EPiC Series in Computing
Volume 46, 2017, Pages 319–337
LPAR-21. 21st International Conference on Logic for Programming, Artificial Intelligence and Reasoning

A Quantitative Partial Model-Checking Function and Its Optimisation

Stefano Bistarelli[1], Fabio Martinelli[2], Ilaria Matteucci[2], Francesco Santini[1]

[1] Dipartimento di Matematica e Informatica, Università di Perugia, Italy
[bista,francesco.santini]@dmi.unipg.it
[2] Istituto di Informatica e Telematica, IIT-CNR, Pisa, Italy
[fabio.martinelli,ilaria.matteucci]@iit.cnr.it

Abstract

Partial Model-Checking (PMC) is an efficient tool to reduce the combinatorial explosion of a state-space, arising in the verification of loosely-coupled software systems. At the same time, it is useful to consider quantitative temporal-modalities. This allows for checking whether satisfying such a desired modality is too costly, by comparing the final score consisting of how much the system spends to satisfy the policy, to a given threshold. We stir these two ingredients together in order to provide a *Quantitative PMC* function (QPMC), based on the algebraic structure of semirings. We design a method to extract part of the weight during QPMC, with the purpose to avoid the evaluation of a modality as soon as the threshold is crossed. Moreover, we extend classical heuristics to be quantitative, and we investigate the complexity of QPMC.

Keyword: Partial Model Checking, Semirings, Optimisation, Quantitative Modal Logic Quantitative Process Algebra, Quantitative Evaluation of Systems.

1 Introduction

When considering large concurrent software or hardware systems that are distributed over several execution points, it is clearly important to check offline if the related model meets a desired specification. For example, whether the overall interaction-design among components contains safety requirements, *e.g.*, the absence of deadlocks, which can lead the system to crash. The considerable amount of decentralisation in large networks of small computational units demands for a rigorous property analysis, which becomes more complex due to their high distribution-degree. Nowadays, well-known examples of such networks are *Cyber-physical Systems* and the *Internet of Things*.

Model Checking is a well-established method to formally verify finite-state concurrent systems. Specifications about the system are expressed as temporal logic formulas (*i.e.*,, a formula ϕ), and efficient symbolic algorithms are used to traverse the model defined by the system and check if the specification holds or not. A key limitation to its use is due to the state explosion problem. For this reason, a variety of techniques to limit state-explosion have been investigated over the years (Sec. 6). The technique we consider in this paper is *Partial Model-Checking* (*PMC*) [1]: parts of the concurrent system are gradually removed while transforming ϕ accordingly (such operation is also known as "quotienting"). When the intermediate specifications constructed in this manner can be kept small, the state-explosion problem is avoided.

Keeping a desired behaviour to verify some property of finite-state systems unavoidably impacts on the non-functional aspects of that system: these quality attributes describe *how* a system behaves while following a certain behaviour; some examples are *availability*, *adaptability*, *efficiency*, and *securability*. Hence, functional aspects add to the overall picture costs, execution times, and rates (for instance). The motivation behind this work is to enhance the existing qualitative approach, whose the answer is a plain "yes/no" (a system satisfies/does not satisfy ϕ), to consider a quantitative score that is definitely more informative to understand how costly it is to satisfy ϕ, in case the answer is "yes". What previously introduced defines the scenario and the motivations behind this work. The first step consists in elaborating on part of the initial ideas presented in [31]: we equip the semiring-based logic introduced in [31] with fix points, thus obtaining an equations system similar to what has been proposed in [1]. We call such a logic *c-semiring Equational μ-calculus*, c-Eμ for short. The main benefits coming from the use of semirings is that we can design a general computational framework that is parametric with respect to this structure: any metric instance complying with the properties demanded by semirings can be cast in the presented framework. Semirings are so expressive to allow partially-ordered values, multiple-criteria and lexicographic orders, and either idempotent or non-idempotent value-composition operators (Sec. 2).

We use c-Eμ to define a quantitative modality ϕ to be checked over processes expressed in an "à la CCS" version of *Generalised Process Algebra* [12]. Transitions of such processes are labelled with a value taken from a semiring, expressing a "cost" associated with each action.

The main result of the paper is the design of a *Quantitative Partial Model-Checking* (*QPMC*) function to verify a ϕ in c-Eμ against a threshold t: ϕ is satisfied if its cost is less than t. This approach both takes into account quantitative aspects of systems, and reduces the number of states (thus mitigating the combinatorial explosion). The QPMC function is able to extract (part of) the weight of actions removed from the system specification and accumulate it into a side variable k. Indeed not all the weight can be extracted, since non-deterministic branches may have different costs. However, such a removal is useful to avoid the evaluation of ϕ when already k crosses t, because the cost of ϕ may only be worse than k (and than t consequently).

On the other hand, while QPMC shrinks the system specification, it moves the system specification to ϕ, which grows in size instead. This effect may counter-balance the former beneficial effect on the computational cost to verify ϕ. Therefore, we have developed some rules to simplify ϕ where possible, before checking its satisfaction. All the boolean heuristics presented in [1] are a subset of ours; hence, we achieve the same efficiency proven in [1], at least on Boolean algebras.

Finally, we present how the application of simplification rules can be fruitfully exploited to decrease the computational upper-bound in case of distributive lattices (where $\otimes$ is idempotent). Moreover, we suggest the existence of upper-bounds for non-distributive semirings as well. The intuition is that, being the computation of fix-points limited by t, this allows us to find a finite lattice of possible evaluations with respect to a finite set of generators, even in case $\otimes$ is non-idempotent. As an example, we provide a result for the weighted semiring.

A first application (to the adaptation of agents) of a similar QPMC function has been proposed in [8, 9]. However such a function is defined without fix-points, heuristics, and complexity considerations: in this paper we focus on these aspects instead, also proposing a novel c-Eμ to represent properties.

The paper is organised as follows: next section introduces the necessary background information on semiring structures (Sec. 2.1) and GPA (Sec. 2.2). Section 3 presents c-Eμ, while Sec. 4 details our QPMC function. The final step in Sec. 5 is represented by the description of all the simplifications that can be used to reduce the size of a formula obtained though QPMC. Finally, Sec. 6 and Sec. 7 respectively discuss about similar results in literature and provides a final discussion and ideas about future work.

2 Background

We recall the fundamental notions about semirings [6, 5] and a CCS-like version of *Generalised Process Algebra* [12], a process algebra based on semiring.

2.1 Semirings

Definition 2.1 (Semiring [19])**.** *A commutative semiring is a five-tuple* $\mathbb{K} = \langle K, \oplus, \otimes, \bot, \top \rangle$ *such that* K *is a set,* $\top, \bot \in K$*, and* $\oplus, \otimes : K \times K \to K$ *are binary operators making the triples* $\langle K, \oplus, \bot \rangle$ *and* $\langle K, \otimes, \top \rangle$ *commutative monoids (semigroups with identity), satisfying (distributivity)* $\forall a, b, c \in K. a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$ *and (annihilator)* $\forall a \in A. a \otimes \bot = \bot$.

Definition 2.2 (Absorptive semiring [19])**.** *Let* $\mathbb{K}$ *be a commutative semiring. An absorptive semiring is a semiring where it holds*

1) (absorptiveness) $\forall a, b \in K. a \oplus (a \otimes b) = a$,
2) ($\top$ *absorbing element of* $+$*)* $\forall a \in K. a \oplus \top = \top$.

Absorptive semirings are referred also as *simple*, and their $\oplus$ operator is necessarily idempotent [19, Ch. 1, pp. 14]. Semirings where $\oplus$ is idempotent are defined as *tropical* semirings, or *diods*.

Definition 2.3 (C-semiring [6])**.** *C-semirings are commutative and absorptive semirings.*

The idempotency of $\oplus$ leads to the definition of a partial ordering $\leqslant_K$ over the set K (K is a poset). Such partial order is defined as $a \leqslant b$ if and only if $a \oplus b = b$, and $\oplus$ becomes the *least upper bound* (*lub*, or $\Cup$) of the lattice $\langle K, \leqslant_K \rangle$. This intuitively means that b is "better" than a. Some more properties can be derived on c-semirings [6]: *i)* both $\oplus$ and $\otimes$ are monotone over $\leqslant_K$, *ii)* $\otimes$ is intensive (*i.e.*, $a \otimes b \leqslant_K a$), iii) $\otimes$ is closed (*i.e.*, $a \otimes b \in K$), and *iv)* $\langle K, \leqslant_K \rangle$ is a complete lattice. When also $\otimes$ is idempotent, *i)* $\oplus$ distributes over $\otimes$, *ii)* $\otimes$ is the *greater lower bound* (*glb*, or $\Cap$), and *iii)* $\langle K, \leqslant_K \rangle$ is a distributive lattice.

C-semirings have been often adopted in combinatorial problems as a very simple but yet expressive optimisation structure [32, Ch. 9]. Some well-known instances are: the *boolean* $\langle \{F, T\}, \vee, \wedge, F, T \rangle$[1], *fuzzy* $\langle [0, 1], \max, \min, 0, 1 \rangle$, Viterbi $\langle [0, 1], \max, \times, 0, 1 \rangle$, *weighted* $\langle \mathbb{N}^+ \cup \{+\infty\}, \min, +, +\infty, 0 \rangle$ semirings.

Although c-semirings have been historically [32] used as monotonic structures where to aggregate costs (and find best solutions), the need of removing values has raised in local consistency algorithms and non-monotonic algebras using constraints (*e.g.*, [5, 32]). A solution comes from *residuation theory* [10], a standard tool on tropical arithmetic that allows for obtaining a division operator via an approximate solution to the equation $b \otimes x = a$.

[1] Boolean c-semirings can be used to model crisp (*i.e.*, unweighted) problems.

