Counter machines, Petri Nets, and Consensual Computation[☆]

Stefano Crespi Reghizzi[a,b], Pierluigi San Pietro[a,b]

[a]*Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Italia*
[b]*CNR IEEIT, Milano*

Abstract

We establish the relation between two language recognition models that use counters and operate in real-time: Greibach's partially blind machines operating in real time (RT-PBLIND), which recognize Petri Net languages, and the consensually regular (CREG) language model of the authors. The latter is based on synchronized computational threads of a finite automaton, where at each step one thread acts as the leader and all other threads as followers. We introduce two new normal forms of RT-PBLIND machines (and Petri Nets), such that counter operations are scheduled and rarefied, and transitions are quasi-deterministic, i.e., the finite automaton obtained by eliminating counter moves is deterministic. We prove that the CREG family can simulate any normalized RT-PBLIND machine, but it also contains the non-RT-PBLIND language $\{a^n b^n \mid n > 1\}^*$.

Keywords: formal languages, multi-counter machine, real-time partially blind machine, Petri Net language, Petri Net normal form, quasi-deterministic counter machine, consensual language, multiset machine, modulo scheduling.

1. Introduction

Multi-Counter Machines (MCM) have been used since half a century to model integer computer programs [20, 21] and to recognize formal languages, yet their theory is less established than, say, the theory of push-down machines and context-free languages. Several MCM types exist depending on the operations permitted on counters, on determinism, and on other constraints about reversals and spontaneous moves (see e.g., [11, 13, 17]). Other language families offer a double characterization by grammars and by machines, but only the most restricted MCM subfamilies (such as the one-counter languages) enjoy some sort of generative model.

[☆]Work partially supported by MIUR project PRIN 2010LYA9RH-006.

Email addresses: stefano.crespireghizzi@polimi.it (Stefano Crespi Reghizzi), pierluigi.sanpietro@polimi.it (Pierluigi San Pietro)

Our nontraditional approach to MCM languages offers some promise to clarify the complex computations that have so far hindered their full understanding. Notably, we obtain the possibility to specify counter languages by means of regular expressions that describe interacting computational threads in a rather perspicuous way.

This paper focuses on a classical and important MCM family: the *real-time partially blind machines* [13] to be denoted by RT-PBLIND. Counters are non-negative integers that can be tested for zero only in the final configuration; if the machine attempts to decrement a zero counter, it crashes. In terms of languages, such machines are equivalent to a natural family of Petri Nets firing sequences [13]. They are more powerful than the more popular reversal-bounded MCM [17]. The RT-PBLIND language family is closed with respect to union and intersection, concatenation, and alphabetic (direct or inverse) homomorphism; it includes also languages that have a nonsemilinear Parikh image.

The other model, the *consensually regular* languages (CREG), was introduced by the authors in [5] and further studied in [6, 7, 9, 8, 10]. Deferring its formal presentation to Sect. 3, we intuitively describe its functioning. Consider a nondeterministic real-time finite-state automaton (NFA), and classify each edge of the transition graph as *leader* or *follower*. To *consensually* recognize an input word w of length $|w|$, the NFA may need one or more computations, called threads, each of them accepting the word, and such that at each step $1 \leq i \leq |w|$, one, and only one, thread traverses a leader edge. Therefore at time i all remaining threads traverse a follower edge. In some sense, this *matching* discipline is similar to a token-passing scheduling policy, the temporary leader being the thread with the token. The standard finite-state recognition corresponds to the case of a thread that, for all moves, is a leader. The adjective *consensual* expresses the metaphor that the leader thread reads the current input character, but without the consent of all other threads the computation cannot proceed.

The number of threads i.e., the parallelism of the consensual device, is bounded by the input length, since a thread that at all times is a follower would be useless. The memory of the consensual device is encoded in the current states of all active threads, and can be represented by a multi-set of states, which motivates the name of *multiset machine*. Such machine could be rightly called a multi-counter machine, but we prefer to reserve this name to the classical models.

Word recognition for an RT-PBLIND (Petri Net) language requires logarithmic space complexity, and the same property holds for the multi-set machine. Apart from this similarity, the two models look at first glance very different, but, in an effort to assess the power of CREG, we recently proved that some RT-PBLIND languages are consensually regular: the deterministic RT-PBLIND [8] languages, and the commutative languages that have a semilinear Parikh image [10]. Here we

prove a much more general and unexpected result: the strict inclusion of nondeterministic RT-PBLIND (therefore also of Petri net) languages in CREG. While the two mentioned inclusions had been proved by special transformations of consensual languages, here we directly simulate an RT-PBLIND machine on a multiset machine. An RT-PBLIND move may be nondeterministic in two manners: by moving to multiple next states (as an NFA) and by performing different increments or decrements of counters. For the latter manner, we prove that any RT-PBLIND machine can be transformed to an equivalent RT-PBLIND machine, called *quasi-deterministic* (QD), such that the underlying finite-state automaton is deterministic. Then we show how to simulate any QD RT-PBLIND machine on a multiset machine. At last, the strict inclusion RT-PBLIND $\subset$ CREG and the incomparability of non-RT PBLIND machines and CREG are proved by means of witnesses.

To obtain the QD normal form, we reschedule each counter operation at an ordinal time, which belongs to a specific congruence class, modulo a large enough integer. Machine moves are in this way simplified (at most one counter is affected in one move) and counter operations are rarefied. Such transformations are not new for the more powerful MCM types that are allowed to test counters for zero [11], but in our case we had to develop a very thorough analysis.

We briefly mention similarities and differences of CREG with respect to some language families other than the multi-counter ones, which have been already considered.

Essential features of concurrent and parallel computations are captured in various, old and recent, formal models. Several models compose sequential threads, represented by strings, by means of the *interleaving* or *shuffle* operation. The concurrent regular expressions of [12] are an example; a more recent one is the work on *shuffled languages* [2]. To augment the effect of the pure shuffle operation on regular languages, various directions have been explored. Several authors, e.g., the two latter citations, add the transitive closure of the shuffle. Another possibility for enlarging the language family, exploits the synchronized shuffle operation, which exists in different versions, see [26]. Our consensual model is based on a lock-step synchronization called matching, that bears some similarity to a synchronized shuffle.

We notice another interesting analogy between the multiset consensual device and other devices based on multiple coordinated runs of a finite-state machine. Such devices have been proposed in different settings, as in the machine that recognizes the shuffle closure of regular languages [19]. A recent more complex device is the concurrent machine that recognizes the shuffled languages of [2]: such machine stores in its configuration an unbounded number of states, organized as a tree, that assigns a partial order to state activation. In contrast, the consensual device simply stores a multiset of states, and requires that all states consensually

fire at each step.

At last, there is an enduring interest for models that are able to represent the commutation or partial commutation of words, starting from classical language families. Two recent examples are the restarting automata [23] and the context-free grammars enhanced with rules that interchange adjacent subrees [22]. The consensual model is not entirely dissimilar and is able to recognize a subfamily of commutative languages [10].

To sum up, in our opinion the consensual model is not directly comparable with any existing proposals, and can be considered as a minimalist attempt to capture some language features pertinent to concurrent and parallel computations, using a simple yet intriguing extension to the finite-state model.

Concerning applications, our new normal forms may be interesting for proving properties of multi-counter machines and Petri Nets. Since consensual languages can be defined by finite automata or by regular expressions on an alphabet, which enriches the terminal alphabet with leader/follower roles, it becomes possible to define multi-counter and Petri Net languages using such established notations. We hope that this novel specification style will be convenient in some areas, such as formal specifications, where counter machines are used.

The paper is organized as follows. Sect. 2 deals with RT-PBLIND machines and Petri Nets. It defines the classical models and introduces two new normal forms: the rarefied and modulo-scheduled form, and the QD form. It describes the corresponding normal forms of Petri Nets. Sect. 3 defines the CREG languages and devices and recalls known properties. Sect. 4 constructs the multiset machine recognizing CREG languages, and proves that family CREG strictly contain family RT-PBLIND but is incomparable with PBLIND. Sect. 5 presents some open problems. The Appendix includes further examples of the main constructions.

2. Partially Blind Multi-counter Machines

We define the nondeterministic, real time multi-counter machine known as partially blind. Then we define transformations affecting the parallelism and timing of counter operations. Then, we define the new quasi-deterministic machine model, and we prove that it can simulate any machine of the general types. This simulation is interesting *per se* and is the key to the main result in the Sect. 4.

Let Σ denote a finite terminal alphabet and ε the empty word. For a word w, $|w|$ denotes the length of w; the i-th letter of w is $w(i)$, $1 \leq i \leq |w|$, i.e., $w = w(1)w(2)\dots w(|w|)$; the number of occurrences in w of a letter $a \in \Sigma$ is denoted as $|w|_a$.

A *finite automaton* (FA) A is a tuple $(\Sigma, Q, \delta, q_0, F)$ where: Q is a finite set of states; $\delta : Q \times \Sigma \to 2^Q$ is the state-transition function, q_0 is the initial state,

and $F \subseteq Q$ is the set of final states. If for every pair q, a, $|\delta(q,a)| \leq 1$, then A is deterministic (DFA), otherwise is nondeterministic (NFA). For a DFA we write $\delta(q,a) = q'$ instead of $\{q'\}$.

2.1. Models of multi-counter real-time automata

We deal almost exclusively with machines operating in *real-time* (RT), i.e., such that each move reads an input character. We list the classic definitions for counter machines, see for instance [13, 16]. A counter is just an integer variable. Given an integer $m > 0$, a *counter valuation* is a vector $\overrightarrow{X} \in \mathbb{N}^m$. Each element of the vector is denoted by $\overrightarrow{X}(i)$ and is called the *value of the i-th counter*. Let I^m be the set of all words in $\{-1, 0, 1\}^m$; each word is interpreted as an *increment vector* $\overrightarrow{Y}$ operating on $\mathbb{N}^m$. Intuitively, $\overrightarrow{Y}$ encodes an m-tuple of simultaneous operations, such that if $\overrightarrow{Y}(i) = 1$, 0 or -1, then the i-th counter is, respectively, incremented, left unchanged or decremented.

Definition 2.1 (Real-Time Nondeterministic Partially Blind machine)**.** *A nondeterministic,* real-time, partially blind, multi-counter machine *(RT-PBLIND) is a tuple* $\mathcal{M} = \langle \Sigma, S, \gamma, m, s_1, S_{fin} \rangle$*, where:*

- Σ *is a finite alphabet;*
- S *is a finite set of* states*,* $s_1 \in S$ *is* initial*, and* $S_{fin} \subseteq S$ *is the set of* final *states;*
- $m \geq 0$ *is the number of counters;*
- $\gamma \subseteq S \times \Sigma \times I^m \times S$ *is a set of* transitions*, called the transition relation.*

Notice that the domain of γ does not refer to counter values, i.e., $\mathcal{M}$ cannot check them: zero-testing is not allowed (hence, $\mathcal{M}$ is "blind"), except at the end of computation. Moreover, only nonnegative counter values are allowed, therefore if $\mathcal{M}$ attempts to decrement a null counter during a run, then the run is non-accepting – informally, the machine "crashes".

A *configuration* of $\mathcal{M}$ is a pair in $S \times \mathbb{N}^m$. The *initial* configuration is $(s_1, \overrightarrow{0}^m)$; a configuration $(s, \overrightarrow{0}^m)$, with $s \in S_{fin}$, is *final*. A *move* of $\mathcal{M}$ is an element of relation

$$\longrightarrow_{\mathcal{M}} \subseteq (S \times \mathbb{N}^m) \times \Sigma \times (S \times \mathbb{N}^m)$$

defined, $\forall a \in \Sigma, \forall s_i, s_j \in S, \forall \overrightarrow{X}_i, \overrightarrow{X}_j \in \mathbb{N}^m$, as follows:

$$(s_i, \overrightarrow{X}_i) \xrightarrow{a}_{\mathcal{M}} (s_j, \overrightarrow{X}_j) \text{ if } \exists \overrightarrow{Y} \in I^m \mid (s_i, a, \overrightarrow{Y}, s_j) \in \gamma \text{ and } \overrightarrow{X}_j = \overrightarrow{X}_i + \overrightarrow{Y} \qquad (1)$$

Relation $\longrightarrow_{\mathcal{M}}$ is extended as usual to $(S \times \mathbb{N}^m) \times \Sigma^* \times (S \times \mathbb{N}^m)$: if $w \in \Sigma^n$ for $n > 0$, then $(s, \overrightarrow{X}) \xrightarrow{w}_{\mathcal{M}} (s_n, \overrightarrow{X}_n)$ if there exist $\overrightarrow{X}_1, \ldots \overrightarrow{X}_{n-1} \in \mathbb{N}^m, s_{i_1}, \ldots, s_{i_n} \in S$ such that:

$$(s, \overrightarrow{X}) \xrightarrow{w(1)}_{\mathcal{M}} (s_{i_1}, \overrightarrow{X}_1) \xrightarrow{w(2)}_{\mathcal{M}} \cdots \xrightarrow{w(n)}_{\mathcal{M}} (s_{i_n}, \overrightarrow{X}_n). \qquad (2)$$

When $(s, \overrightarrow{X})$ is the initial configuration, sequence (2) is called a *run* of $\mathcal{M}$ with *label* w and *length* n. If, moreover, $(s_{i_n}, \overrightarrow{X}_n)$ is a final configuration, the run is *accepting*. The *language accepted* by an RT-PBLIND $\mathcal{M}$ is the set $L(\mathcal{M})$ of labels of accepting runs of $\mathcal{M}$. An example of machine is shown in Fig. 1 (left). The *family of languages* accepted by RT-PBLIND machines will be denoted by the same acronym, and similarly for the other devices considered. We list some properties.

Proposition 2.2 (Properties of RT-PBLIND language family)**.** *[14]*
• *RT-PBLIND is the least family of languages that contains the Dyck language over two letters* $D_2 = \{w \in \{a, b\}^* \mid |w|_a = |w|_b$ *and if* $w = xy$ *then* $|x|_a \geq |x|_b\}$ *and that is closed under intersection, union, non-erasing alphabetic homomorphism, inverse alphabetic homomorphism;*
• *RT-PBLIND is not closed under Kleene* $*$*, since it includes* $\{a^n b^n \mid n \geq 1\}$ *but not*

$$L_3 = (\{a^n b^n \mid n \geq 1\})^* \tag{3}$$

• *RT-PBLIND is closed under concatenation and reversal operation; it strictly includes reversal-bounded counter languages [17]; it is incomparable with context-free languages; it includes also non-semilinear (in the Parikh sense) languages.*

At the end of Sect. 4, we consider a more general machine model, denoted by PBLIND, that may perform *spontaneous* moves, i.e., the domain of the transition function is changed to $\gamma \subseteq S \times \Sigma \cup \{\varepsilon\} \times I^m \times S$.

2.2. Petri Nets

Petri Nets (PN) are a widespread formal model of concurrent systems and can be viewed also as a device for recognizing languages. Various PN language families have been studied (e.g., [18]), and we focus on the definition in [14], which was intended to be as close as possible to MCM. We recall the classical result that this family of PN languages coincides with RT-PBLIND. Later, we discuss the significance of our new normal forms for PNs'.

Definition 2.3. *[14] A k-place Petri Net (PN)* $\mathcal{P} = (k, \Sigma, T, F)$ *consists of a finite number k of* places *denoted as* $P(1), \ldots, P(k)$*, a finite set* Σ *of input symbols, a subset* $F \subseteq \{P(1), \ldots, P(k)\}$ *of* accepting places*, and a finite set* $T \subseteq \mathbb{N}^k \times \Sigma \times \mathbb{N}^k$ *of* labelled transitions*. An* instantaneous description *(ID) of* $\mathcal{P}$ *is a member of* $\Sigma^* \times \mathbb{N}^k$*. In an ID* $(w, n_1, \ldots, n_k)$*, we call w the input to be processed, and* n_i *the number of tokens in place* P_i*.*
Define the relation $\mapsto \subseteq \Sigma^* \times \mathbb{N}^k \times \Sigma^* \times \mathbb{N}^k$ *as follows. If there exist an ID* $(aw, y_1, \ldots, y_k)$*, with* $a \in \Sigma, w \in \Sigma^*$*, and a transition* $(u_1, \ldots, u_k, a, v_1, \ldots, v_k) \in T$ *such that* $u_i \leq y_i$ *for all* $1 \leq i \leq k$*, then*

$$(aw, y_1, \ldots, y_k) \mapsto (w, y_1 - u_1 + v_1, \ldots, y_k - u_k + v_k)\,.$$

and speak of this move as "firing" the above transition. An ID $(\varepsilon, n_1, \ldots, n_k)$ *is* accepting *if, for some* $P(j) \in F$, $n_j = 1$ *and* $n_i = 0$ *for all* $i \neq j$.

We denote by $\overset{*}{\mapsto}$ *the reflexive and transitive closure of* $\mapsto$. *The language accepted by* $\mathcal{P}$ *is*

$$L(\mathcal{P}) = \left\{ w \in \Sigma^* \mid \textit{there is an accepting ID } I \textit{ such that } (w, 1, 0, \ldots, 0) \overset{*}{\mapsto} I \right\}.$$

$L(\mathcal{P})$ *is called the* computation sequence set *(CSS) of* $\mathcal{P}$, *or* Petri Net language.

Clearly [14], such PN $\mathcal{P}$ is already defined as a state-less RT-PBLIND machine, if we regard places as counters, with the following restrictions and modifications. In one move $\mathcal{P}$ can operate on any number of counters; it can add or subtract integers larger than 1; first it subtracts, then it adds. $\mathcal{P}$ starts with the 1 in the first counter and 0 in all others. $\mathcal{P}$ accepts when all but one final counter (place) contains 0. Such multicounter machine can be easily converted by standard constructions into the RT-PBLIND model of Def. 2.1.

Theorem 2.4. (stated by Greibach [14] without claiming originality)
The family CSS of Petri Net languages coincides with the family of languages, not containing the empty word, recognized by RT-PBLIND machines.

The PN equivalent to an RT-PBLIND machine is shown in Fig. 1.

2.3. A Rarefied Normal Form for RT-PBLIND machines

It is known, at least since [11], that for certain counter machine models, it is possible to rarefy counter operations, i.e., to impose that the machine alters its counters at most every v steps, for some integer $v > 0$. The construction of the equivalent "rarefied" counter machine sketched in [11] (Th. 1.1 and Th. 1.2) relies on a compressed representation of counter values: when in the original machine counter i has value $\overrightarrow{X}(i)$, the new machine stores in counter i the value $\underline{\overrightarrow{X}}(i) = \lfloor \frac{\overrightarrow{X}(i)}{v} \rfloor$ instead, retaining the residue $p = \overrightarrow{X}(i) \mod v$ in finite-state memory. Hence, given a residue p and a compressed counter value $\underline{\overrightarrow{X}}(i)$, the original counter has value $\overrightarrow{X}(i) = v\underline{\overrightarrow{X}}(i) + p$. In the model of [11], counters can be negative, but their signs can also be represented in the finite-state memory (by testing a counter for 0 before decrementing it): one can further assume that counters may store only nonnegative values.

We are going to present a similar construction for RT-PBLIND, which is more complicated because both of the zero testing limitation of RT-PBLIND and of the stronger "rarefaction" condition that we need, introduced in the next definitions.

Definition 2.5 (Scheduled machine)**.** *The i-th counter,* $1 \leq i \leq m$*, of an RT-PBLIND machine* $\mathcal{M} = \langle \Sigma, S, \gamma, m, s_1, S_{fin} \rangle$ *is* scheduled *with module* $v \geq 2$ *and residues* $\boxminus, \boxplus$*, with* $0 \leq \boxminus < \boxplus \leq v - 1$*, if for all* $w \in \Sigma^*$*, for all* $a \in \Sigma$*, for all* $s, s' \in S$*, for all* $\overrightarrow{X}, \overrightarrow{X'} \in \mathbb{N}^m$*, such that* $(s_1, \overrightarrow{0}^m) \xrightarrow{w}_{\mathcal{M}} (s, \overrightarrow{X}) \xrightarrow{a}_{\mathcal{M}} (s', \overrightarrow{X'})$*, the following conditions hold:*

$$\text{if } \overrightarrow{X'}(i) = \overrightarrow{X}(i) + 1, \text{ then } |wa| = \boxplus \mod v;$$
$$\text{if } \overrightarrow{X'}(i) = \overrightarrow{X}(i) - 1, \text{ then } |wa| = \boxminus \mod v.$$

If counter i is scheduled with modulo v and residues $\boxminus, \boxplus$, then it can be incremented only if the length, modulo v, of the input string read so far is $\boxplus$ and it can be decremented only if the above length is $\boxminus$. [1] Since $\boxminus < \boxplus$ and counters start at zero, in the very first move machine $\mathcal{M}$ cannot modify any (scheduled) counter.

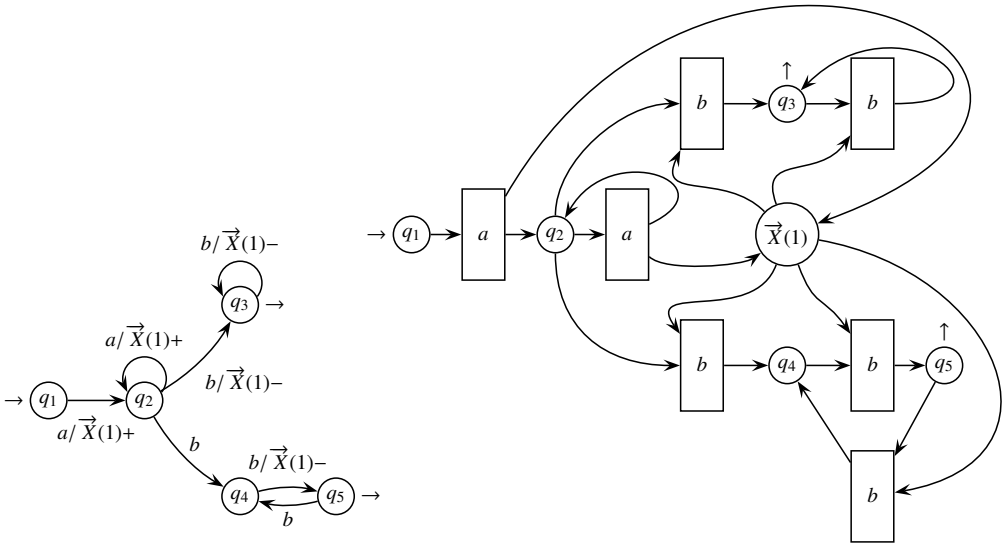


Figure 1: (Left) RT-PBLIND (nondeterministic) 1-counter machine. For readability, increment and decrement operations on counter 1 are denoted by $\overrightarrow{X}(1)+$ and $\overrightarrow{X}(1)-$, respectively. The language recognized is $L_{2.7} = \{a^n b^n \mid n > 0\} \cup \left\{a^n b^{2n} \mid n > 0\right\}$. (Right) Equivalent Petri Net; initial configuration with one token in place q_1 and final configuration with one token either in q_3 or in q_5.

