrow n'$, the *concatenation* of TC-MSCs $\Lambda(n)$, $\Lambda(n')$ is defined with respect to $\Delta(n \rightarrow n')$, and is denoted $\Lambda(n) \circ \Lambda(n')$. Roughly speaking, this consists of putting $\Lambda(n')$ after $\Lambda(n)$ and for every process p, attaching to the pair (e_p, f_p) the constraint $\Delta_p(n \rightarrow n')$, for e_p the last event of $\Lambda(n)$ on process p and f_p the first event of $\Lambda(n')$ on p. Formally, let $\Lambda(n) = (E, (<_p)_{p \in \mathcal{P}}, \mu, \lambda, \delta)$, $\Lambda(n') = (E', (<'_p)_{p \in \mathcal{P}}, \mu', \lambda', \delta')$. Then $\Lambda(n) \circ \Lambda(n') = (E'', (<''_p)_{p \in \mathcal{P}}, \mu'', \lambda'', \delta'')$ where E'' is the disjoint union of E and E', $<''_p$ is the transitive closure of the union of $<_p$, $<'_p$ and $E_p \times E'_p$, and λ'' is given by: $\lambda''(e) = \lambda(e)$ for $e \in E$, $\lambda''(e) = \lambda'(e)$ for $e \in E'$. We also set $\mu''(e) = \mu(e)$ when $\mu(e)$ is defined, and $\mu''(e) = \mu'(e)$ when $\mu'(e)$ is defined. At last, δ'' is given by: $\delta''(e, f) = \delta(e, f)$ for $e \lessdot f$, $\delta''(e, f) = \delta'(e, f)$ for $e \lessdot' f$. For each p, if both E_p and E'_p are nonempty, we set $\delta''(e_p, f_p) = \Delta_p(n \rightarrow n')$ for e_p the last event of E_p and f_p the first event of E'_p.

We emphasize that by definition, $\Delta_p(n \rightarrow n') = [0, \infty)$ if E_p or E'_p is empty. It follows that for $n \rightarrow n' \rightarrow n''$, $(\Lambda(n) \circ \Lambda(n')) \circ \Lambda(n'')$ is the same as $\Lambda(n) \circ (\Lambda(n') \circ \Lambda(n''))$. Thus, we unambiguously define the TC-MSC T^ρ *induced* by a path $\rho = n_0 \dots n_\ell$ of G to be $\Lambda(n_0) \circ \dots \circ \Lambda(n_\ell)$. A path ρ of G is called *consistent* if $\mathcal{L}(T^\rho) \neq \emptyset$. From now on, we will speak interchangeably of a node n and its associated TC-MSC $\Lambda(n)$. We write $\mathcal{L}(G)$ for the union of $\mathcal{L}(T^\rho)$, ρ ranging over *final* paths of G. We call a dated MSC in $\mathcal{L}(G)$ a *timed execution* of G. An example of a TC-MSC graph G_1 is in Figure 1. The TC-MSC T_1 is induced by path $n_1 \cdot n_1 \cdot n_2$ of G_1, i.e., $T_1 = T^{n_1 \cdot n_1 \cdot n_2}$. Further, M_1 is a dated MSC generated by T_1. As n_2 is final, $M_1 \in \mathcal{L}(G_1)$.

The emptiness problem for TC-MSC graphs is: given a TC-MSC graph G, determine whether $\mathcal{L}(G) = \emptyset$, that is, whether it has no *consistent and final* path. This is a fundamental verification problem that must be addressed. Indeed, a TC-MSC graph with an empty language should be considered ill-specified and such an exception should be caught at an early stage of design. In [11], it is shown that this problem is undecidable in general, and decidable for some regular specifications. We show in the following that checking emptiness for TC-MSC graphs is decidable under an arguably mild restriction on time constraints which does not impose regularity. Furthermore, we will show that one can test whether a given TC-MSC graph satisfies this condition.

3 Drift-Boundedness

In this section we define our mild restriction, namely *drift-boundedness*. Let us fix a TC-MSC graph G. Let $\rho = n_0 \dots n_\ell$ be a consistent path of G and $M = (E, (<_p)_{p\in\mathcal{P}}, \mu, \lambda, d)$ be a dated MSC generated by T^ρ. For an integer K, we say that M is a *K-drift-bounded* dated MSC of ρ iff for each $i = 0, \dots, \ell$, for any two events e, e' in $\Lambda(n_i)$, it is the case that $|d(e) - d(e')| \leq K$. We say that ρ is *K-drift-bounded* iff there *exists* a K-drift-bounded dated MSC in $\mathcal{L}(T^\rho)$. We emphasize that $\mathcal{L}(T^\rho)$ may also contain dated MSCs which *are not* K-drift-bounded. We say that G is K-drift-bounded iff every *consistent* (but not necessarily final) path of G is K-drift-bounded. In other words, for each *consistent* path ρ, we can find a dated MSC in $\mathcal{L}(T^\rho)$ such that the difference between the dates of any two events from the same instance of a node is at most K. Notice that we can have $\mathcal{L}(G) = \emptyset$ even though G is K-drift-bounded. In fact, G is vacuously K-drift-bounded for any K if it has no consistent path.

As an example, consider the TC-MSC graph G_1 from Figure 1. G_1 is 3-drift-bounded since in every timed execution, we can be sure that all events in node n_1 or n_2 can be completed within a delay of 3 time units. But if we change the constraints on the loop on n_1 from $([0,1]_r, [0,1]_s)$ to $([4,5]_r, [1,2]_s)$ then for any integer K, G_1 is not K-drift-bounded. Note that G_1 is not locally synchronized (as defined in [16, 5], and lifted in [3] to a timed setting). In fact, we can simulate the producer-consumer protocol and obtain non-regular behaviors. Thus, this example cannot be handled by the decidability result in [3].

We believe that drift-boundedness is a practical notion. Interpreting a node of a TC-MSC graph as a phase or a transaction of a distributed protocol, we expect any scenario labeling the node to be executable in a bounded time, say K. A protocol specified as a TC-MSC graph that is not K-drift-bounded should thus be considered as ill-formed. Indeed, while a TC-MSC graph specification is usually incomplete (as it abstracts away some events and constraints used in the actual implementation), if it is not K-drift-bounded, then every implementation of this specification will not be K-drift-bounded either.

