Fundamenta Informaticae XX (2008) 1–24
IOS Press

Nets with tokens which carry data

Ranko Lazić* [C]
Department of Computer Science, University of Warwick, UK

Tom Newcomb, Joël Ouaknine, A.W. Roscoe, James Worrell
Computing Laboratory, University of Oxford, UK

Abstract. We study data nets, a generalisation of Petri nets in which tokens carry data from linearly-ordered infinite domains and in which whole-place operations such as resets and transfers are possible. Data nets subsume several known classes of infinite-state systems, including multiset rewriting systems and polymorphic systems with arrays.

We show that coverability and termination are decidable for arbitrary data nets, and that boundedness is decidable for data nets in which whole-place operations are restricted to transfers. By providing an encoding of lossy channel systems into data nets without whole-place operations, we establish that coverability, termination and boundedness for the latter class have non-primitive recursive complexity. The main result of the paper is that, even for unordered data domains (i.e., with only the equality predicate), each of the three verification problems for data nets without whole-place operations has non-elementary complexity.

Keywords: Petri nets, infinite-state systems, program verification, computational complexity

1. Introduction

Petri nets (e.g., [21]) are a fundamental model of concurrent systems. Being more expressive than finite-state machines and less than Turing-powerful, Petri nets have an established wide range of applications and a variety of analysis tools (e.g., [13]).

The analysis tools are based on the extensive literature on decidability and complexity of verification problems ([10] is a comprehensive survey). In this paper, we focus on three basic decision problems, to which a number of other verification questions can be reduced:

*Supported by the EPSRC (GR/S52759/01) and the Intel Corporation.
[C]Corresponding author

Coverability: Is a marking reachable which is greater than or equal to a given marking?

Termination: Are all computations finite?

Boundedness: Is the set of all reachable markings finite?

By the results in [17, 20], each of coverability, termination and boundedness is EXPSPACE-complete for Petri nets.

Many extensions of Petri nets preserve decidability of various verification problems. Notably, affine well-structured nets were formulated in [11] as an elegant extension of Petri nets by whole-place operations. The latter are resets, which empty a place, and transfers, which take all tokens from a place and put them onto one or more specified places (possibly several times). Hence, two subclasses of affine WSNs are reset nets and transfer nets, in which whole-place operations are restricted to resets and to transfers, respectively. As shown in [11], coverability and termination for affine WSNs, and boundedness for transfer nets, are decidable. However, compared with Petri nets, there is a dramatic increase in complexity: it follows from the results on lossy channel systems in [23] that coverability and termination for reset nets and transfer nets, and boundedness for transfer nets, are not primitive recursive.[1] It was proved in [9] that boundedness for reset nets is undecidable.

Another important direction of extending Petri nets is by allowing tokens to carry data from infinite domains. (Data from finite domains do not increase expressiveness.) For example, in timed Petri nets [4], each token is equipped with a real-valued clock which represents the age of the token. Multiset rewriting specifications over constraint systems $\mathcal{C}$ [8, 1] can be seen as extensions of Petri nets in which tokens may carry data from the domain of $\mathcal{C}$ and transitions can be constrained using $\mathcal{C}$. In mobile synchronizing Petri nets [22], tokens may carry identifiers from an infinite domain, and transitions may require that an identifier be fresh (i.e., not currently carried by any token).

In this paper, we focus on the following two questions:

(1) Is there a general extension of Petri nets in which tokens carry data from infinite domains, in which whole-place operations are possible, and such that coverability, termination and boundedness are decidable (either for the whole class of extended nets or for interesting subclasses)?

(2) If the answer to the previous question is positive, and if we restrict to the subclass without whole-place operations, do coverability, termination and boundedness remain EXPSPACE-complete (as for Petri nets), or are their complexities greater? What happens if we restrict further to the simplest data domains, i.e. those with only the equality predicate?

Data nets. To answer question (1), we define data nets, in which tokens carry data from linearly-ordered infinite domains. As in Petri nets, transitions consume and produce tokens. For a transition to be firable, we can require that the data which are carried by the tokens to be consumed are ordered in a certain way. In addition to such data, transitions can choose finitely many other data, which satisfy further ordering constraints and which may or may not be present in the current marking. In the production phase, tokens which carry either kind of data can be put into the marking. Data nets also support whole-place operations.

[1] Recall the Ritchie-Cobham property [19, page 297]: a decision problem (i.e. a set) is primitive recursive iff it is solvable in primitive recursive time/space.

In the next few paragraphs, we introduce data nets in an informal but detailed manner, for clarity of the subsequent discussion of contributions of the paper and relations to the literature. As an alternative order of presentation, the reader may wish to postpone the following and read it in conjunction with Section 2.2, where data nets are defined formally.

Data nets are based on affine WSNs [11]. Markings of an affine WSN are vectors in $\mathbb{N}^P$, where P is the finite set of all places. A transition t of an affine WSN is given by vectors $F_t, H_t \in \mathbb{N}^P$ and a square matrix $G_t \in \mathbb{N}^{P\times P}$. Such a transition is firable from a marking m iff $m \geq F_t$, and in that case it produces the marking $(m - F_t)G_t + H_t$. Whole-place operations are performed by the multiplication with G_t.

Since a linear ordering $\preceq$ is the only operation available on data, markings of data nets are finite sequences of vectors in $\mathbb{N}^P \setminus \{\mathbf{0}\}$. Each index j of such a marking s corresponds to an implicit datum d_j, and we have that $j \leq j'$ iff $d_j \preceq d_{j'}$. For each $p \in P$, $s(j)(p)$ is the number of tokens which carry d_j and are at place p. We say that such tokens are at index j. Now, each transition t has an arity $\alpha_t \in \mathbb{N}$. For a transition t to be fired from a marking s, we choose nondeterministically α_t mutually distinct data. Some of those data may be fresh (i.e., not carried by any token in s), so picking the α_t data is formalised by first expanding s to a finite sequence $s_\dagger$ by inserting the vector $\mathbf{0}$ at arbitrary positions, and then picking an increasing (in particular, injective) mapping

$$\iota : \{1, \ldots, \alpha_t\} \to \{1, \ldots, |s_\dagger|\}$$

such that each occurrence of $\mathbf{0}$ is in its range. Now, such a mapping ι partitions $\{1, \ldots, |s_\dagger|\}$ into α_t singletons and $\alpha_t + 1$ contiguous "regions" as follows, where the $Reg_{(i,i+1)}$ are region identifiers:

$$\underbrace{1, \ldots, \iota(1) - 1}_{Reg_{(0,1)}}, \iota(1), \underbrace{\iota(1) + 1, \ldots, \iota(2) - 1}_{Reg_{(1,2)}}, \ldots, \iota(\alpha_t), \underbrace{\iota(\alpha_t) + 1, \ldots, |s_\dagger|}_{Reg_{(\alpha_t,\alpha_t+1)}}$$

The action of t on s with respect to $s_\dagger$ and ι is determined by vectors F_t and H_t, and a square matrix G_t, whose elements are natural numbers, and which are indexed by

$$(\{1, \ldots, \alpha_t\} \cup \{Reg_{(i,i+1)} \,:\, 0 \leq i \leq \alpha_t\}) \times P$$

It consists of the following stages, where $i, i' \in \{1, \ldots, \alpha_t\}$, $R, R' \in \{Reg_{(i,i+1)} \,:\, 0 \leq i \leq \alpha_t\}$ and $p, p' \in P$.

subtraction: for each i and p, $F_t(i, p)$ tokens at index $\iota(i)$ are taken from p;[2]

multiplication: all tokens are taken simultaneously, and then:

- for each token taken from p at index $\iota(i)$, $G_t(i, p, i', p')$ tokens are put onto p' at index $\iota(i')$, and for each j' in region R', $G_t(i, p, R', p')$ tokens are put onto p' at index j';
- for each token taken from p at index j in region R, $G_t(R, p, i', p')$ tokens are put onto p' at index $\iota(i')$, and $G_t(R, p, R, p')$ tokens are put onto p' at index j;

addition: for each i and p, $H_t(i, p)$ tokens are put onto p at index $\iota(i)$, and for each j in region R and p, $H_t(R, p)$ tokens are put onto p at index j.

[2] In order to have well-structuredness (see Proposition 2.6) and for simplicity, entries $F_t(R, p)$ are not used, and neither are entries $G_t(R, p, R', p')$ with $R \neq R'$, so they are assumed to be 0.