Definition 2.6 (Rarefied machine)**.** *An RT-PBLIND machine with m counters is* h-rarefied*,* $h > 0$*, if there exist an integer* $v \geq 2hm$ *and a sequence R of* $2m$ *non-negative integers* $\langle \boxminus_1, \boxplus_1, \ldots, \boxminus_m, \boxplus_m \rangle$ *such that for all* r, r' *in R, with* $r' > r$*, we*

[1] Some readers may notice the analogy with the cyclic process scheduling techniques applied in real-time operating systems [25].

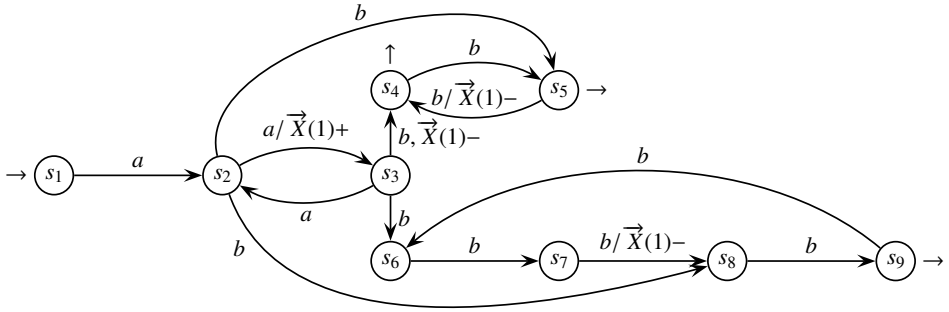


Figure 2: RT-PBLIND (nondeterministic) machine with the counter scheduled by module $v = 2$ and residues $\boxminus = 0, \boxplus = 1$. The machine is equivalent to the machine and the PN in Fig. 1.

have both $r' - r \geq h$ and $v - r' + r \geq h$, and such that each counter i, $1 \leq i \leq m$, is scheduled with module v and residues $\boxminus_i, \boxplus_i$.

Intuitively, if a machine is h-rarefied, then any two counter modifications are separated by at least $h - 1$ steps that do not modify counters. (Such steps will be exploited in a later construction that needs to place further operations on extra counters.) In particular, even in a 1-rarefied machine, at most one counter can be modified in a single move. Clearly, if $h > 1$, a h-rarefied machine is also $(h - 1)$-rarefied.

Example 2.7. Scheduled multi-counter machine
We show in Fig. 1 (left) an RT-PBLIND machine with one counter and the equivalent Petri Net (right). Then Fig. 2 shows an equivalent machine with counting operations scheduled in accordance with Def. 2.5. We choose the minimum value $v = 2$ for the module, and we schedule decrement and increment actions respectively at residue $\boxminus = 0, \boxplus = 1$. The machine is thus 1-rarefied.

Theorem 2.8. *For every RT-PBLIND machine, for all $h \geq 1$ there exists an equivalent h-rarefied RT-PBLIND machine.*

Proof. Let $\mathcal{M} = \langle \Sigma, S, \gamma, m, s_1, S_{fin} \rangle$ be an RT-PBLIND, and let v be an integer such that $v \geq 2hm$. We may assume that S admits a partition into v subsets, denoted $[\![0]\!], [\![1]\!], \cdots, [\![v-1]\!]$ such that: for all $w \in \Sigma^*, s \in S, \overrightarrow{X} \in \mathbb{N}^m$, if $(s_1, \overrightarrow{0}^m) \xrightarrow{w}_{\mathcal{M}} (s, \overrightarrow{X})$, then $s \in [\![|w| \mod v]\!]$. If this is not the case, just define an equivalent RT-PBLIND machine as the product of $\mathcal{M}$ with a FA $(\Sigma, Q, \delta, q_0, Q)$, where $Q = \{q_0, \ldots, q_{v-1}\}$, and for all $a \in \Sigma$, $\delta(q_i, a) = q_j$ with $j = i + 1 \mod v$. The standard construction of the product is omitted.

We show that given a counter i, $1 \leq i \leq m$, for every $0 \leq \boxminus < \boxplus \leq v - 1$, there exists an equivalent RT-PBLIND machine, called $\mathcal{M}(v, \boxminus, \boxplus)$, such that it

is scheduled for the i-th counter, with module v and residues $\boxminus, \boxplus$, and does not modify the behavior of the remaining counters. The main statement follows then immediately. In fact, by applying repeatedly the construction (shown below), we eventually obtain a machine which is h-rarefied for every $h > 0$ such that $2hm \leq v$, by making each one of its counters scheduled and by selecting for each counter a distinct pair of residues.

Define the finite set of relative integers $P = \{-v+1, \ldots, 2v-1\}$. Let $\mathcal{M}(v, \boxminus, \boxplus)$ be $\langle \Sigma, T, \eta, m, t_0, T_{fin} \rangle$, where $T = S \times P \times \{0, -1\}$, $t_0 = \langle s_1, 0, 0 \rangle$, $T_{fin} = S_{fin} \times \{0\} \times \{0\}$ and the transition relation η is defined below, after an intuitive explanation.

Intuition about the definition of $\mathcal{M}(v, \boxminus, \boxplus)$. The idea is that $\mathcal{M}(v, \boxminus, \boxplus)$ is a "compressed" version of $\mathcal{M}$, i.e., it uses a compressed representation of counter $\overrightarrow{X}(i)$, together with its finite-state memory, to avoid counter operations when they are not scheduled. In particular, if $(s, \overrightarrow{X})$ is a configuration of $\mathcal{M}$, for a state $s \in S$ and a vector $\overrightarrow{X} \in \mathbb{N}^m$, then the "compressed" version of this configuration for $\mathcal{M}(v, \boxminus, \boxplus)$ is $\left(\langle s, p, b \rangle, \underline{\overrightarrow{X}}\right)$, where $p \in P$ is a "pseudo-residue", $b \in \{0, -1\}$ is a "flag" needed to detect if the counter may have become negative (as detailed below), and $\underline{\overrightarrow{X}} \in \mathbb{N}^m$ is a vector such that $\overrightarrow{X}(i) = v\underline{\overrightarrow{X}}(i) + p, \overrightarrow{X}(j) = \underline{\overrightarrow{X}}(j), \forall j \neq i$. Thus, $\underline{\overrightarrow{X}}(i)$ contains the value of $\overrightarrow{X}(i)$, but "reduced" by factor v, with p acting as a sort of "residue" to store the result of original counter increments/ decrements executed in unwanted positions. For instance, if the pseudo-residue $p = 2$, $\overrightarrow{X}(i)$ is decremented in $\mathcal{M}$ and, at the current position, the counter is not scheduled for a decrement, then $\underline{\overrightarrow{X}}$ is left unchanged and the pseudo-residue becomes 1; if the pseudo-residue $p = 0$, $\overrightarrow{X}(i)$ is decremented in $\mathcal{M}$ and, at the current position, the counter is scheduled for a decrement then $\underline{\overrightarrow{X}}$ is decremented and the new pseudo-residue is $v - 1$.

The pseudo-residue is needed to keep track of the value of counter i between two subsequent scheduled operations on i. Since such interval can be larger than $v-1$ and filled with original increments/decrements of counter i, the pseudo-residue is not a residue in the arithmetical sense, since it can be greater than $v - 1$ or even negative. Next, we consider cases where increments or a decrements of $\overrightarrow{X}(i)$ must be delayed further to comply with the schedule.

First, the pseudo-residue can become greater than v. In fact, suppose that the pseudo-residue has value $p = v - 1$, after reading a number of symbols equal to $\boxplus$ mod v. Therefore, p is too small for $\mathcal{M}(v, \boxminus, \boxplus)$ to make an increment of $\underline{\overrightarrow{X}}(i)$. If $\mathcal{M}$, while reading the next v symbols in the input string, goes through a sequence of up to $v - 1$ further increments of $\overrightarrow{X}(i)$, then the run of $\mathcal{M}(v, \boxminus, \boxplus)$ may arrive at a configuration $(\langle s', p', b \rangle, \underline{\overrightarrow{X}})$, p' as large as $v - 1 + v - 1 = 2v - 2$, since $\underline{\overrightarrow{X}}(i)$ may

only be incremented when in state s' (at that time it will decrement p' of the value v).

Second, the pseudo-residue can become negative; it may happen that a crash of $\mathcal{M}$ remains hidden for a finite number of steps. To manage this situation, we use the third component of $\mathcal{M}(v, \boxminus, \boxplus)$ states, i.e., the flag. Consider the case when $p = 0$, after reading a number of symbols equal to $\boxminus \mod v$. Therefore, $\underline{\overrightarrow{X}}$ cannot be decremented before reading v input symbols more: in between, $\mathcal{M}$ may go through a sequence of up to $v - 1$ consecutive decrements of $\overrightarrow{X}(i)$. Therefore, $\mathcal{M}(v, \boxminus, \boxplus)$ must be able to store negative values in the pseudo-residue, as small as $-(v - 1)$. When a pseudo-residue becomes negative, it may be that machine $\mathcal{M}$ has actually crashed: if $\underline{\overrightarrow{X}}(i) = 0$, then $\mathcal{M}$ must have tried to decrement a zero counter, while if $\underline{\overrightarrow{X}}(i) > 0$, then $\mathcal{M}$ has not crashed but just decremented $\overrightarrow{X}(i)$ and then continued. However, $\mathcal{M}(v, \boxminus, \boxplus)$, as described so far, cannot distinguish these two cases, since it can neither test if $\underline{\overrightarrow{X}}(i) = 0$ nor decrement $\underline{\overrightarrow{X}}(i)$, since the counter is not scheduled for the current position. Therefore, $\mathcal{M}(v, \boxminus, \boxplus)$ cannot check if its configuration is "valid", in the sense that it corresponds to a reachable configuration of $\mathcal{M}$: for instance, $(\langle s, -1, 0\rangle, \overrightarrow{0}^m)$ is clearly not valid, since it would correspond to a negative value for $\overrightarrow{X}(i)$. Since the only way for $\mathcal{M}(v, \boxminus, \boxplus)$ to check whether $\underline{\overrightarrow{X}}(i) = 0$ is to decrement $\underline{\overrightarrow{X}}(i)$, $\mathcal{M}(v, \boxminus, \boxplus)$ must decrement counter $\underline{\overrightarrow{X}}(i)$ as soon as possible (i.e., in the next scheduled position). The necessity of such future decrement is stored in the finite-memory control in the flag b, where the value -1 "remembers" that $\mathcal{M}(v, \boxminus, \boxplus)$ must still decrement counter $\underline{\overrightarrow{X}}(i)$, since the pseudo-residue has turned negative. Notice that it would not be enough to check if the pseudo-residue is negative, since successive increments may hide a preceding negative value: for instance, $(\langle s, 1, -1\rangle, \overrightarrow{0}^m)$ is not valid, although it might apparently correspond to a configuration of $\mathcal{M}$ where $\overrightarrow{X}(i) = 1$, since to reach $(\langle s, 1, -1\rangle, \overrightarrow{0}^m)$, machine $\mathcal{M}(v, \boxminus, \boxplus)$ must have gone through the invalid configuration $(\langle s, -1, 0\rangle, \overrightarrow{0}^m)$.

Definitions and properties. As explained intuitively before, not all configurations of $\mathcal{M}(v, \boxminus, \boxplus)$ are "valid", i.e., they actually correspond to configurations of $\mathcal{M}$. There are two cases of invalid configurations: first, when $\underline{\overrightarrow{X}}(i) = 0$ and $p < 0$, which correspond to an actual negative value of the counter $\overrightarrow{X}(i)$; second, when $\underline{\overrightarrow{X}}(i) = 0$ and $b = -1$, which models the case where the counter $\overrightarrow{X}(i)$ should have become negative before possibly turning nonnegative again. Formally, a configuration $(\langle s, p, b\rangle, \underline{\overrightarrow{X}})$ of $\mathcal{M}(v, \boxminus, \boxplus)$ is *valid* when:

$$\underline{\overrightarrow{X}}(i) > 0 \vee (p \geq 0 \wedge b = 0). \tag{4}$$

The correspondence from valid configurations of $\mathcal{M}(v, \boxminus, \boxplus)$ to configurations of $\mathcal{M}$ is formalized by the *partial mapping* $\rightarrowtail: T \times \mathbb{N}^m \to S \times \mathbb{N}^m$:

for all $s \in S, \overrightarrow{X} \in \mathbb{N}^m, p \in P, b \in \{0, -1\}$, if $(\langle s, p, b\rangle, \underline{\overrightarrow{X}})$ is valid, then

$$\left(\langle s, p, b\rangle, \underline{\overrightarrow{X}}\right) \rightarrowtail (s, \overrightarrow{X}), \text{ with } \overrightarrow{X}(i) = v\underline{\overrightarrow{X}}(i) + p, \overrightarrow{X}(j) = \underline{\overrightarrow{X}}(j) \text{ for } j \neq i.$$

We now define relation η. For all $a \in \Sigma$, for all $s, s' \in S$, for all $\overrightarrow{Y}, \underline{\overrightarrow{Y}} \in I^m$, for all $p, p' \in P, b, b' \in \{0, -1\}$, if $(s, a, \overrightarrow{Y}, s') \in \gamma$ then

$$(\langle s, p, b\rangle, a, \underline{\overrightarrow{Y}}, \langle s', p', b'\rangle) \in \eta \text{ if, and only if:}$$

1. $\underline{\overrightarrow{Y}}(j) = \overrightarrow{Y}(j)$ for all $j \neq i$;
2. $p' = p + \overrightarrow{Y}(i) - v\underline{\overrightarrow{Y}}(i)$;

and

(I) if $s \in [\![\boxminus]\!]$ and $(p < 0 \vee p + \overrightarrow{Y}(i) < 0 \vee b = -1)$, then $\underline{\overrightarrow{Y}}(i) = 1$ and $b' = 0$;

(II) else if $s \notin [\![\boxminus]\!]$ and $(p < 0 \vee p + \overrightarrow{Y}(i) < 0)$, then $\underline{\overrightarrow{Y}}(i) = 0$ and $b' = -1$;

(III) else if $s \in [\![\boxplus]\!]$ and $(p + \overrightarrow{Y}(i) \geq v \wedge b = 0)$, then $\underline{\overrightarrow{Y}}(i) = -1$ and $b' = b$;

(IV) otherwise, $\underline{\overrightarrow{Y}}(i) = 0$ and $b' = b$;

Nothing else is in η.

The following properties derive from the definition of η and $\rightarrowtail$, for all $s, s' \in S, p, p' \in P, b, b' \in \{0, -1\}, \underline{\overrightarrow{X}} \in \mathbb{N}^m, \underline{\overrightarrow{Y}} \in I^m, a \in \Sigma, \left(\langle s, p, b\rangle, a, \underline{\overrightarrow{Y}}, \langle s', p', b'\rangle\right) \in \eta$:

(A) Let $(\langle s, p, b\rangle, \underline{\overrightarrow{X}})$ be valid, with $(\langle s, p, b\rangle, \underline{\overrightarrow{X}}) \rightarrowtail (s, \overrightarrow{X})$. If also configuration $(\langle s', p', b'\rangle, \underline{\overrightarrow{X}} + \underline{\overrightarrow{Y}})$ is valid, then there exists $\overrightarrow{Y} \in I^m$ such that $(s, a, \overrightarrow{Y}, s') \in \gamma$ and $(\langle s', p', b'\rangle, \underline{\overrightarrow{X}} + \underline{\overrightarrow{Y}}) \rightarrowtail (s', \overrightarrow{X} + \overrightarrow{Y})$.

(B) If $(\langle s, p, b\rangle, \underline{\overrightarrow{X}})$ is not valid, then $(\langle s', p', b'\rangle, \underline{\overrightarrow{X}} + \underline{\overrightarrow{Y}})$ is not valid as well.

For the proof of (A), it is enough to show that: $\overrightarrow{X}(i) + \overrightarrow{Y}(i) = v\left(\underline{\overrightarrow{X}}(i) + \underline{\overrightarrow{Y}}(i)\right) + p'$.

By definition of η, p' is $p + \overrightarrow{Y}(i) - v\underline{\overrightarrow{Y}}(i)$ hence,

$$v\left(\underline{\overrightarrow{X}}(i) + \underline{\overrightarrow{Y}}(i)\right) + p' = v\underline{\overrightarrow{X}}(i) + v\underline{\overrightarrow{Y}}(i) + p + \overrightarrow{Y}(i) - v\underline{\overrightarrow{Y}}(i),$$

which, since by hypothesis $\overrightarrow{X}(i) = v\underline{\overrightarrow{X}}(i) + p$, is equal to $\overrightarrow{X}(i) + \overrightarrow{Y}(i)$.

The proof of (B) follows from negating the definition of a valid configuration: we have $\underline{\overrightarrow{X}}(i) = 0 \wedge (p < 0 \vee b = -1)$. First notice that, if $s \in [\![\boxminus]\!]$, then there is no configuration reachable from $(\langle s, p, b\rangle, \underline{\overrightarrow{X}})$ while reading a. In fact, by definition

of η (since $p < 0 \vee b = -1$) there exists $\overrightarrow{Y} \in I^m$ such that $\underline{\overrightarrow{Y}}(i) = -1$ and $p' = p + \overrightarrow{Y}(i) - v\underline{\overrightarrow{Y}}(i) = p + \overrightarrow{Y}(i) + v$; but $\underline{\overrightarrow{X}}(i) = 0$, hence $\mathcal{M}(v, \boxminus, \boxplus)$ must "crash" by reading a (since it tries to decrement a zero counter). Therefore, $s \notin [\![\boxminus]\!]$. If $b = -1$, then by definition of η also $b' = -1$ (since the case $b = -1, b' = 0$ in η is possible only when $s \in [\![\boxminus]\!]$). If $p < 0$, then again in the definition of η the only possible case is (II), hence also $b' = -1$.

An immediate consequence of (B) is that only a valid configuration can lead to acceptance. Hence, invalid configurations are not productive, in the sense that there is no final configuration reachable from an invalid one.

Proof of $L(\mathcal{M}(v, \boxminus, \boxplus)) \subseteq L(\mathcal{M})$. We show, by induction on the length $k \geq 0$ of a word $w \in \Sigma^k$, that for all $\langle s, p, b\rangle \in T, \overrightarrow{X} \in \mathbb{N}^m$, if $(\langle s, p, b\rangle, \underline{\overrightarrow{X}})$ is valid and $(\langle s_1, 0, 0\rangle, \overrightarrow{0}^m) \xrightarrow{w}_{\mathcal{M}(v,\boxminus,\boxplus)} (\langle s, p, b\rangle, \underline{\overrightarrow{X}})$, then there exists $\overrightarrow{X} \in \mathbb{N}^m$ such that $(\langle s, p, b\rangle, \underline{\overrightarrow{X}}) \rightarrowtail (s, \overrightarrow{X})$ and $(s_1, \overrightarrow{0}^m) \xrightarrow{w}_{\mathcal{M}} (s, \overrightarrow{X})$.

The inclusion of $L(\mathcal{M}(v, \boxminus, \boxplus))$ in $L(\mathcal{M})$ follows then immediately: a final configuration has the form $(\langle s, 0, 0\rangle, \overrightarrow{0}^m)$, with $s \in S_{fin}$, hence it is valid: therefore, $(\langle s, 0, 0\rangle, \overrightarrow{0}^m) \rightarrowtail (s, \overrightarrow{0}^m)$, which means that also $\mathcal{M}$ reaches a final configuration.

The base case $w = \varepsilon$ is obvious, since $(\langle s_1, 0, 0\rangle, \overrightarrow{0}^m) \rightarrowtail (s_1, \overrightarrow{0}^m)$. Assume the induction hypothesis for $w \in \Sigma^k$, hence $(\langle s, p, b\rangle, \underline{\overrightarrow{X}}) \rightarrowtail (s, \overrightarrow{X})$, and let $a \in \Sigma$: $(\langle s, p, b\rangle, \underline{\overrightarrow{X}}) \xrightarrow{a}_{\mathcal{M}(v,\boxminus,\boxplus)} (\langle s', p', b'\rangle, \underline{\overrightarrow{X}} + \underline{\overrightarrow{Y}})$, for some $s', p'b', \underline{\overrightarrow{Y}}$. Hence, $(\langle s, p, b\rangle, a, \underline{\overrightarrow{Y}}, \langle s', p', b'\rangle) \in \eta$. By definition of η, there exists $\overrightarrow{Y} \in I^m$ such that $(s, a, \overrightarrow{Y}, s') \in \gamma$, $p' = p + \overrightarrow{Y}(i) - v\underline{\overrightarrow{Y}}(i)$ and $\underline{\overrightarrow{Y}}(j) = \overrightarrow{Y}(j)$ for $j \neq i$. Therefore, if it is not the case that $\overrightarrow{X}(i) = 0$ and $\overrightarrow{Y}(i) = -1$ (which make $\mathcal{M}$ crash), $(s, \overrightarrow{X}) \xrightarrow{a}_{\mathcal{M}} (s', \overrightarrow{X} + \overrightarrow{Y})$. By Property (A) above, if $(\langle s', p', b'\rangle, \underline{\overrightarrow{X}} + \underline{\overrightarrow{Y}})$ is valid then $(\langle s', p', b'\rangle, \underline{\overrightarrow{X}} + \underline{\overrightarrow{Y}}) \rightarrowtail (s', \overrightarrow{X} + \overrightarrow{Y})$, thus proving the induction hypothesis for wa.

If, instead, $\overrightarrow{X}(i) = 0$ and $\overrightarrow{Y}(i) = -1$ (i.e., if $\mathcal{M}$ crashes), then we show that $\langle s', p', b'\rangle, \underline{\overrightarrow{X}} + \underline{Y}(i))$ is not valid, thus ending the proof. By induction hypothesis, $\underline{\overrightarrow{X}}(i) = \overrightarrow{X}(i) = 0$, hence $p = 0$. If $\overrightarrow{X}(i) = 0$, then, since $\mathcal{M}(v, \boxminus, \boxplus)$ does not crash reading a, it must be $s \notin [\![\boxminus]\!]$. Since $p = 0$ and $\overrightarrow{Y}(i) = -1$, then $p + \overrightarrow{Y}(i) < 0$: we are in Case (II) of the definition of η. Therefore $b' = -1$ and $\underline{\overrightarrow{X}}(i) + \underline{Y}(i) = \underline{\overrightarrow{X}}(i) = 0$: $\langle s', p', b'\rangle, \underline{\overrightarrow{X}} + \underline{Y}(i))$ is not valid.