3.1 The main results

We can now state our main results. The first result establishes the decidability of the emptiness problem for K-drift-bounded TC-MSC graphs.

Theorem 1. *Let $K \in \mathbb{N}$ and G be a K-drift-bounded TC-MSC graph. Then checking whether $\mathcal{L}(G)$ is empty is decidable in PSPACE.*

We next show that the drift-boundedness hypothesis of Theorem 1 can be effectively checked, giving rise to an effective decidability procedure.

Theorem 2. *Let $K \in \mathbb{N}$ and G be a TC-MSC graph. Then checking whether G is K-drift-bounded is decidable in PSPACE.*

We can show that the decidability result in Theorem 2 is in fact at the boundary of undecidability. Recall that the definition of K-drift-bounded considers every path of a TC-MSC graph, including paths that cannot be extended to consistent final paths. Instead, if we consider the problem of checking whether every consistent *final* path of a TC-MSC graph is K-drift-bounded, this turns out to be undecidable. We assume K fixed for the next proposition.

Proposition 1. *It is undecidable, given a TC-MSC graph G, to determine whether every consistent final path of G is K-drift-bounded.*

Proof. The proof is by a reduction from the emptiness problem of TC-MSC graphs, shown undecidable in [11]. Let G be a TC-MSC graph. We construct another TC-MSC graph G' from G such that there does *not* exist a consistent final path of G iff every consistent final path of G' is K-drift-bounded, which shows the result. G' is obtained from G with the following modifications. Firstly, add a new node n_{new} and for every final state n_f of G, add a transition (n_f, n_{new}). Secondly, define the set of final nodes of G' to be the singleton set $\{n_{new}\}$. Thirdly, n_{new} is labeled with a TC-MSC consisting of a single message (e, f) from p to q. The time constraint on (e, f) is $[K+1, K+1]$. Lastly, for every final state n_f of G and every process, the time constraint of transition (n_f, n_{new}) is $[0, \infty)$. If there does not exist a consistent final path of G, then there does not exist a consistent final path of G', and it is vacuously true that every consistent final path of G' is K-drift-bounded. On the other hand, assume that there exists some consistent final path ρ of G. Then $\rho \cdot n_{new}$ is a consistent final path of G' (timing of a consistent dated MSC of ρ can be easily extended). But it is not K-drift-bounded because of the constraint $[K+1, K+1]$ on the last node n_{new} of the path, which impose e, f to be $K+1$ time units away. Hence not every consistent final path of G' is K-drift-bounded. □

Next, we introduce *full* TC-MSC graphs and show that any TC-MSC graph can be transformed into a *full* TC-MSC graph, while preserving consistency and drift-boundedness of paths. This enables us to check both the emptiness of a K-drift-bounded TC-MSC graph G, and the K-drift-boundedness of any TC-MSC graph G, by working with a full TC-MSC graph constructed from G.

3.2 Full TC-MSC Graphs

We call a TC-MSC graph G *full* if each node of G has at least one event on each process $p \in \mathcal{P}$. We will now show how to "augment" a TC-MSC graph G to obtain a full TC-MSC graph $\widehat{G}$ by adding "dummy events" to nodes of G. For notational convenience, we assume that TC-MSCs may contain internal events. We denote by $p(int)$ the label of such an internal event on process $p \in \mathcal{P}$.

Given $G = (N, \mathcal{T}, \Lambda, n_{in}, N_{fi}, \longrightarrow, \Delta)$, the *augmented graph of G* is defined as $\widehat{G} = (N, \widehat{\mathcal{T}}, \widehat{\Lambda}, n_{in}, N_{fi}, \longrightarrow, \Delta)$ differing only in the labeling set of "augmented" TC-MSCs and the labeling function assigning nodes to them. More precisely, any TC-MSC $T = (E, (<_p)_{p\in\mathcal{P}}, \mu, \lambda, \delta)$ in $\mathcal{T}$ is replaced by the TC-MSC $\widehat{T} = (E', (<_p)_{p\in\mathcal{P}}, \mu, \lambda', \delta)$ in $\widehat{\mathcal{T}}$ where E' is obtained from E by adding a new event e_p with $\lambda(e_p) = p(int)$ for each process p such that $E_p = \emptyset$. Every $<_p$ and δ are unchanged, so e_p is an isolated point in the partial order $\leq$. Such events e_p will be called *dummy events*. Events already in $\Lambda(n)$ will be called *concrete events*. Note that $\Delta(n \rightarrow m)$ is unchanged for each transition $n \rightarrow m$. In particular, recall that for each transition (n, m) in G, if either n or m has no concrete event on p, then $\Delta_p(n, m) = [0, \infty)$. For each $\Lambda(n) = T$, we set $\widehat{\Lambda}(n) = \widehat{T}$. Obviously, $\widehat{G}$ is full for any G.

Let H be any full TC-MSC graph with events partitioned as dummy or concrete. That is, in every TC-MSC labeling a node of H, there is a mapping from the set of events to $\{dummy, concrete\}$. For instance, $\widehat{G}$ is such a full TC-MSC graph. Let $Y \leq Y' \in \mathbb{N}$. Now, for a path $\rho = n_0 \dots n_\ell$ of H, we say that a dated MSC $M = (E, (<_p)_{p \in \mathcal{P}}, \mu, \lambda, d)$ generated by T^ρ is (Y, Y')*-drift-bounded* if for each $i = 0, \dots, \ell$, for any two events e, f in the TC-MSC $\Lambda(n_i)$, we have: (i) if both e and f are concrete events, then $|d(e) - d(f)| \leq Y$; (ii) if one or both of e, f are dummy events, then $|d(e) - d(f)| \leq Y'$. We say that a consistent path ρ of H is (Y, Y')*-drift-bounded* if there exists a (Y, Y')-drift-bounded dated MSC generated by ρ. At last, H is (Y, Y')-drift-bounded if all its *consistent* paths are.