Example 1.1. Consider $P = \{p_1, p_2\}$ and a transition t with $\alpha_t = 1$ given by:

F_t	$Reg_{(0,1)}$		1		$Reg_{(1,2)}$	
	0	0	1	1	0	0
	p_1	p_2	p_1	p_2	p_1	p_2

H_t	$Reg_{(0,1)}$		1		$Reg_{(1,2)}$	
	0	0	2	1	6	0
	p_1	p_2	p_1	p_2	p_1	p_2

G_t	$Reg_{(0,1)}$		1		$Reg_{(1,2)}$		
$Reg_{(0,1)}$	0	1	0	0	0	0	p_1
	1	0	0	0	0	0	p_2
1	0	0	2	0	3	0	p_1
	0	0	0	1	3	0	p_2
$Reg_{(1,2)}$	0	0	0	0	1	0	p_1
	0	0	0	2	0	1	p_2
	p_1	p_2	p_1	p_2	p_1	p_2	

From a marking s, in terms of data represented by the indices of s, transition t is fired as follows:

1. a datum d is chosen nondeterministically, such that each of p_1 and p_2 contain at least 1 token carrying d (so, d cannot be fresh);
2. for each datum $d' \prec d$, all tokens at p_1 carrying d' are transferred to p_2, and vice-versa;
3. for each token at p_1 or p_2 carrying d, and each $d' \succ d$, 3 tokens carrying d' are put onto p_1;
4. the number of tokens at p_1 carrying d is multiplied by 2;
5. for each token at p_2 carrying $d' \succ d$, 2 tokens carrying d are put onto p_2.

Since $H_t = F_t G_t$, the addition stage of performing t exactly "undoes" the subtraction stage, so t performs only whole-place operations.

In Section 2.2, the above will be formalised so that t is firable from s with respect to $s_\dagger$ and ι iff $s_\dagger \geq [\![F_t]\!]_\iota^{|s_\dagger|}$, and in that case it produces the marking obtained from $(s_\dagger - [\![F_t]\!]_\iota^{|s_\dagger|})[\![G_t]\!]_\iota^{|s_\dagger|} + [\![H_t]\!]_\iota^{|s_\dagger|}$ by removing each entry $\mathbf{0}$, where $[\![F_t]\!]_\iota^{|s_\dagger|}$, $[\![G_t]\!]_\iota^{|s_\dagger|}$ and $[\![H_t]\!]_\iota^{|s_\dagger|}$ are appropriate "expansions" of F_t, G_t and H_t, indexed by $\{1, \ldots, |s_\dagger|\} \times P$.

Since vectors $\mathbf{0}$ which correspond to fresh data can be inserted at arbitrary positions to fire a transition, the linear ordering on data is assumed to be dense and without least and greatest elements. Having a least or greatest element can easily be simulated, and density is not a restriction when considering only finite computations (as is the case for the coverability problem).

We show that affine WSNs [11] are equivalent to a class of data nets whose transitions have arity 1. Data nets also subsume timed Petri nets [4] and timed networks [3], in the sense that systems obtained after quotienting by time regions can be simulated by data nets, where the data domain is fractional parts of clock values. Monadic multiset rewriting specifications over order constraints on rationals or reals [8] and over gap-order constrains on integers [1] can be translated to data nets, subject to the remarks above about density. Mobile synchronizing Petri nets [22], lossy channel systems [2], and polymorphic systems with one array of type $\langle X, \leq \rangle \to \{1, \ldots, n\}$ or with two arrays of types $\langle X, = \rangle \to \langle Y, \leq \rangle$ and $\langle X, = \rangle \to \{1, \ldots, n\}$ [16, 15], can also be expressed using data nets.

Decidability. Using the theory of well-structured transition systems [12], we prove that coverability and termination for arbitrary data nets, and boundedness for data nets in which whole-place operations are restricted to transfers, are decidable. Thus, question (1) posed above is answered positively. The decidability of coverability for data nets subsumes the results in [11, 4, 3, 8, 1, 22, 2, 16, 15] that coverability is decidable for the respective classes of infinite-state systems mentioned above, and in most cases the proof in this paper is more succinct.

Hardness. To question (2) above, we obtain the following answers. We say that a data net is *Petri* iff it does not contain whole-place operations, and *unordered* iff it makes use only of equality between data (and not of the linear ordering).

- By providing a translation from lossy channel systems to Petri data nets, we establish that coverability, termination and boundedness for the latter class are not primitive recursive. The encoding uses the linear ordering on the data domain, for picking fresh data which are employed in simulating writes to channels.
- The main result of the paper is that coverability, termination and boundedness for unordered Petri data nets are not elementary, i.e., their computational complexities cannot be bounded by towers of exponentials of fixed heights. That is a surprising result, since unordered Petri data nets are highly constrained systems. In particular, they do not provide a mechanism for ensuring that a datum chosen in a transition is fresh (i.e., not present in the current marking). The result is proved by simulating a hierarchy of bounded counters, which is reminiscent of the "rulers" construction of Meyer and Stockmeyer (e.g., [18]).

Therefore, this paper shows that, when Petri nets are generalised to allow tokens to carry data from infinite domains, standard verification problems which were in EXPSPACE become non-elementary, even when data can only be compared for equality and whole-place operations are not allowed.

By translating Petri data nets and unordered Petri data nets to subclasses of systems in [8, 1, 22, 16, 15], the two hardness results yield the same lower bounds for corresponding decision problems for such subclasses. In particular, we obtain non-elementariness of verifying monadic multiset rewriting specifications with only equality constraints [8] and of verifying polymorphic systems with two arrays of types $\langle X, = \rangle \to \langle Y, = \rangle$ and $\langle X, = \rangle \to \{1, \ldots, n\}$ [16].

Paper organisation. Section 2 contains preliminaries, including definitions of data nets and of several relevant subclasses, some basic results, and an example. In Section 3, we present the translation from lossy channel systems to Petri data nets. Sections 4 and 5 contain the decidability and hardness results. Some remaining open problems are discussed in Section 6.

2. Preliminaries

Sets, quasi-orders and mappings. For $n \in \mathbb{N}$, let $[n] = \{1, \ldots, n\}$. We write $\mathbb{N}_\omega$ for $\mathbb{N} \cup \{\omega\}$. The linear ordering $\leq$ on $\mathbb{N}$ is extended to $\mathbb{N}_\omega$ by having $n < \omega$ for each $n \in \mathbb{N}$.

A set A and a relation $\preceq$ on A form a *quasi-order* iff $\preceq$ is reflexive and transitive. We write $a_1 \prec a_2$ iff $a_1 \preceq a_2$ and $a_2 \not\preceq a_1$.

For any $A' \subseteq A$, its upward closure is $\uparrow A' = \{a \in A : \exists a' \in A' \cdot a' \preceq a\}$. We say that A' is upwards-closed iff $A' = \uparrow A'$. A *basis* of an upwards-closed set A' is a subset A'' such that $A' = \uparrow A''$. Downward closure (written $\downarrow A'$), closedness and bases are defined symmetrically.

A mapping f from a quasi-order $\langle A, \preceq \rangle$ to a quasi-order $\langle A', \preceq' \rangle$ is *increasing* iff $a_1 \prec a_2 \Rightarrow f(a_1) \prec' f(a_2)$.

Vectors and matrices. For sets A and B, let A^B denote the set of all B-indexed vectors of elements of A, i.e., the set of all mappings $B \to A$. For example, $\mathbb{N}^{[n]\times[n']}$ is the set of all $n \times n'$ matrices of natural numbers. For $a \in A$, let $\mathbf{a} \in A^B$ denote the vector whose each entry equals a. Let $Id \in \mathbb{N}^{B\times B}$ denote the identity square matrix.

A quasi-ordering $\preceq$ on A induces the following quasi-ordering on A^B: $v \preceq v'$ iff $v(b) \preceq v'(b)$ for all $b \in B$.

Sequences and bags. For a set A, let $Seq(A)$ denote the set of all finite sequences of elements of A. For $s \in Seq(A)$, let $|s|$ denote the length of s, and $s(1), \ldots, s(|s|)$ denote its elements.

For $s, s' \in Seq(A)$ and $a \in A$, we say that s' is an *a-expansion* of s (equivalently, s is the *a-contraction* of s') iff s is obtained by removing each occurrence of a from s'.

For $s, s' \in Seq(A)$, we write $s \sim s'$ iff s' can be obtained from s by permuting its entries. We define the set $Bag(A)$ of all finite bags (i.e., multisets) of elements of A as the set of all equivalence classes of $\sim$. Let $\overline{s}$ denote the equivalence class of s, i.e., the bag with the same elements as s.