Proof of $L(\mathcal{M}) \subseteq L(\mathcal{M}(v, \boxminus, \boxplus))$. For all integers k, r, with $k \geq 0$ and $0 \leq r \leq v-1$, define $[k]_{v,r} = k$ if $k \leq r$, $[k]_{v,r} = 1 + (k + v - r - 1 \mod v)$. Hence, for a word w of length $k > r$, the value $[k]_{v,r}$ is the distance between the rightmost position of w (i.e., k) and the rightmost position (excluding k itself) equal, modulo v, to r. For

instance, if $k = 18, r = 6, v = 10$, then we have $[18]_{10,6} = 1 + (18 + 10 - 6 - 1) \bmod 10 = 2$, which is the distance between positions 18 and 16; if $k = 44, r = 6, v = 10$, then we have $[44]_{10,6} = 1 + (44 - 6 - 1) \bmod 10 = 8$, which is the distance between positions 44 and 36. Notice that if k is equal to r modulo v then by definition $[k]_{v,r} = v$ since the smallest value equal to r modulo v is in this case $k - v$, e.g., for $k = 18, r = 8, v = 10$, $[18]_{10,8} = 10$.

We show by induction on the length $k \geq 0$ of a word $w \in \Sigma^k$ that for all $s \in S, \overrightarrow{X} \in \mathbb{N}^m$ if

$$(s_1, \overrightarrow{0}^m) \xrightarrow{w}_{\mathcal{M}} (s, \overrightarrow{X})$$

then there exists a configuration $\phi = (\langle s, p, b\rangle, \underline{\overrightarrow{X}})$, for some $p \in P, b \in \{0, -1\}, \underline{\overrightarrow{X}} \in \mathbb{N}^m$, such that all the following conditions hold:

$$\phi \text{ is valid}, \phi \rightarrowtail (s, \overrightarrow{X}), \tag{5}$$

$$-[k]_{v,\boxminus} < p < v + [k]_{v,\boxplus} \text{ and} \tag{6}$$

$$(\langle s_1, 0, 0\rangle, \overrightarrow{0}^m) \xrightarrow{w}_{\mathcal{M}(v,\boxminus,\boxplus)} \phi. \tag{7}$$

The claim that $L(\mathcal{M}) \subseteq L(\mathcal{M}(v, \boxminus, \boxplus))$ follows then immediately. In fact, if $\underline{\overrightarrow{X}}(i) = 0$, then $\overrightarrow{X}(i) = 0 \wedge p = 0 \wedge b = 0$, because $p > -[k]_{v,\boxminus} > -v$, hence it is impossible that, e.g., $\overrightarrow{X}(i) = 1 \wedge p = -v$ holds. By definition of $\mathcal{M}(v, \boxminus, \boxplus)$, if $s \in S_{fin}$, then $\langle s, 0, 0\rangle \in T_{fin}$.

The base case $w = \varepsilon$ is obvious. Let $w \in \Sigma^k$ and let $(s_1, \overrightarrow{0}^m) \xrightarrow{w}_{\mathcal{M}} (s, \overrightarrow{X}) \xrightarrow{a}_{\mathcal{M}} (s', \overrightarrow{X} + \overrightarrow{Y})$ for $a \in \Sigma, s' \in S, \overrightarrow{Y} \in I^m$. Hence, $(s, a, \overrightarrow{Y}, s') \in \gamma$.

By induction hypothesis, $(\langle s_1, 0, 0\rangle, \overrightarrow{0}^m) \xrightarrow{w}_{\mathcal{M}(v,\boxminus,\boxplus)} (\langle s, p, b\rangle, \underline{\overrightarrow{X}})$, with $(\langle s, p, b\rangle, \underline{\overrightarrow{X}}) \rightarrowtail (s, \overrightarrow{X})$. We show that

$$(\langle s, p, b\rangle, \underline{\overrightarrow{X}}) \xrightarrow{a}_{\mathcal{M}(v,\boxminus,\boxplus)} \phi' \tag{8}$$

for some configuration $\phi' = (\langle s', p', b'\rangle, \underline{\overrightarrow{X}} + \underline{\overrightarrow{Y}})$, satisfying Cond. (5) and (6), thus also verifying Cond. (7). By definition of η, since $(s, a, \overrightarrow{Y}, s') \in \gamma$ there exists $\underline{\overrightarrow{Y}} \in \mathbb{N}^m$ such that $(\langle s, p, b\rangle, a, \underline{\overrightarrow{Y}}, \langle s', p', b'\rangle) \in \eta$, with $\underline{\overrightarrow{Y}}(j) = \overrightarrow{Y}(j)$ for all $j \neq i$ and $p' = p + \overrightarrow{Y}(i) + v\underline{\overrightarrow{Y}}(i)$.

We first argue that Cond. (6) is verified for p' if $k < \boxplus$, no matter the case of η. In fact, no transition defined in Parts (III) and (I) of the definition of η applies, since for $k \leq \boxminus$ we have $p \geq 0$ (else since $\underline{X} = 0$ the configuration would not be valid), and for $k < \boxplus < v$ we have $p < v - 1$. Therefore, for $0 \leq k \leq \boxminus$ we have $0 \leq p \leq k$ and for $\boxminus < k < \boxplus$, clearly $-[k]_{v,\boxminus} < p \leq k$. By definition, for $k < \boxplus$,

since $p' = p + \overrightarrow{Y}(i)$, it must be $p - 1 \leq p' \leq p + 1$, i.e., $-[k]_{v,\boxminus} - 1 < p' \leq k + 1$, and since $[k+1]_{v,\boxminus} = [k]_{v,\boxminus} + 1$, $[k]_{v,\boxplus} = k$, and $[k+1]_{v,\boxplus} = k + 1$, we have $-[k+1]_{v,\boxminus} < p' \leq [k+1]_{v,\boxplus} < v + [k+1]_{v,\boxplus}$, which is Cond. (6). Therefore, in the following proof when showing that Cond. (6) holds we always assume $k \geq \boxplus$.

We first claim that:

(-) if $s \notin [\![\boxminus]\!]$, then $-[k+1]_{v,\boxminus} < p'$.

(+) if $s \notin [\![\boxplus]\!]$, then $p' < v + [k+1]_{v,\boxplus}$.

Claim (-) is immediate, since if $s \notin [\![\boxminus]\!]$, then $[k]_{v,\boxminus} < v$, hence $[k+1]_{v,\boxminus} = 1 + [k]_{v,\boxminus}$. Then, $p' = p + \overrightarrow{Y}(i) \geq p - 1 > -[k]_{v,\boxminus} - 1 = -[k+1]_{v,\boxminus}$. Claim (+) is also immediate, since in this case $[k]_{v,\boxplus} < v$, hence $[k+1]_{v,\boxplus} = 1 + [k]_{v,\boxplus}$. Therefore, $p' = p + \overrightarrow{Y}(i) \leq p + 1 < v + [k]_{v,\boxminus} + 1 = v + [k+1]_{v,\boxminus}$.

I) *Case* $s \in [\![\boxminus]\!]$ *and* $p < 0 \vee p + \overrightarrow{Y}(i) < 0 \vee b = -1$.
Hence, $\underline{\overrightarrow{Y}}(i) = 1, b' = 0, p' = p + \overrightarrow{Y}(i) + v$. We claim that $\underline{\overrightarrow{X}} > 0$. By contradiction, assume $\underline{\overrightarrow{X}} = 0$. In this case, if $p < 0 \vee b = -1$, then $(\langle s, p, b\rangle, \underline{\overrightarrow{X}})$ would not be valid, contradicting the induction hypothesis, while if $p = 0$ and $p + \overrightarrow{Y}(i) < 0$, then $\overrightarrow{Y}(i) = -1$, therefore $\overrightarrow{X'} = \overrightarrow{X}(i) + \overrightarrow{Y}(i) = v\underline{\overrightarrow{X}}(i) + p + \overrightarrow{Y}(i) = v\underline{\overrightarrow{X}}(i) - 1 = -1$, which is impossible. Since $\underline{\overrightarrow{X}} > 0$, it follows from the definition of η that Eq. (8) holds, with $p' = p + \overrightarrow{Y}(i) + v \geq 0$ (since $p \geq -v + 1$ and thus $p + \overrightarrow{Y}(i) \geq -v$) and $\underline{\overrightarrow{X}}' = \underline{\overrightarrow{X}} - \overrightarrow{i}$.
By definition, $(\langle s', p', 0\rangle, \underline{\overrightarrow{X}}')$ is valid, since $p' \geq 0 \wedge b' = 0$. Clearly, for $j \neq i$, $\underline{\overrightarrow{X}}'(j) = \underline{\overrightarrow{X}}(j)$, while $\underline{\overrightarrow{X}}'(i) = \underline{\overrightarrow{X}}(i) - 1$. But $\overrightarrow{X'}(i) = \overrightarrow{X}(i) + \overrightarrow{Y}(i) = v\underline{\overrightarrow{X}}(i) + p + \overrightarrow{Y}(i) = v(\underline{\overrightarrow{X}}'(i) + 1) + p + \overrightarrow{Y}(i) = v\underline{\overrightarrow{X}}'(i) + v + p + \overrightarrow{Y}(i) = v\underline{\overrightarrow{X}}'(i) + p'$, i.e., $(\langle s', p', 0\rangle, \underline{\overrightarrow{X}}') \rightarrowtail (s, \overrightarrow{X'})$, thus showing Cond. (5) for ϕ'.
We now show that Cond. (6) holds for p'. Since $s \notin [\![\boxplus]\!]$, by (+) we only need to prove $-[k+1]_{v,\boxminus} < p'$. We notice that, since $s \in [\![\boxminus]\!]$, $[k]_{v,\boxminus} = v$ and $[k+1]_{v,\boxminus} = 1$: condition $-[k+1]_{v,\boxminus} < p'$ trivially holds since we have shown above that $p' \geq 0$.

II) *Case* $s \notin [\![\boxminus]\!]$ *and* $p < 0 \vee p + \overrightarrow{Y}(i) < 0$. Hence, $\underline{\overrightarrow{Y}}(i) = 0, b' = -1, p' = p + \overrightarrow{Y}(i)$. Since this transition does not modify the value of $\underline{\overrightarrow{X}}(i)$, we have that Eq. (8) holds, with $\underline{\overrightarrow{X}}' = \underline{\overrightarrow{X}}$. We claim that $\underline{\overrightarrow{X}}(i) > 0$, therefore $(\langle s', p', -1\rangle, \underline{\overrightarrow{X}}')$ is valid. By contradiction, assume $\underline{\overrightarrow{X}}(i) = 0$. Hence, $p \geq 0 \wedge b = 0$, since configuration $(\langle s, p, b\rangle, \underline{\overrightarrow{X}})$ must be valid, therefore, $p = 0 \wedge p + \overrightarrow{Y}(i) < 0$. This means that $\overrightarrow{Y}(i) = -1$; since by induction hypothesis $\overrightarrow{X}(i) = v\underline{\overrightarrow{X}}(i) + p = 0$, then $\overrightarrow{X'}(i) = \overrightarrow{X}(i) + \overrightarrow{Y}(i) < 0$, which is impossible.
Clearly, $\overrightarrow{X'}(i) = \overrightarrow{X}(i) + \overrightarrow{Y}(i) = v\underline{\overrightarrow{X}}(i) + p + \overrightarrow{Y}(i) = v\underline{\overrightarrow{X}}(i) + p'$, i.e, we have

$(\langle s', p', -1\rangle, \underline{\overrightarrow{X}}') \rightarrowtail (s, \overrightarrow{X'})$, thus completing Cond. (5) for ϕ'. We now show that Cond. (6) holds for $p' = p + \overrightarrow{Y}(i)$. By (-), we only need to prove $p' < v + [k+1]_{v,\boxplus}$. Since $p < 0 \vee p + \overrightarrow{Y}(i) < 0$, we have $p' = p + \overrightarrow{Y}(i) \leq p + 1 \leq 0$, therefore $p' < v < v + [k+1]_{v,\boxplus}$.

III) *Case* $s \in [\![\boxplus]\!]$ *and* $p + \overrightarrow{Y}(i) \geq v \wedge b = 0$.
Hence, $\underline{\overrightarrow{Y}} = -1, b' = b = 0, p' = p + \overrightarrow{Y}(i) - v \geq 0$. Then, Eq. (8) holds, with $\underline{\overrightarrow{X}}' = \underline{\overrightarrow{X}} + 1 > 0$. It follows that $(\langle s', p', b'\rangle, \underline{\overrightarrow{X}}')$ is valid. Also, $\overrightarrow{X'}(i) = \overrightarrow{X}(i) + \overrightarrow{Y}(i) = v\underline{\overrightarrow{X}}(i) + p + \overrightarrow{Y}(i) = v(\underline{\overrightarrow{X}}'(i) - 1) + p + \overrightarrow{Y}(i) = v\underline{\overrightarrow{X}}'(i) - v + p + \overrightarrow{Y}(i) = v\underline{\overrightarrow{X}}'(i) + p'$, thus showing Cond. (5) for ϕ'. We show that Cond. (6) holds for $p' = p + \overrightarrow{Y}(i)$. Since $s \notin [\![\boxminus]\!]$, by (-) we only need to prove $p' < v + [k+1]_{v,\boxplus}$. Also, since $s \in [\![\boxplus]\!]$, we have $[k]_{v,\boxplus} = v$ and $[k+1]_{v,\boxplus} = 1$. Therefore, $p' = p + \overrightarrow{Y}(i) - v < (v + [k]_{v,\boxplus}) + \overrightarrow{Y}(i) - v = [k]_{v,\boxplus} + \overrightarrow{Y}(i) = v + \overrightarrow{Y}(i) \leq v + 1 = v + [k+1]_{v,\boxplus}$, i.e, $p' < v + [k]_{v,\boxplus}$.

IV) *All remaining cases.* Since this transition does not modify the value of $\underline{\overrightarrow{X}}(i)$, we have that Eq. (8) holds, with $p' = p + \overrightarrow{Y}(i)$, $b' = b$ and $\underline{\overrightarrow{X}}' = \underline{\overrightarrow{X}}$.
Since $(\langle s, p, b\rangle, \underline{\overrightarrow{X}})$ is valid, then we have two cases: if $\underline{\overrightarrow{X}} > 0$, then $\underline{\overrightarrow{X}}' > 0$, therefore ϕ' is valid. If $\underline{\overrightarrow{X}} = \underline{\overrightarrow{X}}' = 0$, then $p \geq 0 \wedge b = 0$. We show that in this case also $p' \geq 0$, therefore ϕ' is valid. By contradiction, let $p' < 0$. Hence, $p = 0$ and $\overrightarrow{Y}(i) = -1$. Clearly, $\overrightarrow{X'}(i) = \overrightarrow{X}(i) + \overrightarrow{Y}(i) = v\underline{\overrightarrow{X}}(i) + p + \overrightarrow{Y}(i) = v\underline{\overrightarrow{X}}(i) + p'$. Therefore, also $\overrightarrow{X}(i) = v\underline{\overrightarrow{X}}(i) + p = 0$, hence $\overrightarrow{X'}(i) = \overrightarrow{X}(i) + \overrightarrow{Y}(i) = \overrightarrow{X}(i) - 1 < 0$, which is impossible. Cond. (5) is verified for ϕ', since $\overrightarrow{X'}(i) = \overrightarrow{X}(i) + \overrightarrow{Y}(i) = v\underline{\overrightarrow{X}}(i) + p + \overrightarrow{Y}(i) = v\underline{\overrightarrow{X}}(i) + p'$, i.e, $(\langle s', p', -1\rangle, \underline{\overrightarrow{X}}') \rightarrowtail (s, \overrightarrow{X'})$. We now show that Cond. (6) holds for $p' = p + \overrightarrow{Y}(i)$. We consider the only three possible subcases for s: $s \in [\![\boxminus]\!]$, $s \in [\![\boxplus]\!]$ and $s \notin [\![\boxminus]\!] \wedge s \notin [\![\boxplus]\!]$.

Subcase $s \in [\![\boxminus]\!]$. Since $s \notin [\![\boxplus]\!]$, by (+) we need to prove only $-[k+1]_{v,\boxminus} < p'$. Since $s \in [\![\boxminus]\!]$ and we are not in Case (I) of the definition of η, we have $[k]_{v,\boxminus} = v$ and $p + \overrightarrow{Y}(i) \geq 0$. Then, $[k+1]_{v,\boxminus} = 1$. Therefore, $p' = p + \overrightarrow{Y}(i) \geq 0 > -1 = -[k+1]_{v,\boxminus}$.

Subcase $s \in [\![\boxplus]\!]$. Since $s \notin [\![\boxminus]\!]$, by (-) we only need to prove $p' < v + [k+1]_{v,\boxplus}$. From $s \in [\![\boxplus]\!]$, it follows that $k \mod v = \boxplus$, hence $[k]_{v,\boxplus} = v$, i.e., $[k+1]_{v,\boxplus} = 1$. Therefore, $p' = p + \overrightarrow{Y}(i) \leq p + 1 < v + [k]_{v,\boxplus} + 1 = v + [k+1]_{v,\boxplus}$. Since $s \in [\![\boxplus]\!]$, but we are not in Case (III) of the definition of η, we have two cases: $p + \overrightarrow{Y}(i) < v$ or $p + \overrightarrow{Y}(i) \geq v \wedge b = -1$. In the former case, $p' = p + \overrightarrow{Y}(i) < v < v + [k+1]_{v,\boxplus}$, i.e., Cond. (6) holds. We claim that the

latter case is impossible, thus ending the proof. Assume by contradiction $p + \overrightarrow{Y}(i) \geq v \wedge b = -1$. Since $(\langle s_1, 0, 0\rangle, \overrightarrow{0}^m) \xrightarrow{w}_{\mathcal{M}(v,\boxminus,\boxplus)} \phi$, there exist $s_{i_1}, \ldots, s_{i_k} \in S, p_1, \ldots, p_k \in P, b_1, \ldots, b_k \in \{0, -1\}, \overrightarrow{\underline{X}}_1 \ldots \overrightarrow{\underline{X}}_k \in \mathbb{N}^m$, with $s_{i_k} = s, p_k = p, b_k = b = -1$ and $\overrightarrow{\underline{X}}_k = \underline{X}$, such that:

$$(\langle s_1, 0, 0\rangle, \overrightarrow{0}^m) \xrightarrow{w(1)}_{\mathcal{M}(v,\boxminus,\boxplus)} (\langle s_{i_1}, p_1, b_1\rangle, \overrightarrow{\underline{X}}_1) \xrightarrow{w(2)} \cdots \xrightarrow{w(k)} (\langle s_{i_k}, p_k, b_k\rangle, \overrightarrow{\underline{X}}_k).$$

Let t, with $1 \leq t < k$, be the largest value such that $b_t = 0, b_{t+1} = -1$, i.e., $(\langle s_{i_t}, p_t, b_t\rangle, \overrightarrow{\underline{X}}_t) \xrightarrow{w(t)}_{\mathcal{M}(v,\boxminus,\boxplus)} (\langle s_{i_{t+1}}, p_{t+1}, b_{t+1}\rangle, \overrightarrow{\underline{X}}_{t+1})$, applying a move in the form of Case (II) in the definition of η. Hence, $p_t \leq 0$. Since $k \mod v = \boxplus$ and $t \mod v = \boxminus$, then $t < k$. We claim that the largest position j in w such that $j \mod v = \boxminus$ is just equal to t. By contradiction, assume that there is $j > t$, but $j \neq k$, such that $s_{i_j} \in [\![\boxminus]\!]$. Since by definition of t we have $b_{t+1} = b_{t+2} = \cdots = b_k = -1$, also $b_j = -1$, i.e, s_{i_j}, p_j, b_j verify the conditions of Case (I) of η (which are mutually exclusive with Cases (II), (III) and (IV)). Therefore, the step from s_{i_j}, p_j, b_j to $s_{i_{j+1}}, p_{j+1}, b_{j+1}$ applies a move in the form of Case (II), hence $b_{j+1} = 0$, a contradiction.
Since $k \mod v = \boxplus \neq \boxminus$ and $t \mod v = \boxminus$, it follows that $k - t \leq v - 1$. Every transition applied while reading the $k - (t + 1)$ symbols $w(t + 1), \ldots, w(k)$, must have the form of Case (II) or the form of Case (IV) of the definition of η. Hence, p_t may incremented only of at most 1 in each of the following $k - (t + 1)$ steps, i.e., $p_k \leq p_t + (k - t - 1) \leq v - 2$. It then follows that $p = p_k \leq v - 2$. Therefore, $p + \overrightarrow{Y}(i) \leq p + 1 \leq v - 1$. However, we assumed at the beginning that $p + \overrightarrow{Y}(i) \geq v \wedge b = -1$, a contradiction with $p + \overrightarrow{Y}(i) \leq v - 1$.

Subcase $s \notin [\![\boxminus]\!] \wedge s \notin [\![\boxplus]\!]$. Cond. (6) then holds for p', by Claims (-) and (+).

□

2.4. Quasi-deterministic RT-PBLIND machines

We precisely define the quasi-deterministic form of counter machines and prove that it is a normal form for RT-PBLIND machines.

We need some further notation on vectors. For all integers i, m, $1 \leq i \leq m$, let $\overrightarrow{i} \in I^m$ be the vector with value 1 in position i and value 0 elsewhere. Then, $-\overrightarrow{i}$ is the vector with value -1 in position i and 0 elsewhere, and, for $j \neq i$, $\overrightarrow{j} - \overrightarrow{i} \in I^m$ is the vector with value -1 in position i, value $+1$ in position j and value 0 elsewhere.

We want to restrict the allowed increment vectors $\overrightarrow{Y}$ to the case that at most one increment and one decrement are present in a single move. Define:

$$J^m = \{\overrightarrow{i} \mid 1 \leq i \leq m\} \cup \{-\overrightarrow{i} \mid 1 \leq i \leq m\} \cup \{\overrightarrow{j} - \overrightarrow{i} \mid 1 \leq i, j \leq m\} \qquad (9)$$

Notice that $\overrightarrow{0}^m \in J^m$ (for $i = j$). An RT-PBLIND machine $\mathcal{M} = \langle \Sigma, S, \gamma, m, s_1, S_{fin} \rangle$ has the *simple operation* property if $\gamma \subseteq S \times \Sigma \times J^m \times S$. A simple-operation, real-time machine is as powerful as one performing unrestricted operations. This can be shown by using the same rarefaction technique of the proof of Th. 2.8. Notice that a rarefied RT-PBLIND enjoys the simple operation property too.

Let $\mathcal{M} = \langle \Sigma, S, \gamma, m, s_1, S_{fin} \rangle$ be a simple-operation RT-PBLIND machine.