Proposition 2. *For a TC-MSC graph G, a path ρ of G and $K \in \mathbb{N}$, (i) ρ is consistent in G iff $\widehat{\rho}$ is consistent in $\widehat{G}$, (ii) ρ is K-drift-bounded in G iff $\widehat{\rho}$ is $(K, \widehat{K})$-drift-bounded in $\widehat{G}$, with $\widehat{K} = (|\mathcal{P}| - 1) \cdot K$.*

Hence, we are able to restrict to full TC-MSC graphs when checking for emptiness using (i), and when checking for K-drift boundedness using (ii):

Corollary 1. *Given a TC-MSC graph G, (i) $\mathcal{L}(G) \neq \emptyset$ iff $\mathcal{L}(\widehat{G}) \neq \emptyset$, and (ii) G is K-drift-bounded iff $\widehat{G}$ is $(K, \widehat{K})$-drift-bounded, where $\widehat{K} = (|\mathcal{P}| - 1) \cdot K$.*

4 Emptiness for *K*-Drift-Bounded TC-MSC Graphs

We now prove Theorem 1. We assume G to be a K-drift-bounded TC-MSC graph. By Corollary 1, we can build $\widehat{G}$, a $(K, \widehat{K})$-drift-bounded full TC-MSC graph with $\mathcal{L}(\widehat{G}) \neq \emptyset$ iff $\mathcal{L}(G) \neq \emptyset$. It then suffices to check the emptiness of a finite automaton that accepts the set of $(K, \widehat{K})$-drift-bounded *final* paths of $\widehat{G}$.

Let H be a full TC-MSC graph, with events partitioned as dummy or concrete. To avoid clutter, we assume that constraints in H are only of the form $[a, b]$ and $[a, \infty)$. Extending proofs to handle other constraints is straightforward and all statements hold in general, but additional notations are needed to remember whether each inequality is strict or not. We first describe intuitively the key ingredients of the proof, which will be developed in the rest of this section.

- First, we observe that checking consistency of a path ρ of H, i.e., $\mathcal{L}(T^\rho) \neq \emptyset$, is equivalent to checking for the existence of a solution to a system of inequalities over (real-valued) variables x_e depicting the dates of events e of T^ρ.
- Next, we show that checking whether a dated MSC can be extended by a node by assigning appropriate dates to events of this node can be done with information only on the relative difference of dates of the last event of the

dated MSC on each process. This motivates us to associate a symbolic profile $PF(\rho)$ to each path ρ. A symbolic profile is a system of inequalities whose solutions correspond to the dates of final events of dated MSCs generated by T^ρ, and vice versa. In particular, $PF(\rho)$ has a solution iff ρ is consistent.

- We remark that constants appearing in symbolic profiles can be chosen as integers. Restricting constants to be within $[-\widehat{K}, \widehat{K}]$ does not exclude any consistent $(K, \widehat{K})$-*drift-bounded* path of H. We can then represent with a finite automaton the set of consistent $(K, \widehat{K})$-*drift-bounded* paths of H.

Systems of inequalities and Fourier-Motzkin elimination. We first fix basic terminologies for systems of difference inequalities. Let X be a finite nonempty set of real-valued variables. A *(difference) inequality* is an inequality of the form $x - y \leq a$, where x, y are two different variables in X.

Definition 5. *A* system of (difference) inequalities ϕ *over* X *is* $\wedge_{(x,y) \in R}\, x - y \leq a_{xy}$ *where* $R \subseteq X \times X$ *is an irreflexive relation. We say that* ϕ *has* integral coefficients *whenever* a_{xy} *is a* (possibly negative) integer *for all* $(x, y) \in R$.

From now on, we assume that the system is *simplified*, that is, for each $x, y \in X$, there is at most one inequality of the form $x - y \leq a$. This involves no loss of generality as $x - y \leq a \wedge x - y \leq a'$ is equivalent with $x - y \leq min(a, a')$. If $x - y \leq a$ appears in ϕ, we say that ϕ contains an *edge* (x, y), and the weight of this edge is a. We say that two systems ϕ, ψ of inequalities are *equivalent* when ϕ has a solution (in the real domain) iff ψ has a solution (in the real domain).

A key idea is to propagate constraints concerning variables in a subset $Y \subsetneq X$ on variables in $X \setminus Y$, and then safely remove variables in Y while keeping an equivalent system. This is done using the *Fourier-Motzkin* elimination method (see extended version, or [9, 14]).

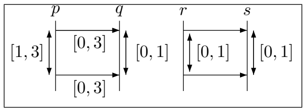
For $F \subseteq X$, let $\phi_{|F}$ denote the (unique) system of inequalities over variables F obtained by performing Fourier-Motzkin elimination of variables in $X \setminus F$ following a fixed order. We have that ϕ and $\phi_{|F}$ are equivalent. If ϕ has *integral coefficients*, then so does $\phi_{|F}$.

Symbolic Profiles. Let $T^\rho = (E, (<_p), \mu, \lambda, \delta)$ be the TC-MSC associated with some path $\rho = n_0 \dots n_\ell$ of H. We denote by x_e a $\mathbb{R}_{\geq 0}$-valued variable, standing for the date of event $e \in E$, and let $X_E = \{x_e \mid e \in E\}$. We associate path ρ with a system of linear inequalities $\Phi(\rho)$ *with integral coefficients* as follows:

Definition 6. *The system* $\Phi(\rho)$ *associated with* ρ *is the smallest system of inequalities over the set of variables* X_E *such that, for any* $e, f \in E$ *with* $e \lessdot f$,

- *if* $\delta(e, f) = [L, U]$, *then* $\Phi(\rho)$ *contains both* $x_f - x_e \leq U$ *and* $x_e - x_f \leq -L$;
- *if* $\delta(e, f) = [L, \infty)$, *then* $\Phi(\rho)$ *contains* $x_e - x_f \leq -L$.

We easily have that ρ is consistent iff $\Phi(\rho)$ has a solution. Let e_p be the last event of T^ρ on p, for each process p. Let E_{last} be the set $\{e_p \mid p \in \mathcal{P}\}$. Using Fourier-Motzkin elimination of variables $X' = \{x_e \mid e \notin E_{last}\}$, we obtain a



$$= T^{n_1 \cdot n_1}, \quad \phi(n_1 \cdot n_1)|_{X_{last}} = \left\{ \begin{array}{lcl} x_q - x_p & \leq & 3 \\ x_p - x_q & \leq & 0 \\ x_r - x_s & \leq & 0 \end{array} \right\}$$

Fig. 2. The TC-MSC induced by path $n_1 \cdot n_1$ of G_1 and its profile

system $\Phi(\rho)|_{X_{last}}$ over variables $X_{last} = \{x_e \mid e \in E_{last}\}$, with integral coefficients, equivalent with $\Phi(\rho)$. Once simplified, this system has at most $|\mathcal{P}|^2$ inequalities with integral coefficients. We encode this system as a *symbolic profile*.