Suppose $\langle A, \preceq \rangle$ is a quasi-order. The quasi-ordering $\preceq$ induces quasi-orderings on $Seq(A)$ and $Bag(A)$ as follows. For $s, s' \in Seq(A)$, we write $s \preceq s'$ iff there exists an increasing $\iota : [|s|] \to [|s'|]$ such that $s(i) \preceq s'(\iota(i))$ for all $i \in [|s|]$. For $b, b' \in Bag(A)$, we write $b \preceq b'$ iff there exist $s \in b$ and $s' \in b'$ such that $s \preceq s'$.

Well-quasi-orderings. A quasi-ordering $\preceq$ on a set A is a well-quasi-ordering iff, for every infinite sequence $a_1, a_2, \ldots \in A$, there exist $i < j$ such that $a_i \preceq a_j$.

Proposition 2.1. ([14])
Whenever $\preceq$ is a well-quasi-ordering on a set A, the induced orderings on $Seq(A)$ and $Bag(A)$ also are well-quasi-orderings.

2.1. Affine well-structured nets

We recall the notion of affine well-structured net [11].[3] Such a net is a tuple $\langle P, T, F, G, H \rangle$ such that P is a finite set of places, T is a finite set of transitions, and for each $t \in T$, F_t and H_t are vectors in $\mathbb{N}^P$, and G_t is a matrix in $\mathbb{N}^{P\times P}$.

Markings of an affine WSN $\langle P, T, F, G, H \rangle$ are vectors in $\mathbb{N}^P$. A marking m' can be obtained from a marking m by firing a transition $t \in T$, written $m \xrightarrow{t} m'$, iff $m \geq F_t$ and $m' = (m - F_t)G_t + H_t$.

As was shown in [11], Petri nets and many of their known extensions are special cases of affine WSNs. In particular, Petri nets and their extensions by (generalised) resets and transfers are equivalent to the classes of affine WSNs $\langle P, T, F, G, H \rangle$ determined by the following restrictions:

[3] For technical reasons, the formalisation of affine WSNs in this paper is slightly different, but equivalent.

Petri nets: $\forall t \in T \cdot G_t = Id$

reset nets: $\forall t \in T \cdot G_t \leq Id$

transfer nets: $\forall t \in T, p \in P \cdot \exists p' \in P \cdot G_t(p, p') > 0$

2.2. Data nets

Given $n \in \mathbb{N}$, let $Regs(n) = \{Reg_{(i,i+1)} : 0 \leq i \leq n\}$. For each $0 \leq i \leq n$, $m \geq n$ and increasing $\iota : [n] \to [m]$, let $[\![Reg_{(i,i+1)}]\!]^m_\iota = \{j \in [m] : \iota(i) < j < \iota(i+1)\}$, where by convention $\iota(0) = 0$ and $\iota(n+1) = m+1$.

A *data net* is a tuple $\langle P, T, \alpha, F, G, H\rangle$ such that:

- P is a finite set of places;
- T is a finite set of transitions;
- for each $t \in T$, $\alpha_t \in \mathbb{N}$ specifies the arity of t;
- for each $t \in T$, $F_t \in \mathbb{N}^{([\alpha_t] \cup Regs(\alpha_t)) \times P}$, and $F_t(R, p) = 0$ whenever $R \in Regs(\alpha_t)$ and $p \in P$;
- for each $t \in T$, $G_t \in \mathbb{N}^{(([\alpha_t] \cup Regs(\alpha_t)) \times P)^2}$, and $G_t(R, p, R', p') = 0$ whenever $R, R' \in Regs(\alpha_t)$, $R \neq R'$ and $p, p' \in P$;
- for each $t \in T$, $H_t \in \mathbb{N}^{([\alpha_t] \cup Regs(\alpha_t)) \times P}$.

Suppose $\langle P, T, \alpha, F, G, H\rangle$ is a data net, and $t \in T$. Any $m \geq \alpha_t$ and increasing $\iota : [\alpha_t] \to [m]$ determine the following instances of F_t, G_t and H_t:

- $[\![F_t]\!]^m_\iota \in \mathbb{N}^{[m] \times P}$ is defined by

$$[\![F_t]\!]^m_\iota(\iota(i), p) = F_t(i, p) \qquad [\![F_t]\!]^m_\iota(j, p) = F_t(R, p) \text{ for } j \in [\![R]\!]^m_\iota$$

- $[\![G_t]\!]^m_\iota \in \mathbb{N}^{([m] \times P)^2}$ is defined by

$$\begin{array}{rcll} [\![G_t]\!]^m_\iota(\iota(i), p, \iota(i'), p') & = & G_t(i, p, i', p') & \\ [\![G_t]\!]^m_\iota(\iota(i), p, j', p') & = & G_t(i, p, R, p') & \text{for } j' \in [\![R]\!]^m_\iota \\ [\![G_t]\!]^m_\iota(j, p, \iota(i'), p') & = & G_t(R, p, i', p') & \text{for } j \in [\![R]\!]^m_\iota \\ [\![G_t]\!]^m_\iota(j, p, j, p') & = & G_t(R, p, R, p') & \text{for } j \in [\![R]\!]^m_\iota \\ [\![G_t]\!]^m_\iota(j, p, j', p') & = & 0 & \text{otherwise} \end{array}$$

- $[\![H_t]\!]^m_\iota \in \mathbb{N}^{[m] \times P}$ is defined in the same way as $[\![F_t]\!]^m_\iota$.

A *marking* of a data net $\langle P, T, \alpha, F, G, H\rangle$ is a finite sequence of vectors in $\mathbb{N}^P \setminus \{\mathbf{0}\}$. A marking s' can be obtained from a marking s by firing a transition $t \in T$, written $s \xrightarrow{t} s'$, iff there exist a $\mathbf{0}$-expansion $s_\dagger$ of s and an increasing $\iota : [\alpha_t] \to [|s_\dagger|]$ such that:[4]

[4] In (ii) and (iii), $s_\dagger$ is treated as a vector in $\mathbb{N}^{[|s_\dagger|] \times P}$.

(i) $\{j : s_\dagger(j) = \mathbf{0}\} \subseteq Range(\iota)$;

(ii) $s_\dagger \geq [\![F_t]\!]_\iota^{|s_\dagger|}$;

(iii) s' is the $\mathbf{0}$-contraction of $(s_\dagger - [\![F_t]\!]_\iota^{|s_\dagger|})[\![G_t]\!]_\iota^{|s_\dagger|} + [\![H_t]\!]_\iota^{|s_\dagger|}$.

We may also write $s \xrightarrow{t,s_\dagger,\iota} s'$, or just $s \to s'$.

Proposition 2.2. For any data net, its transition system $\langle Seq(\mathbb{N}^P \setminus \{\mathbf{0}\}), \to\rangle$ is finitely branching.

2.3. Decision problems

We consider the following standard problems:

Coverability: Given a data net, and markings s and s', to decide whether some marking $s'' \geq s'$ is reachable from s.

Termination: Given a data net, and a marking s, to decide whether all computations from s are finite.

Boundedness: Given a data net, and a marking s, to decide whether the set of all markings reachable from s is finite.

Coverability, termination and boundedness for affine WSNs are defined in the same way.

2.4. Classes of data nets

We now define several classes of data nets. Figure 1 shows the inclusions among classes of data nets and affine well-structured nets in Propositions 2.4, 2.5, 2.7 and 3.1 below. In addition, the mapping $\mathcal{N} \mapsto \widetilde{\mathcal{N}}$ and its inverse (see Proposition 2.5) provide a correspondence between unary transfer data nets (resp., unary Petri data nets) and transfer nets (resp., Petri nets). The dashed line represents the fact that Proposition 3.1 does not provide a reduction for the boundedness problem.

Unordered data nets. A data net $\langle P, T, \alpha, F, G, H\rangle$ is unordered iff:

(i) for each $t \in T$, $R, R' \in Regs(\alpha_t)$ and $p, p' \in P$, we have $G_t(R, p, R, p') = G_t(R', p, R', p')$ and $H_t(R, p) = H_t(R', p)$;

(ii) for each $t \in T$ and permutation π of $[\alpha_t]$, there exists $t' \in T$ such that $F_{t'}$, $G_{t'}$ and $H_{t'}$ are obtained from F_t, G_t and H_t (respectively) by applying π to each index in $[\alpha_t]$.