Definition 2.4 (Residuation [5]). *Let $\mathbb{K}$ be a tropical semiring. It is residuated if the set $\{x \in K \mid b \otimes x \leqslant_K a\}$ admits a maximum $\forall\, a, b \in K$, denoted $a \oslash b$.*

Since a complete[2] tropical semiring is also residuated, we have that all the classical instances of c-semiring presented above are residuated, *i.e.*, each element in K admits an "inverse", which is unique in case $\leqslant_K$ is a total order [5]. For instance, the unique "inverse" $a \oslash b$ in the weighted semiring is defined as follows: $a \oslash b = \min\{x \mid b \oplus x \geqslant_K a\}$, which is $a - b$ if $a > b$, and 0 otherwise.

Definition 2.5 (Invertibility uniqueness [5]). *Let $\mathbb{K}$ be an absorptive, invertible semiring. Then, $\mathbb{K}$ is uniquely invertible if and only if it is cancellative,* i.e., $\forall a, b, c \in A.(a \otimes c = b \otimes c) \wedge (c \neq 0) \Rightarrow a = b$.

Note that since all the previously presented instances (*e.g.*, weighted) are cancellative, they are uniquely invertible as well.

2.2 Generalised Process Algebra ("à la CCS")

Process algebras are languages formalisation with precise mathematical semantics, tailored to exhibit and study specific features of computation. Typically, a *process* P, specified by some syntax, may non-deterministically execute several *labelled transitions* of the form $P \xrightarrow{a} P'$, where a is an observable effect and P' is a new process. In a *quantitative process*, observable transitions are labelled with some value, *i.e.*, transitions are labelled by pairs (a, x) where x is a quantity associated to the effect a. In *Generalised Process Algebra* (*GPA*) [12] the authors use semirings to have a general framework to model min/plus, max/plus or min/max systems, which are applied in the area of communication protocols, queueing networks, and real-time systems.

We modify the definition of GPA given in [12] with the purpose to a have an "à la CCS" instead of "à la CSP" synchronisation: our synchronous communication uses a two-way handshake via matching actions and co-actions of two processes (refer to [21] for a recent encoding of CSP into CCS). Nevertheless, we refer to this algebra as GPA as well.

We start by defining transition systems in Def. 2.6. An MLTS is a graph where each transition is labelled by a symbol to define the action, and a cost to perform it. The notion of MLTS is subsequently used to define the semantics of GPA processes.

Definition 2.6 (MLTS). *A (finite) Multi Labelled Transition System (MLTS) is a five-tuple $MLTS = (S, Act, \mathbb{K}, T, s_0)$, where S is the countable (finite) state space, $s_0 \in S$ is the initial state, Act is a finite set of actions, $\mathbb{K}$ is a semiring used to weigh actions, and $T : (S \times Act \times S) \longrightarrow K$ is a transition function.*

We now define the syntax of GPA processes:

Definition 2.7 ([12]). *The set $\mathcal{P}$ of terms in GPA over a set of finite transition labels (a, k) where $a \in Act$ and $k \in K$ from a semiring $\langle K, \oplus, \otimes, \bot, \top\rangle$ is defined by $P ::= 0 \mid (a, k).P \mid P + P \mid P \| P \mid X$, where X is a countable set of* process variables, *coming from a system of co-recursive equations of the form $X \triangleq P$.*

Informally, process 0 describes inaction or termination; $(a, k).P$ performs action a with *value* k and evolves into P; $P + P'$ non-deterministically behaves as either P or P'; $P \| P'$ describes the process in which P and P' proceed concurrently when they perform complementary actions (*e.g.*, $a, \bar{a} \in Act$), and independently on all other actions.

[2] $\mathbb{K}$ is complete if it is closed with respect to infinite sums, and the distributivity law holds also for an infinite number of summands [5].

$$\frac{}{(a,k).P \xrightarrow{a,k} P} \qquad \frac{P \xrightarrow{a,k} P_1}{X \xrightarrow{a,k} P_1} X \triangleq P \qquad \frac{P \xrightarrow{a,k_j} P_1}{P + P' \xrightarrow{a,k_j} P_1} j \in I$$

$$\frac{P \xrightarrow{a,k} P_1 \quad P' \xrightarrow{\bar{a},l} P'_1}{P \| P' \xrightarrow{\tau, k \otimes l} P_1 \| P'_1} \qquad \frac{P \xrightarrow{a,k} P_1}{P \| P' \xrightarrow{a,k} P_1 \| P'} \qquad \frac{P' \xrightarrow{a,k} P'_1}{P \| P' \xrightarrow{a,k} P \| P'_1}$$

Table 1: Semantic rules for *GPA* processes.

The semantics of a process (or agent) $P \in \mathcal{P}$ is a MLTS. As usual in process algebras, we cannot distinguish between a process and a state. A process and all its derivatives, reachable by applying the semantic rules in Tab. 1, form the state space of a system. The set of *derivatives* is defined as $Der(P) = \{P' \mid P \rightarrow^* P'\}$, where $\rightarrow^*$ is the transitive and reflexive closure of T.

3 C-semiring Equational μ-calculus

We propose a quantitative variant of the *Equational μ-calculus*, here named c-Eμ, in such a way to evaluate a given property with a score. In particular, we take into account the weights on the transitions of an MLTS as defined in Def. 2.6. In Def. 3.1, we syntactically define the set Φ_M of formulas over a finite MLTS M.

Definition 3.1 (Syntax). *Given a MLTS $M = \langle S, Act, \mathbb{K}, T \rangle$, and let $k \in K$ and $a \in Act$, the syntax of a formula $\phi \in \Phi_M$ is as follows:*

$$\begin{array}{lll} \phi & ::= & k \mid v \mid \phi_1 \oplus \phi_2 \mid \phi_1 \otimes \phi_2 \mid \phi_1 \Cap \phi_2 \mid \langle a \rangle \phi \mid [a]\phi \\ E & ::= & v =_\mu \phi E \mid v =_\nu \phi E \mid \epsilon \end{array}$$

Hence, we can express more than just *true* (corresponding to $\top \in K$) and *false* ($\bot \in K$) by using all the $k \in K$, which correspond to different degrees of truth less $\top$ (full truth); v belongs to a set of variables V. Semiring operators lub $\oplus$, glb $\Cap$, and $\otimes$ are used in place of classical logic operators $\vee$ and $\wedge$, in order to compose the truth values of two formulas together. As a reminder, when the $\otimes$ operator is idempotent, then $\otimes$ and $\Cap$ coincide (see Sec. 2). Then, we have the two classical modal operators, *i.e.*, "possibly" ($\langle \cdot \rangle$), and "necessarily" ($[\cdot]$).

C-Eμ is based on fix-point equations: $v =_\mu \phi$ is a minimal fixpoint equation, where ϕ is an assertion (*i.e.*, a simple modal formula without recursion operator), and $v =_\nu \phi$ is a maximal fixpoint equation.

We define a *free* variable $v \in V$ of a formula ϕ, (*free(V)*) as a variable without any defining equation, *i.e.*, it is not bound by any $=_\mu$ or $=_\nu$ ($=_{\mu/\nu}$ for short). The meaning of free variables of an equation system is given by an *environment function* $\rho : (V \longrightarrow S) \longrightarrow K$, such that for any free variable $v \in V$ it associates a state $s \in S$, and returns a corresponding semiring value $\rho(v, s) = k \in K$.[3] A formula ϕ is closed if all its variables are bound, *i.e.*, the evaluation of ϕ does not depend on ρ.

The semantics of a formula ϕ is always related to a finite MLTS $M = \langle S, Act, \mathbb{K}, T, s_0 \rangle$, which is related to the semantics of a GPA process P. Note that, while in [1] the semantics of ϕ

[3] This recalls the definition of soft constraints [6], where a constraint is a function $c : (V \longrightarrow D) \longrightarrow K$ and D is the domain of the variables in V.

$$
\begin{array}{ll}
[\![k]\!]_\rho(s) & = k \in K \;\; \forall s \in S \\
[\![v]\!]_\rho(s) & = \rho(v,s) \\
[\![\phi_1 \oplus \phi_2]\!]_\rho(s) & = [\![\phi_1]\!]_\rho(s) \oplus [\![\phi_2]\!]_\rho(s) \\
[\![\phi_1 \otimes \phi_2]\!]_\rho(s) & = [\![\phi_1]\!]_\rho(s) \otimes [\![\phi_2]\!]_\rho(s) \\
[\![\phi_1 \mathbin{\bigcirc\mkern-15mu\sqcap} \phi_2]\!]_\rho(s) & = [\![\phi_1]\!]_\rho(s) \mathbin{\bigcirc\mkern-15mu\sqcap} [\![\phi_2]\!]_\rho(s) \\
[\![\langle a\rangle\phi]\!]_\rho(s) & = \bigoplus_{\{s' \in S \mid s \xrightarrow{a} s' \in T\}} (T(s,a,s') \otimes [\![\phi]\!]_\rho(s')) \\
[\![[a]\phi]\!]_\rho(s) & = \underset{\{s' \in S \mid s \xrightarrow{a} s' \in T\}}{\bigcirc\mkern-15mu\sqcap} (T(s,a,s') \otimes [\![\phi]\!]_\rho(s')) \\
[\![v =_\mu \phi E]\!]_\rho(s) & = \mathit{fix}\ \lambda k'.[\![\phi E]\!]_{\rho[k'/v]}(s) \\
[\![v =_\nu \phi E]\!]_\rho(s) & = \mathit{FIX}\ \lambda k'.[\![\phi E]\!]_{\rho[k'/v]}(s) \\
[\![\epsilon]\!]_\rho(s) & = \top
\end{array}
$$

$$
\mathit{where}\ [\![\phi E]\!]_{\rho[k'/v]}(s) = [\![\phi]\!]_{\rho'}(s), \rho'(y,s) = \begin{cases} \rho(y,s) & \forall y \in \mathit{free}(V) \\ k' & \mathit{if}\ y = v \\ [\![E]\!]_{\rho[k'/v]}(s) & \forall y \notin \mathit{free}(V) \end{cases}
$$

Table 2: Semantics of c-Eμ. $\bigoplus(\emptyset) = \bot$ and $\bigcirc\mkern-15mu\sqcap(\emptyset) = \top$.

computes all the states $U \subseteq S$ that satisfy ϕ, our semantics computes a truth value (in K) for the same U. For instance, if we use the boolean semiring we always obtain $\top$ iff $U \neq \emptyset$, and $\bot$ otherwise. It is not difficult to extend our semantics to also return U, as in [1]; however, in c-Eμ we focus on computing a degree of satisfaction for U (and ϕ). In Tab. 2 we show the function that computes the semantics of ϕ, *i.e.*, $[\![\ ]\!]_\rho(s) : (\Phi_M \times S) \longrightarrow K$.