Definition 7. *A* symbolic profile *σ is a function from $\mathcal{P} \times \mathcal{P}$ to $\mathbb{Z} \cup \{\infty\}$. We denote by $\mathcal{PF}$ the (infinite) set of all profiles.*

Notice that symbolic profiles are *syntactically* similar to Difference Bounded Matrices (DBMs) [6] over $|\mathcal{P}|$ clocks. However, unlike a DBM, a symbolic profile may not correspond to a timed linearization, and the update function (defined below) is very different when compared to DBMs.

Let ϕ be a system of inequalities with integral coefficients over $X_{last} = \{x_p \mid p \in \mathcal{P}\}$. We define the symbolic profile $PF(\phi)$ induced by ϕ as $PF(\phi)[p,q] = a_{pq}$ if $x_p - x_q \leq a_{pq}$ belongs to ϕ, and $PF(\phi)[p,q] = \infty$ otherwise. Intuitively, $PF(\phi)[p,q] = \infty$ means that there is no inequality of the form $x_p - x_q \leq a_{pq}$ in ϕ. We abusively use $PF(\phi)$ as a system of inequalities in the following, and denote x_p for x_{e_p}. For a path ρ, we denote $PF(\rho) = PF((\Phi(\rho))|_{X_{last}})$. We say that a symbolic profile $\sigma \in \mathcal{PF}$ is *satisfiable* if it has a solution. It is easy to check whether $PF(\rho)$ is satisfiable, either by using Fourier-Motzkin elimination till reaching a trivial equation, or by using Shostak characterisation [17].

Proposition 3. *$PF(\rho)$ is satisfiable iff ρ is consistent.*

As an example, consider the TC-MSC $T^{n_1 \cdot n_1}$ in Figure 2, generated by path $n_1 \cdot n_1$ of G_1 from Figure 1. Let e_j^i denote the i^{th} event on process j and E be the set of events of $T^{n_1 \cdot n_1}$. We obtain $\Phi(n_1 \cdot n_1)$ to be the set of inequalities over $X = \{x_e \mid e \in E\}$, where for instance the inequations $x_{e_p^2} - x_{e_p^1} \leq 3$ and $x_{e_p^1} - x_{e_p^2} \leq -1$ capture the timing constraint $[1,3]$ between e_p^1 and e_p^2. Now eliminating variables $x_{e_p^1}, x_{e_q^1}, x_{e_r^1}, x_{e_s^1}$ results in a set of equations on $X_{last} = \{x_{e_p^2}, x_{e_q^2}, x_{e_r^2}, x_{e_s^2}\} = \{x_p, x_q, x_r, x_s\}$ as shown. E.g., $PF(n_1 \cdot n_1))[p,q] = \min(3, -1+3+1) = 3$ and $PF(n_1 \cdot n_1))[s,r] = \infty$. This system of inequalities has many solutions.

Bounded profiles. Notice that the set of symbolic profiles as defined above is not finite in general (the coefficients range over $\mathbb{Z}$), and so, it cannot be recorded by a finite state automaton. Instead, we use the *finite set* of L-bounded profiles, where $L \in \mathbb{N}$ is some integer.

Definition 8. *For $L \in \mathbb{N}$, a L-*bounded profile *σ is a function from $\mathcal{P} \times \mathcal{P}$ to $\mathbb{Z} \cap [-L, L]$. We denote by $\mathcal{PF}_L$ the set of L-bounded profiles.*

Let $Y \leq Y' \in \mathbb{N}$. Notice that the set $\mathcal{PF}_{Y'}$ is finite. We denote by $\Phi_{Y,Y'}(\rho)$ the system of inequalities obtained from $\Phi(\rho)$ by the following modification: for each $i = 0, \dots, \ell$, for any two different events e, f in the same node n of ρ, if $\Phi(\rho)$ contains $x_e - x_f \leq a_{e,f}$, then replace it by $x_e - x_f \leq min(a_{e,f}, Y)$ if both e, f are *concrete*, and by $x_e - x_f \leq min(a_{e,f}, Y')$ otherwise (that is if at least one of e or f is *dummy*); if $\Phi(\rho)$ does not have an edge (e, f), then add the inequality $x_e - x_f \leq Y$ if both e, f are *concrete*, and $x_e - x_f \leq Y'$ otherwise. Clearly, ρ is consistent and (Y, Y')-drift-bounded iff $\Phi_{Y,Y'}(\rho)$ has a solution. If $\Phi_{Y,Y'}(\rho)$ has a solution, we set $PF_{Y,Y'}(\rho) = PF(\Phi_{Y,Y'}(\rho)|_{X_{last}})$. In a full TC-MSC graph H, by definition of $\Phi_{Y,Y'}(\rho)$, we have $PF_{Y,Y'}(\rho) \in \mathcal{PF}_{Y'}$. If $\Phi_{Y,Y'}(\rho)$ has no solution, it is possible that $PF(\Phi_{Y,Y'}(\rho)|_{X_{last}}) \notin \mathcal{PF}_{Y'}$. In this case, we set $PF_{Y,Y'}(\rho)$ to be a particular profile $\bot \in \mathcal{PF}_{Y'}$ without solution, e.g. $\bot[p, q] = 0, \bot[q, p] = -1$ (which would require $1 \leq x_p - x_q \leq 0$).