Given an unordered data net $\langle P, T, \alpha, F, G, H\rangle$, we write $t \sim t'$ iff t and t' have the property in (ii) above. That defines an equivalence relation on T, and we write $\overline{t}$ for the equivalence class of t. From the following proposition, the same-bag relation $\sim$ between markings is a bisimulation on the transition system of $\langle P, T, \alpha, F, G, H\rangle$.[5]

[5]Conditions (i) and (ii) in the definition of unordered data nets suggest an alternative formalisation, where only one region is used for indexing F, G and H, and only one transition from each equivalence class is represented. Such a formalisation is more succinct (exponentially in transition arities), but that issue is not important in this paper. In addition, by Proposition 2.3, markings of unordered data nets can be regarded as bags.

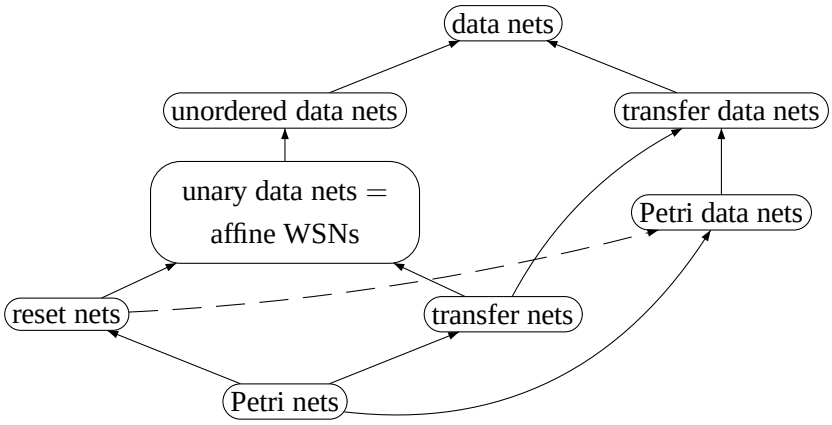

Figure 1. Inclusions among classes of data nets

Proposition 2.3. For any unordered data net, whenever $s_1 \xrightarrow{t} s_2$ and $s_1' \sim s_1$, we have $s_1' \xrightarrow{t'} s_2'$ for some $t' \sim t$ and $s_2' \sim s_2$.

Unary data nets. A data net $\langle P, T, \alpha, F, G, H\rangle$ is unary iff:

(i) for each $t \in T$, $\alpha_t = 1$;

(ii) for each $t \in T$, there exists $p \in P$ such that $F_t(1, p) > 0$;

(iii) for each $t \in T$, $R \in Regs(1)$ and $p, p' \in P$, we have $G_t(1, p, R, p') = 0$, $G_t(R, p, 1, p') = 0$, $G_t(R, p, R, p) = 1$, $G_t(R, p, R, p') = 0$ if $p \neq p'$, and $H_t(R, p) = 0$.

Proposition 2.4. Any unary data net is an unordered data net.

Given a unary data net $\mathcal{N} = \langle P, T, \alpha, F, G, H\rangle$, let $\widetilde{\mathcal{N}} = \langle P, T, \tilde{F}, \tilde{G}, \tilde{H}\rangle$ be the affine WSN such that $\tilde{F}$, $\tilde{G}$ and $\tilde{H}$ are obtained from F_t, G_t and H_t (respectively) by removing entries which involve indices from $Regs(1)$. Observe that, conversely, for each affine WSN $\mathcal{N}'$ in which no transition is firable from $\mathbf{0}$, there is a unique unary data net $\mathcal{N}$ such that $\widetilde{\mathcal{N}} = \mathcal{N}'$. Both $\mathcal{N} \mapsto \widetilde{\mathcal{N}}$ and its inverse are computable in logarithmic space.

Proposition 2.5.

(a) For any unary data net $\mathcal{N}$, we have that $s \xrightarrow{t} s'$ iff $|s'| = |s|$ and there exists $i \in [|s|]$ with $s(i) \xrightarrow{t} s'(i)$ in $\widetilde{\mathcal{N}}$ and $s'(j) = s(j)$ for all $j \neq i$.

(b) Coverability of s' from s in a unary data net $\mathcal{N}$ is equivalent to existence of an increasing $\iota : [|s'|] \to [|s|]$ such that $s'(i)$ is coverable from $s(\iota(i))$ in $\widetilde{\mathcal{N}}$ for each $i \in [|s'|]$.

Termination (resp., boundedness) from s in a unary data net $\mathcal{N}$ is equivalent to $\widetilde{\mathcal{N}}$ being terminating (resp., bounded) from $s(i)$ for each $i \in [|s|]$.

(c) Coverability of m' from m, termination from m and boundedness from m in an affine well-structured net $\widetilde{\mathcal{N}}$ are equivalent to coverability of $\langle m' \rangle$ from $\langle m \rangle$, termination from $\langle m \rangle$ and boundedness from $\langle m \rangle$ (respectively) in $\mathcal{N}$.

Note that Proposition 2.5 (c) can be extended to affine WSN with transitions firable from $\mathbf{0}$ by adding an auxiliary place in which a single token is kept.

Transfer data nets. A data net $\langle P, T, \alpha, F, G, H \rangle$ is transfer iff:

(i) for each $t \in T$, $i \in [\alpha_t]$ and $p \in P$, we have $G_t(i, p, i', p') > 0$ for some $i' \in [\alpha_t]$ and $p' \in P$;

(ii) for each $t \in T$, $R \in \mathit{Regs}(\alpha_t)$ and $p \in P$, either we have $G_t(R, p, i', p') > 0$ for some $i' \in [\alpha_t]$ and $p' \in P$, or we have $G_t(R, p, R, p') > 0$ for some $p' \in P$.

Observe that (i) and (ii) are satisfied by the transition t in Example 1.1.

Proposition 2.6.

(a) Whenever $s_1 \xrightarrow{t} s_2$ in a data net and $s_1' \geq s_1$, then $s_1' \xrightarrow{t} s_2'$ for some $s_2' \geq s_2$.

(b) Whenever $s_1 \xrightarrow{t} s_2$ in a transfer data net and $s_1' > s_1$, then $s_1' \xrightarrow{t} s_2'$ for some $s_2' > s_2$.

Petri data nets. In Petri data nets, whole-place operations are not allowed, and transitions can produce tokens carrying only data which were chosen during the firing. Formally, a data net $\langle P, T, \alpha, F, G, H \rangle$ is Petri iff:

- for each $t \in T$, $G_t = \mathit{Id}$;
- for each $t \in T$, $R \in \mathit{Regs}(\alpha_t)$ and $p \in P$, $H_t(R, p) = 0$.

Proposition 2.7. Any Petri data net is a transfer data net.

2.5. Example: a file system

As an illustration, we now show how a file system which permits unboundedly many users, user processes and files can be modelled as a data net. A variety of other examples of systems expressible using data nets can be found in [4, 3, 8, 1, 22, 2, 16], including a real-timed mutual exclusion protocol, a distributed authentication protocol, a communication protocol over unreliable channels, and a leader election algorithm.

We suppose there are two user categories: administrators and staff members. Let `Administrator` be a finite set consisting of all possible states which an administrator process can be in, let `Staff` be such a set for staff-member processes, and let `Contents` be a finite set of all possible file contents. In case file contents is unbounded, the `Contents` set may consist of finitary abstractions, which include information such as file names. We assume that `Administrator`, `Staff` and `Contents` are mutually disjoint.

The set of places is

$$P = \texttt{Administrator} \cup \texttt{Staff} \cup \texttt{Contents}$$

Tokens represent user processes and files, and data which they carry represents user identities. More specifically:

- a token at place $a \in \texttt{Administrator}$ carrying datum d represents a process of administrator d and which is in state a;
- a token at place $b \in \texttt{Staff}$ carrying datum d represents a process of staff member d and which is in state b;
- a token at place $c \in \texttt{Contents}$ carrying datum d represents a file owned by user d and with contents c.