The semantics of a system of equational assertions is inductively defined on the solution of each equational assertion, as described in Tab. 2: let $v =_{\mu/\nu} \phi E$ be a system of equational assertions, the first step consists of finding a first value k for v. Then, using that value, it is possible to inductively solve E. Indeed, the solution for v is the fix-point determined by the meaning of ϕ in an environment ρ' where the meaning of the free variables of the equation system is given by ρ, the meaning of v is k, and the meaning of the remaining bound variables is given by the inductive solution of E; ϵ is the empty assertion.

We consider a system of equations E as *well-defined* if each variable $v \in V$ appears only once on the left side of assertions. In the following we always suppose to work with a well-defined E. As in [1], we consider a *top assertion* $E_{\downarrow v}$, which identifies the solution for our system achieved by projecting on part of the assertions. Hence, the semantics of $E_{\downarrow v}$ is $[\![v =_{\mu/\nu} \phi E]\!]_\rho(s_0)$ (refer to Tab. 2), where $v =_{\mu/\nu} \phi E$ is one of the assertions in a system.

Let $\tau(k') = \lambda k'.[\![\phi E]\!]_{\rho[k'/v]}$ be order-preserving functions over a complete lattice K (*i.e.*, $\lambda k' \geqslant_K \lambda(\lambda k')$) [16, pp. 50]. Referring to [27], τ functions are monotone because they are composed by $\oplus$, $\bigcirc\mkern-15mu\sqcap$, and $\otimes$, which are all monotone over a poset. Functions ρ are updated accordingly to τ functions (see last line in Tab. 2), also being well defined on each variable (free and bound). Hence, by Knaster-Tarski Theorem [33] the fix points are well-defined. They can be computed as $\mathit{fix} = \bigcirc\mkern-15mu\sqcap\{k \mid k \leqslant_K \tau(k)\}$, $\mathit{FIX} = \bigoplus\{k \mid k \leqslant_K \tau(k)\}$.

We are now ready to rephrase the notion of satisfiability to take into account a threshold t (t-satisfiability):

Definition 3.2 (t-satisfiability: $\vDash_t$)**.** *A process P satisfies a c-Eμ formula ϕ with a threshold-value*

t, i.e., $P \vDash_t \phi$, if and only if the evaluation of ϕ on P is not worse than t, considering the order $\leqslant_K$. Formally, $P \vDash_t \phi \Leftrightarrow t \leqslant_K [\![\phi]\!]_\rho(P)$.

This means that P is a model for a formula ϕ with respect to a certain value t if and only if the interpretation of ϕ on P is not worse than t in the partial order defined by $\oplus$ in a given $\mathbb{K}$.

Remark 3.1. *If P does not satisfy a formula ϕ then $[\![\phi]\!]_\rho(P) = \bot$. Hence, the only t s.t. $P \vDash_t \phi$ is $t = \bot$. If $[\![\phi]\!]_\rho(P) \neq \bot$, then ϕ is t-satisfiable for some $t \neq \bot$.*

4 Quantitative PMC for c-Eμ formulas

The *Partial Model checking* function has been firstly introduced in [1] as a mechanism that, by partially removed the specification of the model in such a way that the formula expressed requirement is transformed accordingly, deals with the state explosion problem that affects model checking problems.

In this section, we present a quantitative version of the PMC function, named QPMC, with respect to the parallel composition of GPA processes, *e.g.*, $P \parallel Q$. This function extracts a weight $k_{P,\phi}$ that represents an upper bound on the cost to satisfy $\phi_{//_P}$, which is the result of QPMC over ϕ: the function in Tab. 3 totally removes the specification of P from $P \parallel Q$, and moves it to ϕ. The computation of $k_{P,\phi}$ is presented in Tab. 4 for each different ϕ in Tab. 3. The benefit behind using QPMC is that the evaluation of $\phi_{//_P}$ gets simpler than the one of ϕ by applying the strategies described in Sec. 5 (as also proved in [1]).

We now introduce the main formal result of the paper. Theorem 4.1 brings to say that, if the evaluation of $\phi_{//_P}$ over the remaining process Q composed with $k_{P,\phi}$ (*i.e.*, $k_{P,\phi} \otimes [\![\phi_{//_P}]\!]_\rho(Q)$) is t-satisfied, then also the original ϕ is t-satisfied: if $k_{P,\phi} \otimes [\![\phi_{//_P}]\!]_\rho(Q) \geqslant_K t$, then also $[\![\phi]\!]_\rho(P \parallel Q) \geqslant_K t$ (transitivity of $\geqslant_K$). On the other hand, if $k_{P,\phi} \otimes [\![\phi_{//_P}]\!]_\rho(Q) \not\geqslant_K t$ then ϕ could be anyhow satisfied: therefore, $k_{P,\phi} \otimes [\![\phi_{//_P}]\!]_\rho(Q)$ represents a lower bound on the evaluation of ϕ over $P \parallel Q$.

Theorem 4.1. *Let P and Q two processes in GPA, $\mathbb{K}$ a c-semiring with $k \in K$, and ϕ a c-Eμ formula, then:*

$$[\![\phi]\!]_\rho(P \parallel Q) \geqslant_K k_{P,\phi} \otimes [\![\phi_{//_P}]\!]_\rho(Q)$$

Proof. The theorem is proved by induction on the complexity of a formula. To lighten the notation, we write $[\![\phi]\!]$ instead of $[\![\phi]\!]_\rho$ (see Tab. 2). We show the proof for the base case and for the $\oplus$, the diamond modality, and the minimum fixpoint operators. The omitted cases are similar to the presented ones.

Base case ($\phi = k$). According to Tab. 2, $[\![k]\!]_{P\parallel Q} = k = k_{//_P} = \top \otimes [\![k_{//_P}]\!]_Q$.

Inductive step. Let us consider all the possible structures of formulas:

$\phi = \phi_1 \oplus \phi_2$**:** According to the semantics interpretation of the $\oplus$ formula (Tab. 2),

$$[\![\phi]\!]_{P\parallel Q} = [\![\phi_1 \oplus \phi_2]\!]_{P\parallel Q} = [\![\phi_1]\!]_{P\parallel Q} \oplus [\![\phi_2]\!]_{P\parallel Q}$$

By inductive hypothesis,

$$[\![\phi_1]\!]_{P\parallel Q} = k_{P,\phi_1} \otimes [\![(\phi_1)_{//_P}]\!]_Q \textit{ and } [\![\phi_2]\!]_{P\parallel Q} = k_{P,\phi_2} \otimes [\![(\phi_2)_{//_P}]\!]_Q$$

Then

$$[\![\phi_1]\!]_{P\parallel Q} \oplus [\![\phi_2]\!]_{P\parallel Q} = (k_{P,\phi_1} \otimes [\![(\phi_1)_{//_P}]\!]_Q) \oplus (k_{P,\phi_2} \otimes [\![(\phi_2)_{//_P}]\!]_Q)$$

$$
\begin{array}{ll}
(1) & k_{//_P} = k \\
(2) & v_{//_P} = v_P \\
(3) & (\phi_1 \otimes \phi_2)_{//_P} = (k_{P,\phi_1} \oslash k_{P,\phi})(\phi_1)_{//_P} \otimes (k_{P,\phi_2} \oslash k_{P,\phi})(\phi_2)_{//_P} \\
(4) & (\phi_1 \oplus \phi_2)_{//_P} = (k_{P,\phi_1} \oslash k_{P,\phi})(\phi_1)_{//_P} \oplus (k_{P,\phi_2} \oslash k_{P,\phi})(\phi_2)_{//_P} \\
(5) & (\phi_1 \sqcap \phi_2)_{//_P} = (k_{P,\phi_1} \oslash k_{P,\phi})(\phi_1)_{//_P} \sqcap (k_{P,\phi_2} \oslash k_{P,\phi})(\phi_2)_{//_P}
\end{array}
$$

$$
(6) \quad (\langle a \rangle \phi_1)_{//_P} \;=\; (k_{P,\phi_1} \oslash k_{P,\phi}) \otimes \langle a \rangle (\phi_1)_{//_P} \oplus \bigoplus_{P \overset{a,k_a}{\rightarrow} P'} (k_a \otimes (k_{P',\phi_1} \oslash k_{P,\phi}) \otimes (\phi_1)_{//_{P'}})
$$

$$
(7) \quad \begin{array}{rl}
(\langle \tau \rangle \phi_1)_{//_P} = & (k_{P,\phi_1} \oslash k_{P,\phi}) \otimes \langle \tau \rangle (\phi_1)_{//_P} \oplus \bigoplus_{P \overset{\tau,k_\tau}{\rightarrow} P'} (k_\tau \otimes (k_{P',\phi_1} \oslash k_{P,\phi}) \otimes (\phi_1)_{//_{P'}}) \\
\oplus & \bigoplus_{P \overset{a,k_a}{\rightarrow} P'} ((k_a \otimes k_{P',\phi_1}) \oslash k_{P,\phi}) \otimes \langle \bar{a} \rangle (\phi_1)_{//_{P'}})
\end{array}
$$

$$
(8) \quad ([a]\phi_1)_{//_P} \;=\; (k_{P,\phi_1} \oslash k_{P,\phi}) \otimes [a](\phi_1)_{//_P} \sqcap \bigsqcap_{P \overset{a,k_a}{\rightarrow} P'} (k_a \otimes (k_{P',\phi_1} \oslash k_{P,\phi}) \otimes (\phi_1)_{//_{P'}})
$$

$$
(9) \quad \begin{array}{rl}
([\tau]\phi_1)_{//_P} = & (k_{P,\phi_1} \oslash k_{P,\phi}) \otimes [\tau](\phi_1)_{//_P} \sqcap \bigsqcap_{P \overset{\tau,k_\tau}{\rightarrow} P'} (k_\tau \otimes (k_{P',\phi_1} \oslash k_{P,\phi}) \otimes (\phi_1)_{//_{P'}}) \\
\sqcap & \bigsqcap_{P \overset{a,k_a}{\rightarrow} P'} ((k_a \otimes k_{P',\phi_1}) \oslash k_{P,\phi}) \otimes [\bar{a}](\phi_1)_{//_{P'}})
\end{array}
$$

$$
(10) \quad (v =_\mu \phi_1 E)_{//_P} = \begin{cases} v_{P_1} =_\mu \quad \phi_{1//_{P_1}} \\ \vdots \\ v_{P_n} =_\mu \quad \phi_{1//_{P_n}} \\ E_{//_P} \end{cases}
$$

$$
(11) \quad (v =_\nu \phi_1 E)_{//_P} = \begin{cases} v_{P_1} =_\nu \quad \phi_{1//_{P_1}} \\ \vdots \\ v_{P_n} =_\nu \quad \phi_{1//_{P_n}} \\ E_{//_P} \end{cases}
$$

$$
(12) \quad \epsilon_{//_P} = \epsilon
$$

Table 3: The QPMC function. $k_{P,\phi}$ is computed as given in Tab. 4 for each ϕ of this table.