Proposition 4. *Let ρ be a path of a full TC-MSC graph H. Then $PF_{Y,Y'}(\rho) \in \mathcal{PF}_{Y'}$, and $PF_{Y,Y'}(\rho)$ is satisfiable iff ρ is consistent and (Y, Y')-drift-bounded.*

Notice that $PF_{Y,Y'}(\rho)$ *cannot* be obtained from $PF(\rho)$. An intuitive (but wrong) idea would be to set $PF_{Y,Y'}(\rho)[p, q] = Y'$ for all $PF(\rho)[p, q] > Y'$ and else $PF_{Y,Y'}(\rho)[p, q] = PF(\rho)[p, q]$. However, setting $PF_{Y,Y'}(\rho)[p, q] = Y'$ for all $PF(\rho)[p, q] > Y'$ only constrains the dates of the last events on each process. So, the bound Y' in $\Phi_{Y,Y'}(\rho)$ must be imposed for every node of ρ, and these constraints on past nodes can have implications for the profile of ρ.

We now explain how to compute $PF_{Y,Y'}(\rho)$ in an inductive way, by defining an extension function $\theta^{n^- \to n}_{Y,Y'} : \mathcal{PF}_{Y'} \to \mathcal{PF}_{Y'}$ for all transitions $n^- \to n$. For $\sigma \in \mathcal{PF}_{Y'}$ and a transition $n^- \to n$, we define the profile $\theta^{n^- \to n}_{Y,Y'}(\sigma)$ as follows:

- Form the system $\Psi = \psi_\sigma \wedge \psi_{n^- \to n} \wedge \psi_n$ over $X = \{x_p \mid p \in \mathcal{P}\} \cup \{x_e \mid e \in E_n\}$ (x_p represents the date of process p in σ, E_n the events of T^n), where:
  - ψ_σ consists of $x_p - x_q \leq \sigma[p, q]$ for every $p, q \in \mathcal{P}$, such that $\sigma[p, q] \neq \infty$.
  - $\psi_{n^- \to n}$ contains, for each p with $\Delta_p(n^- \to n) = [L, U]$, two inequalities $x_p - x_{f_p} \leq -L$ and $x_{f_p} - x_p \leq U$, where f_p is the first event of n on p. For each p with $\Delta_p(n^- \to n) = [L, \infty)$, $\psi_{n^- \to n}$ contains $x_p - x_{f_p} \leq -L$.
  - ψ_n is $\Phi_{Y,Y'}(n)$, the system associated with the singleton path n.
- Perform Fourier-Motzkin elimination on Ψ to remove all variables but $\{x_{\hat{e}_p}\}_{p \in \mathcal{P}}$ where $\hat{e}_p$ is the last event of $\rho \cdot n$ on p. Denote by Π the resulting system (after simplification) of inequalities over $\{x_{\hat{e}_p} \mid p \in \mathcal{P}\}$. Set $\theta^{n^- \to n}_{Y,Y'}(\sigma) = PF(\Pi)$. If at any stage of Fourier-Motzkin elimination, the system is not satisfiable, then set $\theta^{n^- \to n}_{Y,Y'}(\sigma)$ to be the un-satisfiable profile $\bot \in \mathcal{PF}_{Y'}$.

Lemma 1. *For a path ρ ending in n^- and a transition $n^- \to n$, we have that $PF_{Y,Y'}(\rho \cdot n)$ and $\theta^{n^- \to n}_{Y,Y'}(PF_{Y,Y'}(\rho))$ have the same set of solutions.*

Construction of a Symbolic Automaton. We now construct a symbolic automaton $\mathcal{A}(H)$ accepting the final (Y, Y')-drift-bounded paths of H.

Proposition 5. *Let H be a full TC-MSC graph with $|H|$ nodes. Then there exists an automaton $\mathcal{A}(H)$ with at most $|H| \times (2 \cdot Y' + 1)^{|\mathcal{P}|^2}$ states, such that $\mathcal{L}(\mathcal{A}(H)) \neq \emptyset$ iff H has a (consistent) final (Y, Y')-drift-bounded path.*

Proof (sketch). The states of $\mathcal{A}(H)$ are pairs (n, σ), with n a state of H and $\sigma \in \mathcal{PF}_{Y'}$. The initial state is $(n_{in}, PF_{Y,Y'}(n_{in}))$. A state (n, σ) is final if n is final, and σ is satisfiable. There is a transition labeled by n' from (n, σ) to (n', σ') iff both σ, σ' are satisfiable, there is a transition from n to n', and $\sigma' = \theta^{n \to n'}_{Y,Y'}(\sigma)$. The proof now follows from Lemma 1 and Proposition 4. □

The proof of Theorem 1 follows from this: as every path of $\widehat{G}$ is $(K, \widehat{K})$-drift-bounded, taking $H = \widehat{G}, Y = K, Y' = \widehat{K}$ implies $\mathcal{L}(\mathcal{A}(\widehat{G})) \neq \emptyset$ iff $\mathcal{L}(\widehat{G}) \neq \emptyset$ (iff $\mathcal{L}(G) \neq \emptyset$ by Corollary 1). Now, checking that $\mathcal{L}(\mathcal{A}(\widehat{G})) \neq \emptyset$ is decidable in space logarithmic in $|G|, K$ and polynomial in $|\mathcal{P}|$.

Compared with [3], which builds an automaton accepting every timed linearizations of a regular TC-MSC graph, we end up with a much smaller automaton in the worst case (exponential in $|\mathcal{P}|^2$ instead of exponential in $|G|$ for [3]). Further, being symbolic, we believe that the worst case is seldom reached, contrary to constructions based on zones of timed automata [3, 1, 2, 18]. Indeed, consider a path ρ made of one node, labeled by a TC-MSC with one event e_p for every $p \in \mathcal{P}$, and without constraints, hence allowing events to occur at any date. Without symbolic encoding, this path would give rise to $|2K|^{|\mathcal{P}|}$ configurations of the form $(x_p)_{p \in \mathcal{P}}$, with $x_p \in \{0, (0, 1), 1, \cdots, K\}$ the clock associated with e_p. Our solution only memorizes the unique symbolic profile $PF_{K,\widehat{K}}(\rho)$ such that $\forall p, q \in \mathcal{P}$, $PF_{K,\widehat{K}}(\rho)[p, q] = \widehat{K}$, meaning that $-\widehat{K} \leq x_p - x_q \leq \widehat{K}$ for all p, q.

