

# Some undecidable properties of SOS specifications<sup>☆</sup>

Bartek Klin

*University of Warsaw*

Beata Nachyła

*Institute of Computer Science, Polish Academy of Sciences*

---

## Abstract

Several properties of SOS specifications with negative premises, often used to define what it means for a specification to be meaningful, are proved undecidable. This includes the existence of least or unique supported or stable models, and specification completeness.

*Keywords:* structural operational semantics, negative premises, supported model, stable model, complete ntyft/ntyxt  
*2000 MSC:* 68Q85, 68Q55

---

## 1. Introduction

Structural Operational Semantics (SOS, see [1] for the original account and [2, 3] for surveys) is a standard way of defining labeled transition systems (LTSs) whose states are terms over some algebraic signature. A typical SOS specification consists of rules such as

$$\frac{x \xrightarrow{a} x'}{\mathbf{f}(x, y) \xrightarrow{a} \mathbf{g}(y, x')}$$

---

<sup>☆</sup>This work was supported by the Polish National Science Centre (NCN) grant 2012/07/E/ST6/03026.

where  $\mathbf{f}$  and  $\mathbf{g}$  are operation symbols from the signature and  $a$  is a transition label. This rule says that for all terms  $t$  and  $s$ , if  $t$  makes an  $a$ -labeled transition to a term  $t'$  then the term  $\mathbf{f}(t, s)$  makes a similar transition to  $\mathbf{g}(s, t')$ .

The general intuition is that a set of rules defines an LTS that consists of all transitions that can be proved by composing (closed instances of) those rules. As long as the specification is *positive*, i.e., it conditions the presence of some transitions only on the presence (and not on the absence) of other transitions, this intuition can be made entirely formal. However, if rules contain so-called *negative premises*, matters become complicated [4]. For example, for the specification

$$\frac{\mathbf{C} \not\rightarrow}{\mathbf{C} \xrightarrow{a} \mathbf{D}} \quad (1)$$

(where  $\mathbf{C}, \mathbf{D}$  are constants in the signature), it is not clear whether an  $a$ -labeled transition from  $\mathbf{C}$  to  $\mathbf{D}$  should be included in the LTS under consideration or not: if no  $a$ -labeled transitions from  $\mathbf{C}$  are present then it should be included, but its presence invalidates the only reason for its inclusion.

Another problematic example is:

$$\frac{}{\mathbf{C} \xrightarrow{a} \mathbf{q}(\mathbf{C})} \quad \frac{x \xrightarrow{a} x' \quad x' \not\rightarrow}{\mathbf{q}(x) \xrightarrow{b} \mathbf{D}} \quad (2)$$

Here it is intuitively clear that the only transition from  $\mathbf{C}$  should be  $\mathbf{C} \xrightarrow{a} \mathbf{q}(\mathbf{C})$ . However, it is less clear whether a transition  $\mathbf{q}(\mathbf{C}) \xrightarrow{b} \mathbf{D}$  should be included. If the rule on the right is instantiated with  $x = \mathbf{C}$  and  $x' = \mathbf{q}(\mathbf{C})$  then the first premise is satisfied, but the second premise is invalidated by the rule conclusion, much as in (1).

The questions of whether an SOS specification is meaningful, and if it is, what is the LTS it defines, can be answered in several different ways. A variety of possible answers is surveyed in [2], and more elaborately in [5]. For example, it may be required that a specification have a least model, or a unique supported model [6], or a stable model [7], or that it is complete [5] or stratifiable [4]. Specifications (1) and (2) satisfy none of these conditions.

For the answers surveyed in [5], it is not immediately apparent whether a given specification satisfies them. More syntactic, but considerably more restrictive conditions are provided by the so-called SOS rule formats [2].

These formats are usually defined with more ambitious purposes in mind, such as to guarantee that bisimilarity on the defined LTS is a congruence, but in particular they guarantee that an LTS is meaningfully defined. One such format is GSOS [6], where (i) no terms other than variables are allowed in rule premises, and (ii) no *lookahead*, where variables can be tested for transitions of depth more than one, is allowed. Another is coGSOS [8], called “safe ntree” in [9], where lookahead is allowed, but only one operation symbol is allowed on the right-hand side of a rule conclusion. For example, rule (1) is neither GSOS nor coGSOS, the first rule in (2) is GSOS but not coGSOS, and the second rule is coGSOS but not GSOS.

If a specification consists entirely of GSOS rules or entirely of coGSOS rules, most answers surveyed in [5] hold and the specification defines an LTS very unambiguously. However, as the example in (2) shows, this fails if one allows both GSOS and coGSOS rules in a single specification.

GSOS and coGSOS formats may be considered somewhat restrictive, and several generalizations have been proposed in the literature (see [2] for a survey), with the so-called ntyft/ntyxt format [4] as a rather expressive example. Since these formats typically generalize both GSOS and coGSOS, to avoid specifications such as (2) they are normally stated with additional semantic conditions such as completeness, that guarantee LTSs to be meaningfully defined.

This paper shows that those additional conditions are problematic. Technically, we prove that most conditions surveyed in [5] that guarantee SOS specifications to be meaningful, are undecidable. Specifically, given an SOS specification, it is undecidable whether it (i) has a supported model, (ii) has a least supported model, (iii) has a unique supported model, (iv) has a unique stable model, or (v) is complete. This remains true even if only GSOS and coGSOS rules are allowed in specifications. From this one may conclude that formats such as “complete ntyft/ntyxt” [10] are not *bona fide* syntactic formats, as there is no algorithmic way to tell whether a given specification fits such a format.

Specifications used in our undecidability proofs are actually quite simple, barely extending both GSOS and coGSOS. There seems to remain little space for properly syntactic rule formats that would generalize both these formats. One available option is to consider stratifiable specifications (see [2, 4]), as

the specifications that we use fail to be stratifiable.<sup>1</sup> Another option is to consider one of several existing extensions of the basic SOS framework, such as formats for weak process equivalences [11], ordered specifications [12], processes with name binding features [13], etc. One could perhaps use their additional features to weed out problematic examples and still be able to cover a wide range of useful ones. For example, one typically applies ordered SOS to avoid using negative premises in rules. In this paper we deal only with the basic SOS framework as presented in [2], and we leave other approaches for future consideration.

This paper is a follow-up to the workshop presentation [14]. There, we proved that it is undecidable whether an SOS specification defines a distributive law of a monad over a comonad, a property useful in the bialgebraic theory of operational semantics [9, 8]. Most proofs in this paper are variations of the one in [14], but we present them here in much more detail. We also avoid categorical terminology altogether and present all results in the classical parlance of SOS theory.

## 2. Queue machines

All our undecidability results will be by a reduction of the halting problem of queue machines. Such a machine is similar to a classical pushdown automaton, except that it maintains a queue (rather than a stack) of symbols. In the standard definition (see e.g. [15, Exercise 99]), in each transition step a machine, before moving to a new state:

- removes exactly one element from the start of the queue, and
- adds zero, one or more elements to the end of the queue.

For our purposes, it will be convenient to consider a different variant of queue machine which, in each step:

- removes zero, one or two elements from the queue, and
- adds exactly one element to the end of the queue.

The following definition formalizes this notion.

---

<sup>1</sup>We are grateful to Pedro D'Argenio for pointing this out.

**Definition 1.** A *queue machine* (QM)  $\mathcal{M} = (Q, \Gamma, a_1, q_1, \delta_0, \delta_1, \delta_2)$  consists of a finite set  $Q$  of states with a chosen initial state  $q_1 \in Q$ , a finite alphabet  $\Gamma$  with a chosen initial symbol  $a_1 \in \Gamma$ , and three partial transition functions:

$$\delta_0 : Q \rightarrow \Gamma \times Q \quad \delta_1 : Q \times \Gamma \rightarrow \Gamma \times Q \quad \delta_2 : Q \times \Gamma \times \Gamma \rightarrow \Gamma \times Q$$

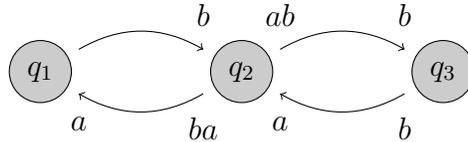
that are disjoint and jointly total, i.e., such that for each  $q \in A$  and  $a, b \in \Gamma$ , exactly one of  $\delta_0(q)$ ,  $\delta_1(q, a)$  and  $\delta_2(q, a, b)$  is defined. A *configuration* of  $\mathcal{M}$  is a pair  $(q, w) \in Q \times \Gamma^*$ . The machine induces a transition function on the set of configurations by:

$$\begin{aligned} (q, w) &\rightarrow_{\mathcal{M}} (q', wc) && \text{if } \delta_0(q) = (c, q') \\ (q, aw) &\rightarrow_{\mathcal{M}} (q', wc) && \text{if } \delta_1(q, a) = (c, q') \\ (q, abw) &\rightarrow_{\mathcal{M}} (q', wc) && \text{if } \delta_2(q, a, b) = (c, q'). \end{aligned}$$

A QM never makes a queue empty, and it *halts* if and only if, starting from the initial configuration  $(q_1, a_1)$ , it reaches a configuration  $(q, a)$  with a single letter  $a$  in the queue, such that  $\delta_0(q)$  and  $\delta_1(q, a)$  are undefined.

Note that if a machine  $\mathcal{M}$ , starting from the initial configuration  $(q_1, a_1)$ , reaches a configuration  $(q, w)$  in  $n$  steps, then necessarily  $|w| \leq n$ .

**Example 2.** A queue machine is naturally depicted as a directed graph whose vertices are states and edges transitions between them. Each edge can have two labels: at the beginning of an edge are symbols that the machine removes from the queue to make the corresponding transition. The label at the end of an edge is the symbol inserted to the queue in the transition. For example, consider a queue machine  $\mathcal{M}_\bullet$  given by the graph:



Its transitions are formally defined by the following transition functions:

$$\begin{aligned} \delta_0(q_1) &= (b, q_2) \\ \delta_2(q_2, a, b) &= (b, q_3) \\ \delta_2(q_2, b, a) &= (a, q_1) \\ \delta_1(q_3, b) &= (a, q_2) \end{aligned} \tag{3}$$

Formally, to make the transition functions jointly total, one needs to consider another “error” state  $q_{\perp}$ , omitted in the drawing above, and extend the above definition with e.g.:

$$\delta_2(q_2, a, a) = \delta_2(q_2, b, b) = \delta_1(q_3, a) = \delta_0(q_{\perp}) = (a, q_{\perp})$$

The machine  $\mathcal{M}_{\bullet} = (\{q_1, q_2, q_3, q_{\perp}\}, \{a, b\}, a, q_1, \delta_0, \delta_1, \delta_2)$  makes three steps from the initial configuration  $(q_1, a)$ :

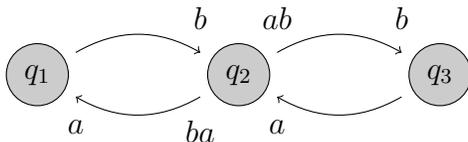
$$(q_1, a) \rightarrow_{\mathcal{M}_{\bullet}} (q_2, ab) \rightarrow_{\mathcal{M}_{\bullet}} (q_3, b) \rightarrow_{\mathcal{M}_{\bullet}} (q_2, a)$$

and then stops.