A *state covering* is a family $\mathcal{B}$ of non-empty distinct subsets of S, also called *blocks*, such that $\bigcup_{B \in \mathcal{B}} B = S$. Given a covering $\mathcal{B}$ for an RT-PBLIND machine $\mathcal{M} = \langle \Sigma, S, \gamma, m, s_1, S_{fin} \rangle$, one can define a finite nondeterministic automaton, called a *covering automaton* $\mathcal{M}_{\mathcal{B}} = \left(\Sigma, \mathcal{B}, \delta, B_0, \mathcal{B}_{fin}\right)$, where:

- B_0 is a block of $\mathcal{B}$, that contains s_1;
- $\mathcal{B}_{fin}$ is the set of blocks in $\mathcal{B}$ which include a state in S_{fin};
- for all $a \in \Sigma, B', B'' \in \mathcal{B}$, the triple $\langle B', a, B'' \rangle \in \delta$ if, and only if, for all $s' \in B'$, for all $s'' \in S, \overrightarrow{Y} \in J^m$ if $\langle s', a, \overrightarrow{Y}, s'' \rangle \in \gamma$, then $s'' \in B''$.

It is obvious that for some $w \in \Sigma^*, \overrightarrow{X} \in \mathbb{N}^m$, if $(s_1, \overrightarrow{0}^m) \xrightarrow{w}_{\mathcal{M}} (s', \overrightarrow{X})$ is a run, then the covering automaton $\mathcal{M}_{\mathcal{B}}$ has a run $B_0 \xrightarrow{w}_{\mathcal{M}_{\mathcal{B}}} B'$, for some B' with $s' \in B'$.

Definition 2.9 (QD covering)**.** *Let $\mathcal{M}$ be a simple-operation RT-PBLIND machine and let $\mathcal{B}$ be a covering for $\mathcal{M}$. $\mathcal{B}$ is* quasi-deterministic *(QD) for $\mathcal{M}$ if the covering automaton $\mathcal{M}_{\mathcal{B}}$ is deterministic and, for all $a \in \Sigma$, $B, B_1 \in \mathcal{B}, s', s'' \in B, s'_1, s''_1 \in B_1, \overrightarrow{Y}_1, \overrightarrow{Y}_2 \in J^m$:*

$$\begin{aligned} &\text{if } \left(s', a, \overrightarrow{Y}_1, s'_1\right) \in \gamma \text{ and } \left(s'', a, \overrightarrow{Y}_2, s''_1\right) \in \gamma, \text{ then:} \\ &\qquad \overrightarrow{Y}_1 = \overrightarrow{Y}_2 \text{ or } \overrightarrow{Y}_1 = \overrightarrow{0}^m \text{ or } \overrightarrow{Y}_2 = \overrightarrow{0}^m. \end{aligned} \tag{10}$$

Cond. (10) says that from any state belonging to the same block $\mathcal{M}$ either performs the same simple operation on counters, or does not modify the counters. This is designed to rule out cases where the same input word $w = w(1) \dots w(|w|)$ may lead to two or more runs that, at the same time position j, $1 \leq j \leq |w|$, execute different (simple) operations.

The following corollary is derived immediately from the definition of scheduled machines. Since a partition of the state set is also a covering, it defines a covering automaton.

Corollary 2.10. *Let $\mathcal{M} = \langle \Sigma, S, \gamma, m, s_1, S_{fin} \rangle$ be a h-rarefied RT-PBLIND machine, $h > 0$, of module $v \geq h \cdot 2m$. Then, there exists a partition of S into v subsets, denoted $\mathcal{B} = \{S_0, S_1, \dots, S_{v-1}\}$ such that:*

1. for all $w \in \Sigma^*$, if $(s_1, \overrightarrow{0}^m) \xrightarrow{w}_{\mathcal{M}} (s, \overrightarrow{X})$, then $s \in S_j$, with $j = |w| \mod v$;

2. *the covering automaton $\mathcal{M}_{\mathcal{B}}$ induced by partition $\mathcal{B}$ is QD.*

We now introduce a machine model characterized by the property that the state covering where every block is just a single state is QD.

Definition 2.11 (Quasi-Deterministic RT-PBLIND machine)**.** *A RT-PBLIND machine $\mathcal{M} = \langle \Sigma, S, \gamma, m, s_1, S_{fin} \rangle$ is called* Quasi-Deterministic*, or a QD-RT-PBLIND machine, if $\mathcal{M}$ has the simple operation property and the trivial state covering $\mathcal{B} = \{\{s_i\} \mid s_i \in S\}$ is quasi-deterministic for $\mathcal{M}$.*

We show that a QD real-time machine can simulate a nondeterministic one, in spite of the limited counter testing capability available in PBLIND operations. For that we have to assume that the input word is terminated by an end marker, denoted by symbol $\dashv \notin \Sigma$, since otherwise the real-time constraint would prevent the QD machine to terminate correctly in some cases.

Lemma 2.12. *For every 3-rarefied RT-PBLIND machine $\mathcal{M} = \langle \Sigma, S, \gamma, m, s_1, S_{fin} \rangle$, there exists an equivalent QD-RT-PBLIND machine $\mathcal{M}_{qd} = \langle \Sigma \cup \{\dashv\}, T, \eta, m + |S|, t_0, \{t_{fin}\} \rangle$, accepting $L(\mathcal{M})\dashv$. Moreover, machine $\mathcal{M}_{qd}$ performs counter operations in every move, i.e., $\eta \subseteq T \times \Sigma \times \left(J^m - \{\overrightarrow{0}^m\} \right) \times T$.*

Proof. To ease understanding, we intuitively explain how $\mathcal{M}_{qd}$ simulates $\mathcal{M}$. The $|S|$ extra counters serve as binary flags, to represent the current state s_h of $\mathcal{M}$. When $\mathcal{M}$ moves from s_h to s_i, $\mathcal{M}_{qd}$ decrements the flag of s_h (thus testing that the current state is s_h) and increments the flag of s_i to set the new state. Unfortunately, if flag updating is simultaneous with a counter operation on one of the original m counters of $\mathcal{M}$, machine $\mathcal{M}_{qd}$ cannot do it and must defer updating to the following step; more precisely, when $\mathcal{M}$ moves from s_i to s_j, machine $\mathcal{M}_{qd}$ decrements the flag of s_h (which was left behind) and increments the flag of s_j. The latter counter operation is always feasible thanks to the rarefaction hypothesis.
Another situation worth explaining occurs on a final accepting move of $\mathcal{M}$ from state s_h to a final state. If machine $\mathcal{M}_{qd}$ has flag s_h set to 1, it must simply set it to 0. But if the s_h flag is still unset (its setting having been deferred to the next move as explained before), then machine $\mathcal{M}_{qd}$ needs the extra move on end marker $\dashv$, to test that the current state is correct, and to simulate the accepting move of $\mathcal{M}$. An example of the construction is presented after the formal proof.

Let $\mathcal{B}$ be the partition of S in Cor. 2.10 and let $B_0 \in \mathcal{B}$ be the block including s_1. Let $n = |S|$, with $S = \{s_1, s_2, \ldots, s_n\}$. Consider the covering DFA $\mathcal{M}_{\mathcal{B}} = \left(\Sigma, \mathcal{B}, \delta, B_0, \mathcal{B}_{fin}\right)$. We may assume, since $\mathcal{M}_{\mathcal{B}}$ is a finite automaton, that all blocks in $\mathcal{B}$ are reachable from B_0, i.e., if $B \in \mathcal{B}$, then there exists $w \in \Sigma^*$ such that $B_0 \xrightarrow{w}_{\mathcal{M}_{\mathcal{B}}} B$. We now define $\mathcal{M}_{qd}$. Let $T = \{t_0, t_{fin}\} \cup (\mathcal{B} \times (\Sigma \cup \{\dashv\}))$.

Therefore, apart from t_0 and t_{fin}, the states of $\mathcal{M}_{qd}$ are pairs in $\mathcal{B} \times (\Sigma \cup \{\dashv\})$.
For every $\langle B, a\rangle$, $B \in \mathcal{B}, a \in \Sigma$, let

$$\lfloor\langle B, a\rangle\rfloor = \{s' \in S \mid \exists s \in B, \overrightarrow{Y} \in J^m, (s, a, \overrightarrow{Y}, s') \in \gamma\}.$$

By quasi-determinism, there exists one, and only one, block $B' \in \mathcal{B}$ such that $\lfloor\langle B, a\rangle\rfloor \subseteq B'$: the block B' such that $\langle B, a, B'\rangle \in \delta$.

State $\langle B, a\rangle$ represents all states of $\mathcal{M}$ which can (potentially) be reached from a state in B by reading a (i.e., the set $\lfloor\langle B, a\rangle\rfloor$).

To make the definition of η more perspicuous, we classify the states of T as *checking* and *normal*. A state of the form $\langle B', a\rangle$, $a \in \Sigma \cup \{\dashv\}$, is *checking* if there exist $\overrightarrow{Y} \neq \overrightarrow{0}^m$, $s' \in B'$, $s'' \in \lfloor\langle B', a\rangle\rfloor$ such that $(s', a, \overrightarrow{Y}, s'') \in \gamma$, i.e., if reading a in state s' the machine $\mathcal{M}$ may modify a counter with an increment vector $\overrightarrow{Y}$, upon entering a state s'', which is represented in $\mathcal{M}_{qd}$ by $\langle B', a\rangle$. In other words, a state is checking if it is representative of one or more states of $\mathcal{M}$ which can be entered by a transition that modifies a counter; every other state is classified as *normal*.

The transition relation $\eta \subseteq T \times (\Sigma \cup \{\dashv\}) \times J^{n+m} \times T$ to be next defined, makes use of vectors in J^{n+m} of the form $\overrightarrow{i}$, $\overrightarrow{h}$ etc. Recall that, by Def. 2.5, a move from the initial state s_1 cannot modify counters.

Here and in what follows the dot "$\cdot$" denotes vector concatenation, e.g., $\overrightarrow{0}^n \cdot \overrightarrow{X}$, for $\overrightarrow{X} \in \mathbb{N}^m$, is the vector in $\mathbb{N}^{n+m}$ with n 0's followed by the content of vector $\overrightarrow{X}$. With an abuse of notation, $\overrightarrow{i} \cdot \overrightarrow{X}$, for $1 \leq i \leq n$, denotes the vector $\overrightarrow{i} + (\overrightarrow{0}^n \cdot \overrightarrow{X})$ in $\mathbb{N}^{n+m}$, i.e., the vector equal to $\overrightarrow{i}$ in the leftmost n positions and equal to $\overrightarrow{X}$ in the rightmost m positions.

The transition relation is defined by the following cases.

Initialization. For all $a \in \Sigma$, for all $s_i \in S$, $1 \leq i \leq n$, if $(s_1, a, \overrightarrow{0}^m, s_i) \in \gamma$, then $(t_0, a, \overrightarrow{i}, \langle B_0, a\rangle) \in \eta$.

State Transition. For all $a \in \Sigma$, for all $b \in \Sigma \cup \{\dashv\}$, for all $B', B'', B''' \in \mathcal{B}$ such that $\langle B', a, B''\rangle, \langle B'', b, B'''\rangle \in \delta$, for all i, j such that $s_i \in B''$, $s_j \in B'''$:

1. if $(s_i, b, \overrightarrow{0}^m, s_j) \in \gamma$, then $(\langle B', a\rangle, b, \overrightarrow{j} - \overrightarrow{i}, \langle B'', b, \rangle) \in \eta$.
2. for all $\overrightarrow{Y} \in J^m$, $\overrightarrow{Y} \neq \overrightarrow{0}^m$:
   (a) if $(s_i, b, \overrightarrow{Y}, s_j) \in \gamma$, then $(\langle B', a\rangle, b, \overrightarrow{0}^n \cdot \overrightarrow{Y}, \langle B'', b\rangle) \in \eta$;
   (b) for all $s_h \in B'$ such that $(s_h, a, \overrightarrow{Y}, s_i) \in \gamma$, if $(s_i, b, \overrightarrow{0}^m, s_j) \in \gamma$, then $(\langle B', a\rangle, b, \overrightarrow{j} - \overrightarrow{h}, \langle B'', b\rangle) \in \eta$;

Acceptance. For all $a \in \Sigma \cup \{\dashv\}$, for all $B', B'' \in \mathcal{B}$, such that $\langle B', a, B'' \rangle \in \delta$, for all $s_h \in B'' \cap S_{fin}$, $1 \leq h \leq n$,

1. $(\langle B', a \rangle, \dashv, -\overrightarrow{h}, t_{fin}) \in \eta$ and
2. for all $s_i \in B'$, $\overrightarrow{Y} \in J^m$, $\overrightarrow{Y} \neq 0$, such that $(s_i, a, \overrightarrow{Y}, s_h) \in \gamma$, also: $(\langle B', a \rangle, \dashv, -\overrightarrow{i}, t_{fin}) \in \eta$.

Nothing else is in η.

Notice that all cases in the definition of η are distinct, i.e., each transition in η can only be produced by one case. In particular, the Initialization and the Acceptance definitions are the only ones to introduce transitions with third component of the form $\overrightarrow{i}$ and, respectively, $-\overrightarrow{i}$ and $-\overrightarrow{h}$. Part (1) and (2) of the Acceptance are distinct since the former requires that state $s_h \in S_{fin}$, while the latter requires that there is a transition from state s_i to a state in S_{fin}: by Corollary 2.10, Part (1), $s_i \notin S_{fin}$, hence $i \neq h$. Part (2.a) in the definition of State Transition is the only one to introduce a transition with third component of the form $\overrightarrow{0}^n \cdot \overrightarrow{Y}$ for $\overrightarrow{Y} \neq \overrightarrow{0}^m$. Part (1) and Part (2.b) both introduce transitions with third component of the form $\overrightarrow{j} - \overrightarrow{h}$, but Part (1) requires that $(s_h, b, \overrightarrow{0}^m, s_j) \in \gamma$ while Part (2.b) requires that $(s_h, a, \overrightarrow{Y}, s_i) \in \gamma$ and $(s_i, b, \overrightarrow{0}^m, s_j) \in \gamma$: both conditions are impossible to be simultaneously verified by γ, by Corollary 2.10, Part (1).

Proof of $L(\mathcal{M}_{qd}) \subseteq L(\mathcal{M})$ $\dashv$. We list a few obvious facts on $\mathcal{M}_{qd}$ based on the above definition of η.

I) If $(t_0, \overrightarrow{0}^{n+m}) \xrightarrow{a}_{\mathcal{M}_{qd}} (t, \overrightarrow{\Xi})$, then state t is normal and $\overrightarrow{\Xi} = \overrightarrow{j}$ for $1 \leq j \leq n$, because of the *Initialization* definition and of the rarefaction of $\mathcal{M}$ (no move from t_0 may modify a counter). State t_0 itself is not the endpoint of any transition, hence an *Initialization* transition is used only at the first move in a run.

II) Every transition $(t, a, \overrightarrow{W}, t') \in \eta$, for $t, t' \in T, t \neq t_0, t' \neq t_{fin}, a \in \Sigma$, is such that either $\overrightarrow{W} = \overrightarrow{i} - \overrightarrow{j}$, for some $1 \leq i, j \leq n$, or $\overrightarrow{W} = \overrightarrow{0}^n \cdot \overrightarrow{Y}$ for some $\overrightarrow{Y} \in J^m$, $\overrightarrow{Y} \neq \overrightarrow{0}^m$).

III) Every configuration $(t, \overrightarrow{\Xi})$ of $\mathcal{M}_{qd}$ which can be reached from the initial configuration is such that if $t \neq t_0, t \neq t_{fin}$, then $\overrightarrow{\Xi} = \overrightarrow{j} \cdot \overrightarrow{X}$ for $1 \leq j \leq n$, $\overrightarrow{X} \in \mathbb{N}^m$. In fact, by (I), $(t_0, \overrightarrow{0}^{n+m})$ may only move to a configuration of the form $(t, \overrightarrow{j})$, and, by Part (II) above, the other transitions available in η may only either change the value of j, i.e., the position of the value 1 in the first n components, or the content of $\overrightarrow{X}$.

IV) If $(t, \overrightarrow{\Xi}) \xrightarrow{a}_{\mathcal{M}_{qd}} (t', \overrightarrow{\Xi}')$, then at least one of t and t' is normal, since η does not contain transitions among checking states ($\mathcal{M}$ is rarefied).

V) If $(t, \overrightarrow{\Xi}) \xrightarrow{\dashv}_{\mathcal{M}_{qd}} (t', \overrightarrow{\Xi}')$, then $t' = t_{fin}$, and $\overrightarrow{\Xi}' = \overrightarrow{0}^n \cdot \overrightarrow{X}$, for $\overrightarrow{X} \in \mathbb{N}^m$, because of the transitions in the Acceptance definition.

We now show by induction on $k > 0$ that if $w \in \Sigma^k$, $t_1, \ldots, t_k \in T$, $\overrightarrow{\Xi}_1, \ldots, \overrightarrow{\Xi}_k \in \mathbb{N}^{n+m}$, and

$$(t_0, \overrightarrow{0}^{n+m}) \xrightarrow{w(1)}_{\mathcal{M}_{qd}} (t_1, \overrightarrow{\Xi}_1) \xrightarrow{w(2)}_{\mathcal{M}_{qd}} \cdots \xrightarrow{w(k)}_{\mathcal{M}_{qd}} (t_k, \overrightarrow{\Xi}_k) \tag{11}$$

then there exist $s_{i_0} \in \lfloor t_0 \rfloor$, $s_{i_1} \in \lfloor t_1 \rfloor, \ldots, s_{i_k} \in \lfloor t_k \rfloor$, and $\overrightarrow{X}_1, \ldots, \overrightarrow{X}_k \in \mathbb{N}^m$, such that:

$$(s_{i_0}, \overrightarrow{0}^m) \xrightarrow{w(1)}_{\mathcal{M}} (s_{i_1}, \overrightarrow{X}_1) \xrightarrow{w(2)}_{\mathcal{M}} \cdots \xrightarrow{w(k)}_{\mathcal{M}} (s_{i_k}, \overrightarrow{X}_k) \tag{12}$$

with $s_{i_0} = s_1$, and such that, by setting $\overrightarrow{W} = \overrightarrow{\Xi}_k - \overrightarrow{\Xi}_{k-1}$, the following conditions are verified:

1. if $\overrightarrow{W} = \overrightarrow{0}^n \cdot \overrightarrow{Y}$, for some $\overrightarrow{Y} \in J^m, \overrightarrow{Y} \neq \overrightarrow{0}^m$, then $\overrightarrow{\Xi}_k = \overrightarrow{i_{k-1}} \cdot \overrightarrow{X}_k$ and $(s_{i_{k-1}}, w(k), \overrightarrow{Y}, s_{i_k}) \in \gamma$.
2. else $\overrightarrow{\Xi}_k = \overrightarrow{i_k} \cdot \overrightarrow{X}_k$ and $(s_{i_{k-1}}, w(k), \overrightarrow{0}^m, s_{i_k}) \in \gamma$.

The base case $k = 1$ is obvious: if $(t_0, \overrightarrow{0}^{n+m}) \xrightarrow{w}_{\mathcal{M}_{qd}} (t_1, \overrightarrow{\Xi}_1)$, then, for some $s_{i_1} \in \lfloor t_1 \rfloor$. We have $(t_0, w(1), \overrightarrow{i_1}, t_1\rangle) \in \eta$. Hence, $\overrightarrow{\Xi}_1 = \overrightarrow{i_1} \cdot \overrightarrow{0}^m$, and by definition of η it must be $(s_{i_0}, w(1), \overrightarrow{0}^m, s_{i_1}) \in \gamma$: Cond. (2) of the induction hypothesis is verified and $(s_{i_0}, \overrightarrow{0}^m) \xrightarrow{w(1)}_{\mathcal{M}} (s_{i_1}, \overrightarrow{X}_1)$.

Let now $k > 1$, and assume that (11) holds for $w \in \Sigma^{k-1}$, hence $(s_{i_0}, \overrightarrow{0}^m) \xrightarrow{w}_{\mathcal{M}} (s_{i_{k-1}}, \overrightarrow{X}_{k-1})$. We show that the induction hypothesis holds also for wa, for all $a \in \Sigma$.

Consider first the case where $\overrightarrow{W} = \overrightarrow{0}^n \cdot \overrightarrow{Y}$, for some $\overrightarrow{Y} \neq \overrightarrow{0}^m, \overrightarrow{Y} \in J^m$. The transition applied in the move $(t_{k-1}, \overrightarrow{\Xi}_{k-1}) \xrightarrow{a}_{\mathcal{M}_{qd}} (t_k, \overrightarrow{\Xi}_k)$ must be $(t_{k-1}, a, \overrightarrow{0}^n \cdot \overrightarrow{Y}, t_k)$. This transition can only be defined by Part 2.a of the *State Transition* definition, hence $(s_{i_{k-1}}, a, \overrightarrow{Y}, s_{i_k}) \in \gamma$, for some $s_{i_k} \in \lfloor t_k \rfloor$. Therefore, $(s_{i_{k-1}}, \overrightarrow{X}_{k-1}) \xrightarrow{a}_{\mathcal{M}} (s_{i_k}, \overrightarrow{X}_k)$ and $\overrightarrow{\Xi}_k = \overrightarrow{i_{k-1}} \cdot \overrightarrow{X}_k$, since: $\overrightarrow{\Xi}_k = \overrightarrow{0}^n \cdot \overrightarrow{Y} + \overrightarrow{\Xi}_{k-1} = (\overrightarrow{0}^n \cdot \overrightarrow{Y}) + (\overrightarrow{i_{k-1}} \cdot \overrightarrow{X}_{k-1}) = \overrightarrow{i_{k-1}} \cdot (\overrightarrow{Y} + \overrightarrow{X}_{k-1}) = \overrightarrow{i_{k-1}} \cdot \overrightarrow{X}_k$. Therefore, the induction hypothesis (12) and Cond. (1) are verified.

If $\overrightarrow{W}$ does not have the form $\overrightarrow{0}^n \cdot \overrightarrow{Y}$ for some $\overrightarrow{Y} \neq \overrightarrow{0}^m, \overrightarrow{Y} \in J^m$, then there are two sub-cases to be considered, depending on $\overrightarrow{W}' = \overrightarrow{\Xi}_{k-1} - \overrightarrow{\Xi}_{k-2}$.

- $\overrightarrow{W}' = \overrightarrow{0}^n \cdot \overrightarrow{Y'}$, for some $\overrightarrow{Y'} \neq \overrightarrow{0}^m, \overrightarrow{Y'} \in J^m$. By Cond. (1) of the induction hypothesis, $\overrightarrow{\Xi}_{k-1} = \overrightarrow{i_{k-2}} \cdot \overrightarrow{X}_{k-1}$ and $(s_{i_{k-2}}, w(k-1), \overrightarrow{Y'}, s_{i_{k-1}}) \in \gamma$.