5 Checking K-Drift-Boundedness of TC-MSC Graphs

The construction of automaton $\mathcal{A}(\widehat{G})$ in Section 4 allows to decide the emptiness of $\mathcal{L}(\widehat{G})$ (and hence of $\mathcal{L}(G)$), under the hypothesis that G is K-drift-bounded. We show here that given K, one can decide whether G is K-drift-bounded. We use Proposition 2 to create a full TC-MSC graph $\widehat{G}$. The main idea is that if $\widehat{G}$ is not $(K, \widehat{K})$-drift-bounded, then there must be a path of "minimal" length which is consistent but not $(K, \widehat{K})$-drift-bounded. The idea is then to look for such a *minimal witness*. We call a path $\rho \cdot n$ of $\widehat{G}$ *a minimal witness* iff:

1. The path ρ is $(K, \widehat{K})$-drift-bounded, and
2. The path $\rho \cdot n$ is not $(K, \widehat{K})$-drift-bounded, and
3. The path $\rho \cdot n$ is consistent.

Remark 1. G is not K-drift-bounded iff $\widehat{G}$ is not $(K, \widehat{K})$-drift-bounded iff there exists a minimal witness in $\widehat{G}$.

Now we build a finite automaton recognizing exactly the set of minimal witnesses of $\widehat{G}$ which from the remark above immediately proves Theorem 2. Requirements 1. and 2. are easy to check with the automaton built in the previous section. Requirement 3. is harder to check on its own as there is no effectively constructible finite state automaton accepting all consistent paths, (since it is undecidable to know whether there exists a consistent final path [11]). However, we will prove that thanks to requirement 1., requirement 3. can be replaced by: the path $\rho \cdot n$ is consistent and K_2-drift-bounded, for some contant K_2 depending on G and $\widehat{K}$. Notice that fixing $K_2 = \widehat{K}$ may not be enough.

The bound K_2 is chosen as follows. For a node n in $\widehat{G}$, set D_n to be the sum of lower bounds of $\delta(e, f)$, for every pair (e, f) with $e \lessdot f$. For a transition (n, n') in $\widehat{G}$, set $D_{(n,n')}$ to be the sum of the lower bounds of $\Delta_p(n, n')$ for p ranging over $\mathcal{P}$. Set $D(\widehat{G})$ to be the maximum of $D_{(n,n')} + D_{n'}$ where (n, n') ranges over transitions of $\widehat{G}$. Finally, we let $K_2 = (|\mathcal{P}|/2 + 1) \cdot \widehat{K} + D(\widehat{G})$.

Proposition 6. *Let $\rho \cdot n$ be a path of $\widehat{G}$ such that ρ is $(K, \widehat{K})$-drift-bounded. Then $\rho \cdot n$ is consistent iff $\Phi_{K_2,K_2}(\rho \cdot n)$ is satisfiable.*

The technical proof uses the characterization of consistent systems of equations given by Shostak lemma [17], which we explain now.

Recall that consistency of a path ρ in $\widehat{G}$ is equivalent to satisfiability of the associated system of inequalities $PF(\rho)$. Let φ be a (simplified) system of inequalities. A *cycle* in φ is a sequence $x_1 \dots x_m$ such that for all $i \in \{1, \dots, m-1\}$, $x_{i+1} - x_i \leq a_i$ appears in ϕ for some a_i, and $x_m = x_1$. The *weight* of this cycle is $\sum_{i \in \{1,\dots,m-1\}} a_i$. A cycle is simple if all variables, except the first and last one, are pairwise distinct. According to Shostak lemma [17], *a system of inequalities φ has a solution iff every cycle in φ has non-negative weight iff every* simple *cycle in φ has non-negative weight.* Detection of cycles of negative weight can be efficiently performed with the Bellman-Ford algorithm.

Proof (of Prop. 6.). We will consider three systems of inequalities.

1. The first one is $\phi_1 = \Phi(\rho \cdot n)$.
2. The second one is $\phi_2 = \Phi_{K_2,K_2}(\rho \cdot n)$. By definition, ϕ_2 is obtained from ϕ_1 by adding inequalities $x_e - x_f \leq K_2$ for all e, f from the same node of $\rho \cdot n$.
3. Finally, $\phi_3 = \Phi_{K,\widehat{K}}(\rho)$. Since $K \leq \widehat{K} \leq K_2$, ϕ_3 can be obtained from ϕ_2 by deleting the events from n, and adding inequalities $x_e - x_f \leq K$ for all concrete e, f from the same node of ρ, and adding inequalities $x_e - x_f \leq \widehat{K}$ for all events e, f from the same node of ρ s.t. e or f or both are dummy.

We know that $\rho \cdot n$ is consistent iff ϕ_1 is satisfiable. Hence, we just need to prove that ϕ_2 has a solution iff ϕ_1 has a solution to yield the statement of the proposition. Clearly, if ϕ_2 has a solution, then this solution is also a solution for ϕ_1. Conversely, assume that ϕ_1 has a solution. By Shostak lemma, it implies that every cycle in ϕ_1 has weight at least 0. Now to prove that ϕ_2 has a solution, it suffices to show that every simple cycle of ϕ_2 has weight at least 0. Let $x_1 \dots x_m$ be a simple cycle in ϕ_2. That is, for all $i \in \{1, \dots, m-1\}$, $x_{i+1} - x_i \leq b_i$ appears in ϕ_2 for some b_i, and $x_m = x_1$. We want to prove that $\sum_i b_i \geq 0$.

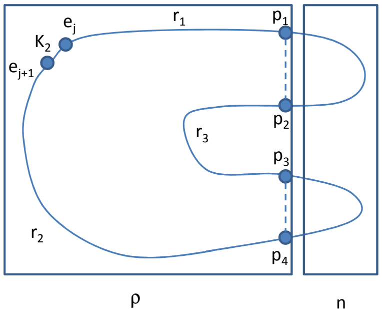
Let a_i be the associated coefficients in ϕ_1, i.e such that there is an inequality in ϕ_1 of the form, $x_{i+1} - x_i \leq a_i$ (if a_i does not exist, fix $a_i = +\infty$). Let c_i be the associated coefficients of ϕ_3 (we fix $c_i = -\infty$ if it corresponds to events in n, i.e., events not represented in ϕ_3).