If the transition function in state  $q_3$  is modified, replacing (3) with:

$$\delta_0(q_3) = (a, q_2)$$

then the graph changes to:



This machine, which we call  $\mathcal{M}_{\circ}$ , after four steps returns to the initial configuration:

$$(q_1, a) \rightarrow_{\mathcal{M}_{\circ}} (q_2, ab) \rightarrow_{\mathcal{M}_{\circ}} (q_3, b) \rightarrow_{\mathcal{M}_{\circ}} (q_2, ba) \rightarrow_{\mathcal{M}_{\circ}} (q_1, a)$$

therefore it does not halt.

**Theorem 3.** It is undecidable whether a given queue machine halts from the initial configuration.

PROOF. See Appendix A.

### 3. Structural Operational Semantics

We begin by recalling some standard notions related to Structural Operational Semantics (SOS); for more information see [2].

**Definition 4.** A *labelled transition system* (LTS) consists of a set  $X$  of *states*, a set  $A$  of *labels*, and a *transition relation*  $\rightarrow \subseteq X \times A \times X$ . We write  $x \xrightarrow{a} y$  for  $\langle x, a, y \rangle \in \rightarrow$ .

In SOS, one considers LTSs where states are terms over some algebraic signature.

**Definition 5.** A *signature*  $\Sigma$  is a set of *operation symbols*, where each symbol  $\mathbf{f} \in \Sigma$  is equipped with an *arity*, a natural number. Given a set  $X$ , the set  $\Sigma^*X$  of  $\Sigma$ -*terms over*  $X$  is the least set such that

- every  $x \in X$  is a term, and
- if  $t_1, \dots, t_n$  are terms and  $\mathbf{f} \in \Sigma$  with an arity  $n$ , then  $\mathbf{f}(t_1, \dots, t_n)$  is a term.

A term of the form  $\mathbf{f}(x_1, \dots, x_n)$  where  $x_1, \dots, x_n \in X$  is called *flat*. A term is *closed* if it contains no variables; the set of all such terms is denoted  $\Sigma^*\emptyset$ .

From now on, fix a countably infinite set of variables  $\mathcal{V} \ni \mathbf{x}, \mathbf{y}, \mathbf{z}, \dots$ , a signature  $\Sigma$ , and a set of labels  $A$ .

**Definition 6.** An *SOS rule* is an expression of the form

$$\frac{P_1 \quad P_2 \quad \dots \quad P_m}{Q}$$

where:

- each  $P_i$ , called a *premise*, is either of the form  $t \xrightarrow{a} t'$  for some  $t, t' \in \Sigma^*\mathcal{V}$  and  $a \in A$  (a *positive* premise) or of the form  $t \not\xrightarrow{a}$  for some  $t \in \Sigma^*\mathcal{V}$  and  $a \in A$  (a *negative* premise), and
- $Q$ , the *conclusion*, is of the form  $t \xrightarrow{a} t'$  for some  $t, t' \in \Sigma^*\mathcal{V}$  and  $a \in A$ .

The left-hand side and the right-hand side of a premise (or the conclusion) are called the *source* and the *target* of that premise (resp. the conclusion).

We shall write  $t \not\xrightarrow{a}$  for the set of negative premises  $\{t \not\xrightarrow{a} \mid a \in A\}$ .

An *SOS specification* is a set of SOS rules. A specification is *finite* if  $\Sigma$  and  $A$  are finite, there are finitely many rules, and each rule has finitely many premises. Throughout this paper we consider only finite specifications.

A *closed instance* of a rule arises by replacing variables with closed terms according to some substitution  $\sigma : \mathcal{V} \rightarrow \Sigma^*\emptyset$ .

In this paper we consider only SOS rules of two rather specific shapes.

**Definition 7.** A *GSOS rule* [2, 6] is an SOS rule where:

- all terms in premises are variables from  $\mathcal{V}$ ,
- the source of the conclusion is a flat term with all variables pairwise different,
- variables from the source of the conclusion are not allowed as targets of premises,
- no variable appears as the target of more than one premise,
- only variables from the source of the conclusion are allowed as sources of premises,
- every variable that appears in the target of the conclusion must appear elsewhere in the rule.

An example of a GSOS rule is

$$\frac{x_1 \xrightarrow{a} y \quad x_1 \xrightarrow{b} z \quad x_1 \xrightarrow{e} \quad x_2 \xrightarrow{c} w}{f(x_1, x_2) \xrightarrow{d} f(g(z, x_2, x_2), y)} \quad (4)$$

**Definition 8.** A *coGSOS rule* [8] is an SOS rule where:

- all terms in premises are variables from  $\mathcal{V}$ ,
- the source of the conclusion is a flat term with all variables pairwise different,
- the target of the conclusion is a flat term or a variable,
- variables from the source of the conclusion are not allowed as targets of premises,
- no variable appears as the target of more than one premise,
- for each variable  $z$  present in a premise, either  $z$  appears in the source of the conclusion or there is a sequence of positive premises:

$$x_1 \xrightarrow{a_1} x_2 \quad x_2 \xrightarrow{a_2} x_3 \quad \cdots \quad x_n \xrightarrow{a_n} z$$

where  $x_1$  appears in the source of the conclusion,

- every variable that appears in the target of the conclusion must appear elsewhere in the rule.

An example of a coGSOS rule is

$$\frac{x \xrightarrow{a} y \quad y \xrightarrow{b} z \quad y \not\xrightarrow{c}}{f(x) \xrightarrow{d} g(y, x)} \quad (5)$$

Note that premises  $y \xrightarrow{b} z$  and  $y \not\xrightarrow{c}$ , whose sources do not appear in the source of the conclusion, prevent this from being a GSOS rule. On the other hand, rule (4) is not a coGSOS rule, as its conclusion is neither a flat term nor a variable.

Generally, the purpose of SOS specifications is to define LTSs in some way. Intuitively, an LTS defined from a set of rules should contain those transitions that can be “inferred” from those rules. However, the presence of negative premises complicates matters.

**Definition 9.** A *positive literal* is an expression  $t \xrightarrow{a} t'$ , where  $t, t' \in \Sigma^*\emptyset$  and  $a \in A$ . A *negative literal* is an expression  $t \not\xrightarrow{a}$ , where  $t \in \Sigma^*\emptyset$  and  $a \in A$ .

Note that a set of positive literals is simply an LTS with  $\Sigma^*\emptyset$  as the set of states. From now on, we focus on LTSs of this form.

**Definition 10.** A positive literal and a negative literal *contradict* each other if they have the same source and transition label. For an LTS  $L$  and a set of literals  $H$ , we write  $L \models H$  (read:  $L$  *entails*  $H$ ) if:

- every positive literal in  $H$  is also in  $L$ ,
- for every negative literal in  $H$ , there are no literals in  $L$  that contradict it.

**Definition 11.** Given an SOS specification  $\mathcal{S}$ , a *proof* of a literal  $\alpha$  from a set of literals  $H$  is a finite, upwardly branching tree with nodes labeled by literals, where:

- the root is labeled by  $\alpha$ , and
- for any node (labeled by some  $\beta$ ), if  $\gamma_1, \dots, \gamma_m$  are the labels of the nodes directly above it then either  $m = 0$  and  $\beta \in H$ , or  $\frac{\gamma_1 \dots \gamma_m}{\beta}$  is a closed instance of some rule from  $\mathcal{S}$ .

If such a proof exists, we say that  $\alpha$  is *provable* from  $H$ . A proof where  $H$  is empty is called a *closed proof*.

**Definition 12.** An LTS  $L$  is a *model* of an SOS specification  $\mathcal{S}$  if, for every closed instance  $\frac{\gamma_1 \dots \gamma_m}{\beta}$  of a rule in  $\mathcal{S}$ , if  $L \models \{\gamma_1, \dots, \gamma_m\}$  then  $L \models \{\beta\}$ . It is a *supported model* [6] if, additionally, every literal in  $L$  is provable from a set  $H$  of literals such that  $L \models H$ . If  $H$  is restricted to a set of negative literals only, then  $L$  is a *stable model* [7].

**Definition 13.** A disjoint pair of sets of positive literals  $\langle L, U \rangle$  is a *three-valued stable model* for an SOS specification  $\mathcal{R}$  if:

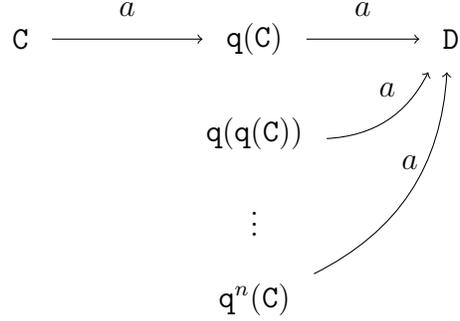
- a positive literal is in  $L$  if and only if it is provable from a set  $H$  of negative literals such that  $L \cup U \models H$ ,
- a positive literal is in  $L \cup U$  if and only if it is provable from a set  $H$  of negative literals such that  $L \models H$ .

The set  $U$  in a three-valued model is, intuitively, the set of unknown transitions whose presence in a model is undecided. Each specification has an (*information-*)*least* three-valued stable model [5], i.e., one where  $L$  is the smallest and  $U$  the largest among all three-valued stable models. The specification  $\mathcal{R}$  is *complete* if its least three-valued stable model does not contain unknown transitions. Equivalently,  $\mathcal{R}$  is complete iff it has a three-valued stable model  $\langle L, U \rangle$  with  $U = \emptyset$ , but no three-valued stable models with  $U \neq \emptyset$ .

**Example 14.** The specification:

$$\frac{}{C \xrightarrow{a} q(C)} \quad \frac{x \xrightarrow{a} x' \quad x' \not\xrightarrow{b}}{q(x) \xrightarrow{a} D} \quad (6)$$

has a supported model  $L$ :



The transition  $\mathbf{C} \xrightarrow{a} \mathbf{q}(\mathbf{C})$  has a closed proof. Every other transition in  $L$  is provable from the set of negative literals  $H = \{\mathbf{q}(\mathbf{C}) \not\xrightarrow{b}, \mathbf{D} \not\xrightarrow{b}\}$  which is entailed by  $L$ . As a result  $L$  is stable, and the pair  $\langle L, \emptyset \rangle$  is a three-valued stable model.