  Therefore, the transition applied in $(t_{k-1}, \overrightarrow{\Xi}_{k-1}) \xrightarrow{a}_{\mathcal{M}_{qd}} (t_k, \overrightarrow{\Xi}_k)$ is necessarily $(t_{k-1}, a, \overrightarrow{i_k} - \overrightarrow{i_{k-2}}, t_k)$ for some $s_{i_k} \in \lfloor t_k \rfloor$ (hence, $\overrightarrow{W} = \overrightarrow{i_k} - \overrightarrow{i_{k-2}}$). This transition

can only be defined by Part (2.b) of the *State Transition* definition (as no other part of that definition can define a transition of the form $(t_{k-1}, a, \overrightarrow{i_j} - \overrightarrow{i_h}, t_k)$ with $s_{i_j} \in \lfloor t_k \rfloor, s_{i_h} \in \lfloor t_{k-2} \rfloor$). Hence, it is necessary that also $(s_{i_{k-1}}, a, \overrightarrow{0}^m, s_{i_k}) \in \gamma$. Therefore, $(s_{i_{k-1}}, \overrightarrow{X}_{k-1}) \xrightarrow{a}_{\mathcal{M}} (s_{i_k}, \overrightarrow{X}_k)$, with $\overrightarrow{X}_k = \overrightarrow{X}_{k-1}$. Also, $\overrightarrow{\Xi}_k = \overrightarrow{i_k} \cdot \overrightarrow{X}_k$ since: $\overrightarrow{X}_k = \overrightarrow{X}_{k-1}$ and $\overrightarrow{\Xi}_k = \overrightarrow{i_k} - \overrightarrow{i_{k-2}} + \overrightarrow{\Xi}_{k-1} = \overrightarrow{i_k} - \overrightarrow{i_{k-2}} + (\overrightarrow{i_{k-2}} \cdot \overrightarrow{X}_{k-1}) = \overrightarrow{i_k} \cdot \overrightarrow{X}_{k-1} = \overrightarrow{i_k} \cdot \overrightarrow{X}_k$. Thus, Cond. (2) of the induction hypothesis is verified.

- W' has not the form $\overrightarrow{0}^n \cdot \overrightarrow{Y'}$. By Cond. (2) of the induction hypothesis, $\overrightarrow{\Xi}_{k-1} = \overrightarrow{i_{k-1}} \cdot \overrightarrow{X}_{k-1}$, with $s_{i_{k-1}} \in \lfloor t_{k-1} \rfloor$. Therefore, the rightmost transition applied in the above run of M_{qd} is $(t_{k-1}, a, \overrightarrow{i_k} - \overrightarrow{i_{k-1}}, t_k)$, for some $s_{i_k} \in \lfloor t_k \rfloor$, i.e., $\overrightarrow{W} = \overrightarrow{i_k} - \overrightarrow{i_{k-1}}$. This transition cannot be defined by Part (2) of the *State Transition* definition: in fact, Part (2.a) may only define transitions with increment vector $\overrightarrow{Y} \neq \overrightarrow{0}^m$, while Part (2.b) may only define transitions of the form $(t_{k-1}, a, \overrightarrow{j} - \overrightarrow{h}, t_k)$, with $s_j \in \lfloor t_k \rfloor, s_h \in \lfloor t_{k-2} \rfloor$: but $s_{i_{k-1}} \in \lfloor t_{k-1} \rfloor$, i.e., $s_h, s_{i_{k-1}}$ are in different blocks, so the transition $(t_{k-1}, a, \overrightarrow{i_k} - \overrightarrow{i_{k-1}}, t_k)$ cannot be defined by this part either. Therefore, it must be defined in Part (1) of the *State Transition* definition, hence $(s_{i_{k-1}}, a, \overrightarrow{0}^m, s_{i_k}) \in \gamma$. Therefore, $(s_{i_{k-1}}, \overrightarrow{X}_{k-1}) \xrightarrow{a}_{\mathcal{M}} (s_{i_k}, \overrightarrow{X}_k)$, with $\overrightarrow{X}_k = \overrightarrow{X}_{k-1}$. Also, $\overrightarrow{\Xi}_k = \overrightarrow{i_k} \cdot \overrightarrow{X}_k$ since: $\overrightarrow{\Xi}_k = \overrightarrow{W} + \overrightarrow{\Xi}_{k-1} = \overrightarrow{i_k} - \overrightarrow{i_{k-1}} + \overrightarrow{\Xi}_{k-1} = \overrightarrow{i_k} - \overrightarrow{i_{k-1}} + (\overrightarrow{i_{k-1}} \cdot \overrightarrow{X}_{k-1}) = \overrightarrow{i_k} \cdot \overrightarrow{X}_{k-1} = \overrightarrow{i_k} \cdot \overrightarrow{X}_k$ (because $\overrightarrow{X}_k = \overrightarrow{X}_{k-1}$). Thus, Cond. (2) of the induction hypothesis is verified.

We now show that if a word $w \dashv$, with $w \in \Sigma^*$, is accepted by $\mathcal{M}_{qd}$, then w is accepted by $\mathcal{M}$, thus ending the proof. Let $(t_0, \overrightarrow{0}^{n+m}) \xrightarrow{w}_{\mathcal{M}_{qd}} (t_k, \overrightarrow{\Xi}_k) \xrightarrow{\dashv}_{\mathcal{M}_{qd}} (t_{fin}, \overrightarrow{0}^{n+m})$. Hence, $(s_{i_0}, \overrightarrow{0}^m) \xrightarrow{w}_{\mathcal{M}} (s_{i_k}, \overrightarrow{X}_k)$. Also, if $\overrightarrow{\Xi}_k - \overrightarrow{\Xi}_{k-1}$ is not of the form $\overrightarrow{0}^m \cdot \overrightarrow{Y}$, for $\overrightarrow{Y} \neq \overrightarrow{0}^m, \overrightarrow{Y} \in J^m$, then $\overrightarrow{\Xi}_k = \overrightarrow{i_k} \cdot \overrightarrow{X}_k$, else $\overrightarrow{\Xi}_k = \overrightarrow{i_{k-1}} \cdot \overrightarrow{X}_k$. In the former case, the transition applied in the last move of the run of $\mathcal{M}_{qd}$ is $(t_k, \dashv, -\overrightarrow{i_k}, t_{fin})$. Since $s_{i_k} \in \lfloor t_k \rfloor$, we are in case (1) of the *Acceptance* definition. Hence, $s_k \in S_{fin}$ and $\overrightarrow{\Xi}_k - \overrightarrow{i_k} = \overrightarrow{0}^{n+m}$, therefore $\overrightarrow{X}_k = \overrightarrow{0}^m$: w is recognized by $\mathcal{M}$. In the latter case, the transition applied in the last move of the run of $\mathcal{M}_{qd}$ is $(t_k, \dashv, -\overrightarrow{i_{k-1}}, t_{fin})$. Since $s_{i_{k-1}} \in \lfloor t_{k-1} \rfloor$, we are in case (2) of the *Acceptance* definition. Hence, it must be $s_k \in S_{fin}$. Also, $\overrightarrow{\Xi}_k - \overrightarrow{i_{k-1}} = \overrightarrow{0}^{n+m}$, therefore $\overrightarrow{X}_k = \overrightarrow{0}^m$, and w is recognized by $\mathcal{M}$.

Proof of $L(\mathcal{M}) \dashv \subseteq L(\mathcal{M}_{qd})$. We first prove by induction on $k \geq 1$ that for all $w \in \Sigma^k, k \geq 1$, for all $s_1, \ldots, s_{i_k} \in S, \overrightarrow{X}_1, \ldots, \overrightarrow{X}_k \in \mathbb{N}^m$, if

$$(s_{i_0}, \overrightarrow{0}^m) \xrightarrow{w(1)}_{\mathcal{M}} (s_{i_1}, \overrightarrow{X}_1) \cdots \xrightarrow{w(k)}_{\mathcal{M}} (s_{i_k}, \overrightarrow{X}_k) \tag{13}$$

with $s_{i_0} = s_1$, then there exist $t_k \in T$, $\overrightarrow{\Xi}_k \in \mathbb{N}^{n+m}$ such that:

$$(t_0, \overrightarrow{0}^{n+m}) \xrightarrow{w}_{\mathcal{M}_{qd}} (t_k, \overrightarrow{\Xi}_k), \tag{14}$$

$$\text{if } \overrightarrow{X}_k = \overrightarrow{X}_{k-1}, \text{ then } \overrightarrow{\Xi}_k = \overrightarrow{i_k} \cdot \overrightarrow{X}_k \tag{15}$$

$$\text{if } \overrightarrow{X}_k \neq \overrightarrow{X}_{k-1}, \text{ then } \overrightarrow{\Xi}_k = \overrightarrow{i_{k-1}} \cdot \overrightarrow{X}_k. \tag{16}$$

The base case $k = 1$, i.e., $w = a \in \Sigma$, follows from the *Initialization* case in the definition of η. If $(s_1, \overrightarrow{0}^m) \xrightarrow{a}_{\mathcal{M}} (s_{i_1}, \overrightarrow{X}_1)$, then $(s_1, a, \overrightarrow{0}^m, s_{i_1}) \in \gamma$, hence $(t_0, a, \overrightarrow{i_1}, \langle B_0, a\rangle) \in \eta$. Let $t_1 = \langle B_0, a\rangle$. Hence, $(t_0, \overrightarrow{0}^{n+m}) \xrightarrow{a}_{\mathcal{M}_{qd}} (t_1, \overrightarrow{i_1} \cdot \overrightarrow{X}_1)$, thus Conditions (14) and (15) in the induction hypothesis are verified.

Let now $k > 1$, and assume that (13) holds for $w \in \Sigma^{k-1}$, i.e., there exist $t_{k-1} \in T$, $\overrightarrow{\Xi}_{k-1} \in \mathbb{N}^{n+m}$ such that $(t_0, \overrightarrow{0}^{n+m}) \xrightarrow{w}_{\mathcal{M}_{qd}} (t_{k-1}, \overrightarrow{\Xi}_{k-1})$.

We need to show that for all $a \in \Sigma$ there exist $t_k \in T$, $\overrightarrow{\Xi}_k \in \mathbb{N}^{n+m}$, verifying the correct case among Conditions (15) and (16), and such that $(t_{k-1}, \overrightarrow{\Xi}_{k-1}) \xrightarrow{a}_{\mathcal{M}_{qd}} (t_k, \overrightarrow{\Xi}_k)$. Since $(s_{i_{k-1}}, \overrightarrow{X}_{k-1}) \xrightarrow{a}_{\mathcal{M}} (s_{i_k}, \overrightarrow{X}_k)$, then $(s_{i_{k-1}}, a, \overrightarrow{Y}, s_{i_k}) \in \gamma$, with $\overrightarrow{Y} = \overrightarrow{X}_k - \overrightarrow{X}_{k-1}$. Let $\overrightarrow{Y'} = \overrightarrow{X}_{k-1} - \overrightarrow{X}_{k-2}$.

Consider first the case $\overrightarrow{Y} = \overrightarrow{0}^m$. Hence, $(s_{i_{k-1}}, a, \overrightarrow{0}^m, s_{i_k}) \in \gamma$ and by Part (1) of *State Transition*, $(t_{k-1}, a, \overrightarrow{i_k} - \overrightarrow{i_{k-1}}, t_k) \in \eta$ (just let $j = i_k, i = i_{k-1}$). Therefore, $(t_{k-1}, \overrightarrow{\Xi}_{k-1}) \xrightarrow{a}_{\mathcal{M}_{qd}} (t_k, \overrightarrow{\Xi}_{k-1} + \overrightarrow{i_k} - \overrightarrow{i_{k-1}})$. Let $\overrightarrow{\Xi}_k = \overrightarrow{\Xi}_{k-1} + \overrightarrow{i_k} - \overrightarrow{i_{k-1}}$. If $\overrightarrow{Y'} = \overrightarrow{0}^m$, by Cond. (15) of the induction hypothesis, $\overrightarrow{\Xi}_{k-1} = \overrightarrow{i_{k-1}} \cdot \overrightarrow{X}_{k-1}$. Therefore, $\overrightarrow{\Xi}_k = (\overrightarrow{i_{k-1}} \cdot \overrightarrow{X}_{k-1}) + \overrightarrow{i_k} - \overrightarrow{i_{k-1}} = (\overrightarrow{0}^n \cdot \overrightarrow{X}_{k-1}) + \overrightarrow{i_k} = \overrightarrow{i_k} \cdot \overrightarrow{X}_{k-1} = \overrightarrow{i_k} \cdot \overrightarrow{X}_k$, which is exactly Cond. (15) of the induction hypothesis. If $\overrightarrow{Y'} \neq \overrightarrow{0}^m$, by Cond. (16) of the induction hypothesis, $\overrightarrow{\Xi}_{k-1} = \overrightarrow{i_{k-2}} \cdot \overrightarrow{X}_{k-1}$. We are in case of Part (2.a) of the *State Transition* definition, therefore $(t_{k-1}, a, \overrightarrow{i_k} - \overrightarrow{i_{k-2}}, t_k) \in \eta$. Hence, $(t_{k-1}, \overrightarrow{\Xi}_{k-1}) \xrightarrow{a}_{\mathcal{M}_{qd}} (t_k, \overrightarrow{\Xi}_{k-1} + \overrightarrow{i_k} - \overrightarrow{i_{k-2}})$. Let $\overrightarrow{\Xi}_k = \overrightarrow{\Xi}_{k-1} + \overrightarrow{i_k} - \overrightarrow{i_{k-2}}$. Therefore, $\overrightarrow{\Xi}_k = (\overrightarrow{i_{k-2}} \cdot \overrightarrow{X}_{k-1}) + \overrightarrow{i_k} - \overrightarrow{i_{k-2}} = \overrightarrow{i_k} \cdot \overrightarrow{X}_{k-1} = \overrightarrow{i_k} \cdot \overrightarrow{X}_k$, which is again exactly Cond. (15) (which is the correct one, since if t_{k-1} is checking, then t_k is normal).

Consider now the case $\overrightarrow{Y} \neq \overrightarrow{0}^m$. Since $\mathcal{M}$ is 3-rarefied, $\overrightarrow{Y'} = 0$. By induction hypothesis, $\overrightarrow{\Xi}_{k-1} = (\overrightarrow{i_{k-1}} \cdot \overrightarrow{X}_{k-1})$. Therefore, by Part (2.a) of the *State Transition* case (t_k must be checking), $(t_{k-1}, a, \overrightarrow{0}^n \cdot \overrightarrow{Y}, t_k) \in \eta$. Therefore, $(t_{k-1}, \overrightarrow{\Xi}_{k-1}) \xrightarrow{a}_{\mathcal{M}_{qd}} (t_k, \overrightarrow{\Xi}_{k-1} + (\overrightarrow{0}^n \cdot \overrightarrow{Y}))$. Let $\overrightarrow{\Xi}_k = \overrightarrow{\Xi}_{k-1} + (\overrightarrow{0}^n \cdot \overrightarrow{Y}) = (\overrightarrow{i_{k-1}} \cdot \overrightarrow{X}_{k-1}) + (\overrightarrow{0}^n \cdot \overrightarrow{Y}) = \overrightarrow{i_{k-1}} \cdot (\overrightarrow{X}_{k-1} + \overrightarrow{Y}) = \overrightarrow{i_{k-1}} \cdot \overrightarrow{X}_k$. But this is exactly Cond. (16).

We now show that if $w \in \Sigma$ is accepted by $\mathcal{M}$, then $w \dashv$ is also accepted by $\mathcal{M}_{qd}$, thus ending the proof. Let $(s_{i_0}, \overrightarrow{0}^m) \xrightarrow{w}_{\mathcal{M}} (s_{i_k}, \overrightarrow{X}_k)$, with s_{i_k} being final and $\overrightarrow{X}_k =$

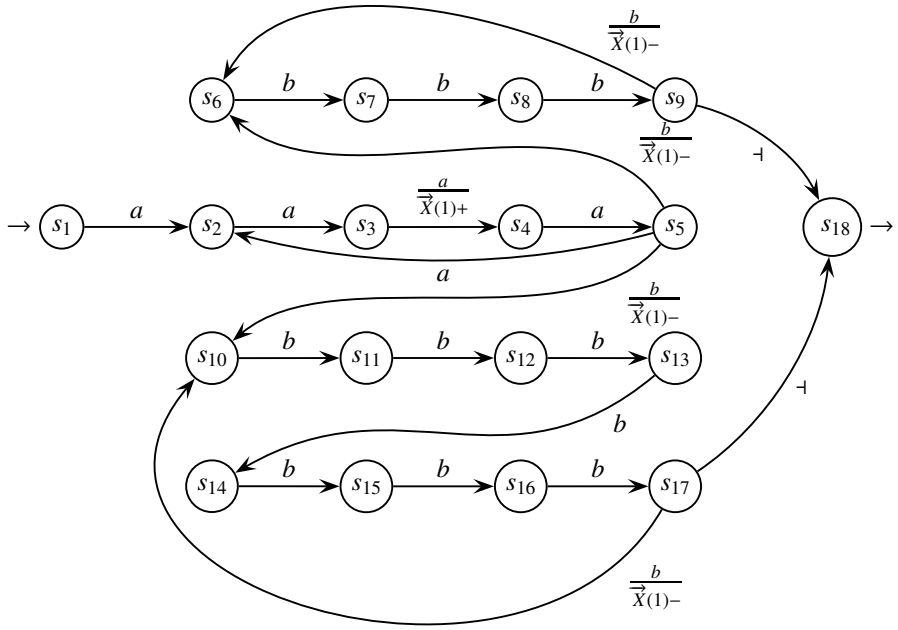


Figure 3: RT-PBLIND rarefied nondeterministic machine with the counter scheduled by module $v = 4$, for language $L_{\text{four}} = \{a^{4n}b^{4n} \mid n \geq 1\} \cup \{a^{4n}b^{8n} \mid n \geq 1\}$.

$\overrightarrow{0}^m$. Hence, $(t_0, \overrightarrow{0}^{n+m}) \xrightarrow{w}_{\mathcal{M}_{qd}} (t_k, \overrightarrow{\Xi}_k)$, with $t_k, \overrightarrow{\Xi}_k$ verifying the conditions of the induction hypothesis above. If Cond. (15) is verified, then $\overrightarrow{\Xi}_k = \overrightarrow{i_k} \cdot \overrightarrow{X}_k = \overrightarrow{i_k} \cdot \overrightarrow{0}^m$: the *Acceptance* definition, Part (1), makes sure that in this case $(t_k, \overrightarrow{\Xi}_k) \longrightarrow_{\mathcal{M}_{qd}} (t_{fin}, \overrightarrow{0}^{n+m})$ (by letting $h = i_k$). If Cond. (16) holds, then t_k is checking and $\overrightarrow{\Xi}_k = \overrightarrow{i_{k-1}} \cdot \overrightarrow{0}^m$: Part (2) of the *Acceptance* definition ensures again that $(t_k, \overrightarrow{\Xi}_k) \longrightarrow_{\mathcal{M}_{qd}} (t_{fin}, \overrightarrow{0}^{n+m})$, by letting $t_k = \langle B', w(k) \rangle$, $i = i_{k-1}, h = i_k$. □

Example 2.13. Quasi-deterministic machine.
To illustrate, we show the QD-RT-PBLIND simple-operation machine obtained from a rarefied nondeterministic machine. We observe that drawing the machine equivalent to the one in Fig. 1 would be unwieldy, because the minimum scheduling module that permits to apply the construction in the proof of Th. 2.8 is $v = 4$; the value $v = 2$ used in Fig. 2 does not leave any slots (residues) for scheduling the actions on the extra counters (flags) used for simulating the original states of the nondeterministic machine. To reduce size, but not significance, we illustrate in Fig. 3 and Fig. 4 the case for the subset of language $L_{2.7}$ that contains the strings having a number of a's multiple of 4, i.e., for language $L_{\text{four}} = \{a^{4n}b^{4n} \dashv \mid n \geq 1\} \cup \{a^{4n}b^{8n} \dashv \mid n \geq 1\}$.

Choose the state partition $\{s_1\}$, $R_0 = \{s_2, s_6, s_{10}, s_{14}\}$, $R_1 = \{s_3, s_7, s_{11}, s_{15}\}$,

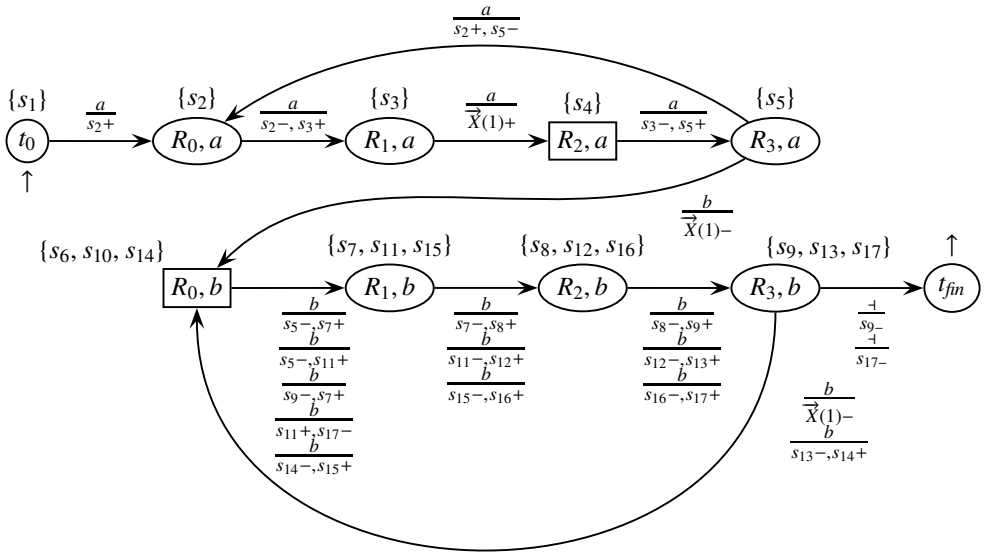


Figure 4: Quasi-deterministic RT-PBLIND machine for language $L_{\text{four}} = \{a^{4n}b^{4n} \mid n \geq 1\} \cup \{a^{4n}b^{8n} \mid n \geq 1\}$. Checking states are framed.

$R_2 = \{s_4, s_8, s_{12}, s_{16}\}, R_3 = \{s_5, s_9, s_{13}, s_{17}\}, \{s_{18}\}$. Fig. 4 depicts the QD machine; a state named by a pair, say (R_3, b), simulates a set of s_i states of the nondeterministic machine, namely the states s_9, s_{13}, s_{17} that are entered upon reading b from each s_j state contained in a predecessor state – here the only predecessor (R_2, b), namely the states s_8, s_{12}, s_{16}. For explanatory reasons, the simulated s_i states are written next to the states and the *checking* states are rectangular. There are, according to the general construction, 17 + 1 counters denoted by $s_1, \ldots, s_{17}, \overrightarrow{X}(1)$; the first 17 counter values are limited to 0 or 1, and are called *flags*. To illustrate how the QD machine simulates the rarefied one, the runs recognizing a^4b^4 and a^4b^8 are tabulated in Fig. 6 of the Appendix.