Observe that $c_i \leq b_i \leq a_i$ by definition of ϕ_1, ϕ_2, ϕ_3. Now, if $a_i = b_i$ for all i, then the cycle $x_1 \dots x_m$ in ϕ_2 is also a cycle in ϕ_1 and $\sum_i b_i = \sum_i a_i$. As every cycle in ϕ_1 has weight at least 0, we are done. Else, we have $a_j \neq b_j$ for some j. Let $J \neq \emptyset$ be the set of indices j such that $a_j \neq b_j$. Hence $|J| \geq 1$. Further, e_j, e_{j+1} are in the same node m of $\rho \cdot n$ for all $j \in J$, because ϕ_2 only adds constraints on pairs of events of the *same node*. Last, $b_j = K_2$ for all $j \in J$, as the only additional constraints in ϕ_2 w.r.t. ϕ_1 are of the form $x_e - x_f \leq K_2$.

Now, we partition the indices $\{1, \dots, m\} = I_\rho \cup I_n \cup J_\rho \cup J_n$ where,
$J_\rho = \{j \mid b_j = K_2$ and both x_j and x_{j+1} belong to $\rho\}$,
$J_n = \{j \mid b_j = K_2$ and at least one of x_j or x_{j+1} belongs to $n\}$.
$I_n = \{j \mid b_j \neq K_2$ and at least one of x_j or x_{j+1} belongs to $n\}$, and
$I_\rho = \{j \mid b_j \neq K_2$ and both x_j and x_{j+1} belong to $\rho\}$.

With this $\sum_i b_i = \sum_{i \in I_n} b_i + \sum_{i \in J_n} b_i + \sum_{i \in I_\rho} b_i + \sum_{i \in J_\rho} b_i$. Now, observing that $J = J_\rho \cup J_n$, we have $\sum_{i \in (J_n \cup J_\rho)} b_i = K_2 \cdot (|J_\rho| + |J_n|) = K_2 \cdot |J| \geq K_2 \cdot 1$. Further, we also have $\sum_{i \in I_n} b_i \geq -D(\widehat{G})$ by definition of $D(\widehat{G})$ and because the cycle is simple. Now, we bound the sum $\sum_{i \in I_\rho} b_i$ (the remaining weights) using ϕ_3. Indeed, since each $i \in I_\rho$ is an index such that x_i and x_{i+1} are events of ρ, we have $b_i \geq c_i$ where c_i is the coefficient of ϕ_3. And therefore it suffices to bound $\left(\sum_{i \in I_\rho} c_i\right)$. It immediately yields the bound for $\sum_i b_i$.

For this, the set I_ρ is first partitioned into *pieces*. Each *piece* $I' \subseteq I_\rho$ is made of "consecutive" indices, i.e., either $I' = \{i, i+1 \dots, j\}$ or $I' = \{i, \dots, m, 1, \dots, j\}$, such that ($e_{i-1} \in n$ or $b_i = K_2$) and ($e_{j+1} \in n$ or $b_j = K_2$). There are at most $|J_\rho| + |\mathcal{P}|/2$ pieces (because the cycle is simple). Each piece begins and ends either with the last event on some process of the node before n or with an event e_i or e_{i+1} such that $b_i = K_2$.



For instance, the picture above depicts a cycle (in ϕ_2) with 3 pieces r_1, r_2, r_3, involving 4 processes. r_1 begins with the last event on some process p_1 of ρ and ends with an event e_j such that $b_j = K_2$. r_2 begins with e_{j+1} and ends with the last event on some process p_4 of ρ. r_3 begins and ends with the last events on some processes p_2, p_3 of ρ.

As ρ is $(K, \widehat{K})$-drift-bounded and consistent, we know that ϕ_3 has a solution, that is every cycle in ϕ_3 has weight at least 0 by Shostak lemma. Let $I_1, \cdots, I_r$ be the pieces of I_ρ. Recall that $r \leq |J_\rho| + |\mathcal{P}|/2$. For all $i \leq r$, denoting $I_i = \{s, \ldots, t\}$, we rename $e_s \cdots e_t$ into $y_1^i \cdots y_{m^i}^i$. We now build a cycle of ϕ_3 using every piece, and with some additional edges connecting these pieces. More precisely, we define $\xi = y_1^1 \cdots y_{m^1}^1 \cdots y_1^r \cdots y_{m^r}^r y_1^1$, made by gluing all the pieces together. Comparing the weight of ξ with $\sum_{i \in I_\rho} c_i$, there is an additional weight d_i in ξ with $y_1^{i+1} - y_{m^i}^i \leq d_i$, for each i. We have that both $y_{m^i}^i$ and y_1^{i+1} are in the same node (either the last node before n, or some node where there were a K_2 edge). In ϕ_3, there is an edge between any two events of the same node, hence this connecting edge $y_1^{s+1} - y_{m^s}^s \leq c_s$ exists (that is ξ is a cycle), and $c_s \leq \widehat{K}$, by definition of ϕ_3. By Shostak lemma, the weight w of ξ in ϕ_3 is at least 0. We thus have $\left(\sum_{i \in I_\rho} c_i\right) + (|\mathcal{P}|/2 + |J_\rho|) \cdot \widehat{K} \geq w \geq 0$. We then have $\sum_{i \in I_\rho} b_i \geq \sum_{i \in I_\rho} c_i \geq -(|\mathcal{P}|/2 + |J_p|) \cdot \widehat{K}$. Thus, we get $\sum_i b_i \geq K_2 \cdot (|J_\rho| + |J_n|) - (|\mathcal{P}|/2 + |J_\rho|) \cdot \widehat{K} - D(\widehat{G}) = K_2 + (|J_\rho| + |J_n| - 1) \cdot K_2 - (D(\widehat{G}) + |\mathcal{P}|/2K) - |J_\rho|) \cdot \widehat{K} = \widehat{K} + (|J_\rho| + |J_n| - 1) \cdot K_2 - |J_\rho| \cdot \widehat{K} = (|J_\rho| + |J_n| - 1) \cdot K_2 - (|J_\rho| - 1) \cdot \widehat{K} \geq 0$, as $|J_\rho| - 1 \leq |J_\rho| + |J_n| - 1$, $0 \leq |J_\rho| + |J_n| - 1$ and $\widehat{K} \leq K_2$. □

We can now build an automaton accepting minimal witness paths of $\widehat{G}$.