Noting that $k_{P,\phi_1} \geqslant_K (k_{P,\phi_1} \oslash k_{P,\phi}) \otimes k_{P,\phi}$ holds and that the same holds also for k_{P,ϕ_2}, we have

$$
(k_{P,\phi_1} \otimes [\![(\phi_1)_{//_P}]\!]_Q) \oplus (k_{P,\phi_2} \otimes [\![(\phi_2)_{//_P}]\!]_Q) \geqslant_K \begin{array}{l} (((k_{P,\phi_1} \oslash k_{P,\phi}) \otimes k_{P,\phi}) \otimes [\![(\phi_1)_{//_P}]\!]_Q) \oplus \\ \oplus(((k_{P,\phi_2} \oslash k_{P,\phi}) \otimes k_{P,\phi}) \otimes [\![(\phi_2)_{//_P}]\!]_Q) \end{array}
$$

For the associativity and commutativity of the $\otimes$ operation, this is equal to $(k_{P,\phi} \otimes ((k_{P,\phi_1} \oslash k_{P,\phi}) \otimes [\![(\phi_1)_{//_P}]\!]_Q)) \oplus (k_{P,\phi} \otimes ((k_{P,\phi_2} \oslash k_{P,\phi}) \otimes [\![(\phi_2)_{//_P}]\!]_Q))$. For the distributivity of the product with respect to the sum, this is equal to $k_{P,\phi} \otimes (((k_{P,\phi_1} \oslash k_{P,\phi}) \otimes [\![(\phi_1)_{//_P}]\!]_Q) \oplus$

(1)	$\top$	(2)	$\top$
(3)	$k_{P,\phi_1} \oplus k_{P,\phi_2}$	(4)	$k_{P,\phi_1} \oplus k_{P,\phi_2}$
(5)	$k_{P,\phi_1} \oplus k_{P,\phi_2}$	(6)	$k_{P,\phi_1} \oplus \bigoplus_{P'} k_{P',\phi_1}$
(7)	$k_{P,\phi_1} \oplus (\bigoplus_{P'} k_{P',\phi_1}) \oplus \bigoplus_{P'} (k_a \otimes k_{P',\phi_1})$	(8)	$k_{P,\phi_1} \oplus \bigoplus_{P'} k_{P',\phi_1}$
(9)	$k_{P,\phi_1} \oplus (\bigoplus_{P'} k_{P',\phi_1}) \oplus \bigoplus_{P'} (k_a \otimes k_{P',\phi_1})$	(10)	$k_{P,E} \oplus \bigoplus_{P_i \in DerP} k_{P_i,\phi_1} \oplus k_{P,E}$
(11)	$k_{P,E} \oplus \bigoplus_{P_i \in DerP} k_{P_i,\phi_1}$	(12)	$\top$

Table 4: $k_{P,\phi}$ is an amount of weight that QPMC can safely extract from each ϕ in Tab. 3.

$((k_{P,\phi_2} \oslash k_{P,\phi}) \otimes \llbracket (\phi_2)_{//P} \rrbracket_Q))$. Hence,

$$\begin{array}{lll} \llbracket \phi_1 \oplus \phi_2 \rrbracket_{P\|Q} & \geqslant_K & k_{P,\phi} \otimes \llbracket ((k_{P,\phi_1} \oslash k_{P,\phi}) \otimes (\phi_1)_{//P}) \\ & \oplus & ((k_{P,\phi_2} \oslash k_{P,\phi}) \otimes (\phi_2)_{//P}) \rrbracket_Q = k_{P,\phi} \otimes \llbracket (\phi_1 \oplus \phi_2)_{//P} \rrbracket_Q. \end{array}$$

$\phi = \langle a \rangle \phi_1$: According to Tab. 2, $\llbracket \phi \rrbracket_{P\|Q} =$

$$\llbracket \langle a \rangle \phi_1 \rrbracket_{P\|Q} = \bigoplus_{P\|Q \xrightarrow{(a,k_a)} (P\|Q)'} (T(s,a,s') \otimes \llbracket \phi_1 \rrbracket_\rho((P\|Q)')) = \bigoplus_{P\|Q \xrightarrow{(a,k_a)} (P\|Q)'} (k_a \otimes \llbracket \phi_1 \rrbracket_\rho((P\|Q)'))$$

The semantic of the parallel operator $\|$ in Tab. 1 leads to the following reduction,

$$\bigoplus_{P\|Q \xrightarrow{(a,k_a)} (P\|Q)'} (k_a \otimes \llbracket \phi_1 \rrbracket_\rho((P\|Q)')) = \bigoplus_{P \xrightarrow{(a,k_a)} P'} (k_a \otimes \llbracket \phi_1 \rrbracket_\rho(P'\|Q)) \oplus \bigoplus_{Q \xrightarrow{(a,k_a)} Q'} (k_a \otimes \llbracket \phi_1 \rrbracket_\rho(P\|Q'))$$

Being the formula simpler than the initial one, we can apply the inductive hypothesise. As consequence, we obtain

$$\bigoplus_{P \xrightarrow{(a,k_a)} P'} (k_a \otimes \llbracket \phi_1 \rrbracket_\rho(P'\|Q)) \oplus \bigoplus_{Q \xrightarrow{(a,k_a)} Q'} (k_a \otimes \llbracket \phi_1 \rrbracket_\rho(P\|Q')) \geqslant \begin{array}{l} \bigoplus_{P \xrightarrow{(a,k_a)} P'} (k_a \otimes K_{P',\phi_1} \otimes \llbracket (\phi_1)_{//P'} \rrbracket_\rho(Q)) \oplus \\ \oplus \bigoplus_{Q \xrightarrow{(a,k_a)} Q'} (k_a \otimes K_{P,\phi_1} \llbracket (\phi_1)_{//P} \rrbracket_\rho(Q')) \end{array}$$

Note that, $K_{P',\phi_1} \geqslant (K_{P',\phi_1} \oslash K_{P,\phi}) \otimes K_{P,\phi}$ and $K_{P,\phi_1} \geqslant (K_{P,\phi_1} \oslash K_{P,\phi}) \otimes K_{P,\phi}$. Hence,

$$\bigoplus_{P \xrightarrow{(a,k_a)} P'} (k_a \otimes K_{P',\phi_1} \otimes \llbracket (\phi_1)_{//_{P'}} \rrbracket_\rho(Q)) \oplus \bigoplus_{Q \xrightarrow{(a,k_a)} Q'} (k_a \otimes K_{P,\phi_1} \llbracket (\phi_1)_{//_P} \rrbracket_\rho(Q')) \geqslant$$

(*due to previous disequalities*)

$$\bigoplus_{P \xrightarrow{(a,k_a)} P'} (k_a \otimes ((K_{P',\phi_1} \oslash K_{P,\phi}) \otimes K_{P,\phi}) \otimes \llbracket (\phi_1)_{//_{P'}} \rrbracket_\rho(Q))$$
$$\oplus \bigoplus_{Q \xrightarrow{(a,k_a)} Q'} (k_a \otimes ((K_{P,\phi_1} \oslash K_{P,\phi}) \otimes K_{P,\phi}) \llbracket (\phi_1)_{//_P} \rrbracket_\rho(Q')) =$$

(*being $K_{P,\phi}$ not regulated by $\bigoplus$*)

$$K_{P,\phi} \otimes (\bigoplus_{P \xrightarrow{(a,k_a)} P'} (k_a \otimes (K_{P',\phi_1} \oslash K_{P,\phi})) \otimes \llbracket (\phi_1)_{//_{P'}} \rrbracket_\rho(Q)) \oplus (K_{P,\phi_1} \oslash K_{P,\phi})$$
$$\bigoplus_{Q \xrightarrow{(a,k_a)} Q'} (k_a \otimes \llbracket (\phi_1)_{//_P} \rrbracket_\rho(Q'))) =$$

(*due to semantics definition of modality operators*)

$$K_{P,\phi} \otimes ((k_{P,\phi_1} \oslash k_{P,\phi}) \otimes \llbracket \langle a \rangle (\phi_1)_{//_P} \rrbracket_\rho(Q) \oplus \bigoplus_{P \xrightarrow{(a,k_a)} P'} (k_a \otimes (k_{P',\phi_1} \oslash k_{P,\phi}) \otimes \llbracket (\phi_1)_{//_{P'}} \rrbracket_\rho(Q))) =$$

(*due to definition of the QPMC function Rule* (6) *of Tab.* 3)

$$K_{P,\phi} \otimes \llbracket \langle a \rangle \phi_1 \rrbracket_\rho(Q).$$

$v =_\nu \phi E$: The base case, $E = \epsilon$ trivially holds being always $k_P = \top \geqslant_K t$. Let us consider $E = v =_\nu \phi E'$. $P \| Q \models E \Leftrightarrow \llbracket v =_\nu \phi E \rrbracket \geqslant_K t$. According to Tab. 2, this is equivalent to say that there exists a minimum value k_1 such that $\llbracket \phi E' \rrbracket_{\rho'} = k'$ and $P \| Q \models_t \phi E' \wedge t' \geqslant_K t$. We can now apply the inductive hypothesis because $k_P = \top \geqslant_K k$. Considering the definition of ρ', we recall that ϕ has to be satisfied when t', *i.e.*, the minimum value that has to be substituted to v, is substituted to v in ϕ, hence $Q \models_k (v)_{//P} =_\nu (\phi)_{//P} E'_{//P}$.