From the preceding lemmas we have:

Theorem 2.14 (Quasi-deterministic normal form)**.** *For every RT-PBLIND machine $\mathcal{M}$ there exists a QD-RT-PBLIND machine $\mathcal{M}_{QD}$ recognizing $L(\mathcal{M})$ ⊣.*

We recall that machine $\mathcal{M}_{QD}$ has the simple operation property and, at every move, modifies a counter.

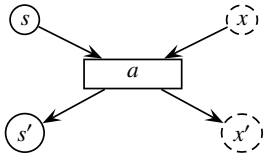
2.5. Application of normal forms to Petri Nets

It is natural and interesting to see how the previous normal forms of multi-counter machines, when applied to Petri Nets, induce a particular network structure. The following informal discussion has three parts corresponding to the cases

of simple operation property, rarefied and scheduled form, and quasi-deterministic form.

Given an RT-PBLIND machine $\mathcal{M}$, with states $S = \{s_1, \ldots, s_n\}$ and m counters, the corresponding PN $\mathcal{P}$ outlined in Th. 2.4 has $n + m$ places that we denote by $s_1, \ldots, s_n, x_1, \ldots, x_m$. The first n places are limited to hold one or zero tokens, and are named *state places*; the other places, called *counter-places*, may contain unbounded values.

Simple operation form. If $\mathcal{M}$ has the simple operation property, all transitions take the form below:

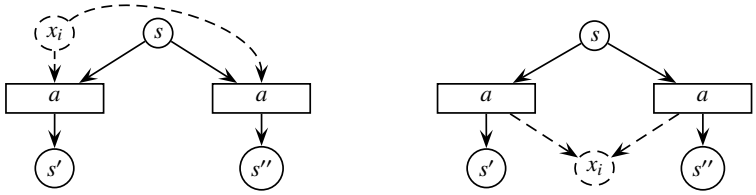


where $s, s' \in S$ are the state-places and x, x', the counter-places (dashed nodes). This PN is in the so called *semi-bounded normal form* of [4], there proved in the much simpler case of a PN model permitting also unlabelled transitions, which is equivalent to PBLIND non-RT machines.

Rarefied normal form with counter operations scheduled. Rarefaction imposes further restrictions on the transitions of the simple operation form. All transitions take one of the forms below:

The left (respectively right) transition is scheduled at positions having as ordinal residue the constant $\boxminus_i$ (respectively $\boxplus_i$).

Quasi-deterministic normal form. The constraints caused by the QD form, assuming the MCM to be also rarefied, are better explained by listing below the subnets that are forbidden:



Notice that, according to Eq. (10) in Def. 2.9, one or both counter operations (dashed arcs) may be absent from each subnet.

We are not aware of any existing similar normal forms for Petri Nets, but some analogy can be found with certain program control-flow graph transformations familiar to compiler designers (e.g., [1]), such as "loop unrolling" and "modulo scheduling" of the instructions contained in a loop.

3. Consensual finite-state computation

The consensual language model was first proposed in [5, 6] and further developed until present. We introduce it by means of an intuitive operational model based on NFA computations, highlighting its difference from the classical nondeterministic FA. Imagine an NFA that performs one of several possible accepting runs to recognize an input word w. In contrast, to recognize "consensually" input w, the NFA performs up to $|w|$ concurrent accepting computations, such that, for each letter $w(1), \ldots, w(|w|)$, exactly one computation - the "leader", asserts the validity of the letter. The remaining computations - the "followers", give their consent to the letter chosen by the leader. Out of metaphor, such dual behavior follower/leader can be represented in the NFA transition graph by two different edge types, say, thin for follower and thick for leader. But we prefer to use instead two distinct copies of each letter $a \in \Sigma$: a dotted letter $\mathring{a}$ denotes a follower type move, and a plain (undotted) letter a a leader type move.

3.1. State vectors of NFA computations

Let $\mathring{\Sigma}$ be the alphabet obtained by *marking* each letter $a \in \Sigma$ as $\mathring{a}$. The union $\Sigma \cup \mathring{\Sigma}$ is named the *double* alphabet and denoted by $\widetilde{\Sigma}$. A language $R \subseteq \left(\widetilde{\Sigma}\right)^*$ over the double alphabet is called a *base language*, because it provides the support needed to define a consensual language.

Consider an NFA recognizing a base language: $A = (\widetilde{\Sigma}, Q, \delta, q_0, F)$. In the following, assume, without loss of generality, that the transition relation δ is total, i.e, for every state q and every terminal a, $|\delta(q, a)| \geq 1$. For all $k > 0$, a k-tuple $\vec{V} \in Q^k$ is called a *state vector* of A of cardinality k. We define a transition relation on state vectors.

Definition 3.1. *The* state vector transition relation *of an NFA A, denoted by*

$$\longrightarrow_{SV_A} \subset_{fin} \; Q^{\mathbb{N}} \times \Sigma \times Q^{\mathbb{N}}$$

is defined, for $a \in \Sigma$, for all $k > 0$, and for all state vectors $\vec{V}, \vec{V}' \in Q^k$, as:

$$\vec{V} \xrightarrow{a}_{SV_A} \vec{V}' \text{ if } \exists j, 1 \leq j \leq k : \; \vec{V}'(j) \in \delta\left(\vec{V}(j), a\right) \quad \text{and} \quad \forall i \neq j, 1 \leq i \leq k : \; \vec{V}'(i) \in \delta\left(\vec{V}(i), \mathring{a}\right) \tag{17}$$

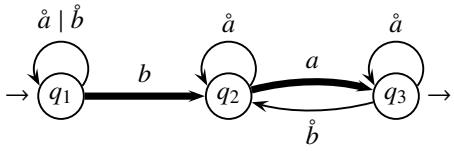


Figure 5: NFA $A_{3.4}$ accepting the base language $L(A_{3.4}) = \left(å \cup \mathring{b}\right)^* b å^* a å^* \left(\mathring{b} å^* a å^*\right)^*$, and consensually recognizing $C(A_{3.4}) = \{ba\, ba^2 \dots ba^k \mid k \geq 1\}$. Leader (follower) transitions are evidenced as thick (thin) edges.

Notice that the definition asserts that exactly one of the k computations reads a (i.e., its next state is in $\delta\left(\vec{V}(j), a\right)$ thus acting as *leader*, while all others read $å$ (i.e., their next state is in $\delta\left(\vec{V}(i), å\right)$ acting as *followers*. We also say that the leader *places* letter a and the followers *consent* to it.

Relation $\xrightarrow{a}_{\mathrm{SV}_A}$ can be extended, as usual, from a letter a to a word $w \in \Sigma^*$. A state vector of A of cardinality k is said to be *initial* if $\vec{V}(i) = q_0$ for all $1 \leq i \leq k$, *final* if $\vec{V}(i) \in F$ for all $1 \leq i \leq k$.

Definition 3.2 (Consensual recognition)**.** *A word $w \in \Sigma^*$ is* consensually recognized *by NFA A if there exist an initial state vector $\vec{V}_0$ and a final state vector $\vec{V}_f$ such that $\vec{V}_0 \xrightarrow{w}_{SV_A} \vec{V}_f$. The* language consensually recognized *by A, denoted by $C^{rec}(A)$ is $\{x \in \Sigma^* \mid x$ is consensually recognized by $A\}$.*

Since at each step there is one and only one leader that places a terminal letter, the following proposition immediately follows:

Proposition 3.3. *The state vectors that are needed to consensually recognize a word x have cardinality upper bounded by $|x|$.*

Example 3.4. The language $L_{3.4} = \{ba^1 ba^2 \dots ba^k \mid k \geq 1\}$ is consensually recognized by the NFA in Fig. 5.

A state-vector transition relation recognizing *babaa* is:

$$\begin{bmatrix} q_1 \\ q_1 \end{bmatrix} \xrightarrow{b}_{\mathrm{SV}_A} \begin{bmatrix} q_2 \\ q_1 \end{bmatrix} \xrightarrow{a}_{\mathrm{SV}_A} \begin{bmatrix} q_3 \\ q_1 \end{bmatrix} \xrightarrow{b}_{\mathrm{SV}_A} \begin{bmatrix} q_2 \\ q_2 \end{bmatrix} \xrightarrow{a}_{\mathrm{SV}_A} \begin{bmatrix} q_3 \\ q_2 \end{bmatrix} \xrightarrow{a}_{\mathrm{SV}_A} \begin{bmatrix} q_3 \\ q_3 \end{bmatrix}$$

Notice that the above state-vector transitions result from the following finite-state computations (equivalent ones exist):

$$\begin{array}{lllll} q_1 \xrightarrow{b} q_2 & \xrightarrow{a} q_3 & \xrightarrow{\mathring{b}} q_2 & \xrightarrow{a} q_3 & \xrightarrow{å} q_3 \\ q_1 \xrightarrow{\mathring{b}} q_1 & \xrightarrow{å} q_1 & \xrightarrow{b} q_2 & \xrightarrow{å} q_2 & \xrightarrow{a} q_3 \end{array}$$

For instance, at step three, the bottom computation is the leader placing letter b, whereas at step four the top computation leads and places letter a. We observe that language $C^{rec}(A_{3,4})$ has a nonsemilinear Parikh image.

Although the configurations reached by an NFA in consensual mode have been represented by a state vector, the order of vector components is irrelevant. It is immediate to see that, if $\vec{V} \xrightarrow{a}_{\mathrm{SV}_A} \vec{V}'$, then, for every permutation $\vec{V}_p$ of $\vec{V}$, there exists a permutation $\vec{V}'_p$ of $\vec{V}'$ such that $\vec{V}_p \xrightarrow{a}_{\mathrm{SV}_A} \vec{V}'_p$. This suggests to formalize the concurrent and matching NFA computations of A by means of multisets of states. In [6] a nondeterministic machine was defined, called a *multiset machine*, having as auxiliary memory a multiset of states of a DFA that recognizes the base language. To avoid confusion, we do not call such device a "counter machine" although it really is one; also, such machine should not be confused with the "multiset" automata studied in [3]. We will define and use the multiset machine in Sect. 4 to prove our main result.

It is straightforward, but important, to observe that the language consensually recognized by an NFA A does not depend on A itself, but only on the *base* language $L(A)$ over the double alphabet. Therefore any formal device apt to define regular languages, say, regular expressions, can be used to specify a consensual language. Next we give a more abstract and general definition of consensual languages.

3.2. Match relation

To express an "agreement" or consensus between words over the double alphabet, we introduce a binary relation, called *match*, over $\widetilde{\Sigma}$, then extended to words.

Definition 3.5 (Match)**.** *The partial symmetric associative binary operator* match $@ : \widetilde{\Sigma} \times \widetilde{\Sigma} \to \widetilde{\Sigma}$*, is defined for all* $a \in \Sigma$*,, then for all words* $w, w' \in (\widetilde{\Sigma})^n, n \geq 0$ *as:*

$$\begin{cases} a@\mathring{a} = \mathring{a}@a = a \\ \mathring{a}@\mathring{a} = \mathring{a} \\ \textit{undefined}, \ \textit{otherwise} \end{cases} \qquad \begin{cases} \varepsilon \,@\, \varepsilon = \varepsilon \\ w@w' = (w(1)@w'(1)) \cdot \ldots \cdot (w(n)@w'(n)) \end{cases}$$

In words, the match is undefined if $|w| \neq |w'|$, or whenever in some position i the match $w(i)@w'(i)$ is undefined, which happens when both letters are in Σ, when both are in $\mathring{\Sigma}$ and differ, and when either one is dotted but is not the dotted copy of the other. For instance, $\mathring{a}ab\mathring{b} \,@\, \mathring{a}\mathring{a}\mathring{b}b = \mathring{a}abb$ while $\mathring{a}ab\mathring{b} \,@\, \mathring{a}\mathring{a}bb$ is undefined.
The *match* of a finite nonempty set of words $w_1, \ldots, w_m \in (\widetilde{\Sigma})^*$ is denoted by $w = w_1@w_2@ \ldots @w_m$ and is a partially defined function. The number m is called the *degree* of the match. The match result is further qualified as *strong* if $w \in \Sigma^*$, or as *weak* otherwise. By Def. 3.5, if w is a strong match, in each position $1 \leq i \leq |w|$, exactly one word, say w_h, is undotted, i.e., $w_h(i) \in \Sigma$, and $w_j(i) \in \mathring{\Sigma}$ for all $j \neq h$;

we say that word w_h *places* the letter at position i and the other words *consent* to it; i.e., w_h is the leader at position i and the others are followers. The match operator is extended to two (or more) languages $L', L'' \subseteq (\widetilde{\Sigma}^*)$ by means of

$$L' \mathbin{@} L'' = \{w' \mathbin{@} w'' \mid w' \in L', w'' \in L''\}$$

and its repeated application to a language is defined by

$$L^{1@} = L\,, \qquad L^{i@} = L \mathbin{@} L^{(i-1)@},\ i \geq 2.$$

Definition 3.6 (Consensual language.)**.** *The* closure under match*, or* @-closure*, of a language* $L \subseteq (\widetilde{\Sigma})^*$ *is* $L^{@} = \bigcup_{i\geq 1} L^{i@}$*. Let* $B \subseteq (\widetilde{\Sigma})^*$*. The* consensual language with base B *is* $C(B) = B^{@} \cap \Sigma^*$*. The family of* consensually regular *languages, denoted by CREG, is the collection of all languages* $C(B)$ *such that* B *is regular.*

It is straightforward that this definition is equivalent to the previous Def. 3.2 of consensual recognition by means of an NFA:

Proposition 3.7. *If language* $B \subseteq (\widetilde{\Sigma})^*$ *is recognized by NFA* A*, then* $C^{rec}(A) = C(B)$*.*

On the other hand, Def. 3.6 is more general than Def. 3.2 because it permits the base language to be nonregular ([6, 7] consider also context-free and context-sensitive bases); but here we only need regular bases.

To illustrate, we specify by means of "consensual regular expressions" some typical languages recognized by counter machines of different types. This suggests that counting operations performed by a multi-counter machine can be simulated by consensual computations, and sets the directions for the formal development in Sect. 4.

Example 3.8 (CREG specification of some counter languages). The deterministic RT-PBLIND 2-counter language $\{a^n b^n c^n \mid n > 0\}$ is defined by the base

$$\mathring{a}^*\, a\, \mathring{a}^*\, \mathring{b}^*\, b\, \mathring{b}^*\, \mathring{c}^*\, c\, \mathring{c}^*$$

The language $L_3 = (\{a^n b^n \mid n \geq 1\})^*$ is accepted by a RT 1-counter machine that is allowed to test for zero, but not by any RT-PBLIND multi-counter machine, as stated in Prop. 2.2. It is consensually defined by the base:

$$\left(\mathring{a} \cup \mathring{b}\right)^*\, a\, \mathring{a}^*\, \mathring{b}^* b\, \mathring{b}^*\, \left(\mathring{a} \cup \mathring{b}\right)^*$$

Each word in L_3 is the concatenation of an unbounded number of substrings of the form $a^{n_1} b^{n_1}, a^{n_2} b^{n_2}, \ldots$: every word in the base places exactly one a and one b into

one of these substrings and consent to any other word. It is then easy to see that all, and only words, in L_3 can be defined by a strong match. For example, a^3b^3ab is the strong match of, for instance, the four words: $\mathring{a}\mathring{a}\mathring{a}\mathring{b}\mathring{b}\mathring{b}ab$, $\mathring{a}a\mathring{a}\mathring{b}\mathring{b}\mathring{a}\mathring{b}$, $a\mathring{a}\mathring{a}\mathring{b}\mathring{b}\mathring{b}\mathring{a}\mathring{b}$ and $\mathring{a}\mathring{a}ab\mathring{b}\mathring{b}\mathring{a}\mathring{b}$.

For language $L_{2.7} = \{a^nb^n \mid n > 0\} \cup \left\{a^nb^{2n} \mid n > 0\right\}$ of the running example (Fig. 1) a consensual regular expression is in [8].

Remarks on closure properties of CREG. First, we list some properties of CREG, then we discuss some implications of its closure properties.

Proposition 3.9. *Known properties of CREG [6, 7]:*

1. *CREG includes REG, is incomparable with both the context-free and deterministic context-free families, it is included within the context-sensitive family, and it contains non-semilinear (in the sense of Parikh) languages.*
2. *CREG is closed under reversal, union and intersection with regular languages, and under inverse (erasing) alphabetic homomorphism.*
3. *CREG is closed under marked union, marked concatenation and marked Kleene closure (as defined, e.g., in [24]).*
4. *The family of regular languages coincides with the family of consensual languages having a strictly locally testable base.*
5. *Membership in CREG can be decided in NLOGSPACE.*
6. *Non-emptiness checking of CREG is undecidable.*

Notice that the homomorphism considered in Item (2) can be *erasing* (the original proof in [6] did not explicitly consider this case but it is straightforward). Although it is not known whether CREG has other closure properties, the classical theory of abstract families of languages (AFL) (some familiarity of the reader is assumed) offers some information on the topics. An immediate consequence of Prop. 3.9 is that the CREG family, although possibly not an AFL, is a *pre-AFL*, i.e., a family of languages closed under inverse homomorphism, intersection with regular languages, union with language $\{\varepsilon\}$, marked concatenation, and marked Kleene closure (see Chapter IV.2 of [24]). Since the closure under (non-erasing) homomorphism of a pre-AFL is an AFL (Prop. 2.1 in Chapter IV.2 of [24]), it follows that closure under non-erasing homomorphism of CREG would entail also closure under union, concatenation and Kleene closure.

A stronger result may be derived from Th. 1 of [15]: any recursively enumerable language may be defined as the (erasing alphabetic) homomorphic image of the intersection of two languages in the family $\mathcal{L}_d$, where $\mathcal{L}_d$ is the smallest family of languages including language $\{a^nb^n \mid n \geq 1\}$ and all regular languages, and closed under marked Kleene + and inverse deterministic Generalized Sequential

Machine mappings (inv. det. GSM). Therefore, any family of languages larger than $\mathcal{L}_d$ and closed under both homomorphic image and intersection includes also all recursively enumerable languages. Since every AFL is closed under inv. det. GSM, it follows that the closure of CREG under non-erasing homomorphism includes the family $\mathcal{L}_d$, since $\{a^n b^n \mid n \geq 1\}$ and all regular languages are in CREG. This leads to the conclusion that CREG cannot be closed under both erasing homomorphism and intersection, else it would include all recursively enumerable languages.

4. Multi-counter and Petri Nets Languages are Consensually Regular

In this section we prove that every RT-PBLIND language is consensually regular. From Th. 2.14 we can assume that the PBLIND machine is quasi-deterministic, a fact that makes it easier to construct the equivalent consensual multiset machine. Then strict inclusion follows from a witness consensual language.

4.1. A multiset notation for consensual languages

First, we define a transition relation based on multisets of states and show its equivalence to the transition relation based on state-vectors. We use the following notation for multisets.

Definition 4.1 (Multiset)**.** *A finite* multiset *over a given set Q is a total mapping $Z : Q \to \mathbb{N}$. The cardinality of multiset Z is $|Z| = \sum_{q \in Q} Z(q)$. If $Z(q) > 0$, then we say that $q \in Z$ with* multiplicity *$Z(q)$. For all multisets Z, Z' over Q, let the* underlying set *be $[\![Z]\!] = \{q \in Q \mid Z(q) > 0\}$, let the* inclusion *$Z \subseteq Z'$ hold if, for every $q \in Q$, $Z(q) \leq Z'(q)$, and let the* sum *$Z \uplus Z'$ and the* difference *$Z - Z'$ be the multisets specified by the following characteristic functions, for all $q \in Q$:*

$$(Z \uplus Z')(q) = Z(q) + Z'(q), \quad (Z - Z')(q) = max(0, Z(q) - Z'(q))$$

Consider a state vector $\vec{V}$ of cardinality $k > 0$. The multiset *Z* associated with *$\vec{V}$ is defined, for all $q \in Q$, as $Z(q) = \left|\left\{1 \leq i \leq k \mid \vec{V}(i) = q\right\}\right|$.*

We now define a condition imposing that all transitions acting as followers are deterministic, with nondeterminism limited to transitions labeled by symbols in Σ.

Definition 4.2. *An NFA $A = (\widetilde{\Sigma}, Q, \delta, q_0, F)$ is called* follower-deterministic *if for every state q and every input symbol $a \in \Sigma$, $|\delta(q, \mathring{a})| \leq 1$.*

In order to define a transition relation on multisets of states, we extend the state transition relation δ to multisets over Q and to symbols in $\mathring{\Sigma}$, by setting for every

multiset Z, for every $\mathring{a} \in \mathring{\Sigma}$, that $\delta(Z, \mathring{a})$ is the multiset whose characteristic function is defined for every $q \in Q$ by:

$$\delta(Z, \mathring{a}) = \sum_{\substack{p \in Q: \\ \delta(p, \mathring{a}) = q}} Z(p).$$

We disregard the action of δ on Σ since it is not required in the following.

Definition 4.3 (multiset transition relation)**.** *The* multiset transition relation *of a follower-deterministic automaton $A = (\overline{\Sigma}, Q, \delta, q_0, F)$, denoted by*

$$\longrightarrow_{MS_A} \subset_{fin} \{Z \mid Z \text{ multiset over } Q\} \times \Sigma \times \{Z \mid Z \text{ multiset over } Q\}$$

is defined, for $a \in \Sigma$, for all $k > 0$, and for all multisets Z, Z' over Q, as:

$$\begin{gathered} Z \xrightarrow{a}_{MS_A} Z' \text{ if, and only if} \\ \exists q \in Z, q' \in \delta(q, a) \mid Z' = \{q'\} \uplus \delta(Z - \{q\}, \mathring{a}). \end{gathered} \tag{18}$$

The relation can be naturally extended from letters to words over Σ.

We apply the multiset transition relation to define languages. A multiset Z_0 such that $Z_0(q_0) > 0$ and $Z_0(q) = 0$ for every $q \neq q_0$ is said to be *initial*. A multiset Z_f such that, for all $q \in Q$ if $Z_f(q) > 0$, then $Q \in F$, is called *final*. A word $w \in \Sigma^*$ is accepted by the multiset transition relation if there exist an initial multiset Z_0 and a final multiset Z_f such that $Z_0 \xrightarrow{w}_{\mathrm{MS}_A} Z_f$. The language of the multiset transition relation is the set of its accepted words.