An automaton for minimal witnesses. We search for a minimal witness path $\rho \cdot n = n_0 \cdots n_\ell \cdot n$ in $\widehat{G}$ using an automaton $\mathcal{B}(\widehat{G})$. The first component of a state of $\mathcal{B}(\widehat{G})$ keeps track of the current node n. The second component will test for $(K, \widehat{K})$-drift-boundedness, which needs to hold for ρ but not for $\rho \cdot n$. This is done by keeping track of a $\widehat{K}$*-bounded profile*. The last component keeps track of $PF_{K_2,K_2}(\rho)$ which is sufficient to check consistency of ρ according to Proposition 6. Theorem 2 is obtained using the following proposition (where $|\widehat{G}|$ is the number of nodes of $\widehat{G}$):

Proposition 7. *Let G be a TC-MSC graph. Then there exists an automaton $\mathcal{B}(\widehat{G})$ such that $\mathcal{L}(\mathcal{B}(\widehat{G})) = \emptyset$ iff $\widehat{G}$ is $(K, \widehat{K})$-drift-bounded. Further, $\mathcal{B}(\widehat{G})$ has at most $|\widehat{G}| \times (2\widehat{K} + 1)^{|\mathcal{P}|^2} \times (2K_2 + 1)^{|\mathcal{P}|^2}$ states, where $K_2 = (|\mathcal{P}|/2 + 1) \cdot \widehat{K} + D(\widehat{G})$.*

Proof (Sketch). The states of $\mathcal{B}(\widehat{G})$ are triples (n, σ, τ), with n a node of $\widehat{G}$, σ a $\widehat{K}$-bounded profile of $\widehat{G}$, and τ a K_2-bounded profile of $\widehat{G}$. The initial state of $\mathcal{B}(\widehat{G})$ is $(n_{in}, PF_{K,\widehat{K}}(n_{in}), PF_{K_2,K_2}(n_{in}))$. A state (n, σ, τ) of $\mathcal{B}(\widehat{G})$ is final if σ is not satisfiable, but τ is. Last, there is a transition labeled by n' from (n, σ, τ) to (n', σ', τ') iff $\widehat{G}$ contains a transition $n \rightarrow n'$, $\sigma' = \theta_{K,\widehat{K}}^{n \rightarrow n'}(\sigma)$, $\tau' = \theta_{K_2,K_2}^{n \rightarrow n'}(\tau)$, and *both σ and τ' are satisfiable.* Notice that τ is satisfiable when σ is, as $K_2 \geq \widehat{K} \geq K$, and that σ' is not required to be satisfiable. □

6 Conclusion

This paper has addressed the emptiness problem for TC-MSC graphs. We have shown that emptiness can be checked under the restriction that a TC-MSC graph is K-drift-bounded, for some K, and we established the decidability of checking this restriction. The decision procedure does not consider linearizations of TC-MSC graphs, nor rely on the seminal result of [4]. Instead, a finite automaton keeps track of a system of inequalities describing symbolically constraints over dates on each process. As future work, we plan to consider checking whether a TC-MSC graph is drift-bounded (without the bound K), and if so computing the bound. It seems that tackling this problem needs new ideas and concepts.

References

1. S. Akshay, B. Genest, L. Hélouët, and S. Yang. Regular set of representatives for time-constrained MSC graphs. *Inf. Proc. Letters*, 112(14-15):592–598, 2012.
2. S. Akshay, M. Mukund, and K. Narayan Kumar. Checking coverage for infinite collections of timed scenarios. In *CONCUR 2007, LNCS 4703, pp. 181–196*. Springer.
3. S. Akshay, P. Gastin, K. Narayan Kumar, and M. Mukund. Model checking time-constrained scenario-based specifications. In *FSTTCS 2010, LNCS 4855, pp. 290–302*. Springer.
4. R. Alur and D. L. Dill. A theory of timed automata. *Theoretical Comp. Sci.*, 126(2):183–235, 1994.
5. R. Alur and M. Yannakakis. Model checking of message sequence charts. In *CONCUR 1999, LNCS 1664, pp. 114–129*. Springer.
6. J. Bengtsson and W. Yi. On clock difference constraints and termination in reachability analysis of timed automata. In *ICFEM 2003, pp. 491–503*.
7. P. Bouyer, S. Haddad, and P.-A. Reynier. Timed unfoldings for networks of timed automata. In *ATVA 2006, LNCS 4218, pp. 292–306*. Springer.
8. F. Cassez, T. Chatain, and C. Jard. Symbolic unfoldings for networks of timed automata. In *ATVA 2006, LNCS 4218, pp. 307–321*. Springer.
9. G. Dantzig and B. C. Eaves. Fourier-Motzkin Elimination and Its Dual. *J. Comb. Theory, Ser. A*, 14(3):288–297, 1973.
10. C. Dima and R. Lanotte. Distributed time-asynchronous automata. In *ICTAC 2007, LNCS 4711, 185–200*. Springer.
11. P. Gastin, K. Narayan Kumar, and M. Mukund. Reachability and boundedness in time-constrained MSC graphs. In *Perspectives in Concurrency – A Festschrift for P. S. Thiagarajan*. Universities Press, India, 2009.
12. J. G. Henriksen, M. Mukund, K. N. Kumar, M. Sohoni, and P. S. Thiagarajan. A theory of regular MSC languages. *Inf. and Comp.*, 202(1):1–38, 2005.
13. ITU-TS Recommendation Z.120: Message Sequence Chart (MSC '99), 1999.
14. B. Korte and J. Vygen. *Combinatorial Optimization: Theory and Algorithms*. Springer, 3rd edition, 2006.
15. D. Lugiez, P. Niebert, and S. Zennou. A partial order semantics approach to the clock explosion problem of timed automata. *Theoretical Comp. Sci.*, 345(1):27–59, 2005.
16. A. Muscholl and D. Peled. Message sequence graphs and decision problems on Mazurkiewicz traces. In *MFCS 1999, LNCS 1672, pp. 81–91*. Springer.
17. R. Shostak. Deciding linear inequalities by computing loop residues. *JACM*, 28(4):769–779, 1981.
18. J. Zhao, H. Xu, X. Li, T. Zheng, and G. Zheng. Partial order path technique for checking parallel timed automata. In *FTRTFT 2002, LNCS 2469, pp. 417–432*.