□

Moreover, if the adopted semiring is uniquely invertible (or cancellative, see Sec. 2.1) as *e.g.*, the weighted semiring, then Th. 4.1 can be further refined and $\geqslant_K$ becomes $=$. This means that the QPMC function does not find only a lower bound of $\llbracket \phi \rrbracket (P \parallel Q)$, but just its exact value. This result is formalised by Cor. 4.1.

Corollary 4.1. *Let P and Q two processes in GPA, $\mathbb{K}$ an ordered and uniquely invertible c-semiring, and ϕ a c-Eμ formula, then:*

$$\llbracket \phi \rrbracket (P \parallel Q) = k_{P,\phi} \otimes \llbracket \phi_{//_P} \rrbracket (Q).$$

Proof. Adding the hypothesis that the c-semiring $\mathbb{K}$ is uniquely invertible and ordered, $k_{P,\phi_1} = (k_{P,\phi_1} \oslash k_{P,\phi}) \otimes k_{P,\phi}$. Hence, the proof is the same of the one of Th. 4.1 with the equality instead of $k_{P,\phi_1} \geqslant_K (k_{P,\phi_1} \oslash k_{P,\phi}) \otimes k_{P,\phi}$. For instance, if $\phi = \phi_1 \oplus \phi_2$, According to Tab. 2, $\llbracket \phi \rrbracket_{P \| Q} = \llbracket \phi_1 \oplus \phi_2 \rrbracket_{P \| Q} = \llbracket \phi_1 \rrbracket_{P \| Q} \oplus \llbracket \phi_2 \rrbracket_{P \| Q}$. By inductive hypothesis, $\llbracket \phi_1 \rrbracket_{P \| Q} = k_{P,\phi_1} \otimes \llbracket (\phi_1)_{//_P} \rrbracket_Q$ and $\llbracket \phi_2 \rrbracket_{P \| Q} = k_{P,\phi_2} \otimes \llbracket (\phi_2)_{//_P} \rrbracket_Q$. Then $\llbracket \phi_1 \rrbracket_{P \| Q} \oplus \llbracket \phi_2 \rrbracket_{P \| Q} = (k_{P,\phi_1} \otimes \llbracket (\phi_1)_{//_P} \rrbracket_Q) \oplus (k_{P,\phi_2} \otimes \llbracket (\phi_2)_{//_P} \rrbracket_Q)$. Since an c-semiring is invertible, the equality $k_{P,\phi_1} = (k_{P,\phi_1} \oslash k_{P,\phi}) \otimes k_{P,\phi}$ holds. The same holds also for k_{P,ϕ_2}. Hence, the previous equation can be written as $(((k_{P,\phi_1} \oslash k_{P,\phi}) \otimes k_{P,\phi}) \otimes \llbracket (\phi_1)_{//_P} \rrbracket_Q) \oplus (((k_{P,\phi_2} \oslash k_{P,\phi}) \otimes k_{P,\phi}) \otimes \llbracket (\phi_2)_{//_P} \rrbracket_Q)$. For the associativity and commutativity of the $\otimes$ operation, this is equal to $(k_{P,\phi} \otimes ((k_{P,\phi_1} \oslash k_{P,\phi}) \otimes \llbracket (\phi_1)_{//_P} \rrbracket_Q)) \oplus (k_{P,\phi} \otimes ((k_{P,\phi_2} \oslash k_{P,\phi}) \otimes \llbracket (\phi_2)_{//_P} \rrbracket_Q))$. For

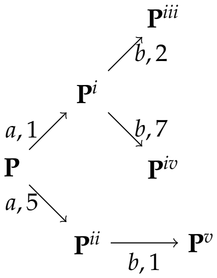

Figure 1: The MLTS of P.

the distributivity of $\otimes$ with respect to $\oplus$, this is equal to $k_{P,\phi} \otimes (((k_{P,\phi_1} \oslash k_{P,\phi}) \otimes [\![(\phi_1)_{//_P}]\!]_Q) \oplus ((k_{P,\phi_2} \oslash k_{P,\phi}) \otimes [\![(\phi_2)_{//_P}]\!]_Q))$. Hence

$$\begin{aligned} [\![\phi_1 \oplus \phi_2]\!]_{P \| Q} &= k_{P,\phi} \otimes [\![((k_{P,\phi_1} \oslash k_{P,\phi}) \otimes (\phi_1)_{//_P}) \oplus ((k_{P,\phi_2} \oslash k_{P,\phi}) \otimes (\phi_2)_{//_P})]\!]_Q \\ &= k_{P,\phi} \otimes [\![(\phi_1 \oplus \phi_2)_{//_P}]\!]_Q \end{aligned}$$

□

Hence, it is clear why it is important to accumulate weight to $k_{P,\phi}$ in the QPMC function. When $k_{P,\phi}$ is already worse than t, *i.e.*, $k_{P,\phi} <_K t$, we can avoid evaluating $[\![\phi_{//_P}]\!]_\rho(Q)$: in case *i)* the semiring is not uniquely invertible (*i.e.*, Th. 4.1) the found lower-bound is not helpful for checking the validity of ϕ, while *ii)* if the semiring is uniquely invertible (*i.e.*, Cor. 4.1), then we can immediately state that $P \parallel Q \not\models_t \phi$. Hence, *i* and *ii* can be checked only by looking at the result of QPMC, consequently saving the time for the evaluation of $\phi_{//_P}$.

The key idea behind extracting $k_{P,\phi}$ is that, as long there is only one branch of ϕ, we remove all the weight along that branch. When two branches are merged, *e.g.*, $\phi_1 \otimes \phi_2$ in Tab. 3 (3), then k_{P,ϕ_1} and k_{P,ϕ_2} from the two branches are composed according to the operator of ϕ (in this case, $k_{P,\phi} = k_{P,\phi_1} \oplus k_{P,\phi_2}$, see Tab. 4 (3)). Finally, the difference is pushed back to each branch, *e.g.*, $k_{P,\phi_1} \oslash k_{P,\phi}$ and $k_{P,\phi_2} \oslash k_{P,\phi}$, in order to not change the evaluation of $\phi_{//_P}$ w.r.t. the one of ϕ.

Referring to the complexity results described in [1], the complexity of the QPMC function is polynomial in both of the dimension of the system E. Note that the dimension of E depends on the number of formulas belonging to E and on their dimension, *i.e.*, the number of operators present in a formula. As it can be directly seen from Tab. 3, the QPMC function consists in a recursive rewriting of ϕ until its end is reached (as a reminder, the associated MLTS is finite).

Example 4.1. *We provide an example on how the QPMC function in Tab. 3 works to obtain $\phi_{//_P}$ from ϕ. We show how to move the specification of the process $P = (a,1).((b,2).0 + (b,7).0) + (a.5).(b,1).0$ (its MLTS is in Fig. 1) to a formula $\phi = [a][b]0$, and accumulate an amount of weight $k_{P,\phi}$. We work with the weighted semiring, $\langle \mathbb{N}^+ \cup \{+\infty\}, \min, \hat{+}, +\infty, 0 \rangle$.*

By considering Tab. 3 and Tab. 4 we have $k_{P,\phi} = (1 \otimes k_{P^i,\phi_1}) \oplus (5 \otimes k_{P^{ii},\phi_1})$ and $\phi_{//_P} = ((k_{P,\phi_1} \oslash k_{P,\phi}) \otimes [a](\phi_1)_{//_P}) \sqcap (((1 \otimes k_{P^i,\phi_1}) \oslash k_{P,\phi}) \otimes [a](\phi_1)_{//_{P^i}}) \sqcap (((5 \otimes k_{P^{ii},\phi_1}) \oslash k_{P,\phi}) \otimes [a](\phi_1)_{//_{P^{ii}}})$ where $\phi_1 = [b]0$. $(\phi_1)//_P = ((\top \oslash k_{P,[0]}) \otimes [b]0)$ where $k_{P,\phi_1} = \top$ Then, both $P^i = (b,2).0 + (b,7).0$ and $P^{ii} = (b,1)$ perform the action b with weights 2 and 7, and 1 respectively.

$$(\phi_1)//_{P^i} = ((\top \oslash k_{P^i,\phi_1}) \otimes [b]0) \sqcap (((2 \otimes \top) \oslash k_{P^i,\phi_1}) \otimes [b]0) \sqcap (((7 \otimes \top) \oslash k_{P^i,\phi_1}) \otimes [b]0)$$

and

$$(\phi_1)//_{P^{ii}} = ((\top \oslash k_{P^i,\phi_1}) \otimes [b]0) \sqcap (((1 \otimes \top) \oslash k_{P^{ii},\phi_1}) \otimes [b]0)$$

where $k_{P^i,\phi_1} = 2 \oplus 7 = 2$ *and* $k_{P^{ii},\phi_1} = 1$. *Finally, we obtain*

$$\underline{k_{P,\phi}} = (1 \otimes 2) \oplus (5 \otimes 1) = 3 \oplus 6 = 3$$

$$\begin{aligned}\underline{\phi_{//_P}} &= (\top \oslash 3 \otimes [a]\top) \circledcap (((1 \otimes 2) \oslash 3) \otimes [a](\phi_1)_{//_{P'}}) \circledcap (((5 \otimes 1) \oslash 3 \otimes [a](\phi_1)_{//_{P''}}) \\ &= ([a]([b]0 \circledcap [b]0 \circledcap (5 \otimes [b]0))) \circledcap (([b]0) \circledcap 3 \otimes [a][b]0)\end{aligned}$$

5 Simplification Rules and Complexity of Evaluating ϕ

When it is not possible to deduce the t-satisfiability of $\phi_{//_P}$ from $k_{P,\phi}$, then $[\![\phi_{//_P}]\!]_\rho(Q)$ needs to be necessarily evaluated. The aim of QPMC is to move P to ϕ and consequently the result $\phi_{//_P}$ may consist in a considerably longer formula. Of course, this can impact on the time needed for its evaluation. To prevent this, in this section we present some strategies that help to reduce the size of $\phi_{//_P}$ while maintaining equivalent formulas.