Since multisets can be represented by counters, the multiset transition relation is easily implemented by a kind of counter machine, called the multiset machine [6]. The original definition assumed, for simplicity, that the finite-state recognizer of the base language is deterministic, a restriction partly lifted here by considering follower-deterministic machines; in fact, for our present purposes, it would be unecessary – though possible – to define the multiset transition relation also for nondeterministic transitions labelled with dotted letters. The state vector and multiset transition relations are equivalent:

Lemma 4.4 (State vector and multiset transition relations)**.** *For all state vectors $\vec{V}$ and $\vec{V}'$, for all words $w \in \Sigma^*$,*

$$\vec{V} \xrightarrow{w}_{SV_A} \vec{V}' \quad \text{if, and only if} \quad \left[\vec{V}\right]_{MS} \xrightarrow{w}_{MS_A} \left[\vec{V}'\right]_{MS}$$

where $\left[\vec{V}\right]_{MS}$ denotes the multiset associated with $\vec{V}$.

Sketch of the proof. For $w = a \in \Sigma$, the two relations are equivalent. This follows immediately by observing that $\vec{V}$ contains a component $\vec{V}(j)$ and $\vec{V}'$ a component $\vec{V}'(j)$ such that $\vec{V}'(j) \in \delta(\vec{V}(j), a)$, and for all other components, with $r \neq j$, of $\vec{V}$ and of $\vec{V}'$, it is $\vec{V}'(r) = \delta(\vec{V}'(r), \mathring{a})$. It suffices to set, in Eq. (18), $Z = \left[\vec{V}\right]_{MS}$, $q = \vec{V}(j)$ and $q' = \vec{V}'(j)$, to obtain the identity $Z' = \left[\vec{V}'\right]_{MS}$.
Then, also the iterated applications of $\longrightarrow_{\mathrm{SV}_A}$ and of $\longrightarrow_{\mathrm{MS}_A}$ are equivalent. □

By definition, a vector $\vec{V_0}$ is initial if, and only if, the multiset $\left[\vec{V_0}\right]_{MS}$ is initial and a vector $\vec{V_F}$ is final if, and only if, the multiset $\left[\vec{V_F}\right]_{MS}$ is final. Therefore, given an automaton A, the state vector and the multiset relation define the same language, i.e.,

$$C^{rec}(A) = \left\{ w \mid \left[\vec{V_0}\right]_{MS} \xrightarrow{w}_{\mathrm{MS}_A} \left[\vec{V_F}\right]_{MS}, \vec{V_0} \text{ initial and } \vec{V_F} \text{ final} \right\}.$$

4.2. From RT-PBLIND to CREG: main result

Theorem 4.5. *Let $\mathcal{M}$ be a real-time partially blind multi-counter machine with input alphabet Σ. Then language $L(\mathcal{M})$ ⊣ is consensually regular.*

Since any RT-PBLIND language can be recognized by a QD-RT-PBLIND machine (Th. 2.14) $\mathcal{M}$ (by adding the end marker ⊣), we are going to construct in Lm. 4.8 an NFA A over the double alphabet, such that $C^{rec}(A) = L(\mathcal{M})$.
To simplify the proof, we reduce the distance between RT-PBLIND machines and consensual devices by means of a technical arrangement, following similar ideas of Greibach [14] for Petri Nets. We wish that every move must increment or decrement a counter i, $1 \leq i \leq m$ and at the same time, respectively, decrement or increment another counter j. This can be formalized by saying that the increment vector is in the set:

$$J_m^m = \left\{ \overrightarrow{j} - \overrightarrow{i} \mid 1 \leq i \neq j \leq m \right\}$$

Notice that vector $\overrightarrow{0}^m$ and all vectors of the forms $\overrightarrow{i}$ or $-\overrightarrow{j}$ are not in J_m^m.

However, since the sum of all counters is now invariant, if all counters started at zero, this machine model would not be able to make any increment! To remedy, we assume that counter $\overrightarrow{X}(m)$, called the *initialized counter*, starts with a nonzero value. The definitions of configuration and transition relation $\longrightarrow$ in Def. 4.3 do not change, but we redefine the initial and final configurations, and thus the recognized language: in the initial and final configuration, counter $\overrightarrow{X}(m)$ must have an identical value $k \geq 0$ (i.e., it starts and ends with a nondeterministically chosen value). Clearly, in every configuration the sum of all counters is equal to k.

Definition 4.6 (Initialized Language)**.** *Let* $\mathcal{M} = \langle \Sigma, S, \gamma, m, s_0, S_{fin} \rangle$ *be an RT-PBLIND machine such that* $\gamma \subseteq S \times \Sigma \times J_m^m \times S$*. A word* $w \in \Sigma^*$ *is an* initialized word *of* $\mathcal{M}$ *if there exist* $k \geq 0$ *and* $s_f \in S_{fin}$ *such that* $(s_0, 0^{m-1}k) \xrightarrow{w}_{\mathcal{M}} (s_f, 0^{m-1}k)$*. The* initialized language *of* $\mathcal{M}$ *is the set* $L_{in}(\mathcal{M})$ *of initialized words of* $\mathcal{M}$*.*

Given a QD-RT-PBLIND machine $\mathcal{M}$ with $m-1$ counters, in simple operation form, it is straightforward to define a QD-RT-PBLIND machine, $\mathcal{M}'$, with m counters and increment vectors in J_m^m, whose initialized language is the same language of $\mathcal{M}$. In fact, by Lm. 2.12, we may assume that each increment vector in the transition relation of $\mathcal{M}$ is not null. Also, the new (initialized) counter m of $\mathcal{M}'$ is incremented (resp. decremented) for every decrement (resp. increment) move of $\mathcal{M}$. On the other hand, the new counter is not modified when the move of $\mathcal{M}$ has the form $\overrightarrow{i} - \overrightarrow{j}$ for some $i \neq j$, $1 \leq i, j \leq m-1$. i.e., a simultaneous increment and decrement. The identity (to be next stated) between $L_{in}(\mathcal{M}')$ and $L(\mathcal{M})$, is trivial, since both machines have essentially the same accepting runs, provided that $\mathcal{M}'$ starts in a configuration such that counter m holds a value k that is at least as large as the maximum sum of all counters during the run, so that counter m cannot try to go below zero. All counters $1, \ldots, m-1$ go back to zero at the end of the run if, and only if, counter m goes back to its initial value k.

Proposition 4.7. *If* $\mathcal{M}$ *is a QD-RT-PBLIND machine with* $m-1$ *counters, then there exists a QD-RT-PBLIND* $\mathcal{M}'$ *machine with m counters and increment vectors in* J_m^m *such that* $L(\mathcal{M}) = L_{in}(\mathcal{M}')$*.*

Next, we prove that QD-RT-PBLIND machines accepts only CREG languages.

Lemma 4.8. *For every QD-RT-PBLIND machine* $\mathcal{M}$*, there exists an NFA A such that* $C^{rec}(A) = L(\mathcal{M})$*.*

Proof. Given a QD-RT-PBLIND machine with $m-1$ counters ($m > 0$) recognizing a language L, we can assume by Prop. 4.7 that there exists a QD-RT-PBLIND machine $\mathcal{M} = \langle \Sigma, S, \gamma, m, s_0, S_{fin} \rangle$ (with m counters) with $\gamma \subseteq S \times \Sigma \times J_m^m \times S$ and such that $L_{in}(\mathcal{M}) = L$. It remains to prove that there exists an NFA A such that $C^{rec}(L(A)) = L_{in}(\mathcal{M})$.

Intuition about the proof. The idea is that A is composed of m copies, numbered $1, \ldots, m$ of the transition graph γ of $\mathcal{M}$. The states of the i-th copy are identified by superscript i. The i-th copy of γ is intended to simulate counter i of $\mathcal{M}$ during a computation on a multiset machine with base language $L(A)$. Let $(s, a, \overrightarrow{Y}, r) \in \gamma$, i.e., $\mathcal{M}$ may go from state s to state r while reading a letter a and with increment vector $\overrightarrow{Y}$. Then, for every copy i there is a (follower) transition from s^i to r^i while

reading $\mathring{a}$, thus in the multiset machine the multiplicity of s^i is transferred to the multiplicity of r^i. If $\overrightarrow{Y}$ increments counter i and decrements counter j (necessarily $i \neq j$), then there is a (leader) transition from s^i to r^j while reading a (i.e., 1 is transferred from s^i to r^j). The initial state of A is s_0^m, which is the copy that corresponds to the initialized counter of $\mathcal{M}$. The formal construction follows.

For all $k > 0$ denote with $Z_{0,k}$ the initial multiset with k occurrences of the initial state s_0^m, i.e., such that $Z_{0,k}(s_0^m) = k, Z_{0,k}(q) = 0$ for all $q \in Q, q \neq s_0^m$. In this way the multiset machine associated with A is able to simulate the original QD-RT-PBLIND machine: if $\mathcal{M}$ is such that $(s_0, \overrightarrow{0}^m k) \xrightarrow{w}_{\mathcal{M}} (s, \overrightarrow{X})$, for some configuration $(s, \overrightarrow{X})$, then there exists a multiset Z such that $Z_{0,k} \xrightarrow{w}_{\mathrm{MS}_A} Z$ with: $Z(s^i) = \overrightarrow{X}(i)$, for every i, $1 \leq i \leq m$, and for all $j, 1 \leq j \leq m$, $Z(r^j) = 0$ for every $r \in S, r \neq s$.

Formal proof. For all $1 \leq i \leq m$, let S^i be a marked copy of S. If $s \in S$, then its marked copy in S^i is denoted by s^i. Define an NFA $A = (\overline{\Sigma}, Q, \delta, s_0^m, F)$, where:

- $Q = \bigcup_{1 \leq i \leq m} S^i$;
- $F = \{s^m \mid s \in S_{fin}\}$;
- The transition function δ is defined for all $a \in \Sigma$, $s \in S$, for all $1 \leq i \leq m$:
  1. $\delta(s^i, a) = \bigcup_{(s,a,\overrightarrow{Y},r) \in \gamma \text{ such that } \overrightarrow{Y}(i) = -1} \left\{ r^j \mid 1 \leq j \leq m, \overrightarrow{Y}(j) = 1 \right\}$.
  2. if there exist $r \in S$ and $\overrightarrow{Y} \in J_m^m$ (Def. 4.6) such that $(s, a, \overrightarrow{Y}, r) \in \gamma$ then $\delta(s^i, \mathring{a}) = r^i$, else $\delta(s^i, \mathring{a}) = \emptyset$.

We first prove:

(1) $L_{in}(\mathcal{M}) \subseteq C(L(A))$.

To show (1), we claim that for all $w \in \Sigma^*, s \in S, \overrightarrow{X} \in \mathbb{N}^m$, if there exists $k > 0$ such that $(s_0, \overrightarrow{0}^{m-1} k) \xrightarrow{w}_{\mathcal{M}} (s, \overrightarrow{X})$, then there exists a multiset $Z \in \mathbb{N}^{|Q|}$ such that:

(1.1) $Z_{0,k} \xrightarrow{w}_{\mathrm{MS}_A} Z$; moreover,

(1.2) for every i, $1 \leq i \leq m$, $Z(s^i) = \overrightarrow{X}(i)$ and $[\![Z]\!] \subseteq \{s^1, \ldots, s^m\}$ (i.e, for every other state $q \in Q$, it is $Z(q) = 0$).

We first show that (1) follows from this claim. Let w be in $L_{in}(\mathcal{M})$. Hence, there exists $k > 0$ such that $(s_0, \overrightarrow{0}^{m-1} k) \xrightarrow{w}_{\mathcal{M}} (s, \overrightarrow{0}^{m-1} k)$ for some $s \in S_{fin}$ By (1.1), $Z_{0,k} \xrightarrow{w}_{\mathrm{MS}_A} Z$, for some $Z \in \mathbb{N}^{|Q|}$. By (1.2), for every i, $1 \leq i \leq m-1$, $Z(s^i) = 0$, $Z(s^m) = k$ while $Z(s'^j) = 0$ for $0 \leq j \leq m, s' \neq s$. Hence, $[\![Z]\!] = \{s^m\}$, i.e., Z is a final multiset, since $s^m \in Q_{fin}$, therefore $w \in C(L(A))$.

The proof of Claim (1) is by induction on $|w| > 0$. The base case considers $w = a \in \Sigma$. Let $(s_0, \overrightarrow{0}^{m-1} k) \xrightarrow{a}_{\mathcal{M}} (s, \overrightarrow{X})$, with $\overrightarrow{X}(j) = 1, \overrightarrow{X}(m) = k - 1$ for some

$1 \leq j < m$ (the case $j = m$ is simpler). Let $\overrightarrow{Y} = \overrightarrow{X} - \overrightarrow{0}^{m-1}k$, i.e., $\overrightarrow{Y}(j) = 1$, $\overrightarrow{Y}(m) = -1$. Hence, $(s_0, a, \overrightarrow{Y}, s) \in \gamma$ and by definition of δ, $s^i \in \delta(s_0^m, a)$ and $\delta(s_0^m, \mathring{a}) = s^m$. Let Z be the multiset such that $[\![Z]\!] = \{s^m, s^j\}$, with $Z(s^m) = k-1$, $Z(s^j) = 1$. Hence, (1.2) holds. By definition of $\longrightarrow_{\mathrm{MS}_A}$, $Z_{0,k} \xrightarrow{a}_{\mathrm{MS}_A} Z$, hence also (1.1) holds.

Assume the induction hypothesis holds for w, $|w| > 0$. For all $a \in \Sigma$, consider a run $(s_0, \overrightarrow{0}^{m-1}k) \xrightarrow{wa}_{\mathcal{M}} (s', \overrightarrow{X'})$, for some $k > 0, s' \in S, \overrightarrow{X'} \in \mathbb{N}^m$, which can be written as $(s_0, \overrightarrow{0}^{m-1}k) \xrightarrow{w}_{\mathcal{M}} (s, \overrightarrow{X}) \xrightarrow{a}_{\mathcal{M}} (s', \overrightarrow{X'})$ for some $s \in S, \overrightarrow{X} \in \mathbb{N}^m$ verifying the induction hypothesis: there exists $Z \in \mathbb{N}^{|Q|}$ such that $Z_{0,k} \xrightarrow{w}_{\mathrm{MS}_A} Z$ and for every i, $1 \leq i \leq m$, $Z(s^i) = \overrightarrow{X}(i)$. By definition of $\xrightarrow{a}_{\mathcal{M}}$, there exists $\overrightarrow{Y} \in J_m^m$ such that $(s, a, \overrightarrow{Y}, s') \in \gamma$ and $\overrightarrow{X'} = \overrightarrow{X} + \overrightarrow{Y}$. By definition of δ, for all i, $0 \leq i \leq m$, $\delta(s^i, \mathring{a}) = s'^i$. For all $1 \leq i, j \leq m$ such that $\overrightarrow{Y}(j) = 1$ and $\overrightarrow{Y}(i) = -1$, then $s'^j \in \delta(s^i, a)$. Let Z'_j be such that $Z'_j(s') = j$ and $[\![Z'_j]\!] = \{s'\}$. Let $Z' = Z'_j \uplus \delta(Z - \{s^i\}, \mathring{a})$.

By Def. 4.3, it follows that $Z \xrightarrow{a}_{\mathrm{MS}_A} Z'$, hence (1.1) holds for Z'. To show (1.2), consider that for every n, if $n \neq j, n \neq i$, then $Z'(s'^n) = Z(s^n)$ (since $\delta(s^n, \mathring{a}) = s'^n$), while $Z'(s'^j) = Z(s^j)+1$, $Z'(s'^i) = Z(s^i)-1$. By induction hypothesis, for every l, $1 \leq l \leq m$, $Z(s^l) = \overrightarrow{X}(l)$ and no other non-zero element is in Z. Hence, for all $l, l \neq j, l \neq i$, $Z'(s'^l) = Z(s^l) = \overrightarrow{X}(l) = \overrightarrow{X'}(l)$, since $\overrightarrow{X'}(l) = \overrightarrow{X}(l)+\overrightarrow{Y}(l)$ with $\overrightarrow{Y}(l) = 0$; $Z'(s'^i) = Z(s^i) - 1 = \overrightarrow{X}(i) - 1 = \overrightarrow{X'}(i)$; $Z'(s'^j) = Z(s^j) + 1 = \overrightarrow{X}(j) + 1 = \overrightarrow{X'}(j)$, and no other element is in Z' (i.e., $[\![Z']\!] \subseteq \{s'^1, \ldots, s'^m\}$).

To prove the converse relation:

(2) $C(L(A)) \subseteq L_{in}(\mathcal{M})$,

we claim that for all $w \in \Sigma^+$, if there exist $k > 0$ and a multiset $Z \in \mathbb{N}^{|Q|}$ such that $Z_{0,k} \xrightarrow{w}_{\mathrm{MS}_A} Z$, then there exist $s \in S, \overrightarrow{X} \in \mathbb{N}^m$ such that

(2.1) $(s_0, \overrightarrow{0}^{m-1}k) \xrightarrow{w}_{\mathcal{M}} (s, \overrightarrow{X})$,

(2.2) for every i, $1 \leq i \leq m$, $Z(s^i) = \overrightarrow{X}(i)$ and $[\![Z]\!] \subseteq \{s^1, \ldots, s^m\}$ (i.e, no other element is in Z).

Thesis (2) follows from this claim: if $w \in C(L(A))$, then by Prop. 3.7 and Lemma 4.4 there exist $k < |w|$ and a final multiset $Z \in \mathbb{N}^{|Q|}$ such that $Z_{0,k} \xrightarrow{w}_{\mathrm{MS}_A} Z$. Hence, (2.1) holds, and since Z is final, only $Z(s^m) = \overrightarrow{X}(m)$ may be greater than 0. Therefore, $\overrightarrow{X}(1) = \overrightarrow{X}(2) \cdots = \overrightarrow{X}(m-1) = 0$, $\overrightarrow{X}(m) = k$ and s is a final state: $(s, \overrightarrow{X})$ is a final configuration, hence $w \in L_{in}(\mathcal{M})$.

The proof of claim (2) is by induction on $n = |w| > 0$. The base case considers $w = a$. Let $k > 0$ be any integer number. Then, $Z_{0,k} \xrightarrow{a}_{\mathrm{MS}_A} Z$ for some $Z \in \mathbb{N}^{|Q|}$. Hence, by definition of $\longrightarrow_{\mathrm{MS}_A}$, $\delta(s_0^m, a)$ must be defined: by definition of A, there exist $s \in S$ and $1 \leq j \leq m$ such $s^j \in \delta(s_0^m, a)$, $\delta(s_0^m, \mathring{a}) = s^m$. Consider the case

$j \neq m$ (the case $j = m$ is simpler). Therefore, $(s_0, a, \overrightarrow{Y}, s) \in \gamma$, with $\overrightarrow{Y} \in J_m^m$, $\overrightarrow{Y}(j) = 1, \overrightarrow{Y}(m) = -1$. Hence, $(s_0, \overrightarrow{0}^{m-1}k) \xrightarrow{a}_{\mathcal{M}} (s, \overrightarrow{X})$, for $\overrightarrow{X}(j) = 1, \overrightarrow{X}(m) = k - 1$, and $\overrightarrow{X}$ equal to zero in every other position. This is (2.1); moreover, clearly by definition of $\longrightarrow_{\mathrm{MS}_A}$ $Z(s^j) = 1, Z(s^m) = k - 1$ and $[\![Z]\!] = \{s^j, s^m\}$, i.e., no other element is in Z, satisfying also (2.2). Assume now that the induction hypothesis holds for all words of length up to $n \geq 0$. Consider a word w' of length $n + 1$ such that there exist $k \leq |w'|$ and a multiset $Z' \in \mathbb{N}^{|Q|}$ such that $Z_{0,k} \xrightarrow{wa}_{\mathrm{MS}_A} Z'$. Hence, there exist a word w, of length n, and a symbol $a \in \Sigma$ such that $w' = wa$, and there exists $Z \in \mathbb{N}^{|Q|}$ such that $Z_{0,k} \xrightarrow{w}_{\mathrm{MS}_A} Z \xrightarrow{a}_{\mathrm{MS}_A} Z'$.

By induction hypothesis, (2.1) holds: there exist $s \in S, \overrightarrow{X} \in \mathbb{N}^m$ such that

$$(s_0^m, \overrightarrow{0}^{m-1}k) \xrightarrow{w}_{\mathcal{M}} (s, \overrightarrow{X}),$$

with $Z, s, \overrightarrow{X}$ satisfying (2.2): for all $1 \leq h \leq m$, $Z(s^h) = \overrightarrow{X}(h)$ and, for every state $r \neq s$, $Z(r^h) = 0$ by induction hypothesis. Hence, there exists i such that $\delta(s^i, a)$ is defined, i.e., there exist j and $s' \in S$ such that $s'^j \in \delta(s^i, a)$, with $Z' = \{s'^j\} \uplus \delta(Z - \{s^i\}, \mathring{a})$. By definition of δ, since $s'^j \in \delta(s^i, a)$, for all $1 \leq h \leq m$ also $s'^h \in \delta(s^h, \mathring{a})$; moreover, there exists $\overrightarrow{Y} \in J_m^m$ such that $(s, a, \overrightarrow{Y}, s') \in \gamma$ with $\overrightarrow{Y}(j) = 1, \overrightarrow{Y}(i) = -1$. Therefore, $(s, \overrightarrow{X}) \xrightarrow{a}_{\mathcal{M}} (s', \overrightarrow{X} + \overrightarrow{Y})$ and $s'^j \in \delta(s^i, a)$.
Then, (2.1) holds: $(s_0, \overrightarrow{0}^{m-1}k) \xrightarrow{wa}_{\mathcal{M}} (s', \overrightarrow{X} + \overrightarrow{Y})$.
We prove that (2.2) holds for $Z', s', \overrightarrow{X} + \overrightarrow{Y}$. Since $Z' = \{s'^j\} \uplus \delta(Z - \{s^i\}, \mathring{a})$, then:

$$\text{for all } h, 1 \leq h \leq m, \text{ if } r \neq s', Z'(r^h) = 0, \text{ and for } h \neq i, h \neq j:$$
$$Z'(s'^h) = Z(s^h) = \overrightarrow{X}(h). \tag{19}$$

For all $h \neq i, h \neq j$, $1 \leq h \leq m$, we have $\overrightarrow{Y}(h) = 0$; hence by (19), being $i \neq j$, $Z'(s'^h) = \overrightarrow{X}(h) = \overrightarrow{X}(h) + \overrightarrow{Y}(h)$. Consider $Z'(s'^i), Z'(s'^j)$, with $\overrightarrow{Y}(j) = 1, \overrightarrow{Y}(i) = -1$: again, being $Z' = \left\{s'^j\right\} \uplus \delta\left(Z - \left\{s^i\right\}, \mathring{a}\right)$ and $s'^h \in \delta(s^h, \mathring{a})$ for all $1 \leq h \leq m$, then $Z'(s'^i) = Z(s^i) - 1 = \overrightarrow{X}(i) - 1 = \overrightarrow{X}(i) + \overrightarrow{Y}(i)$ and $Z'(s'^j) = Z(s^j) + 1 = \overrightarrow{X}(j) + \overrightarrow{Y}(j)$ and $[\![Z']\!] \subseteq \left\{s'^1, \ldots, s'^m\right\}$. □

Example A.2 in Appendix shows the NFA recognizing the base language that consensually defines L_{four} of Fig. 4.