To express a write by a staff-member process in state b to a file with contents c, which changes b to b' and c to c', we define a transition $\texttt{write}(b, b', c, c')$. It involves one user, so $\alpha_{\texttt{write}(b,b',c,c')} = 1$. Firstly, it takes one token from place b and one token from place c. They must carry the same datum, which ensures that the user owns the file.

$$F_{\texttt{write}(b,b',c,c')}(1, b) = 1 \qquad F_{\texttt{write}(b,b',c,c')}(1, c) = 1$$

The transition involves no whole-place operations, so $G_{\texttt{write}(b,b',c,c')} = Id$. Finally, it puts one token onto place b' and one token onto place c', which carry the same datum as the two taken tokens.

$$H_{\texttt{write}(b,b',c,c')}(1, b') = 1 \qquad H_{\texttt{write}(b,b',c,c')}(1, c') = 1$$

The remaining entries of $F_{\texttt{write}(b,b',c,c')}$ and $H_{\texttt{write}(b,b',c,c')}$ are 0.

As a slightly more complex example, we can express a change of ownership of a file with contents c from an administrator to a staff member. It involves an administrator process which changes state from a to a', and a staff-member processes which changes state from b to b'. Since two users are involved, we have $\alpha_{\texttt{change}(c,a,a',b,b')} = 2$. As in the previous example, $G_{\texttt{change}(c,a,a',b,b')} = Id$ and we show only entries which are not 0:

$$\begin{array}{rclrcl}
F_{\texttt{change}(c,a,a',b,b')}(1, c) & = & 1 & H_{\texttt{change}(c,a,a',b,b')}(2, c) & = & 1 \\
F_{\texttt{change}(c,a,a',b,b')}(1, a) & = & 1 & H_{\texttt{change}(c,a,a',b,b')}(1, a') & = & 1 \\
F_{\texttt{change}(c,a,a',b,b')}(2, b) & = & 1 & H_{\texttt{change}(c,a,a',b,b')}(2, b') & = & 1
\end{array}$$

In the $\texttt{change}(c, a, a', b, b')$ transition, it is assumed that the administrator identity is smaller than the staff-member identity. To cover the opposite case, and to have an unordered data net, we define a transition $\texttt{change}(c, b, b', a, a')$. The definition is the same as that of $\texttt{change}(c, a, a', b, b')$, except that indices 1 and 2 are swapped when defining $F_{\texttt{change}(c,b,b',a,a')}$ and $H_{\texttt{change}(c,b,b',a,a')}$.

The data net having the three sets of transitions introduced so far is unordered and Petri. Implementing the following action makes it no longer Petri, in fact not even a transfer data net: all processes and files of a staff member who has a process which is in state b are removed. We have $\alpha_{\texttt{crash}(b)} = 1$, $F_{\texttt{crash}(b)}(1, b) = 1$, the remaining entries of $F_{\texttt{crash}(b)}$ and all entries of $H_{\texttt{crash}(b)}$ are 0, and:

$$\begin{array}{rcll}
G_{\texttt{crash}(b)}(1, p, 1, p') & = & 0 & \text{for } p, p' \in P \\
G_{\texttt{crash}(b)}(1, p, R, p') & = & 0 & \text{for } R \in \mathit{Regs}(1) \text{ and } p, p' \in P \\
G_{\texttt{crash}(b)}(R, p, 1, p') & = & 0 & \text{for } R \in \mathit{Regs}(1) \text{ and } p, p' \in P \\
G_{\texttt{crash}(b)}(R, p, R, p) & = & 1 & \text{for } R \in \mathit{Regs}(1) \text{ and } p \in P \\
G_{\texttt{crash}(b)}(R, p, R', p') & = & 0 & \text{otherwise}
\end{array}$$

Supposing that $\texttt{Administrator} = \{a_1, a_2\}$, $\texttt{Staff} = \{b_1, b_2\}$ and $\texttt{Contents} = \{c_1, c_2\}$, consider the following marking s, in which there are 3 users:

1						2						3					
0	0	0	1	3	2	2	0	0	0	1	3	0	0	1	2	0	2
a_1	a_2	b_1	b_2	c_1	c_2	a_1	a_2	b_1	b_2	c_1	c_2	a_1	a_2	b_1	b_2	c_1	c_2

The transition $\texttt{crash}(b_2)$ is firable from s in exactly two ways: either for user 1 or for user 3. In the notation of Section 2.2, the latter choice is formalised by having $s_\dagger = s$ and $\iota(1) = 3$. We then have $[\![Reg_{(0,1)}]\!]^3_\iota = \{1, 2\}$, $[\![Reg_{(1,2)}]\!]^3_\iota = \emptyset$, and the instances $[\![F_{\texttt{crash}(b_2)}]\!]^3_\iota$, $[\![G_{\texttt{crash}(b_2)}]\!]^3_\iota$ and $[\![H_{\texttt{crash}(b_2)}]\!]^3_\iota$ are as follows, respectively:

1						2						3					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
a_1	a_2	b_1	b_2	c_1	c_2	a_1	a_2	b_1	b_2	c_1	c_2	a_1	a_2	b_1	b_2	c_1	c_2

	1						2						3						
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	a_1
	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	a_2
	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	b_1
	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	b_2
	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	c_1
	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	c_2
2	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	a_1
	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	a_2
	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	b_1
	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	b_2
	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	c_1
	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	c_2
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	a_1
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	a_2
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	b_1
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	b_2
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	c_1
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	c_2
	a_1	a_2	b_1	b_2	c_1	c_2	a_1	a_2	b_1	b_2	c_1	c_2	a_1	a_2	b_1	b_2	c_1	c_2	

1						2						3					
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
a_1	a_2	b_1	b_2	c_1	c_2	a_1	a_2	b_1	b_2	c_1	c_2	a_1	a_2	b_1	b_2	c_1	c_2

Thus, the vector $(s_\dagger - [\![F_{\texttt{crash}(b_2)}]\!]^3_\iota)[\![G_{\texttt{crash}(b_2)}]\!]^3_\iota + [\![H_{\texttt{crash}(b_2)}]\!]^3_\iota$ that results from the firing equals

1						2						3					
0	0	0	1	3	2	2	0	0	0	1	3	0	0	0	0	0	0
a_1	a_2	b_1	b_2	c_1	c_2	a_1	a_2	b_1	b_2	c_1	c_2	a_1	a_2	b_1	b_2	c_1	c_2

and its **0**-contraction is the marking

1						2					
0	0	0	1	3	2	2	0	0	0	1	3
a_1	a_2	b_1	b_2	c_1	c_2	a_1	a_2	b_1	b_2	c_1	c_2

Many interesting properties of the file system can be formalised as coverability, termination or boundedness properties. For example, that there is never a user who is both an administrator and a staff member amounts to none of the markings $s_{a,b}$ for $a \in \texttt{Administrator}$ and $b \in \texttt{Staff}$ being coverable, where $|s_{a,b}| = 1$, $s_{a,b}(1)(a) = s_{a,b}(1)(b) = 1$, and $s_{a,b}(1)(p) = 0$ for all $p \in P \setminus \{a, b\}$.

3. Reset nets and lossy channel systems

In this section, we first show how Petri data nets can express reset nets, which establishes the dashed inclusion in the diagram in Section 2.4. The translation preserves coverability and termination properties of reset nets.

Secondly, we show that Petri data nets can also express lossy channel systems [2]. The translation provides reductions of the location reachability and termination problems for lossy channel systems to the coverability, termination and boundedness problems for Petri data nets. Thus, the latter three problems will be shown non-primitive recursive: see Theorem 5.1.

Proposition 3.1.

(a) Coverability for reset nets is Turing reducible in polynomial space to coverability for Petri data nets.

(b) Termination for reset nets is reducible in polynomial space to termination for Petri data nets, and to boundedness for Petri data nets.

Proof:
We define a translation from reset nets $\mathcal{N} = \langle P, T, F, G, H\rangle$ to Petri data nets $\widehat{\mathcal{N}} = \langle \hat{P}, \hat{T}, \alpha, \hat{F}, \hat{G}, \hat{H}\rangle$. For each $t \in T$, let s^0_t be a sequence consisting of all $p \in P$ which are reset by t, i.e., such that $G(p, p) = 0$ (each occurring once).

The set of places of $\widehat{\mathcal{N}}$ is formed by adding a place to P: $\hat{P} = P \uplus \{\hat{p}\}$. In $\widehat{\mathcal{N}}$, each place $p \in P$ will store a single token, carrying a datum which represents the place p of $\mathcal{N}$. The place $\hat{p}$ will store as many tokens carrying the datum which represents a place p as there are tokens at p in $\mathcal{N}$. More precisely, for markings m of $\mathcal{N}$ and s of $\widehat{\mathcal{N}}$, we write $m \approx s$ iff for each $p \in P$, there exists $j_p \in [|s|]$ such that:

- $s(j_p)(p) = 1$, $s(j')(p) = 0$ for all $j' \neq j_p$, and

- $s(j_p)(\hat{p}) = m(p)$.