In Tab. 5 we show an extension of the simplification rules originally presented in [1]. We classify such rules into *simple evaluation* (*SE*), *constant propagation* (*CP*), *unguardedness removal* (*UR*), *trivial equation elimination* (*TEE*), and *equivalence reduction* (*ER*). The main differences w.r.t [1] are that *i)* in general we have more than just *true* and *false* truth values (*i.e.*, any $k \in K$), and *ii)* satisfiability is w.r.t. a threshold t. Considering i, by using the set in Tab. 5 with the boolean semiring $\langle \{F, T\}, \vee, \wedge, F, T \rangle$ we exactly obtain the rules in [1]. Hence we provide a direct extension of those simplifications. Considering ii instead, for some rules (*e.g.*, **SE1**), when part of a formula ϕ is already evaluated to a $h <_K k$, we can propagate $\bot$ instead of h. This is due to the monotonicity of c-Eμ operators, which are directly based on semiring operators $\oplus, \otimes, \circledcap$. Moreover, due to the distributivity of $\oplus$ and $\circledcap$ with respect to $\otimes$ we can define **TEE5-TEE8**.

With the aim to remain adherent to [1], our set of simplifications in Tab. 5 works with totally ordered semirings (*e.g.*, weighted, Viterbi, fuzzy, and boolean). For instance, in a partial order **SE7** cannot be applied in general, since it could be that given $h <_K t$, $k <_K t$, then $lub(h, k) = t$. If h is removed, than we are left with only $lub(k) = k <_K t$: we rewrite something that satisfied t into something that does not satisfy t anymore, but the rules in Tab. 5 are only required to simplify a modality, not to change its evaluation. Still similarly to Andersen [1], we restrict ourselves to the use of *simple assertions*, *i.e.*, formulas without $\oplus$ between $[\cdot]$ modalities, and $\otimes$ between $\langle \cdot \rangle$ modalities.

Note that the cost of the simplification using the rules in Tab. 5 is linear in the dimension of E, since it just corresponds to a rewriting of each formula in E.

Finally, it is possible to perform a pre-processing *reachability-analysis* step as in [1]: the computation of all bound variables that are not required in the satisfaction of top assertion $E_{\downarrow v}$ can be safely discarded, since their satisfaction is useless with respect to v.

Complexity considerations. As advanced in previous sections, both the application of QPMC and simplification rules have polynomial time-complexity. In this paragraph, we provide the last the complexity consideration, related to the verification of a formula ϕ written in c-Eμ.

The evaluation of ϕ is based on the computation of a system of fix points, starting from a top assertion $E_{\downarrow v}$: in general, their iteration may become infeasible in case the MLTS is infinite (but ours is finite, see Def. 2.6), or if the semiring set K is infinite; hence, this represents an obstacle. The literature only provides some complexity upper-bounds on distributive c-semirings with infinite K [28], *i.e.*, in case $\otimes$ is idempotent (*e.g.*, the fuzzy semiring). The basic idea is that even if the domain of the c-semiring, and hence the corresponding lattice, are infinite, only a

Simple Evaluation			
SE1	$\models_t v =_{\mu/\nu} \bigotimes\{h, \phi_1, \ldots, \phi_n\}$	$\Longleftrightarrow$	$\models_t v =_{\mu/\nu} \bot$ *if* $h <_K t$
SE2	$\models_t v =_{\mu/\nu} \bigotimes\{\top, \phi_1, \ldots, \phi_n\}$	$\Longleftrightarrow$	$\models_t v =_{\mu/\nu} \bigotimes\{\phi_1, \ldots, \phi_n\}$
SE3	$\models_t v =_{\mu/\nu} \text{\textcircled{$\sqcap$}}\{h, \phi_1, \ldots, \phi_n\}$	$\Longleftrightarrow$	$\models_t v =_{\mu/\nu} \bot$ *if* $h <_K t$
SE5	$\models_t v =_{\mu/\nu} \text{\textcircled{$\sqcap$}}\{\top, \phi_1, \ldots, \phi_n\}$	$\Longleftrightarrow$	$\models_t v =_{\mu/\nu} \text{\textcircled{$\sqcap$}}\{\phi_1, \ldots, \phi_n\}$
SE6	$\models_t v =_{\mu/\nu} \bigoplus\{\top, \phi_1, \ldots, \phi_n\}$	$\Longleftrightarrow$	$\models_t v =_{\mu/\nu} \top$
SE7	$\models_t v =_{\mu/\nu} \bigoplus\{h, \phi_1, \ldots, \phi_n\}$	$\Longleftrightarrow$	$\models_t v =_{\mu/\nu} \bigoplus\{\phi_1, \ldots, \phi_n\}$ *if* $h <_K t$
SE8	$\models_t v =_{\mu/\nu} \langle a \rangle h$	$\Longleftrightarrow$	$\models_t v =_{\mu/\nu} \bot$ *if* $h <_K t$
SE9	$\models_t v =_{\mu/\nu} [a]h$	$\Longleftrightarrow$	$\models_t v =_{\mu/\nu} \bot$ *if* $h <_K t$
Constant Propagation			
CP1	$\models_t v =_{\mu/\nu} \phi$ $\vdots$ $\models_t w =_{\mu/\nu} h$	$\Longleftrightarrow$	$\models_t v =_{\mu/\nu} \phi[h/w]$ $\vdots$ $\models_t w =_{\mu/\nu} h$ *if* $h \geqslant_K t$
CP2	$\models_t v =_{\mu/\nu} \phi$ $\vdots$ $\models_t w =_{\mu/\nu} h$	$\Longleftrightarrow$	$\models_t v =_{\mu/\nu} \phi[\bot/w]$ $\vdots$ $\models_t w =_{\mu/\nu} \bot$ *if* $h <_K t$
Unguardedness Removal (w unguarded [1])			
UR	$\models_t v =_{\mu/\nu} \psi$ $\vdots$ $\models_t w =_{\mu/\nu} \phi$	$\Longleftrightarrow$	$\models_t v =_{\mu/\nu} \psi[\phi/w]$ $\vdots$ $\models_t w =_{\mu/\nu} \phi$
Trivial Equation Elimination			
TEE1	$\models_t v =_\mu \langle a \rangle v$	$\Longleftrightarrow$	$\models_t v =_\mu \bot$
TEE2	$\models_t v =_\nu [a]v$	$\Longleftrightarrow$	$\models_t v =_\nu \top$
TEE3	$\models_t \phi \text{\textcircled{$\sqcap$}} \phi$	$\Longleftrightarrow$	$\models_t \phi$
TEE4	$\models_t \phi \oplus \phi$	$\Longleftrightarrow$	$\models_t \phi$
TEE5	$\models_t \phi_1 \oplus (\phi_1 \otimes \phi_2)$	$\Longleftrightarrow$	$\models_t \phi_1$
TEE6	$\models_t \phi_1 \oplus (\phi_1 \text{\textcircled{$\sqcap$}} \phi_2)$	$\Longleftrightarrow$	$\models_t \phi_1$
TEE7	$\models_t \phi_1 \text{\textcircled{$\sqcap$}} (\phi_1 \otimes \phi_2)$	$\Longleftrightarrow$	$\models_k \phi_1 \otimes \phi_2$
TEE8	$\models_t \phi_1 \text{\textcircled{$\sqcap$}} (\phi_1 \text{\textcircled{$\sqcap$}} \phi_2)$	$\Longleftrightarrow$	$\models_t \phi_1 \text{\textcircled{$\sqcap$}} \phi_2$
Equivalence Reduction			
ER1	$\models_t v =_\mu \phi_1$ $\models_t w =_\mu \phi_2$	$\Longleftrightarrow$	$\models_t v =_\mu \phi_1 \oplus \phi_2$ $\models_t w =_\mu v$
ER2	$\models_t v =_\nu \phi_1$ $\models_t w =_\nu \phi_2$	$\Longleftrightarrow$	$\models_t v =_\nu \phi_1 \text{\textcircled{$\sqcap$}} \phi_2$ $\models_t w =_\nu v$

Table 5: Valid transformations on assertion equations: v, w are variables and $h, t \in K$. In equivalene reduction, v, w are variables in the same block, and equal on E [1].

finite sub-lattice is involved in the evaluation of ϕ. Consequently, the number of steps needed to evaluate ϕ is in this case finite.

In Th. 5.1 we provide complexity-considerations that exploit the distributive hypothesis, adapting a result in [27] to our framework.