The specification (2) from the Introduction has the same rule for  $\mathbf{C}$  as (6), and the second rule is almost the same except for the label in the conclusion. The specification (2) has no supported or stable models. It has a model with two literals  $L = \{\mathbf{C} \xrightarrow{a} \mathbf{q}(\mathbf{C}), \mathbf{q}(\mathbf{C}) \xrightarrow{b} \mathbf{D}\}$ , but the model is not supported, as the second literal is not provable from literals entailed by  $L$ . The least three-valued stable model is  $\langle \{\mathbf{C} \xrightarrow{a} \mathbf{q}(\mathbf{C})\}, \{\mathbf{q}(\mathbf{C}) \xrightarrow{b} \mathbf{D}\} \rangle$ , and the specification (2) is not complete.

In the literature, several answers have been proposed as to when an SOS specification with negative premises is meaningful. Most of them are present in the theorem below:

**Theorem 15.** The following questions of a finite SOS specification  $\mathcal{S}$  are undecidable:

- whether  $\mathcal{S}$  has a supported model,
- whether  $\mathcal{S}$  has a least supported model,
- whether  $\mathcal{S}$  has a unique supported model,
- whether  $\mathcal{S}$  has a unique stable model,
- whether  $\mathcal{S}$  is complete.

These questions remain undecidable if every rule in  $\mathcal{S}$  is either a GSOS or a coGSOS rule.

The rest of this paper is devoted to proving Theorem 15.

#### 4. From queue machines to SOS specifications

Given a queue machine  $\mathcal{M} = (Q, \Gamma, a_1, q_1, \delta_0, \delta_1, \delta_2)$ , consider a signature  $\Sigma_{\mathcal{M}}$  with a single constant (i.e., an operation symbol with arity 0)  $\mathbf{C}$  and a family of operation symbols  $\{\mathbf{q} \mid q \in Q\}$  each with arity 1, and a set of rules:

$$\begin{array}{c}
\frac{}{\mathbf{C} \xrightarrow{a_1} \mathbf{q}_1(\mathbf{C})} \quad \frac{}{\mathbf{q}(x) \xrightarrow{c} \mathbf{q}'(x)} \quad (\mathbf{R0}) \quad \frac{x \xrightarrow{a} y}{\mathbf{q}(x) \xrightarrow{c} \mathbf{q}'(y)} \quad (\mathbf{R1}) \\
\frac{x \xrightarrow{a} y \quad y \xrightarrow{b} z}{\mathbf{q}(x) \xrightarrow{c} \mathbf{q}'(z)} \quad (\mathbf{R2}) \quad \frac{x \xrightarrow{a} y \quad y \not\xrightarrow{b}}{\mathbf{q}(x) \xrightarrow{a} \mathbf{q}(x)} \quad (\mathbf{R2n}) \quad (7)
\end{array}$$

for all  $q, q' \in Q$  and  $a, b, c \in \Gamma$  subject to the following conditions:

- **R0** is included whenever  $\delta_0(q) = (c, q')$ ,
- **R1** is included whenever  $\delta_1(q, a) = (c, q')$ , and
- **R2** and **R2n** are included whenever  $\delta_2(q, a, b) = (c, q')$ .

The specification obtained in this way from a machine  $\mathcal{M}$  will be denoted  $S_{\mathcal{M}}$ . Note that rules **R0** are GSOS rules, **R2** and **R2n** are coGSOS rules, and **R0** and **R1** are both GSOS and coGSOS.

**Example 16.** The machine  $\mathcal{M}_{\bullet}$  from Example 2 gives rise to the following specification over the signature  $\Sigma_{\mathcal{M}_{\bullet}} = \{\mathbf{C}, \mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3, \mathbf{q}_{\perp}\}$ :

$$\begin{array}{c}
\frac{}{\mathbf{C} \xrightarrow{a} \mathbf{q}_1(\mathbf{C})} \quad (1) \quad \frac{}{\mathbf{q}_1(x) \xrightarrow{b} \mathbf{q}_2(x)} \quad (2) \quad \frac{x \xrightarrow{b} y}{\mathbf{q}_3(x) \xrightarrow{a} \mathbf{q}_2(y)} \quad (3) \\
\frac{x \xrightarrow{a} y \quad y \xrightarrow{b} z}{\mathbf{q}_2(x) \xrightarrow{b} \mathbf{q}_3(z)} \quad (4) \quad \frac{x \xrightarrow{a} y \quad y \not\xrightarrow{b}}{\mathbf{q}_2(x) \xrightarrow{a} \mathbf{q}_2(x)} \quad (5) \\
\frac{x \xrightarrow{b} y \quad y \xrightarrow{a} z}{\mathbf{q}_2(x) \xrightarrow{a} \mathbf{q}_1(z)} \quad (6) \quad \frac{x \xrightarrow{b} y \quad y \not\xrightarrow{a}}{\mathbf{q}_2(x) \xrightarrow{b} \mathbf{q}_2(x)} \quad (7)
\end{array}$$

These are only the rules that correspond to the transitions presented on the first graph in Example 2. We omit the remaining rules, since they are negligible in further considerations.

The specification  $S_{\mathcal{M}_\circ}$  corresponding to the machine  $\mathcal{M}_\circ$  differs only on the rules for operation  $\mathfrak{q}_3$ . It has only one such rule, replacing rule **(3)** above:

$$\frac{}{\mathfrak{q}_3(x) \xrightarrow{a} \mathfrak{q}_2(x)} \quad (\mathbf{3}')$$

## 5. Machines that do not halt

Assuming that a machine  $\mathcal{M}$  does not halt from its initial configuration, we shall construct a stable model of  $S_{\mathcal{M}}$ . Suppose that  $\mathcal{M}$  executes an infinite computation starting from the initial configuration  $(q_1, w_1)$  where  $w_1 = a_1$  as follows:

$$(q_1, w_1) \rightarrow_{\mathcal{M}} (q_2, w_2) \rightarrow_{\mathcal{M}} (q_3, w_3) \rightarrow_{\mathcal{M}} \cdots \quad (8)$$

Define an infinite sequence of closed terms  $t_0, t_1, t_2, \dots \in \Sigma_{\mathcal{M}}^* \emptyset$  by:

$$\begin{aligned} t_0 &= \mathfrak{C}, \\ t_n &= \mathfrak{q}_n(t_k) \text{ where } k = n - |w_n|, \text{ for } n > 0. \end{aligned} \quad (9)$$

This is well defined, since  $|w_n| \leq n$ . Now define an LTS on the set  $\{t_n \mid n \in \mathbb{N}\}$  with exactly one transition for each term  $t_n$ :

$$t_n \xrightarrow{a_{n+1}} t_{n+1}$$

where  $a_{n+1}$  is the last symbol in  $w_{n+1}$ . We denote this LTS by  $L_{\mathfrak{C}}$ . Note that  $L_{\mathfrak{C}}$  is deterministic, i.e., every state in it has exactly one outgoing transition. In particular,  $L_{\mathfrak{C}}$  contains the transition  $\mathfrak{C} \xrightarrow{a_1} \mathfrak{q}_1(\mathfrak{C})$ .

**Example 17.** Recall the infinite computation of the machine  $\mathcal{M}_\circ$  from Example 2:

$$(q_1, a) \rightarrow_{\mathcal{M}_\circ} (q_2, ab) \rightarrow_{\mathcal{M}_\circ} (q_3, b) \rightarrow_{\mathcal{M}_\circ} (q_2, ba) \rightarrow_{\mathcal{M}_\circ} (q_1, a) \rightarrow_{\mathcal{M}_\circ} \cdots$$

After a sequence of four steps it returns to the initial configuration and it repeats the same transitions again. This corresponds to the infinite stream  $L_{\mathfrak{C}}$  of transitions between closed terms of the form:

$$t_n \xrightarrow{a_{n+1}} t_{n+1}$$

for  $n \geq 0$ , so that  $t_0 = \mathbf{C}$ ,  $a_1 = a$  and:

$$t_n = \begin{cases} \mathbf{q}_1(t_{n-1}) & \text{for every } n \text{ s.t. } n \bmod 4 = 1 \\ \mathbf{q}_2(t_{n-2}) & \text{for every } n \text{ s.t. } n \bmod 4 = 2 \text{ or } n \bmod 4 = 0 \\ \mathbf{q}_3(t_{n-1}) & \text{for every } n \text{ s.t. } n \bmod 4 = 3 \end{cases}$$

for  $n > 0$ . A transition label  $a_{n+1} = b$  if  $n \bmod 4 = 1$  or  $n \bmod 4 = 2$ , otherwise  $a_{n+1} = a$ .

The stream  $L_{\mathbf{C}}$  begins by:

$$\mathbf{C} \xrightarrow{a} \mathbf{q}_1(\mathbf{C}) \xrightarrow{b} \mathbf{q}_2(\mathbf{C}) \xrightarrow{b} \mathbf{q}_3(\mathbf{q}_2(\mathbf{C})) \xrightarrow{a} \mathbf{q}_2(\mathbf{q}_2(\mathbf{C})) \xrightarrow{a} \mathbf{q}_1(\mathbf{q}_2(\mathbf{q}_2(\mathbf{C}))) \xrightarrow{b} \dots$$

The following lemma will be useful later. Intuitively it states that the contents of the queue throughout the infinite computation of  $\mathcal{M}$  can be read from transition labels of  $L_{\mathbf{C}}$ .

**Lemma 18.** Consider the infinite computation in (8). Then, for each  $n > 0$ ,  $w_n = a_{k+1} \cdots a_{n-1} a_n$ , where  $k = n - |w_n|$ .

PROOF. By induction on  $n$ . The base case, for  $n = 1$ , is  $w_1 = a_1$ , true by definition. Since  $w_{n+1}$  arises from  $w_n$  by adding  $a_{n+1}$  at the end (and perhaps removing one or two symbols from the beginning), the inductive step follows easily.  $\square$

We wish to extend  $L_{\mathbf{C}}$  to an LTS on all  $\Sigma_{\mathcal{M}}^*$ -terms. To this end, we introduce an auxiliary function:

**Definition 19.** Define  $\Delta : \Sigma_{\mathcal{M}}^* \emptyset \rightarrow \mathbb{N}$  by:

- $\Delta(t) = 0$  if  $t \in L_{\mathbf{C}}$ , that is if  $t = t_n$  for some  $n \geq 0$ ,
- $\Delta(t) = \Delta(s) + 1$  if  $t \notin L_{\mathbf{C}}$  and  $t = \mathbf{q}(s)$  for some  $\mathbf{q} \in \Sigma_{\mathcal{M}}$ .