The relation $\approx$ will be a bisimulation between $\mathcal{N}$ and $\widehat{\mathcal{N}}$.

The transitions of $\widehat{\mathcal{N}}$ are pairs of transitions of $\mathcal{N}$ and enumerations of P:

$$\hat{T} = \{\hat{t}_\pi \ : \ t \in T \ \wedge \ [|P|] \stackrel{\pi}{\leftrightarrow} P\}$$

Suppose $m \approx s$, and let π be the enumeration of P such that $\pi^{-1}(p) < \pi^{-1}(p')$ iff $j_p < j_{p'}$. We shall have that:

(i) only transitions of the form $\hat{t}_\pi$ are firable from s;

(ii) $m \xrightarrow{t} m'$ implies $s \xrightarrow{\hat{t}_\pi} s'$ for some $m' \approx s'$;

(iii) $s \xrightarrow{\hat{t}_\pi} s'$ implies $m \xrightarrow{t} m'$ for some $m' \approx s'$.

Consider any $\hat{t}_\pi \in \hat{T}$. We set $\alpha_{\hat{t}_\pi} = |P| + |s_t^0|$. Indices $i \in [|P|]$ will be used to pick data which represent the places of $\mathcal{N}$, and indices $|P| + i$ will be used to pick fresh data (which are greater than all existing data) to simulate the resets of t. Since $\hat{G}_{\hat{t}_\pi} = Id$ is required for $\widehat{\mathcal{N}}$ to be a Petri data net, it remains to define $\hat{F}_{\hat{t}_\pi}$ and $\hat{H}_{\hat{t}_\pi}$ so that (i)–(iii) above are satisfied. Each entry not listed below is set to 0:

$$\begin{array}{rclrcll}
\hat{F}_{\hat{t}_\pi}(i, \pi(i)) & = & 1 & \hat{F}_{\hat{t}_\pi}(i, \hat{p}) & = & F_t(\pi(i)) & (i \in [|P|]) \\
\hat{H}_{\hat{t}_\pi}(i, \pi(i)) & = & 1 & \hat{H}_{\hat{t}_\pi}(i, \hat{p}) & = & H_t(\pi(i)) & (\pi(i) \notin s_t^0) \\
\hat{H}_{\hat{t}_\pi}(|P| + i, s_t^0(i)) & = & 1 & \hat{H}_{\hat{t}_\pi}(|P| + i, \hat{p}) & = & H_t(s_t^0(i)) & (i \in [|s_t^0|])
\end{array}$$

Since any enumeration π of P is storable in polynomial space, we have that polynomial space suffices for the translation.

Given a marking m of $\mathcal{N}$, let s be a marking of $\widehat{\mathcal{N}}$ such that $m \approx s$. For (a), we have by (i)–(iii) above that a given marking m' is coverable from m in $\mathcal{N}$ iff some minimal s' such that $m' \approx s'$ is coverable from s in $\widehat{\mathcal{N}}$. For the first half of (b), we have by (i)–(iii) above that $\mathcal{N}$ terminates from m iff $\widehat{\mathcal{N}}$ terminates from s. For the second half, let $\widehat{\mathcal{N}}'$ be obtained from $\widehat{\mathcal{N}}$ (in logarithmic space) by adding a place $\hat{p}'$ and ensuring that each transition increases the number of tokens at $\hat{p}'$. Let s' be an arbitrary extension of s to place $\hat{p}'$. We have that $\mathcal{N}$ terminates from m iff $\widehat{\mathcal{N}}'$ is bounded from s'. □

A *lossy channel system* is a tuple $\mathcal{S} = \langle Q, C, \Sigma, \Delta \rangle$, where Q is a finite set of locations, C is a finite set of channels, Σ is a finite alphabet, and $\Delta \subseteq Q \times C \times \{!, ?\} \times \Sigma \times Q$ is a set of transitions.

A state of $\mathcal{S}$ is a pair $\langle q, w \rangle$, where $q \in Q$ and $w : C \to \Sigma^*$. For each $c \in C$, the word $w(c)$ is the contents of channel c at state $\langle q, w \rangle$.

To define computation steps, we first define perfect computation steps, which either write a letter to the beginning of a channel, or read a letter from the end of a channel. For states $\langle q_1, w_1 \rangle$ and $\langle q_2, w_2 \rangle$, we write $\langle q_1, w_1 \rangle \rightarrow_{perf} \langle q_2, w_2 \rangle$ iff there exist $c \in C$ and $a \in \Sigma$ such that:

- either $\langle q_1, c, !, a, q_2 \rangle \in \Delta$ and $w_2 = w_1[c \mapsto a(w_1(c))]$,
- or $\langle q_1, c, ?, a, q_2 \rangle \in \Delta$ and $w_1 = w_2[c \mapsto (w_2(c))a]$.

Let $\sqsubseteq$ denote the "subword" well-quasi-ordering on Σ^*, obtained by lifting the equality relation on Σ (see Proposition 2.1). For example, we have $abba \sqsubseteq abracadabra$. For states $\langle q, w\rangle$ and $\langle q', w'\rangle$, we write $\langle q, w\rangle \sqsupseteq \langle q', w'\rangle$ iff $q = q'$ and $w(c) \sqsupseteq w'(c)$ for all $c \in C$, i.e., $\langle q', w'\rangle$ is obtained from $\langle q, w\rangle$ by losing zero or more letters.

A computation step $\langle q, w\rangle \to \langle q', w'\rangle$ of $\mathcal{S}$ consists of zero or more losses, followed by a perfect computation step, followed by zero or more losses. Thus, the $\to$ relation is defined by composing the $\to_{perf}$ and $\sqsupseteq$ relations: $\to = \sqsupseteq \to_{perf} \sqsupseteq$.

The following are two key decision problems for lossy channel systems:

Location reachability: Given a lossy channel system, a state $\langle q, w\rangle$ and a location q', to decide whether some state $\langle q', w'\rangle$ is reachable from $\langle q, w\rangle$.

Termination: Given a lossy channel system, and a state $\langle q, w\rangle$, to decide whether all computations from $\langle q, w\rangle$ are finite.

The proof of the next result will be illustrated in Example 3.1.

Proposition 3.2.

(a) Location reachability for lossy channel systems is reducible in logarithmic space to coverability for Petri data nets.

(b) Termination for lossy channel systems is reducible in logarithmic space to termination for Petri data nets, and to boundedness for Petri data nets.

Proof:
Given a lossy channel system $\mathcal{S} = \langle Q, C, \Sigma, \Delta\rangle$, we define a Petri data net $\mathcal{N}_\mathcal{S} = \langle P, T, \alpha, F, G, H\rangle$ as follows. We shall have that $\mathcal{N}_\mathcal{S}$ is computable in logarithmic space.

Let $P = Q \uplus C \uplus (C \times \Sigma)$. States $\langle q, w\rangle$ of $\mathcal{S}$ will be represented by markings $s \in Seq(\mathbb{N}^P \setminus \{\mathbf{0}\})$ as follows. At places in Q, there will be one token, which is at q, and which carries a datum d which is minimal in s. For each $c \in C$, where $w(c) = a_1 \cdots a_k$, there will be data $d \preceq d^c_1 \prec \cdots \prec d^c_k \prec d^c_{k+1}$ such that:

- place c contains one token which carries d^c_{k+1};
- for each $a \in \Sigma$, place $\langle c, a\rangle$ contains one token carrying d^c_i for each $i \in [k]$ with $a_i = a$, and possibly some tokens carrying data not smaller than d^c_{k+1}.

Formally, we write $\langle q, w\rangle \approx s$ iff:

- $s(1)(q) = 1$, and $s(j)(q') = 0$ whenever either $j > 1$ or $q' \in Q \setminus \{q\}$;
- for each $c \in C$, where $w(c) = a_1 \cdots a_k$, there exist $1 \le j^c_1 < \cdots < j^c_k < j^c_{k+1}$ such that $s(j^c_{k+1})(c) = 1$, $s(j')(c) = 0$ for all $j' \neq j^c_{k+1}$, and for each $1 \le j' < j^c_{k+1}$ and $a' \in \Sigma$, we have

$$s(j')(c, a') = \begin{cases} 1, & \text{if there exists } i \in [k] \text{ with } j' = j^c_i \text{ and } a' = a_i \\ 0, & \text{otherwise} \end{cases}$$

For each read transition of $\mathcal{S}$, there will be two transitions of $\mathcal{N}_\mathcal{S}$, depending on whether the datum that points to the letter read is minimal or not:

$$\begin{aligned} T \;=\; & \{\langle q_1, c, !, a, q_2\rangle \,:\, \langle q_1, c, !, a, q_2\rangle \in \Delta\} \cup \\ & \{\langle q_1, c, ?, a, q_2\rangle^1, \langle q_1, c, ?, a, q_2\rangle^{>1} \,:\, \langle q_1, c, ?, a, q_2\rangle \in \Delta\} \end{aligned}$$

When defining α_t, F_t and H_t for $t \in T$ below, we show only entries which are distinct from 0. Since $\mathcal{N}_\mathcal{S}$ is a Petri data net, we have $G_t = Id$ for each $t \in T$.