Theorem 5.1 (Bound for distributive c-semirings)**.** *Given a distributive c-semiring* $\mathbb{K} = \langle K, \oplus, \otimes, \bot, \top \rangle$ *and* $M = (S, Act, \mathbb{K}, T, s_0)$, $\vDash_t E_{\downarrow v}$ *can be computed in* $O(|E| \cdot h(FD(g(\Phi))))$, *where* Φ *collects all the formulas in* $E_{\downarrow v}$ *with only free variables.*

Proof. While the result in [27] reports the complexity of computing just one fix-point, we extend it by considering an equational system of $|E|$ fix-points. $g(\Phi)$ represents the finite set of evaluation-lattice generators given the set Φ of ϕ in $E_{\downarrow v}$ and do not contain bound variable, computed as follows: $g(\Phi) = \{[\![\phi]\!]_\rho(s_0) \mid \phi \in \Phi_M\}$. $FD(K')$ denotes the domain of the free distributive lattice generated by a finite $K' \subseteq K$ by applying semiring operators $\oplus$, $\otimes$, and $\bigcirc\!\!\!\!\frown$. A coarse upper bound on the size of $|FD(K')|$ is $2^{(2^{|K'|})}$ [28]. However, if the lattice order is total (*e.g.*, in the fuzzy semiring), we can have a smaller number of elements, *i.e.*, $|FD(K')| = |K'|$: $a \oplus b$ and $a \otimes b$ always return either a or b. Finally, $h()$ returns the height of a finite lattice (*i.e.*, the longest chain of elements w.r.t. $\geqslant_{K'}$), which gives the maximal number steps to compute the result of a single fix-point over such a semiring. □

The issue with non-distributive semiring, as $\langle \mathbb{N}^+ \cup \{+\infty\}, \min, +, +\infty, 0 \rangle$ (*i.e.*, the weighted semiring), is that to evaluate *e.g.*, $\phi = (v =_\mu v \otimes 2)$, we need infinite iterations to get ∞ as final result: this means the evaluation lattice is infinite. However, taking advantage of what we are interested in, that is t-satisfiability (see Def. 3.2), in some cases we can compute an upper bound for non-distributive semirings as well. Clearly the intuition is that threshold t ($\vDash_t$) limits the number of computation steps to something finite. Theorem 5.2 shows a complexity bound for evaluating ϕ over the weighted c-semiring.

Theorem 5.2 (t-limited upper-bound)**.** *Given the weighted semiring* $\langle \mathbb{N}^+ \cup \{+\infty\}, \min, +, +\infty, 0 \rangle$ *and an* $MLTS = (S, Act, \mathbb{K}, T, s_0)$, $\vDash_t E_{\downarrow v}$ *can be computed in* $O(|E| \cdot N^{|I|})$, *where N is the number of solutions of a* Linear Diophantine Inequality $a_1 + a_2x_2 + \ldots + a_rx_r \leqslant t$; $\{a_1, \ldots, a_n\}$ *is the subset of co-prime generators of the lattice in which the computation happens.*

Proof. According to the considerations made for Th. 5.1, the issue is that the $\otimes$ operation is not idempotent. Following the proof of Th. 5.1, given in [27], the main goal is to identify the dimension of the lattice generated by the elements in $g(\Phi)$. In the case of $\langle \mathbb{N}^+ \cup \{+\infty\}, \min, +, +\infty, 0 \rangle$, this corresponds to solve a *Linear Diophantine Inequality* [29] $a_1x_1 + a_2x_2 + \ldots + a_rx_r \leqslant t$, in which $\{a_1, a_2, \ldots, a_r\}$ is the subset of co-prime generators in $g(\Phi)$, found as in Th. 5.1. The number N of solutions for such kind of inequality with natural coefficient $a_1, a_2, \ldots, a_r$ is finite and computable. Hence, $height(FD(g(\Phi)))$ is equal to N. In [29] the authors propose an approach to exactly find this number. Here we show an estimation of N, which ranges between two values [29]:

$$\frac{t^r}{r! \prod_{i=1}^{r} a_i} \leqslant N \leqslant \frac{(t + a_1 + a_2 + \ldots + a_r)^r}{r! \prod_{i=1}^{r} a_i}$$

□

5.1 A Simplification Example

Let us consider the well-know *Chinese-Wall* access-control policy: given two sets of resources (*e.g.*, files or data) *fd1* and *fd2*, it is possible to access either to *fd1* or to *fd2*, but if an access to *fd1* is performed, then it is no more possible to access to *fd2* anymore. This is expressed by a formula $\phi = \phi_1 \oplus \phi_2$, where

$$\phi_1 = v =_\nu [access_{fd1}]v \otimes [access_{fd2}]\bot \qquad \phi_2 = w =_\nu [access_{fd2}]w \otimes [access_{fd1}]\bot$$

Let us consider a system $P \parallel Q$, where $P = (access_{fd1}, 5).P + (access_{fd1}, 7).P$; Q is not interested by this example (it could be e.g., $(access_{fd2}, 2)$). Each action is weighted by a value expressing how many clock-intervals such an action takes. To model the number of time-slots we use $\langle \mathbb{N}^+ \cup \{+\infty\}, \min, +, +\infty, 0\rangle$. Hence, at the same time we have a well-known modality, and non-functional aspects to be checked. In this example, we require $(P \parallel Q) \vDash_{10} \phi = \phi_1 \oplus \phi_2$. By using the QPMC function, we partially evaluate ϕ w.r.t. process P:

$$\begin{array}{lllll}
\phi_{//P} & = & {\phi_1}_{//P} & \oplus & {\phi_2}_{//P} \\
\\
{\phi_1}_{//P} & = & v_{//P} & =_\nu & ([access_{fd1}]v \otimes [access_{fd2}]\bot)_{//P} \\
 & & & =_\nu & ([access_{fd1}]v)_{//P} \otimes ([access_{fd2}]\bot)_{//P} \\
 & & & =_\nu & ([access_{fd1}]v_{//P} \bigcirc\!\!\!\!\!\!\sqcap\; (v_{//P} \bigcirc\!\!\!\!\!\!\sqcap\; (2 \otimes v_{//P}))) \otimes ([access_{fd2}]\bot \bigcirc\!\!\!\!\!\!\sqcap\; \top)) \\
\\
{\phi_2}_{//P} & = & w_{//P} & =_\nu & ([access_{fd2}]w \otimes [access_{fd1}]\bot)_{//P} \\
 & & & =_\nu & ([access_{fd2}]w)_{//P} \otimes ([access_{fd1}]\bot)_{//P} \\
 & & & =_\nu & ([access_{fd2}]w_{//P} \bigcirc\!\!\!\!\!\!\sqcap\; (\top)) \otimes ([access_{fd1}]\bot \bigcirc\!\!\!\!\!\!\sqcap\; (\bot \bigcirc\!\!\!\!\!\!\sqcap\; (2 \otimes \bot)))
\end{array}$$

Since $K_{P,\phi} = 5$ ($K_{P,\phi_1} \oplus K_{P,\phi_2} = 5 \oplus 7$) and $5 \geqslant_{\mathbb{K}} 10$, we cannot stop the evaluation of ϕ (see Sec. 5) and we need to simplify it by using Tab. 5:

- By rule **TEE7**, $(v_{//P} \bigcirc\!\!\!\!\!\!\sqcap\; (2 \otimes v_{//P}))$ becomes $(2 \otimes v_{//P})$;
- By rule **TEE7**, $(\bot \bigcirc\!\!\!\!\!\!\sqcap\; (2 \otimes \bot))$ becomes $(2 \otimes \bot) = \bot$;
- By semiring property, $([access_{fd1}]\bot \bigcirc\!\!\!\!\!\!\sqcap\; \bot) = \bot$;
- By rule **SE1**, $([access_{fd2}]w_{//P} \bigcirc\!\!\!\!\!\!\sqcap\; (\top)) \otimes ([access_{fd1}]\bot \bigcirc\!\!\!\!\!\!\sqcap\; \bot) = ([access_{fd2}]w_{//P} \bigcirc\!\!\!\!\!\!\sqcap\; (\top)) \otimes \bot$ becomes $\bot$;
- By semiring property $[access_{fd2}]\bot \bigcirc\!\!\!\!\!\!\sqcap\; \top$ is reduced to $[access_{fd2}]\bot$.

Hence, both ${\phi_2}_{//P} = \bot$ and ${\phi_1}_{//P} = v_{//P} =_\nu ([access_{fd1}]v_{//P} \bigcirc\!\!\!\!\!\!\sqcap\; (2 \otimes v_{//P})) \otimes ([access_{fd2}]\bot)$ have been reduced in size after the application of the simplification rules in Tab. 5, and consequently also the size of $\phi_{//P} = {\phi_1}_{//P} \oplus {\phi_2}_{//P}$.

6 Related Work

Partial model checking has been used in several different contexts, such as state-based models [2, 3], synchronous state/event systems [11], and timed systems [13, 25, 26]. It has also been specialised to check security properties [30] and for adapting one process to another without disclosing its full behaviour [8, 9].

The most direct comparison is with the work in [1]. This paper promotes a quantitative view of such a work: our formulas are not evaluated only to true/false, but they are associated with a semiring value. At the same time, the QPMC in [1] has been here revisited by supposing a weighted transition-system. The heuristics in Tab. 5 flatten to those in [1] by adopting (again) the boolean semiring. For this reason, their validity is comparable to the one proved in that work. Moreover, some of them now exploit the threshold as well.

The second most-direct comparison is with [27], where the extension of μ-calculus to semirings is similar to our logic. Differently from [27] however, c-Eμ comes in an equational form, and it is applied to *à la CCS* GPA-processes. We also focus on QPMC, while the model-checking in [27] is not partial, and no simplification rules are provided. Finally, defining t-satisfiability we estimate a bound also in some cases where the semiring is not distributive.

In the literature we can find a plethora of quantitative model-checking approaches and tools, mainly consisting in probabilistic extensions for systems exhibiting a stochastic behaviour. To name a few [14, 17, 20, 22, 23].

In [14], given a continuous-time Markov Chain and a linear real-time property provided as a deterministic timed automaton, the goal is evaluate the probability of the set of accepted paths.

The work in [23] criticises the use of a threshold for stating the truth of a probabilistic CTL formula (*i.e.*, $p \geqslant t$) directly into the specification of the formula. This mainly avoid to range over the full unit interval and the author guesses that "the inherent difficulties of guessing which threshold to use" can be overcome in several (*e.g.*, security-related) scenarios, while other issues are related to the infiniteness of systems (not treated here). Furthermore, our representation of formulas does not use threshold, that is only used in the definition of satisfaction.

In [22] the authors present *PRISM*, a probabilistic model checking that includes the ability to compute cost- and reward-based measures. Real values are assigned to states and transitions of the model. This permits reasoning about a much wider range of quantitative measures of a system.[4]

In [20] a Monte Carlo approximation algorithm for LTL model-checking is presented. The procedure delivers quantitative information about the likelihood that $S \models \phi$.