In words,  $\Delta(t)$  is the least number of operation symbols that need to be removed from the top of the term  $t$  to obtain a subterm from  $L_{\mathbf{C}}$ .

**Example 20.** Consider terms built over the signature  $\Sigma_{\mathcal{M}_{\circ}}$  induced by the machine  $\mathcal{M}_{\circ}$  from Example 2. For each term  $t_n$  for  $n \geq 0$  defined in Example 17  $\Delta(t_n) = 0$ , since  $t_n \in L_{\mathbf{C}}$ . Let us pick one of those terms, say  $t_{16} = \mathbf{q}_2(t_{14})$ . According to Definition 19:

$$\begin{aligned} \Delta(t_{16}) &= 0, & \Delta(\mathbf{q}_3(t_{16})) &= 1, & \Delta(\mathbf{q}_1(\mathbf{q}_3(t_{16}))) &= 2, \\ \Delta(\mathbf{q}_3(\mathbf{q}_2(t_{16}))) &= 0, & \Delta(\mathbf{q}_2(t_{16})) &= 0 \end{aligned}$$

Indeed,  $t = \mathbf{q}_3(t_{16}) \notin L_{\mathbf{c}}$  hence  $\Delta(t) = \Delta(t_{16}) + 1 = 1$ .  $L_{\mathbf{c}}$  does not contain any term with  $t$  as a subterm. In particular,  $t' = \mathbf{q}_{\perp}(\mathbf{q}_3(t_{16})) \notin L_{\mathbf{c}}$ , hence  $\Delta(t') = \Delta(t) + 1 = 2$ . On the other hand  $t_{18} = \mathbf{q}_2(t_{16})$  and  $t_{19} = \mathbf{q}_3(\mathbf{q}_2(t_{16}))$ , hence  $\Delta(t_{18}) = \Delta(t_{19}) = 0$ .

The function  $\Delta$  has the following basic properties:

**Lemma 21.** For any terms  $s, s' \in \Sigma_{\mathcal{M}}^* \emptyset$  and any operations  $\mathbf{q}, \mathbf{q}' \in \Sigma_{\mathcal{M}}$ :

- (i)  $\Delta(\mathbf{q}(s)) \leq \Delta(s) + 1$ ,
- (ii) if  $\Delta(\mathbf{q}(s)) > 0$  then  $\Delta(s) < \Delta(\mathbf{q}(s))$ ,
- (iii) if  $\Delta(\mathbf{q}(s)) > 0$  and  $\Delta(s') \leq \Delta(s)$  then  $\Delta(\mathbf{q}'(s')) \leq \Delta(\mathbf{q}(s))$ ,

PROOF. All properties easily follow from Definition 19: for  $t = \mathbf{q}(s)$ , either  $\Delta(t) = 0$  or  $\Delta(t) = \Delta(s) + 1$ .  $\square$

We shall now extend  $L_{\mathbf{c}}$  to an LTS, denoted  $L_{\mathcal{M}}$ , on all closed  $\Sigma_{\mathcal{M}}$ -terms. The new LTS will be deterministic, i.e., every term  $s$  will have exactly one outgoing transition. We define these transitions by induction on the value of  $\Delta(s)$ . Along the inductive definition, we will prove for all terms  $s$  that:

$$\text{whenever } s \xrightarrow{a} s' \text{ then } \Delta(s) \geq \Delta(s'). \quad (10)$$

Notice that this holds for all transitions in  $L_{\mathbf{c}}$  since by definition  $\Delta(t_n) = 0$  for each term  $t_n$  in  $L_{\mathbf{c}}$ . This is the base case of our inductive construction.

Consider an arbitrary term  $t = \mathbf{q}(s)$  with  $\Delta(t) > 0$  and assume that all transitions starting from terms with smaller values of  $\Delta$  have been defined. Hence, by Lemma 21(ii), there is a unique transition outgoing from  $s$ ; let it be  $s \xrightarrow{a} s'$ . Since (10) holds for  $s$  by the inductive assumption, there is also a unique transition outgoing from  $s'$ ; let it be  $s' \xrightarrow{b} s''$ .

Define a transition from  $t = \mathbf{q}(s)$  as follows:

- (a) if  $\delta_0(q) = (c, q')$ , then add  $\mathbf{q}(s) \xrightarrow{c} \mathbf{q}'(s)$  to  $L_{\mathcal{M}}$ ,
  - (b) if  $\delta_1(q, a) = (c, q')$ , then add  $\mathbf{q}(s) \xrightarrow{c} \mathbf{q}'(s')$  to  $L_{\mathcal{M}}$ ,
  - (c) if  $\delta_2(q, a, b) = (c, q')$ , then add  $\mathbf{q}(s) \xrightarrow{c} \mathbf{q}'(s'')$  to  $L_{\mathcal{M}}$ .
- (11)

Since by Definition 1 exactly one of these three cases holds, this adds a unique transition from  $\mathbf{q}(s)$ .

To complete the inductive step we need to prove the property (10) for the newly added transition. In case (a), since  $q(s) \notin L_{\mathbf{C}}$ , we have  $\Delta(q(s)) = \Delta(s) + 1 \geq \Delta(q'(s))$  (by Lemma 21(i)). In case (b), since  $\Delta(s) \geq \Delta(s')$  by the inductive assumption, we have  $\Delta(q(s)) \geq \Delta(q'(s'))$  by Lemma 21(iii). Similarly in case (c), since  $\Delta(s) \geq \Delta(s') \geq \Delta(s'')$  by the inductive assumption, we have  $\Delta(q(s)) \geq \Delta(q'(s''))$  by Lemma 21(iii).

**Example 22.** For the specification  $S_{\mathcal{M}_{\circ}}$  from Example 16, the LTS  $L_{\mathcal{M}_{\circ}}$  extends  $L_{\mathbf{C}}$  defined in Example 17 with the following additional transitions:

- (a)  $q_1(s) \xrightarrow{b} q_2(s)$  for any term  $s$ , since  $\delta_0(q_1) = (b, q_2)$ ,
- (b)  $q_3(s) \xrightarrow{a} q_2(s)$  for any term  $s$ , since  $\delta_0(q_3) = (a, q_2)$ ,
- (c)  $q_2(s) \xrightarrow{a} q_{\perp}(s'')$  for any  $s \xrightarrow{a} s' \xrightarrow{a} s'' \in L_{\mathcal{M}_{\circ}}$ , since  $\delta_2(q_2, a, a) = (a, q_{\perp})$ ,
- (d)  $q_2(s) \xrightarrow{a} q_{\perp}(s'')$  for any  $s \xrightarrow{b} s' \xrightarrow{b} s'' \in L_{\mathcal{M}_{\circ}}$ , since  $\delta_2(q_2, b, b) = (a, q_{\perp})$ ,
- (e)  $q_2(s) \xrightarrow{b} q_3(s'')$  for any  $s \xrightarrow{a} s' \xrightarrow{b} s'' \in L_{\mathcal{M}_{\circ}}$ , since  $\delta_2(q_2, a, b) = (b, q_3)$ ,
- (f)  $q_2(s) \xrightarrow{a} q_1(s'')$  for any  $s \xrightarrow{b} s' \xrightarrow{a} s'' \in L_{\mathcal{M}_{\circ}}$ , since  $\delta_2(q_2, b, a) = (a, q_1)$ ,
- (g)  $q_{\perp}(s) \xrightarrow{a} q_{\perp}(s)$  for any term  $s$ , since  $\delta_0(q_{\perp}) = (a, q_{\perp})$ .

This completes the construction of the deterministic LTS  $L_{\mathcal{M}}$ , defined under the assumption that  $\mathcal{M}$  does not halt. Continuing under this assumption, we shall now prove that  $L_{\mathcal{M}}$  is a stable model (see Definition 12).

**Lemma 23.** If  $\mathcal{M}$  does not halt then  $L_{\mathcal{M}}$  is a model of  $S_{\mathcal{M}}$ .

PROOF. Consider each rule in (7) in turn:

- The unique instance of the rule  $\frac{}{\mathbf{C} \xrightarrow{a_1} q_1(\mathbf{C})}$  infers a transition from  $\mathbf{C}$  that is included in  $L_{\mathbf{C}}$ , hence in  $L_{\mathcal{M}}$ .
- Consider any instance of **R0** in  $S_{\mathcal{M}}$ :

$$\frac{}{q(s) \xrightarrow{c} q'(s)}.$$

This means that  $\delta_0(q) = (c, q')$ . There are two cases to consider, depending on  $\Delta(q(s))$ .

If  $\Delta(\mathbf{q}(s)) > 0$ , then  $\mathbf{q}(s) \xrightarrow{c} \mathbf{q}'(s)$  is the transition added to  $L_{\mathcal{M}}$  in (11)-(a).

If  $\Delta(\mathbf{q}(s)) = 0$ , then  $\mathbf{q}(s) = t_n \in L_{\mathcal{C}}$  for some  $n > 0$ . Since  $\delta_0(q)$  is defined, in the  $n$ -th step the machine performs a transition  $(q, w_n) \rightarrow_{\mathcal{M}} (q', w_n c)$ . Then, by definition of  $t_n$ , we have that  $s = t_k$  for  $k = n - |w_n|$ . The machine adds  $c$  to the queue, hence  $|w_{n+1}| = |w_n| + 1$  and  $n + 1 - |w_{n+1}| = n - |w_n| = k$ . As a result,  $t_{n+1} = \mathbf{q}'(s)$  and  $\mathbf{q}(s) \xrightarrow{c} \mathbf{q}'(s) \in L_{\mathcal{C}} \subseteq L_{\mathcal{M}}$ .

- Consider any instance of **R1** in  $S_{\mathcal{M}}$ :

$$\frac{s \xrightarrow{a} s'}{\mathbf{q}(s) \xrightarrow{c} \mathbf{q}'(s')}$$

and assume that  $s \xrightarrow{a} s' \in L_{\mathcal{M}}$ . This means that  $\delta_1(q, a) = (c, q')$ . Again, there are two cases to consider.

If  $\Delta(\mathbf{q}(s)) > 0$ , then  $\mathbf{q}(s) \xrightarrow{c} \mathbf{q}'(s')$  is the transition added to  $L_{\mathcal{M}}$  in (11)-(b).