We shall have that, in computations of $\mathcal{N}_\mathcal{S}$, losses can happen only when reads are performed, but that will be sufficient for the result we are proving. Losses will occur when, in order to read from a channel c, the letter to be read is found using a datum d' which is smaller than the datum that points to the last letter in c, and then the datum at place c is replaced by d'. (Observe that, in data nets, we cannot specify that a transition be firable from a marking only if the latter contains no data which is between two particular data. Otherwise, perfect channel systems which are Turing-powerful would be expressible.)

Writes are performed using a new minimal datum:

$$\begin{aligned} \alpha_{\langle q_1,c,!,a,q_2\rangle} &= 2 & H_{\langle q_1,c,!,a,q_2\rangle}(1, q_2) &= 1 \\ F_{\langle q_1,c,!,a,q_2\rangle}(2, q_1) &= 1 & H_{\langle q_1,c,!,a,q_2\rangle}(1, \langle c, a\rangle) &= 1 \end{aligned}$$

Reads from a channel c of a letter to which the minimal datum d points check that place c contains a greater datum, which is then replaced by d:

$$\begin{aligned} F_{\langle q_1,c,?,a,q_2\rangle^1}(1, q_1) &= 1 & \alpha_{\langle q_1,c,?,a,q_2\rangle^1} &= 2 \\ F_{\langle q_1,c,?,a,q_2\rangle^1}(2, c) &= 1 & H_{\langle q_1,c,?,a,q_2\rangle^1}(1, q_2) &= 1 \\ F_{\langle q_1,c,?,a,q_2\rangle^1}(1, \langle c, a\rangle) &= 1 & H_{\langle q_1,c,?,a,q_2\rangle^1}(1, c) &= 1 \end{aligned}$$

The remaining reads from a channel c of a letter a decrease the datum at place c to a value which is not minimal and which points to an occurrence of a in c:

$$\begin{aligned} F_{\langle q_1,c,?,a,q_2\rangle^{>1}}(1, q_1) &= 1 & \alpha_{\langle q_1,c,?,a,q_2\rangle^{>1}} &= 3 \\ F_{\langle q_1,c,?,a,q_2\rangle^{>1}}(3, c) &= 1 & H_{\langle q_1,c,?,a,q_2\rangle^{>1}}(1, q_2) &= 1 \\ F_{\langle q_1,c,?,a,q_2\rangle^{>1}}(2, \langle c, a\rangle) &= 1 & H_{\langle q_1,c,?,a,q_2\rangle^{>1}}(2, c) &= 1 \end{aligned}$$

Now, the definition of $\mathcal{N}_\mathcal{S}$ ensures that the $\approx$ relation is an inverse simulation: whenever $\langle q, w\rangle \approx s$ and $s \to s'$, there exists $\langle q', w'\rangle$ such that $\langle q', w'\rangle \approx s'$ and $\langle q, w\rangle \to \langle q', w'\rangle$.

We write $\langle q, w\rangle \sqsubseteq \approx s$ iff there exists $\langle q^\dagger, w^\dagger\rangle$ such that $\langle q, w\rangle \sqsubseteq \langle q^\dagger, w^\dagger\rangle$ and $\langle q^\dagger, w^\dagger\rangle \approx s$. It is straightforward to check that the $\sqsubseteq \approx$ relation is a simulation: whenever $\langle q, w\rangle \sqsubseteq \approx s$ and $\langle q, w\rangle \to \langle q', w'\rangle$, there exists s' such that $\langle q', w'\rangle \sqsubseteq \approx s'$ and $s \to s'$.

To establish (a), given a state $\langle q, w\rangle$ and a location q' of $\mathcal{S}$, let s be such that $\langle q, w\rangle \approx s$, and let s' be such that $|s'| = 1$, $s'(1)(q') = 1$, and $s'(1)(p) = 0$ for all $p \in P \setminus \{q'\}$. By the properties above, we have that some state $\langle q', w'\rangle$ is reachable from $\langle q, w\rangle$ iff some marking $s'' \geq s'$ is reachable from s.

For the termination part of (b), if s is such that $\langle q, w\rangle \approx s$, then $\mathcal{S}$ has an infinite computation from $\langle q, w\rangle$ iff $\mathcal{N}_\mathcal{S}$ has an infinite computation from s. For the boundedness part, we modify $\mathcal{N}_\mathcal{S}$ by adding an auxiliary place and ensuring that each transition increases the number of tokens at that place. □

Example 3.1. Consider the lossy channel system $\mathcal{S}$ with three locations q_1, q_2 and q_3, one channel c, two letters a and b, and the following three transitions:

$$\langle q_1, c, !, a, q_2\rangle \qquad \langle q_2, c, !, b, q_3\rangle \qquad \langle q_3, c, ?, b, q_1\rangle$$

Thus, $\mathcal{S}$ attempts to write a, write b and read b repeatedly. Since each a can be lost before the next read of b, $\mathcal{S}$ has an infinite computation from state $\langle q_1, [c \mapsto \varepsilon]\rangle$.

The following is a marking of $\mathcal{N}_\mathcal{S}$ which corresponds to $\langle q_1, [c \mapsto \varepsilon]\rangle$:

	q_1	q_2	q_3	c	$\langle c, a\rangle$	$\langle c, b\rangle$
1	1	0	0	1	0	0

Only transition $\langle q_1, c, !, a, q_2\rangle$ of $\mathcal{N}_\mathcal{S}$ can be fired, and in a unique way. The resulting marking is

	q_1	q_2	q_3	c	$\langle c, a\rangle$	$\langle c, b\rangle$
1	0	1	0	0	1	0
2	0	0	0	1	0	0

Similarly, only $\langle q_2, c, !, b, q_3\rangle$ is firable next, and it results in

	q_1	q_2	q_3	c	$\langle c, a\rangle$	$\langle c, b\rangle$
1	0	0	1	0	0	1
2	0	0	0	0	1	0
3	0	0	0	1	0	0

Now, only $\langle q_3, c, ?, b, q_1\rangle^1$ is firable. It produces the marking below, which again corresponds to state $\langle q_1, [c \mapsto \varepsilon]\rangle$, but which contains a "junk" datum that has been left by the loss of letter a:

	q_1	q_2	q_3	c	$\langle c, a\rangle$	$\langle c, b\rangle$
1	1	0	0	1	0	0
2	0	0	0	0	1	0

4. Decidability

The following two lemmas will be used in the proof of Theorem 4.1 below. The first one, due to Valk and Jantzen, provides a sufficient condition for computability of finite bases of upwards-closed sets of fixed-length tuples of natural numbers. The second lemma shows that, for computing a pred-basis of the upward closure of a marking of a data net, it suffices to consider markings up to a certain computable length.

Lemma 4.1. ([24])
Suppose B is a finite set. A finite basis of an upwards-closed set $V \subseteq \mathbb{N}^B$ is computable iff it is decidable, given any $v \in \mathbb{N}_\omega^B$, whether $V \cap {\downarrow}\{v\} \neq \emptyset$.

For a transition system $\langle S, \rightarrow\rangle$ and $S' \subseteq S$, we write $Pred(S')$ for $\{s \in S \,:\, \exists s' \in S' \cdot s \rightarrow s'\}$. If transitions are labelled by $t \in T$, we write $Pred_t(S')$ for $\{s \in S \,:\, \exists s' \in S' \cdot s \xrightarrow{t} s'\}$.

Lemma 4.2. Given a data net $\mathcal{N}$, a transition t of $\mathcal{N}$, and a marking s' of $\mathcal{N}$, a natural number L is computable, such that whenever $s \in Pred_t(\uparrow\{s'\})$ and $|s| > L$, there exists $\bar{s} \leq s$ with $\bar{s} \in Pred_t(\uparrow\{s'\})$ and $|\bar{s}| \leq L$.