In [17] the authors model-check a *Quantaitive Linear Time Logic* (*QLTL*) over *Quantitative Transition Systems* (*QTS*s) and *Quantitative Markov Chains* (*QMC*s). QTSs and QMCs are respectively Kripke structures and Markov chains whose atomic propositions have values in $[0, 1]$, rather than in $\{0, 1\}$.

In [24] the authors associate each transition in *weighted modal transition systems* with an interval of weights, implementing a sort of "loose" specification by using both negative and positive preferences. This can be achieved by using bipolar-semiring structures [18]. In addition, the interval idea suggests a re-phrasal our framework into a *Soft Constraint Satisfaction Problem* (*SCSP*) [6, 18], where weights correspond to explicit constraints on transitions. Hence, finding a solution of a SCSP leads to satisfying all the intervals.

To summarise, we are not aware of proposals weighing a partial variant of model-checking. These two features can be respectively useful in case it is not possible to have a full system specification, and to deal with weights that are not probabilities, but lattices of preferences.

7 Conclusion

Partial Model-Checking [1] consists in incrementally incorporating into a formula ϕ the behavioural information taken from a process P. The new formula $\phi_{//_P}$ can be verified on a smaller composition of processes $P \parallel Q$. Simplification rules are necessary to keep $\phi_{//_P}$ short,

[4] Other tools supporting probabilistic model-checking can be found at http://www.prismmodelchecker.org/other-tools.php.

before performing its evaluation. To summarise, having ϕ and a parallel execution $P \parallel Q$, the steps consist in first *i)* applying QPMC on ϕ obtaining $\phi_{//_P}$, *ii)* then applying the simplification rules to reduce the size of $\phi_{//_P}$, and finally *iii)* evaluating $[\![\phi_{//_P}]\!]$. The first steps have a polynomial time-complexity, and they can be used to reduce the time to evaluate ϕ, which is in general exponential instead.

The PMC function presented in this paper is extended *i)* to consider modalities weighted in a c-semiring algebraic structure, and *ii)* by also extracting a weight $k_{P,\phi}$, which may consist in an upper bound on the evaluation of $k_{P,\phi}$. Hence, by only applying QPMC we can totally avoid to evaluate any formula, if $k_{P,\phi}$ is already worse than a threshold t that defines t-satisfiability. We also provide simplifications by giving a weighted interpretation of what presented in [1]: the result is a set of simplification rules that work at least as good as those tested in [1] in case the boolean c-semiring is used, since they exactly reconnect to [1], where tests are provided.

As the last result of the paper, we show that, by using a threshold t is now possible to have a complexity upper-bound on the verification of a formula with non-distributive semirings, extending the work in [27]: we show the specific case of the weighted semiring, where the evaluation of ϕ becomes computable in a finite number of steps.

Future Work. We would like to provide more general upper-bounds on the verification complexity of formulas in case the adopted semiring is not distributive. In addition, we would like to prototype the QPMC function and the presented simplification rules by using *Maude* [15]. We also plan to test the efficacy of simplifications by generating random formulas, as performed in [1]. Therefore, we would also like to use Maude to also program all the transformations given in Sec. 5, and, finally, collect a random benchmark of formulas and processes and have an empirical proof of their efficacy. As a reminder, this has been already proved in [1], since it corresponds to our framework in case the boolean semiring is adopted.

Finally, we aim to model-check semiring-based *Soft Concurrent Constraint Programming* languages as [4, 7], since their transition systems can be directly labelled with a semiring value representing the best available solution in the current store of constraints.

References

[1] H. R. Andersen. Partial model checking. In *LICS '95*, page 398. IEEE Computer Society, 1995.

[2] H. R. Andersen, J. Staunstrup, and N. Maretti. A comparison of modular verification techniques. In *TAPSOFT'97: Theory and Practice of Software Development, 7th International Joint Conference CAAP/-FASE*, volume 1214 of *LNCS*, pages 550–564. Springer, 1997.

[3] H. R. Andersen, J. Staunstrup, and N. Maretti. Partial model checking with robdds. In *Tools and Algorithms for Construction and Analysis of Systems, Third International Workshop, TACAS*, volume 1217 of *LNCS*, pages 35–49. Springer, 1997.

[4] S. Bistarelli, M. Gabbrielli, M. C. Meo, and F. Santini. Timed soft concurrent constraint programs: An interleaved and a parallel approach. *TPLP*, 15(6):743–782, 2015.

[5] S. Bistarelli and F. Gadducci. Enhancing constraints manipulation in semiring-based formalisms. In *ECAI*, pages 63–67, 2006.

[6] S. Bistarelli, U. Montanari, and F. Rossi. Semiring-based constraint satisfaction and optimization. *J. ACM*, 44(2):201–236, 1997.

[7] S. Bistarelli and F. Santini. A secure non-monotonic soft concurrent constraint language. *Fundam. Inform.*, 134(3-4):261–285, 2014.

[8] S. Bistarelli, F. Santini, F. Martinelli, and I. Matteucci. Automated adaptation via quantitative partial model checking. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, pages 1993–1996. ACM, 2016.

[9] S. Bistarelli, F. Santini, F. Martinelli, and I. Matteucci. A formal and run-time framework for the adaptation of local behaviours to match a global property. In *Formal Aspects of Component Software - 13th International Conference, FACS 2016, Revised Selected Papers*, LNCS. Springer, 2016.

[10] T. S. Blyth and M. F. Janowitz. *Residuation theory*, volume 102. Pergamon press Oxford, 1972.

[11] N. O. Bodentien, J. Vestergaard, Jacob Friis, K. J Kristoffersen, and K. G Petersen. Verification of state/event systems by quotienting. *BRICS Report Series*, 6(41), 1999.

[12] P. Buchholz and P. Kemper. Quantifying the dynamic behavior of process algebras. In *Proceedings of PAPM-PROBMIV '01*, pages 184–199. Springer, 2001.

[13] F. Cassez and F. Laroussinie. Model-checking for hybrid systems by quotienting and constraints solving. In *Computer Aided Verification, 12th International Conference, CAV*, volume 1855 of *LNCS*, pages 373–388. Springer, 2000.

[14] T. Chen, T. Han, J.-P. Katoen, and A. Mereacre. Quantitative model checking of continuous-time markov chains against timed automata specifications. In *Proceedings of the 24th Annual IEEE Symposium on Logic in Computer Science, LICS*, pages 309–318. IEEE Computer Society, 2009.

[15] M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer, and J. F. Quesada. Maude: specification and programming in rewriting logic. *Theor. Comput. Sci.*, 285(2):187–243, 2002.

[16] B. Davey and H. Priestley. *Introduction to lattices and order*. Cambridge university press, 2002.

[17] M. Faella, A. Legay, and M. Stoelinga. Model checking quantitative linear time logic. *ENTCS*, 220(3):61–77, 2008.

[18] F. Gadducci and F. Santini. Residuation for bipolar preferences in soft constraints. *Inf. Process. Lett.*, 118:69–74, 2017.

[19] J. Golan. *Semirings and affine equations over them: theory and applications*. Kluwer Academic Pub., 2003.

[20] R. Grosu and S. A. Smolka. Monte carlo model checking. In *Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 3440 of *LNCS*, pages 271–286. Springer, 2005.

[21] M. Hatzel, C. Wagner, K. Peters, and U. Nestmann. Encoding CSP into CCS. In *Proceedings of the Combined 22th International Workshop on Expressiveness in Concurrency and 12th Workshop on Structural Operational Semantics, EXPRESS/SOS*, volume 190 of *EPTCS*, pages 61–75, 2015.

[22] A. Hinton, M. Z. Kwiatkowska, G. Norman, and D. Parker. PRISM: A tool for automatic verification of probabilistic systems. In *Tools and Algorithms for the Construction and Analysis of Systems, 12th International Conference, TACAS*, volume 3920 of *LNCS*, pages 441–444. Springer, 2006.

[23] M. Huth and M. Z. Kwiatkowska. Quantitative analysis and model checking. In *Proceedings, 12th Annual IEEE Symposium on Logic in Computer Science,*, pages 111–122. IEEE Computer Society, 1997.

[24] L. Juhl, K. G. Larsen, and J. Srba. Modal transition systems with weight intervals. *J. Log. Algebr. Program.*, 81(4):408–421, 2012.

[25] F. Laroussinie and K. G. Larsen. Compositional model checking of real time systems. In *CONCUR '95: Concurrency Theory*, volume 962 of *LNCS*, pages 27–41. Springer, 1995.

[26] F. Laroussinie and K. G. Larsen. CMC: A tool for compositional model-checking of real-time systems. In *Formal Description Techniques and Protocol Specification, Testing and Verification, FORTE XI*, volume 135 of *IFIP Conference Proceedings*, pages 439–456. Kluwer, 1998.

[27] A. Lluch-Lafuente and U. Montanari. Quantitative mu-calculus and CTL defined over constraint semirings. *TCS*, 346(1):135–160, 2005.

[28] F. Lunnon. The IU function: The size of a free distributive lattice. *Combinatorial Mathematics and its Applications, Academic Press, London*, pages 173–181, 1971.

[29] R. Mahmoudvand, H. Hassani, A. Farzaneh, and G. Howell. The exact number of nonnegative integer solutions for a linear diophantine inequality. *IAENG International Journal of Applied Mathematics*, 40(1):1–5, 2010.

[30] F. Martinelli. Symbolic partial model checking for security analysis. In *Computer Network Security, Second International Workshop on Mathematical Methods, Models, and Architectures for Computer Network Security, MMM-ACNS 2003*, volume 2776 of *LNCS*, pages 122–134. Springer, 2003.

[31] F. Martinelli, I. Matteucci, and F. Santini. Semiring-based specification approaches for quantitative security. In *Proceedings Thirteenth Workshop on Quantitative Aspects of Programming Languages and Systems, QAPL*, volume 194 of *EPTCS*, pages 95–109, 2015.

[32] F. Rossi, P. Van Beek, and T. Walsh. *Handbook of constraint programming*. Elsevier, 2006.

[33] A. Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific journal of Mathematics*, 5(2):285–309, 1955.