If  $\Delta(\mathbf{q}(s)) = 0$  then  $\mathbf{q}(s) = t_n \in L_{\mathcal{C}}$  for some  $n > 0$ . Since  $\delta_1(q, a)$  is defined, in the  $n$ -th step the machine takes the first symbol  $a$  from the queue that contains a word  $w_n$  and adds the symbol  $c$  to it. Then, by definition of  $t_n$ , we have that  $s = t_k$  and  $s' = t_{k+1}$  for  $k = n - |w_n|$ . The machine removes  $a$  and adds  $c$  to the queue, hence  $|w_{n+1}| = |w_n|$  and  $n + 1 - |w_{n+1}| = n + 1 - |w_n| = k + 1$ . As a result,  $t_{n+1} = \mathbf{q}'(s')$  and  $\mathbf{q}(s) \xrightarrow{c} \mathbf{q}'(s') \in L_{\mathcal{C}} \subseteq L_{\mathcal{M}}$ .

- Consider any instance of **R2** in  $S_{\mathcal{M}}$ :

$$\frac{s \xrightarrow{a} s' \quad s' \xrightarrow{b} s''}{\mathbf{q}(s) \xrightarrow{c} \mathbf{q}'(s'')}$$

and assume that  $s \xrightarrow{a} s', s' \xrightarrow{b} s'' \in L_{\mathcal{M}}$ . This means that  $\delta_2(q, a, b) = (c, q')$ . Again, there are two cases to consider.

If  $\Delta(\mathbf{q}(s)) > 0$ , then  $\mathbf{q}(s) \xrightarrow{c} \mathbf{q}'(s'')$  is the transition added to  $L_{\mathcal{M}}$  in (11)-(c).

If  $\Delta(\mathbf{q}(s)) = 0$  then  $\mathbf{q}(s) = t_n \in L_{\mathcal{C}}$  for some  $n > 0$ . Since  $\delta_2(q, a, b)$  is defined, in the  $n$ -th step the machine takes two symbols  $a$  and  $b$  from

the queue that contains a word  $w_n$  and adds the symbol  $c$  to it. Then, by definition of  $t_n$ , we have that  $s = t_k$ ,  $s' = t_{k+1}$  and  $s'' = t_{k+2}$  for  $k = n - |w_n|$ . Since  $\mathcal{M}$  does not halt, there are at least two symbols in  $w_n$ , hence  $k + 1 < n$ . The machine removes  $ab$  and adds  $c$  to the queue, hence  $|w_{n+1}| = |w_n| - 1$  and  $n + 1 - |w_{n+1}| = n + 1 - |w_n| + 1 = k + 2$ . As a result  $t_{n+1} = \mathbf{q}'(s'')$ , and  $\mathbf{q}(s) \xrightarrow{c} \mathbf{q}'(s'') \in L_c \subseteq L_{\mathcal{M}}$ .

- Consider any instance of **R2n** in  $S_{\mathcal{M}}$ :

$$\frac{s \xrightarrow{a} s' \quad s' \not\rightarrow}{\mathbf{q}(s) \xrightarrow{a} \mathbf{q}(s)}.$$

This means that  $\delta_2(q, a, b) = (a, q')$ . Since every term in  $L_{\mathcal{M}}$  has an outgoing transition,  $L_{\mathcal{M}}$  does not entail the premise  $s' \not\rightarrow$  and the model condition holds.  $\square$

**Lemma 24.** If  $\mathcal{M}$  does not halt then the model  $L_{\mathcal{M}}$  of  $S_{\mathcal{M}}$  is stable; moreover, every transition in  $L_{\mathcal{M}}$  has a closed proof in  $S_{\mathcal{M}}$ .

PROOF. We will proceed by induction that follows the construction of  $L_{\mathcal{M}}$ . As the base case, consider transitions in  $L_c$ , i.e. those outgoing from terms  $t_n$  for  $n \geq 0$ , as defined in (9). For this case, we proceed by induction on  $n$ :

- for  $t_0 = \mathbf{C}$  the model  $L_{\mathcal{M}}$  contains a single transition  $\mathbf{C} \xrightarrow{a_1} \mathbf{q}_1(\mathbf{C})$ . The transition has a one-step closed proof:

$$\overline{\mathbf{C} \xrightarrow{a_1} \mathbf{q}_1(\mathbf{C})}.$$

- Assume that all transitions  $t_i \xrightarrow{a_{i+1}} t_{i+1}$  for  $0 \leq i < n$  have closed proofs. Consider the transition  $t_n \xrightarrow{a_{n+1}} t_{n+1}$  where  $t_n = \mathbf{q}_n(t_m)$  for some  $m < n$ . This transition, by definition of  $L_c$ , corresponds to the  $n$ -th step of the machine  $\mathcal{M}$ :  $(q_n, w_n) \rightarrow (q_{n+1}, w_{n+1})$ , for some word  $w_n$  such that  $|w_n| = n - m$ . By definition,  $t_{n+1} = \mathbf{q}_{n+1}(t_k)$ , for  $k = n + 1 - |w_{n+1}|$  and  $a_{n+1}$  is the last letter of  $w_{n+1}$ . Then:

- if the machine in the  $n$ -th step just inserts  $a_{n+1}$  to the queue, then  $|w_{n+1}| = |w_n| + 1$ , hence  $k = m$ . Such a transition is defined by  $\delta_0(q_n) = (a_{n+1}, q_{n+1})$ . In this case  $S_{\mathcal{M}}$  has the following instance of **R0**:

$$\overline{\mathbf{q}_n(t_m) \xrightarrow{a_{n+1}} \mathbf{q}_{n+1}(t_m)}$$

that defines the transition from term  $t_n$ . The target of the conclusion is exactly  $t_{n+1} = \mathbf{q}_{n+1}(t_k)$  for  $k = m$ . The above rule has no premises, hence it constitutes a closed proof of the transition  $t_n \xrightarrow{a_{n+1}} t_{n+1}$ .

- If the machine in the  $n$ -th step takes the first symbol from the queue, which by Lemma 18 is  $a_{m+1}$ , and inserts a new one  $a_{n+1}$ , then the length of the queue does not change, i.e.,  $|w_{n+1}| = |w_n|$ , hence  $k = n + 1 - |w_{n+1}| = m + 1$ . Such a transition is defined by  $\delta_1(q_n, a_{m+1}) = (a_{n+1}, q_{n+1})$ . In this case  $S_{\mathcal{M}}$  has the following instance of **R1**:

$$\frac{t_m \xrightarrow{a_{m+1}} t_{m+1}}{\mathbf{q}_n(t_m) \xrightarrow{a_{n+1}} \mathbf{q}_{n+1}(t_{m+1})}$$

The target of the conclusion is exactly  $t_{n+1} = \mathbf{q}_{n+1}(t_k)$  for  $k = m + 1$ . The rule has a single premise  $t_m \xrightarrow{a_{m+1}} t_{m+1}$ . Since  $m < n$  we know by the inductive assumption that this transition has a closed proof. The concatenation of that proof with the above instance forms a closed proof of  $t_n \xrightarrow{a_{n+1}} t_{n+1}$ .

- If the machine in the  $n$ -th step takes two initial elements from the queue, which by Lemma 18 are  $a_{m+1}a_{m+2}$ , and inserts a new one  $a_{n+1}$ , then the length of the queue decreases by 1, hence  $|w_{n+1}| = |w_n| - 1$ . This implies that  $k = n + 1 - (|w_n| - 1) = m + 2$ . Such a transition is defined by  $\delta_2(q_n, a_{m+1}, a_{m+2}) = (a_{n+1}, q_{n+1})$ . In this case  $S_{\mathcal{M}}$  has the following instance of **R2**:

$$\frac{t_m \xrightarrow{a_{m+1}} t_{m+1} \quad t_{m+1} \xrightarrow{a_{m+2}} t_{m+2}}{\mathbf{q}_n(t_m) \xrightarrow{a_{n+1}} \mathbf{q}_{n+1}(t_{m+2})}$$

The target of the conclusion is exactly  $t_{n+1} = \mathbf{q}_{n+1}(t_k)$  for  $k = m+2$ . Since  $\mathcal{M}$  does not halt, there are at least two elements in  $w_n$ , hence  $m + 1 < n$  and, by the inductive assumption, both premises of the rule have closed proofs. The composition of those proofs with the above instance constitutes a closed proof of  $t_n \xrightarrow{a_{n+1}} t_{n+1}$ .

This completes the argument to show that all transitions in  $L_{\mathcal{C}}$  have closed proofs. We shall now show the same for all remaining transitions in  $L_{\mathcal{M}}$ , by induction on  $\Delta$ .

Let us consider a term  $t = \mathbf{q}(s)$  such that  $\Delta(t) > 0$  and assume that each transition  $r \xrightarrow{a} r'$  where  $\Delta(r) < \Delta(t)$  has a closed proof. By construction,  $L_{\mathcal{M}}$  contains a unique transition from  $s$ , say  $s \xrightarrow{a} s'$ . Similarly, there is a unique transition from  $s'$ , say  $s' \xrightarrow{b} s''$ . Also by construction of  $L_{\mathcal{M}}$ , we have  $\Delta(t) > \Delta(s) \geq \Delta(s')$  therefore, by the inductive assumption, transitions  $s \xrightarrow{a} s'$  and  $s' \xrightarrow{b} s''$  have closed proofs.

The unique transition from  $\mathbf{q}(s)$  in  $L_{\mathcal{M}}$  can arise in one of three mutually exclusive ways, according to (11):

- if  $\delta_0(q) = (c, q')$ , then the transition is  $\mathbf{q}(s) \xrightarrow{c} \mathbf{q}'(s)$ . According to the definition of  $S_{\mathcal{M}}$ , since  $\delta_0(q)$  is defined there is the following instance of **R0**:

$$\frac{}{\mathbf{q}(s) \xrightarrow{c} \mathbf{q}'(s)}$$

The instance is a closed proof.

- If  $\delta_1(q, a) = (c, q')$ , then the transition is  $\mathbf{q}(s) \xrightarrow{c} \mathbf{q}'(s')$ . According to the definition of  $S_{\mathcal{M}}$ , since  $\delta_1(q, a)$  is defined there is the following instance of **R1**:

$$\frac{s \xrightarrow{a} s'}{\mathbf{q}(s) \xrightarrow{c} \mathbf{q}'(s')}$$

Since by assumption  $s \xrightarrow{a} s'$  has a closed proof, so does  $\mathbf{q}(s) \xrightarrow{c} \mathbf{q}'(s')$ .