Th. 4.5 follows then immediately. It is also immediate to see that the inclusion in Th. 4.5 is strict, by recalling that language $L_{4.9} = (\{a^n b^n \mid n \geq 1\})^*$ is in CREG (Ex. 3.8) but is not RT-PBLIND (Prop. 2.2):

Theorem 4.9. *There exist languages in the CREG family that are not in the RT-PBLIND family.*

We observe that language $L_{4.9}$ is recognized by a deterministic RT 1-counter machine that tests for zero. On the other hand, CREG also includes the similar language $(\{a^n b^m \mid n \geq m \geq 1\})^*$, defined by the base: $\left(\mathring{a}^+\mathring{b}^+\right)^* \mathring{a}^* a \mathring{a}^* \mathring{b}^* (b \cup \mathring{b})\mathring{b}^* \left(\mathring{a}^+\mathring{b}^+\right)^*$. Moreover, this language cannot be recognized in RT by any MCM performing zero tests if the machine is deterministic (Th. 5.3 of [11]), yet it can be recognized in the nondeterministic case. The relation of such (deterministic or nondeterministic) RT non-PBLIND machines with CREG is not known.

Finally, we compare CREG with PBLIND machines also not operating in RT.

Theorem 4.10. *CREG is incomparable with the family PBLIND of (also non-realtime) partially blind multi-counter languages.*

Proof. First, the non-PBLIND language $L_{4.9} = (\{a^n b^n \mid n \geq 1\})^*$ is consensually regular. Second, let $L_{\text{binary}} = \left\{wc^h \mid w \in \{0,1\}^+, h \leq \text{bin}(1w)\right\}$, where $\text{bin}(1w)$ is the natural number whose binary representation is $1w$ (i.e., the binary string w with an extra most significant 1 bit). In [14] it is proven that $L \in \text{PBLIND} - \text{RT-PBLIND}$. We claim that L_{binary} is not in CREG. Suppose by contradiction that there exists a DFA $A = (\widetilde{\Sigma}, Q, \delta, q_0, F)$, with $k = |Q| > 0$ states such that $C^{rec}(A) = L_{\text{binary}}$. Let $n > 0$ be a number such that $2^n > (n+1)^k$. In what follows, $\{(q)^h\}$, for $q \in Q$ and $h > 0$, denotes the multiset Z over Q such that $Z(q) = h$, $Z(q') = 0$ for $q' \neq q$. Also, let $dot : \Sigma \to \mathring{\Sigma}$ be the homomorphism such that for all $a \in \Sigma$, $dot(a) = \mathring{a}$.
We claim that for all $w \in \{0,1\}^n$ there exist an integer $h \in \mathbb{N}$, with $n \leq h \leq 2^n + n$, a state $q \in Q$, a multiset $\widetilde{Z} : Q \to \mathbb{N}^k$, and a final multiset $Z_F : F \to \mathbb{N}^k$ such that:

$$\{(q_0)^h\} \xrightarrow{w}_{\text{MS}_A} \left(\widetilde{Z} \uplus \left\{q^{h-n}\right\}\right) \overset{c^{\text{bin}(1w)}}{\longrightarrow}_{\text{MS}_A} \widetilde{Z}_F \tag{20}$$

with $q = \delta(q_0, dot(w))$ and $|\widetilde{Z}| = n$. Notice that h is assumed to be greater than n, without loss of generality (else just replace $h - n$ with 0). In fact, since $h > n$ and $n = |w|$, at most n computations of A can make a strong match. Hence, there are $h - n$ computations that, after reading w, are in state $\delta(q_0, dot(w))$.

Consider the subset W_n of L_{binary} defined as $W_n = \left\{wc^{\text{bin}(1w)} \mid w \in \{0,1\}^n\right\}$. The cardinality of W_n is 2^n, while the number of possible values for $\widetilde{Z}$ and q, as defined in (20), is less than n^{k+1}. Hence, there are at least two words $wc^{\text{bin}(1w)}, w'c^{\text{bin}(1w')} \in W_n$ such that there exist two integers h, h', with $n \leq h, h' \leq 2^n + n$, two final multisets Z_f, Z'_f, one state $q \in Q$ and one multiset $\widetilde{Z}$, with $|\widetilde{Z}| = n$, such that relation (20) holds for $wc^{\text{bin}(1w)}$, while for $w'c^{\text{bin}(1w')}$ we have:

$$\{(q_0)^{h'}\} \xrightarrow{w'}_{\text{MS}_A} \left(\widetilde{Z} \uplus \left\{(q)^{h'-n}\right\}\right) \overset{c^{\text{bin}(1w')}}{\longrightarrow}_{\text{MS}_A} \widetilde{Z}'_f. \tag{21}$$

Suppose that $\text{bin}(1w') > \text{bin}(1w)$ (the other case being symmetrical). By combining relations (20) and (21) we obtain

$$\{(q_0)^{h'}\} \xrightarrow{w}_{\text{MS}_A} \left(\widetilde{Z} \uplus \left\{(q)^{h'-n}\right\}\right) \overset{c^{\text{bin}(1w')}}{\longrightarrow}_{\text{MS}_A} \widetilde{Z}'_f$$

which means that $wc^{\text{bin}(1w')} \in L_{\text{binary}}$, hence $\text{bin}(1w') \leq \text{bin}(1w)$, a contradiction with the assumption $\text{bin}(1w') > \text{bin}(1w)$. □

5. Conclusion

The interest for abstract machines using integer counters and for their languages is almost as old as for the Chomsky's families of languages, dating back at least to Minsky work [20]. Yet, unlike the latter, multi-counter languages are not associated with grammars or with other types of declarative specification, and their structural properties are less understood, except in the basic reversal-bounded model. Our research contributes two new normal forms for nondeterministic real-time partially blind machines, therefore also for the languages of a wide class of Petri Nets: the modulo-scheduled rarefied form and the quasi-deterministic form. Both forms should be interesting in their own for proving properties on such machines and Petri Nets.

Exploiting the normal forms, we have been able to simulate such multi-counter machines on the very different device of consensually regular languages. What is interesting, is that the simulation essentially involves a transformation from a sequential device to a parallel one: a single sequential computation is converted to multiple threads specified by a finite automaton, which are step-by-step synchronized by means of the consensual match function. As a consequence, it becomes possible to specify RT-PBLIND (and *a fortiori* reversal bounded multi-counter machines [17, 16]) or Petri Net languages by means of regular expressions, a notation we (subjectively) find quite readable and amenable to language transformation and composition, as already exemplified in past work on consensual languages.

We mention some open questions concerning the relationship of CREG to multi-counter languages. CREG languages strictly include the RT-PBLIND ones, but we still lack a precise characterization of what types of RT multi-counter languages are consensually regular. For instance, the deterministic machines of [11], endowed with zero-testing capability, can in some cases be simulated by a multiset consensual machine, but their precise relation with CREG is unknown (and in Sect. 4 we have shown that there are consensual languages which are not in this deterministic class). Multiset consensual machines bear some similarity to Vitanyi's *augmented counter machines* [27, 28], which have their counters initialized to some value and offer the additional one-step assignment operation: "set counter

i to the value of counter *j*". In real-time, these augmented machines are more powerful than the classical multi-counter machines, but their relationship to CREG is unknown. In fact, the consensual approach to language definition cuts across traditional classifications of counter-based abstract machines and its overall relation with established models is only partially understood.

Aknowledgment. We thank the anonymous reviewers for their accurate and useful suggestions.

References

[1] A. W. Appel and J. Palsberg. *Modern Compiler Implementation in Java*. Cambridge University Press, New York, USA, 2nd edition, 2003.

[2] M. Berglund, H. Björklund, and J. Björklund. Shuffled languages - representation and recognition. *Theor. Comput. Sci.*, 489-490:1–20, 2013.

[3] C. Calude, G. Paun, G. Rozenberg, and A. Salomaa, editors. *Multiset Processing, Mathematical, Computer Science, and Molecular Computing Points of View*, volume 2235 of *LNCS*. Springer, 2001.

[4] S. Crespi Reghizzi and D. Mandrioli. Petri nets and Szilard languages. *Information and Control*, 33(2):177–192, 1977.

[5] S. Crespi Reghizzi and P. San Pietro. Consensual definition of languages by regular sets. In C. Martín-Vide, F. Otto, and H. Fernau, editors, *LATA 2008*, volume 5196 of *LNCS*, pages 196–208. Springer, 2008.

[6] S. Crespi Reghizzi and P. San Pietro. Consensual languages and matching finite-state computations. *RAIRO - Theor. Inf. and Applic*, 45(1):77–97, 2011.

[7] S. Crespi Reghizzi and P. San Pietro. Strict local testability with consensus equals regularity. In N. Moreira and R. Reis, editors, *CIAA 2012*, volume 7381 of *LNCS*, pages 113–124. Springer, 2012.

[8] S. Crespi Reghizzi and P. San Pietro. Deterministic counter machines and parallel matching computations. In S. Konstantinidis, editor, *CIAA 2013*, volume 7982 of *LNCS*, pages 280–291. Springer, 2013.

[9] S. Crespi Reghizzi and P. San Pietro. Strict local testability with consensus equals regularity, and other properties. *Int. J. Found. Comput. Sci.*, 24(6):747–764, 2013.

[10] S. Crespi Reghizzi and P. San Pietro. Commutative languages and their composition by consensual methods. In Z. Ésik and Z. Fülöp, editors, *AFL 2014*, pages 216–230, 2014.

[11] P. C. Fischer, A. R. Meyer, and A. L. Rosenberg. Counter machines and counter languages. *Math. Syst. Theory*, 2(3):265–283, 1968.

[12] V. K. Garg and M. T. Ragunath. Concurrent regular expressions and their relationship to Petri Nets. *Theor. Comput. Sci.*, 96(2):285–304, Apr. 1992.
[13] S. A. Greibach. Remarks on the complexity of nondeterministic counter languages. *Theor. Comput. Sci.*, 1(4):269–288, Apr. 1976.
[14] S. A. Greibach. Remarks on blind and partially blind one-way multicounter machines. *Theor. Comput. Sci.*, 7:311–324, 1978.
[15] J. Hartmanis and J. Hopcroft. What makes some language theory problems undecidable. *J. Comput. and Syst. Sci.*, 4(4):368 – 376, 1970.
[16] J. Hromkovic. Hierarchy of reversal and zerotesting bounded multicounter machines. In M. Chytil and V. Koubek, editors, *MFCS 1984*, volume 176 of *LNCS*, pages 312–321. Springer, 1984.
[17] O. H. Ibarra. Reversal-bounded multicounter machines and their decision problems. *J. ACM*, 25(1):116–133, 1978.
[18] M. Jantzen. On the hierarchy of Petri net languages. *ITA*, 13(1), 1979.
[19] J. Jędrzejowicz and A. Szepietowski. Shuffle languages are in P. *Theor. Comput. Sci.*, 250(1-2):31–53, Jan. 2001.
[20] M. Minsky. Recursive unsolvability of Post's problem of 'tag' and other topics in the theory of Turing machines. *Annals of Math.*, 74:3:437–455, 1961.
[21] M. Minsky. *Computation: Finite and Infinite Machines*. Prentice-Hall, Englewood Cliffs, USA, 1976.
[22] B. Nagy. Languages generated by context-free grammars extended by type AB -> BA rules. *Journal of Automata, Languages and Combinatorics*, 14(2):175–186, 2009.
[23] B. Nagy and F. Otto. On CD-systems of stateless deterministic R-automata with window size one. *J. Comput. Syst. Sci*, 78(3):780–806, 2012.
[24] A. Salomaa. *Formal languages*. Academic Press, New York, NY, 1973.
[25] L. Sha, T. F. Abdelzaher, K. Årzén, A. Cervin, T. P. Baker, A. Burns, G. C. Buttazzo, M. Caccamo, J. P. Lehoczky, and A. K. Mok. Real time scheduling theory: A historical perspective. *Real-Time Systems*, 28(2-3):101–155, 2004.
[26] M. H. ter Beek and J. Kleijn. Shuffles and synchronized shuffles: A survey. In G. Paun, G. Rozenberg, and A. Salomaa, editors, *Discrete Mathematics and Computer Science. In Memoriam Alexandru Mateescu (1952-2005).*, pages 37–50. The Publishing House of the Romanian Academy, 2014.
[27] P. Vitányi. An optimal simulation of counter machines. *SIAM J. Comput.*, 14(1):1–33, 1985.
[28] P. Vitányi. An optimal simulation of counter machines: the ACM case. *SIAM J. Comput.*, 14(1):34–40, 1985.

Appendix

Example A.1. To illustrate how the QD-RT-PBLIND machine simulates the rarefied one, the runs recognizing a^4b^4 and a^4b^8 are tabulated in Fig. 6.

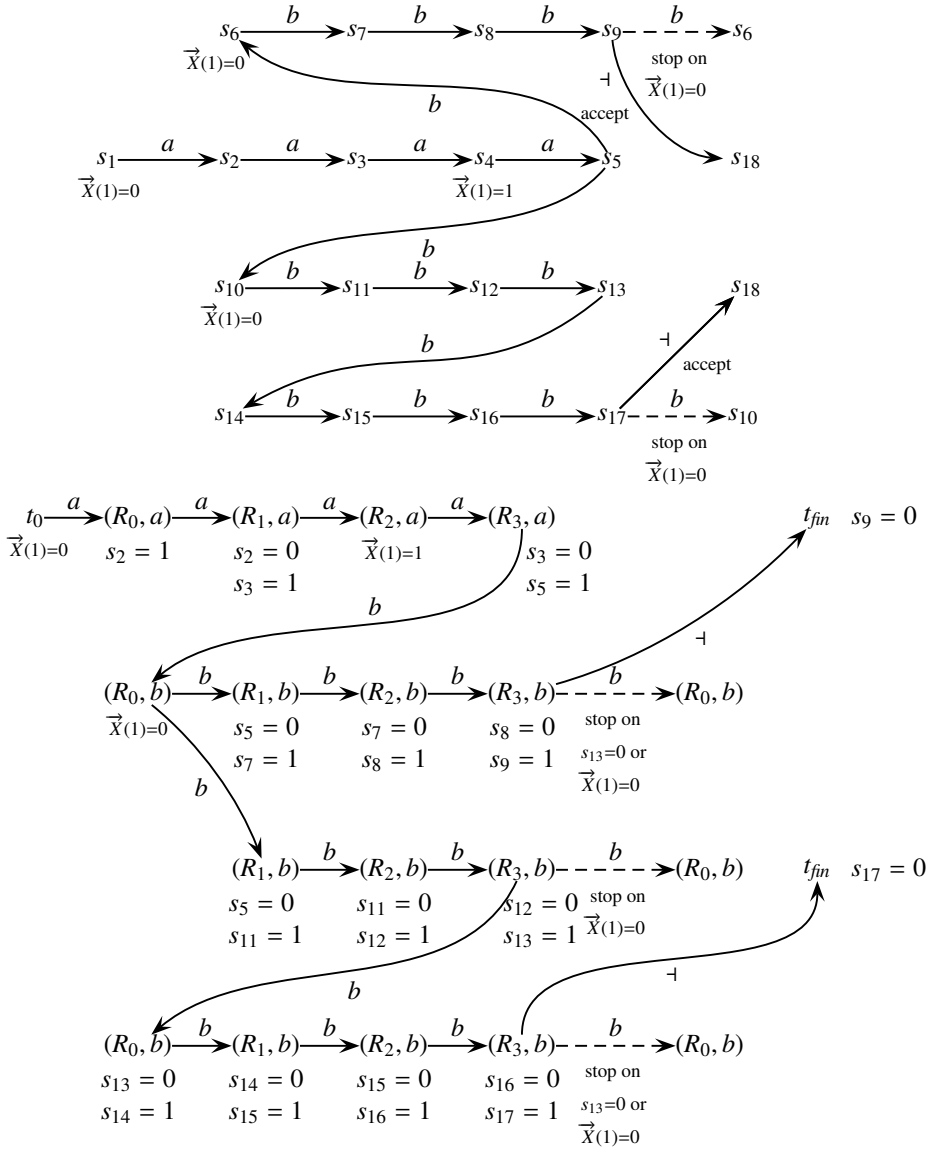


Figure 6: Runs on input a^4b^4, a^4b^8: (top) on the rarefied machine in Fig. 3 and (bottom) on the QD machine in Fig. 4. Counter values are displayed when modified. Dashed transitions stop in error.

Example A.2. From QD-RT-PBLIND machine to CREG Language

We start from the QD machine in Fig. 4, and we add the extra initialized counter, as in Def. 4.6, obtaining a machine with 10 states and 16 counters:

$$\{t_0, (R_0, a), (R_1, a), (R_2, a), (R_3, a), (R_0, b), (R_1, b), (R_2, b), (R_3, b) t_{fin}\} \tag{22}$$
$$\{s_2, s_3, s_4, s_5, s_7, s_8, s_9, s_{11}, s_{12}, s_{13}, s_{14}, s_{15}, s_{16}, s_{17}, \overrightarrow{X}(1), init\}$$

where *init* is the initialized counter. Notice that some states (namely s_1, s_6 and s_{10}) of the nondeterministic machine in Fig. 3 do not have associated flags. The states Q of NFA A serving as base for the consensual language, are the Cartesian product of the two sets in Eq. (22). The transition relation δ of A is deterministic on the sub-domain $Q \times \mathring{\Sigma}$ and nondeterministic on the sub-domain $Q \times \Sigma$. It is represented in Fig. 7. To avoid clogging, states are represented as unnamed points in the Cartesian grid. Leader type and follower type transitions are respectively represented by thick and by thin edges. Notice that we have omitted the thin edges that result from the construction in the proof of Lm. 4.8, but are never traversed by a consensual computation.

To illustrate, strings a^4b^4 and a^4b^8 are the match of the following computations:

$$\begin{array}{cccccccccc} a & a & \mathring{a} & a & \mathring{b} & b & b & b & \dashv \\ \mathring{a} & \mathring{a} & a & \mathring{a} & b & \mathring{b} & \mathring{b} & \mathring{b} & \mathring{\dashv} \end{array} \qquad \begin{array}{cccccccccccc} a & a & \mathring{a} & a & \mathring{b} & b & b & b & b & b & b & \dashv \\ \mathring{a} & \mathring{a} & a & \mathring{a} & b & \mathring{b} & \mathring{b} & \mathring{b} & \mathring{b} & \mathring{b} & \mathring{b} & \mathring{\dashv} \end{array}$$

At last, notice that the edge $(s_{13}, R_{3,b}) \xrightarrow{\mathring{b}} (s_{13}, R_{0,b})$, though obtained by the construction, is never productive.

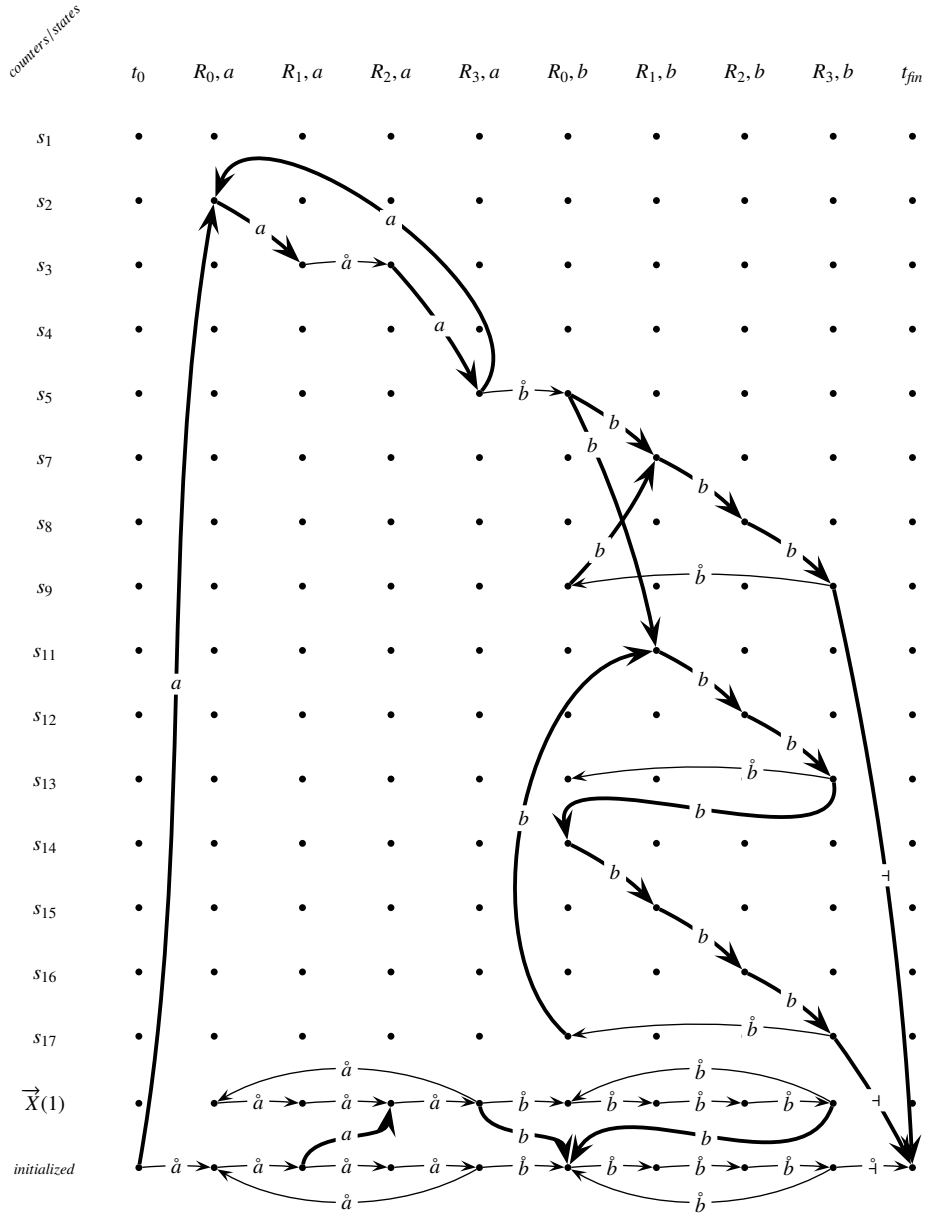


Figure 7: NFA A, constructed according to Lm. 4.8, for the base language B such that $C(B) = L_{\text{four}}$. Follower and leader transitions are respectively represented by thin and thick arrows. Follower transitions that are never executed are not drawn, to avoid clogging.