- If  $\delta_2(q, a, b) = (c, q')$ , then there is a transition  $\mathbf{q}(s) \xrightarrow{c} \mathbf{q}'(s'')$ . According to the definition of  $S_{\mathcal{M}}$ , since  $\delta_2(q, a, b)$  is defined there is the following instance of **R2**:

$$\frac{s \xrightarrow{a} s' \quad s' \xrightarrow{b} s''}{\mathbf{q}(s) \xrightarrow{c} \mathbf{q}'(s'')}$$

By assumption both  $s \xrightarrow{a} s'$  and  $s' \xrightarrow{b} s''$  have closed proofs, hence so does  $\mathbf{q}(s) \xrightarrow{c} \mathbf{q}'(s'')$ .  $\square$

**Lemma 25.** If  $\mathcal{M}$  does not halt then  $S_{\mathcal{M}}$  is complete.

**PROOF.** Assume that a machine  $\mathcal{M}$  does not halt. By Lemma 24,  $L_{\mathcal{M}}$  is a stable model, so according to Definition 13,  $\langle L_{\mathcal{M}}, \emptyset \rangle$  is a three-valued stable model.

Now consider any three-valued stable model  $\langle L, U \rangle$  of  $S_{\mathcal{M}}$ . By Definition 13,  $L$  contains all transitions that are provable in  $S_{\mathcal{M}}$  from negative premises entailed by  $L \cup U$ . By Lemma 24, all transitions in  $L_{\mathcal{M}}$  have closed proofs, which do not use any premises, hence  $L_{\mathcal{M}} \subseteq L$ .

$L_{\mathcal{M}}$ , by construction, has an outgoing transition from every term, therefore so do  $L$  and  $L \cup U$ . As a result, in every closed instance of the rule **R2n**, the negative premise cannot be entailed by  $L$  or  $L \cup U$ . Other rules in  $S_{\mathcal{M}}$  have no negative premises. Therefore, being provable in  $S_{\mathcal{M}}$  from negative premises  $H$  entailed by  $L$  (or by  $L \cup U$ ) is equivalent to being provable with a closed proof. As a result, by Definition 13,  $L = L \cup U$  and  $U = \emptyset$ .  $\square$

## 6. Machines that (may) halt

We defined LTSs  $L_{\mathcal{C}}$  and  $L_{\mathcal{M}}$  under the assumption that the machine  $\mathcal{M}$  does not halt. However, as the following lemma shows, parts of  $L_{\mathcal{C}}$  appear in models of  $S_{\mathcal{M}}$  for *any* machine  $\mathcal{M}$ .

**Lemma 26.** For any  $n \in \mathbb{N}$ , if a machine  $\mathcal{M}$  performs at least  $n$  steps from the initial configuration:

$$(q_1, w_1) \multimap_{\mathcal{M}} (q_2, w_2) \multimap_{\mathcal{M}} \cdots \multimap_{\mathcal{M}} (q_n, w_n) \multimap_{\mathcal{M}} (q_{n+1}, w_{n+1}),$$

then every supported model of  $S_{\mathcal{M}}$  coincides with  $L_{\mathcal{C}}$  on transitions outgoing from terms  $t_0, t_1, \dots, t_n$ :

$$t_0 \xrightarrow{a_1} t_1 \xrightarrow{a_2} t_2 \longrightarrow \cdots \longrightarrow t_n \xrightarrow{a_{n+1}} t_{n+1}$$

as defined in (9).

**PROOF.** Let  $L$  be any supported model of  $S_{\mathcal{M}}$ . We proceed by induction on  $n$ . For  $n = 0$ ,  $t_0 = \mathbf{C}$ , the only possible proof for an outgoing transition is  $\frac{}{\mathbf{C} \xrightarrow{a_1} \mathbf{q}_1(\mathbf{C})}$ , and the resulting transition is in  $L_{\mathcal{C}}$ .

For the inductive step, pick any  $n > 0$  and assume that the statement holds for all numbers smaller than  $n$ . The  $n$ -th step of the machine is the transition  $(q_n, w_n) \multimap_{\mathcal{M}} (q_{n+1}, w_{n+1})$ , where  $a_{n+1}$  is the last symbol in  $w_{n+1}$ . By the inductive assumption  $t_n = \mathbf{q}_n(t_k)$  for  $k < n$  such that  $|w_n| = n - k$ , and  $a_n$  is the last element of  $w_n$ . Hence  $w_n = a_{k+1}, a_{k+2} \cdots, a_n$ . The transition  $t_n \xrightarrow{a_{n+1}} t_{n+1}$  can arise in one of three possible ways:

- if  $\delta_0(q_n) = (a_{n+1}, q_{n+1})$ , then  $w_{n+1} = a_{k+1} \dots a_n a_{n+1}$ , hence  $|w_{n+1}| = |w_n| + 1$ . In this case the only instance of a rule in  $S_{\mathcal{M}}$  that has  $t_n$  in the source is the following instance of **R0**:

$$\overline{\mathbf{q}_n(t_k) \xrightarrow{a_{n+1}} \mathbf{q}_{n+1}(t_k)}$$

Since  $L$  is a model,  $t_n \xrightarrow{a_{n+1}} t_{n+1}$  is a transition in  $L$ , where  $t_{n+1} = \mathbf{q}_{n+1}(t_k)$  as prescribed by (9). Since  $L$  is supported, this is the only transition in  $L$  originating from  $t_n$ .

- if  $\delta_1(q_n, a_{k+1}) = (a_{n+1}, q_{n+1})$ , then  $w_{n+1} = a_{k+2} \dots a_n a_{n+1}$ , hence  $|w_{n+1}| = |w_n|$ . By the inductive assumption, since  $k < n$ , the only transition in  $L$  outgoing from  $t_k$  is  $t_k \xrightarrow{a_{k+1}} t_{k+1}$ . As a result, the only instance of a rule in  $S_{\mathcal{M}}$  that has  $t_n$  in the source and a premise entailed by  $L$ , is the following instance of **R1**:

$$\frac{t_k \xrightarrow{a_{k+1}} t_{k+1}}{\mathbf{q}_n(t_k) \xrightarrow{a_{n+1}} \mathbf{q}_{n+1}(t_{k+1})}.$$

Since  $L$  is a model,  $t_n \xrightarrow{a_{n+1}} t_{n+1}$  is a transition in  $L$ , where  $t_{n+1} = \mathbf{q}_{n+1}(t_{k+1})$  as prescribed by (9). Since  $L$  is supported, this is the only transition in  $L$  originating from  $t_n$ .

- if  $|w_n| = n - k \geq 2$  and  $\delta_2(q_n, a_{k+1}, a_{k+2}) = (a_{n+1}, q_{n+1})$ , then  $w_{n+1} = a_{k+3} \dots a_n a_{n+1}$ , hence  $|w_{n+1}| = |w_n| - 1$ . By the inductive assumption, since  $k+1 < n$ , the only transition in  $L$  outgoing from  $t_k$  is  $t_k \xrightarrow{a_{k+1}} t_{k+1}$  and the only transition in  $L$  outgoing from  $t_{k+1}$  is  $t_{k+1} \xrightarrow{a_{k+2}} t_{k+2}$ . As a result, the only instance of a rule in  $S_{\mathcal{M}}$  that has  $t_n$  in the source and premises entailed by  $L$ , is the following instance of **R2**:

$$\frac{t_k \xrightarrow{a_{k+1}} t_{k+1} \quad t_{k+1} \xrightarrow{a_{k+2}} t_{k+2}}{\mathbf{q}_n(t_k) \xrightarrow{a_{n+1}} \mathbf{q}_{n+1}(t_{k+2})}.$$

(In particular, the following instance of **R2n**:

$$\frac{t_k \xrightarrow{a_{k+1}} t_{k+1} \quad t_{k+1} \not\rightarrow}{\mathbf{q}_n(t_k) \xrightarrow{a_{k+1}} \mathbf{q}_n(t_k)}.$$

has  $t_n$  in the source as required, but the second premise  $t_{k+1} \not\rightarrow$  is not entailed by  $L$ .)

Since  $L$  is a model,  $t_n \xrightarrow{a_{n+1}} t_{n+1}$  is a transition in  $L$ , where  $t_{n+1} = \mathbf{q}_{n+1}(t_{k+2})$  as prescribed by (9). Since  $L$  is supported, this is the only transition in  $L$  originating from  $t_n$ .  $\square$

**Lemma 27.** If  $\mathcal{M}$  does not halt then  $L_{\mathcal{M}}$  is the unique supported model of  $S_{\mathcal{M}}$ .

PROOF.  $L_{\mathcal{M}}$ , being stable by Lemma 24, is supported. Since every transition in  $L_{\mathcal{M}}$  has a closed proof, every model of  $S_{\mathcal{M}}$  contains  $L_{\mathcal{M}}$ . Moreover, by Lemma 26, since  $\mathcal{M}$  performs infinitely many steps, every supported model of  $S_{\mathcal{M}}$  coincides with  $L_{\mathcal{M}}$  on transitions outgoing from terms  $t$  with  $\Delta(t) = 0$ .

Assume, towards a contradiction, that there exists a supported model  $L$  of  $S_{\mathcal{M}}$  such that  $L_{\mathcal{M}} \subsetneq L$ . Pick a transition  $t \xrightarrow{c} t'$  from  $L - L_{\mathcal{M}}$  such that  $\Delta(t)$  is minimal; necessarily  $\Delta(t) > 0$ .

Since  $L$  is supported,  $t \xrightarrow{c} t'$  must have a proof in  $S_{\mathcal{M}}$  from some premises entailed by  $L$ . If  $t = \mathbf{q}(s)$  for some term  $s$  and  $q \in Q$ , the last rule instance in the proof of  $t \xrightarrow{c} t'$  is of one of the three forms:

- If  $t' = \mathbf{q}'(s)$  and the last instance is

$$\frac{}{\mathbf{q}(s) \xrightarrow{c} \mathbf{q}'(s)}$$

then  $\delta_0(q) = (c, q')$  and the resulting transition is in  $L_{\mathcal{M}}$ .

- If  $t' = \mathbf{q}'(s')$  and the last instance is

$$\frac{s \xrightarrow{a} s'}{\mathbf{q}(s) \xrightarrow{c} \mathbf{q}'(s')}$$

then  $\delta_1(q, a) = (c, q')$  and  $s \xrightarrow{a} s'$  is in  $L$ . Since  $\Delta(t) > 0$ , by Lemma 21(ii)  $\Delta(t) > \Delta(s)$ . Then, by the minimality assumption on  $t$ , the transition  $s \xrightarrow{a} s'$  is in  $L_{\mathcal{M}}$ , and since  $L_{\mathcal{M}}$  is a model of  $S_{\mathcal{M}}$  by Lemma 23, the transition  $\mathbf{q}(s) \xrightarrow{c} \mathbf{q}'(s')$  is also in  $L_{\mathcal{M}}$ .

- If  $t' = \mathbf{q}'(s'')$  and the last instance is

$$\frac{s \xrightarrow{a} s' \quad s' \xrightarrow{b} s''}{\mathbf{q}(s) \xrightarrow{c} \mathbf{q}'(s'')}$$

then  $\delta_2(q, a, b) = (c, q')$  and  $s \xrightarrow{a} s'$  and  $s' \xrightarrow{b} s''$  are in  $L$ .

Since  $\Delta(t) > 0$ , by Lemma 21(ii)  $\Delta(t) > \Delta(s)$  and, by the minimality assumption on  $t$ ,  $s \xrightarrow{a} s'$  is in  $L_{\mathcal{M}}$ . Then, by (10),  $\Delta(s) \geq \Delta(s')$  and, again by the minimality assumption on  $t$ ,  $s' \xrightarrow{b} s''$  is in  $L_{\mathcal{M}}$ . As a result, both premises of the above instance are in  $L_{\mathcal{M}}$  and, since  $L_{\mathcal{M}}$  is a model of  $S_{\mathcal{M}}$ , its conclusion  $q(s) \xrightarrow{c} q'(s'')$  is also in  $L_{\mathcal{M}}$ .

- If  $t = q(s)$  and the last instance is

$$\frac{s \xrightarrow{c} s' \quad s' \not\xrightarrow{c}}{q(s) \xrightarrow{c} q(s)}$$

then, since  $L_{\mathcal{M}} \subseteq L$  and  $L_{\mathcal{M}}$  by construction contains an outgoing transition from every term  $s'$ ,  $L$  does not entail the second premise of this instance.

In each case a contradiction is obtained, which means that the assumed transition  $t \xrightarrow{c} t'$  cannot exist and  $L = L_{\mathcal{M}}$ .  $\square$

**Lemma 28.** If  $\mathcal{M}$  halts then  $S_{\mathcal{M}}$  has no supported models.

PROOF. Assume that, for some  $n > 0$ ,  $\mathcal{M}$  terminates after  $n - 1$  steps:

$$(q_1, w_1) \rightarrow_{\mathcal{M}} (q_2, w_2) \rightarrow_{\mathcal{M}} \cdots \rightarrow_{\mathcal{M}} (q_n, w_n),$$

and that there is a supported model  $L$  of  $S_{\mathcal{M}}$ . By Lemma 26,  $L$  contains transitions:

$$t_0 \xrightarrow{a_1} t_1 \xrightarrow{a_2} t_2 \rightarrow \cdots \rightarrow t_{n-1} \xrightarrow{a_n} t_n$$

as defined by (9), and there are no other transitions outgoing from  $t_0, \dots, t_{n-1}$ .

Since  $(q_n, w_n)$  is a configuration where  $\mathcal{M}$  halts, it must be that:

- $|w_n| = 1$  (and  $w_n = a_n$ ), hence  $t_n = q_n(t_{n-1})$ , and
- $\delta_0(q_n)$  and  $\delta_1(q_n, a_n)$  are undefined, therefore  $\delta_2(q_n, a_n, b)$  is defined for all  $b \in \Gamma$ .

Consider two cases:

- $L$  has no outgoing transitions from  $t_n$ . Consider the following instance of **R2n**:

$$\frac{t_{n-1} \xrightarrow{a_n} t_n \quad t_n \not\xrightarrow{a_n}}{t_n \xrightarrow{a_n} t_n}$$

Both premises are entailed by  $L$  so, since  $L$  is a model,  $t_n \xrightarrow{a_n} t_n$  is a transition in  $L$ ; this is a contradiction, as it is a transition outgoing from  $t_n$ .

- $L$  has some outgoing transitions from  $t_n$ . Let  $t_n \xrightarrow{c} s$  be such a transition, chosen so that  $s$  is minimal in terms of size (i.e., number of operation symbols in it). Since  $L$  is supported, this transition must have a proof from premises entailed by  $L$ . This is only possible if:
  - $s = \mathbf{q}'(r)$  for some  $q' \in Q$  and some term  $r$ ,
  - $\delta_2(q_n, a_n, b) = (c, q')$ , for some  $b \in \Gamma$ ,
  - the last rule instance used in the proof is:

$$\frac{t_{n-1} \xrightarrow{a_n} t_n \quad t_n \xrightarrow{b} r}{t_n \xrightarrow{c} s}$$

This means that  $t_n \xrightarrow{b} r$  is a transition in  $L$ , but  $r$  is a smaller term than  $s = \mathbf{q}'(r)$ , which contradicts the minimality of  $s$ .

As a result, no supported model  $L$  exists. □

**Example 29.** The machine  $\mathcal{M}_\bullet$  from Example 2 terminates after three steps:

$$(q_1, a) \rightarrow_{\mathcal{M}_\bullet} (q_2, ab) \rightarrow_{\mathcal{M}_\bullet} (q_3, b) \rightarrow_{\mathcal{M}_\bullet} (q_2, a)$$

By Lemma 26, a supported model of  $S_{\mathcal{M}_\bullet}$  must contain the four-element stream of transitions:

$$\mathbf{C} \xrightarrow{a} \mathbf{q}_1(C) \xrightarrow{b} \mathbf{q}_2(C) \xrightarrow{b} \mathbf{q}_3(\mathbf{q}_2(C)) \xrightarrow{a} \mathbf{q}_2(\mathbf{q}_3(\mathbf{q}_2(C)))$$

Indeed, all these transitions have closed proofs, and no other transitions outgoing from the terms  $t_0 = \mathbf{C}$ ,  $t_1 = \mathbf{q}_1(C)$ ,  $t_2 = \mathbf{q}_2(C)$  and  $t_3 = \mathbf{q}_3(\mathbf{q}_2(C))$  are provable.

What transitions can originate in  $t_4 = \mathbf{q}_2(\mathbf{q}_3(\mathbf{q}_2(C)))$ ? There must be at least one such a transition, otherwise the following instance of **R2n**:

$$\frac{t_3 \xrightarrow{a} t_4 \quad t_4 \not\rightarrow}{t_4 \xrightarrow{a} t_4}$$

would be triggered and infer one.

Each transition from  $t_4$  has to be inferred by an instance of **R2** rule. We have that  $\delta_2(q_2, a, b) = (b, q_3)$  and  $\delta_2(q_2, a, a) = (a, q_\perp)$ , therefore the following instances of **R2** exist and only these, for some term  $r$ , can infer transitions outgoing from  $t_4$ :

$$\frac{t_3 \xrightarrow{a} t_4 \quad t_4 \xrightarrow{a} r}{t_4 \xrightarrow{a} \mathbf{q}_\perp(r)} \quad \frac{t_3 \xrightarrow{a} t_4 \quad t_4 \xrightarrow{b} r}{t_4 \xrightarrow{b} \mathbf{q}_3(r)}$$

However, both these instances prove transitions to terms more complex than  $r$ , from premises that are also transitions from  $t_4$  to  $r$ . As a result, the transition from  $t_4$  to a minimal  $r$  cannot have a proof, which contradicts the assumption that the model is supported.

## 7. Conclusion

The results of the previous sections are brought together in the following:

**Corollary 30.** For any queue machine  $\mathcal{M}$ , the following conditions are equivalent:

- (a)  $\mathcal{M}$  does not halt from the initial configuration  $(q_1, a_1)$ ,
- (b)  $S_{\mathcal{M}}$  has a supported model,
- (c)  $S_{\mathcal{M}}$  has a least supported model,
- (d)  $S_{\mathcal{M}}$  has a unique supported model,
- (e)  $S_{\mathcal{M}}$  has a unique stable model,
- (f)  $S_{\mathcal{M}}$  is complete.

PROOF. • (a)  $\implies$  (d) is Lemma 27. (d)  $\implies$  (c)  $\implies$  (b) are trivial.

- (a)  $\implies$  (e) follows from Lemmas 23, 24 and 27. (e)  $\implies$  (b) is trivial.
- (a)  $\implies$  (f) is Lemma 25. (f)  $\implies$  (b) is easy: if  $\langle L, \emptyset \rangle$  is three-valued stable then  $L$  is stable, therefore supported.
- (b)  $\implies$  (a) is Lemma 28. □

This, together with Theorem 3, completes the proof of Theorem 15.

## Bibliography

- [1] G. D. Plotkin, A structural approach to operational semantics, *J. Log. Algebr. Program.* 60-61 (2004) 17–139. doi:10.1016/j.jlap.2004.05.001.
- [2] L. Aceto, W. J. Fokkink, C. Verhoef, Structural operational semantics, in: J. A. Bergstra, A. Ponse, S. Smolka (Eds.), *Handbook of Process Algebra*, Elsevier, 2002, pp. 197–292.
- [3] M. R. Mousavi, M. A. Reniers, J. F. Groote, SOS formats and meta-theory: 20 years after, *Theoretical Computer Science* 373 (2007) 238–272.
- [4] J. F. Groote, Transition system specifications with negative premises, *Theoretical Computer Science* 118 (2) (1993) 263–299. doi:10.1016/0304-3975(93)90111-6.
- [5] R. J. van Glabbeek, The meaning of negative premises in transition system specifications II, *J. Log. Algebr. Program.* 60-61 (2004) 229–258. doi:10.1016/j.jlap.2004.03.007.
- [6] B. Bloom, S. Istrail, A. Meyer, Bisimulation can't be traced, *Journal of the ACM* 42 (1995) 232–268. doi:10.1145/200836.200876.
- [7] R. Bol, J. F. Groote, The meaning of negative premises in transition system specifications, *J. ACM* 43 (5) (1996) 863–914. doi:10.1145/234752.234756.
- [8] B. Klin, Bialgebras for structural operational semantics: An introduction, *Theoretical Computer Science* 412 (38) (2011) 5043–5069, CMCS Tenth Anniversary Meeting. doi:10.1016/j.tcs.2011.03.023.
- [9] D. Turi, G. D. Plotkin, Towards a mathematical operational semantics, in: *Proc. LICS'97*, IEEE Computer Society Press, 1997, pp. 280–291. doi:10.1109/LICS.1997.614955.
- [10] W. Fokkink, R. J. van Glabbeek, Ntyft/ntyxt rules reduce to ntree rules, *Information and Computation* 126 (1996) 1–10. doi:10.1006/inco.1996.0030.

- [11] W. Fokkink, Rooted branching bisimulation as a congruence, *Journal of Computer and System Sciences* 60 (1) (2000) 13 – 37. doi:10.1006/jcss.1999.1663.
- [12] M. Mousavi, I. Phillips, M. A. Reniers, I. Ulidowski, The meaning of ordered SOS, in: *Procs. FSTTCS 2006*, 2006, pp. 333–344. doi:10.1007/11944836\_31.
- [13] M. P. Fiore, S. Staton, A congruence rule format for name-passing process calculi, *Inf. Comput.* 207 (2) (2009) 209–236. doi:10.1016/j.ic.2007.12.005.
- [14] B. Klin, B. Nachyła, Distributive laws and decidable properties of SOS specifications, in: *Procs. EXPRESS/SOS 2014*, Vol. 160 of *EPTCS*, 2014, pp. 79–93. doi:10.4204/EPTCS.160.8.
- [15] D. Kozen, *Automata and computability*, Springer, 1997. doi:10.1007/978-1-4612-1844-9.

## Appendix A. Queue machines

A standard definition of queue machine (see e.g. [15, Exercise 99]) is somewhat different from our Definition 1:

**Definition 31.** A *standard queue machine* (SQM)  $\mathcal{M} = (Q, \Gamma, a_1, q_1, \delta)$  consists of a finite set  $Q$  of states with a chosen initial state  $q_1 \in Q$ , a finite alphabet  $\Gamma$  with a chosen initial symbol  $a_1 \in \Gamma$ , and a transition function

$$\delta : Q \times \Gamma \rightarrow Q \times \Gamma^*.$$

A *configuration* of  $\mathcal{M}$  is a pair  $(q, w) \in Q \times \Gamma^*$ . The machine induces a transition function on the set of configurations by:

$$(q, aw) \rightarrow_{\mathcal{M}} (q', wv) \quad \text{if} \quad \delta(q, a) = (q', v).$$

The machine *halts* if it reaches a configuration with the queue empty.

It is easy to prove that it is undecidable whether a given SQM halts from the initial configuration  $(q_1, a_1)$ . To prove Theorem 3, for a given SQM

$\mathcal{M} = (Q, \Gamma, a_1, q_1, \delta)$  we construct a QM  $\overline{\mathcal{M}} = (\overline{Q}, \overline{\Gamma}, a_1, q_1, \delta_0, \delta_1, \delta_2)$  so that  $\overline{\mathcal{M}}$  halts on  $(q_1, a_1)$  if and only if  $\mathcal{M}$  does.

The construction proceeds as follows:

- $\overline{Q} = Q \cup \tilde{Q}$ , where  $\tilde{Q}$  is a finite set of fresh additional states used to simulate transitions  $\delta(q, a) = (q', v)$  for  $|v| \geq 2$ , as specified below. The initial state  $q_1$  remains the same.
- $\overline{\Gamma} = \Gamma \cup \{\$\}$  for some  $\$ \notin \Gamma$ , and the initial symbol  $a_1$  remains the same.
- The transition functions  $\delta_0, \delta_1$  and  $\delta_2$  of  $\overline{\mathcal{M}}$  are defined as follows. For any  $q \in Q$  and  $a \in \Gamma$ , denote  $\delta(q, a) = (q', v)$  and reason by cases:
  - if  $v = \epsilon$ , then define  $\delta_1(q, a) = (\$, q')$ ,
  - if  $v = b$  for  $b \in \Gamma$ , then define  $\delta_1(q, a) = (b, q')$ ,
  - $v = b_1 b_2 \cdots b_n$  where  $n \geq 2$ , then define the following sequence of transitions:

$$\delta_1(q, a) = (b_1, p_1) \quad \delta_0(p_1) = (b_2, p_2) \quad \cdots \quad \delta_0(p_{n-1}) = (b_n, q')$$

where  $p_1, p_2, \dots, p_{n-1} \in \tilde{Q}$  are freshly chosen states.

Additionally, define the following transitions for the new symbol  $\$ \in \overline{\Gamma}$ :

- $\delta_2(q, \$, a) = \delta_1(q, a)$  for  $a \in \Gamma$ , and
- $\delta_2(q, \$, \$) = (\$, q)$ .

Note that the first case is well defined, since  $\delta_1(q, a)$  is defined for all  $q \in Q$  and  $a \in \Gamma$ .

The intuition is that the new symbol  $\$$  represents a “no symbol” in the queue. For a word  $w \in \overline{\Gamma}^*$ , denote by  $\underline{w} \in \Gamma^*$  the word obtained from  $w$  by removing all occurrences of  $\$$ .

We shall now prove that  $\mathcal{M}$  halts from the initial configuration if and only if  $\overline{\mathcal{M}}$  does.

First, we show that if  $\mathcal{M}$  does not halt from the initial configuration then  $\overline{\mathcal{M}}$  does not halt either. To this end, it is enough to prove that for any transition

$(q, aw) \rightarrow_{\mathcal{M}} (q', wv)$  arising from some  $\delta(q, a) = (q', v)$  in  $\mathcal{M}$ , and for any  $u \in \Gamma^*$  such that  $\underline{u} = aw$ ,  $\overline{\mathcal{M}}$  makes a nonempty sequence of transitions:

$$(q, u) \rightarrow_{\overline{\mathcal{M}}}^+ (q', u')$$

such that  $\underline{u}' = wv$ . There are three cases to consider, depending on how many symbols  $\$$  occur at the beginning of  $u$ :

A. Assume  $u = au''$  and  $\underline{u}'' = w$ . There are three subcases, depending on the length of  $v$ :

– if  $v = \epsilon$  then  $\delta_1(q, a) = (\$, q')$  hence  $\overline{\mathcal{M}}$  makes a transition:

$$(q, au'') \rightarrow_{\overline{\mathcal{M}}} (q', u''\$)$$

and  $\underline{u}''\$ = \underline{u}'' = w$ ;

– if  $v = b$  for  $b \in \Gamma$ , then  $\delta_1(q, a) = (b, q')$  hence  $\overline{\mathcal{M}}$  makes a transition:

$$(q, au'') \rightarrow_{\overline{\mathcal{M}}} (q', u''b)$$

and  $\underline{u}''b = wb$ ;

– if  $v = b_1b_2 \cdots b_n$  where  $n \geq 2$ , then according to our construction  $\overline{\mathcal{M}}$  makes a sequence of transitions:

$$(q, au'') \rightarrow_{\overline{\mathcal{M}}} (p_1, u''b_1) \rightarrow_{\overline{\mathcal{M}}} (p_2, u''b_1b_2) \rightarrow_{\overline{\mathcal{M}}} \cdots (q', u''b_1 \cdots b_n)$$

and  $\underline{u}''b_1b_2 \cdots b_n = wb_1b_2 \cdots b_n = wv$ .

B. Assume  $u = \$au''$  and  $\underline{u}'' = w$ . Since by definition  $\delta_2(q, \$, a) = \delta_1(q, a)$ ,  $\overline{\mathcal{M}}$  makes the same transition from configurations  $(q, \$au'')$  and  $(q, au'')$ , and the result follows from case A.

C. Assume  $u = \$\$u''$  and  $\underline{u}'' = aw$ . Since by definition  $\delta_2(q, \$, \$) = (\$, q)$ ,  $\overline{\mathcal{M}}$  makes the transition

$$(q, \$\$u'') \rightarrow_{\overline{\mathcal{M}}} (q, u''\$)$$

Note that  $\underline{u}''\$ = aw$  and the number of  $\$$ 's at the beginning of  $u''\$$  is smaller than at the beginning of  $\$\$u''$ ; formally, the result follows by induction on the number of  $\$$ 's at the beginning of  $u$ .

Conversely, we now prove that if  $\overline{\mathcal{M}}$  does not halt from the initial configuration then  $\mathcal{M}$  does not halt either. To this end, we show that from any configuration,  $\overline{\mathcal{M}}$  can make only finitely many transitions before it makes a transition that can be simulated by  $\mathcal{M}$  from a corresponding configuration. Formally, we show that for every transition  $(q, u) \rightarrow_{\overline{\mathcal{M}}} (q', u')$  with  $q \in Q$ , one of the following three conditions holds:

- A.  $\mathcal{M}$  makes a transition  $(q, \underline{u}) \rightarrow_{\mathcal{M}} (q', \underline{u}')$ , or
- B.  $\overline{\mathcal{M}}$  makes a finite sequence of transitions  $(q', u') \rightarrow_{\overline{\mathcal{M}}}^+ (q'', u'')$  such that  $\mathcal{M}$  makes a transition  $(q, \underline{u}) \rightarrow_{\mathcal{M}} (q'', \underline{u}'')$ , or
- C.  $q = q'$ ,  $\underline{u} = \underline{u}'$  and  $|u| > |u'|$ .

By the construction of  $\overline{\mathcal{M}}$ , a transition  $(q, u) \rightarrow_{\overline{\mathcal{M}}} (q', u')$  with  $q \in Q$  may arise from one of the following situations:

- $u = aw$  for  $a \in \Gamma$ ,  $u' = w\$$ , and  $\delta(q, a) = (q', \epsilon)$ . Then

$$(q, \underline{u}) = (q, \underline{aw}) \rightarrow_{\mathcal{M}} (q', \underline{w}) = (q', \underline{u}')$$

and condition A holds.

- $u = aw$  for  $a \in \Gamma$ ,  $u' = wb$ , and  $\delta(q, a) = (q', b)$ . Then

$$(q, \underline{u}) = (q, \underline{aw}) \rightarrow_{\mathcal{M}} (q', \underline{wb}) = (q', \underline{u}')$$

and condition A holds.

- $u = aw$  for  $a \in \Gamma$ ,  $u' = wb_1$ , and  $\delta(q, a) = (q'', b_1 b_2 \cdots b_n)$  for some  $q'' \in Q$  and  $n \geq 2$ , and  $q' = p_1 \notin Q$  is the fresh state such that  $\delta_1(q, a) = (b_1, p_1)$ . Then

$$(p_1, \underline{wb_1}) \rightarrow_{\overline{\mathcal{M}}}^+ (q'', \underline{wb_1 b_2 \cdots b_n})$$

and

$$(q, \underline{u}) = (q, \underline{aw}) \rightarrow_{\mathcal{M}} (q'', \underline{wb_1 b_2 \cdots b_n}) = (q'', \underline{u}')$$

and condition B holds.

- $u = \$aw$  for  $a \in \Gamma$ , and  $(q, aw) \rightarrow_{\overline{\mathcal{M}}} (q', u')$ . Then, since  $\underline{u} = \underline{aw}$ , one of the above three situations occurs and condition A or B holds.
- $u = \$\$w$ ,  $u' = w\$$  and  $q = q'$ . Then condition C holds.

This completes the proof of Theorem 3.