Proof:
Suppose $\mathcal{N} = \langle P, T, \alpha, F, G, H\rangle$, and let

$$L = \alpha_t + |s'| + (\alpha_t + 1) \times (2^{|P|} - 1) \times M$$

where $M = \max\{s'(i)(p) \,:\, i \in [|s'|] \wedge p \in P\}$.

Consider $s \in Pred_t(\uparrow\{s'\})$ with $|s| > L$. For some $s_\dagger$, ι and $s'' \geq s'$, we have $s \xrightarrow{t,s_\dagger,\iota} s''$. Let $s''_\dagger = (s_\dagger - [\![F_t]\!]_\iota^{|s_\dagger|})[\![G_t]\!]_\iota^{|s_\dagger|} + [\![H_t]\!]_\iota^{|s_\dagger|}$. Since s'' is the $\mathbf{0}$-contraction of $s''_\dagger$, there exists an increasing $\iota' : [|s'|] \rightarrow [|s_\dagger|]$ such that $s'(i) \leq s''_\dagger(\iota'(i))$ for all $i \in [|s'|]$.

For each nonempty $P_+ \subseteq P$, let

$$s_\dagger^{P_+} = \{i \in [|s_\dagger|] \,:\, \forall p \in P \cdot s_\dagger(i)(p) > 0 \Leftrightarrow p \in P_+\}$$

Since $|s_\dagger| \geq |s|$, there exist $0 \leq j \leq \alpha_t$ and nonempty $P_+ \subseteq P$ such that $|I_j^{P_+}| > M$, where $I_j^{P_+} = ([\![Reg_{(j,j+1)}]\!]_\iota^{|s_\dagger|} \setminus Range(\iota')) \cap s_\dagger^{P_+}$.

Pick an index $i_\dagger^1 \in I_j^{P_+}$ of $s_\dagger$, and let $i^1 \in [|s|]$ be the corresponding index of s. Let $\tau_\dagger$ be the increasing mapping $[|s_\dagger| - 1] \rightarrow [|s_\dagger|]$ with $i_\dagger^1 \notin Range(\tau_\dagger)$, and τ be the increasing mapping $[|s| - 1] \rightarrow [|s|]$ with $i^1 \notin Range(\tau)$. Then let $s_\dagger^1$ (resp., s^1) be obtained from $s_\dagger$ (resp., s) by removing the entry $i_\dagger^1$ (resp., i^1), $\iota_1 = \tau_\dagger^{-1} \circ \iota$, and $s''^1_\dagger = (s_\dagger^1 - [\![F_t]\!]_{\iota_1}^{|s_\dagger^1|})[\![G_t]\!]_{\iota_1}^{|s_\dagger^1|} + [\![H_t]\!]_{\iota_1}^{|s_\dagger^1|}$. By the definition of $I_j^{P_+}$ and $|I_j^{P_+}| > M$, we have that $s''^1_\dagger(i)(p) \geq M$ whenever $s''^1_\dagger(i)(p) \neq s''_\dagger(\tau_\dagger(i))(p)$. Hence, $s''^1_\dagger \geq s'$, so $s^1 \in Pred_t(\uparrow\{s'\})$.

By repeating the above, we obtain $s \geq s^1 \geq s^2 \geq \cdots s^{|s|-L} \in Pred_t(\uparrow\{s'\})$ such that $|s^k| = |s| - k$ for all k. Setting $\bar{s} = s^{|s|-L}$ completes the proof. □

Theorem 4.1.

(a) Coverability and termination for data nets are decidable.

(b) Boundedness for transfer data nets is decidable.

Proof:
Suppose $\mathcal{N} = \langle P, T, \alpha, F, G, H\rangle$ is a data net. By Propositions 2.1, 2.2 and 2.6, the transition system of $\mathcal{N}$ is finitely-branching and well-structured with strong compatibility, and also with strict compatibility if $\mathcal{N}$ is transfer (using the terminology of [12]). Moreover, $\leq$ between markings of $\mathcal{N}$ is a decidable partial ordering, and $Succ(s) = \{s' \,:\, s \rightarrow s'\}$ is computable for markings s. Hence, termination for data nets and boundedness for transfer data nets are decidable by [12, Theorems 4.6 and 4.11].

To establish decidability of coverability by [12, Theorem 3.6], it suffices to show that, given any $t \in T$ and a marking s', a finite basis of $Pred_t(\uparrow\{s'\})$ is computable. (By Proposition 2.6 (a), $Pred_t(\uparrow\{s'\})$ is upwards-closed.)

First, we compute L as in Lemma 4.2. For any $0 \leq l \leq L$, increasing $\eta : [l] \to [l_\dagger]$ and increasing $\iota : [\alpha_t] \to [l_\dagger]$ such that $[l_\dagger] = Range(\eta) \cup Range(\iota)$, let

$$Pred^l_{t,\eta,\iota}(\uparrow\{s'\}) = \{s \,:\, l = |s| \,\wedge\, \exists s'' \geq s' \cdot s \xrightarrow{t,\eta,\iota} s''\}$$

where $s \xrightarrow{t,\eta,\iota} s''$ means that $s \xrightarrow{t,s_\dagger,\iota} s''$ for some $s_\dagger$ such that $Range(\eta) = \{j \,:\, s_\dagger(j) \neq \mathbf{0}\}$ (necessarily, $l_\dagger = |s_\dagger|$). From the definition of transition firing, we have that $s \xrightarrow{t,s_\dagger,\iota} s''$ iff $s_\dagger \geq [\![F_t]\!]^{l_\dagger}_\iota$ and s'' is the $\mathbf{0}$-contraction of $(s_\dagger - [\![F_t]\!]^{l_\dagger}_\iota)[\![G_t]\!]^{l_\dagger}_\iota + [\![H_t]\!]^{l_\dagger}_\iota$. Hence, each $Pred^l_{t,\eta,\iota}(\uparrow\{s'\})$ is an upwards-closed subset of $\mathbb{N}^{P\times[l]}$. By Lemma 4.2, it remains to compute a finite basis of each $Pred^l_{t,\eta,\iota}(\uparrow\{s'\})$.

Suppose that l, η and ι are as above. Given any $s \in \mathbb{N}^{P\times[l]}_\omega$, we have as in [11] that $Pred^l_{t,\eta,\iota}(\uparrow\{s'\}) \cap \downarrow\{s\} \neq \emptyset$ iff $s_\dagger \geq [\![F_t]\!]^{l_\dagger}_\iota$ and $s'' \geq s'$, where $s_\dagger$ is the $\mathbf{0}$-expansion of s such that $l_\dagger = |s_\dagger|$ and $Range(\eta) = \{j \,:\, s_\dagger(j) \neq \mathbf{0}\}$, s'' is the $\mathbf{0}$-contraction of $(s_\dagger - [\![F_t]\!]^{l_\dagger}_\iota)[\![G_t]\!]^{l_\dagger}_\iota + [\![H_t]\!]^{l_\dagger}_\iota$, and the required operations are extended to ω by taking limits: $\omega \geq n$, $\omega + n = n + \omega = \omega + \omega = \omega$, $\omega - n = \omega$, $0 \times \omega = 0$, and $n \times \omega = \omega$ for $n > 0$. Therefore, by Lemma 4.1, a finite basis of $Pred^l_{t,\eta,\iota}(\uparrow\{s'\})$ is computable. □

5. Hardness

Theorem 5.1. Coverability, termination and boundedness for Petri data nets are not primitive recursive.

Proof:
As shown in [23], location reachability and termination for lossy channel systems are not primitive recursive. It remains to apply Proposition 3.2. □

Theorem 5.2. Coverability, termination and boundedness for unordered Petri data nets are not elementary.

Proof:
For $k \in \mathbb{N}$, the *tetration* operation $a \Uparrow k$ is defined by $a \Uparrow 0 = 1$ and $a \Uparrow (k+1) = a^{a \Uparrow k}$. We shall establish that the three verification problems are not elementary by showing that, given a deterministic machine $\mathcal{M}$ of size n with finite control and two $2 \Uparrow n$-bounded counters, an unordered Petri data net $\mathcal{N}_\mathcal{M}$ which simulates $\mathcal{M}$ is constructible in logarithmic space.

We consider machines $\mathcal{M}$ which are tuples $\langle Q, q_I, q_F, \delta\rangle$ such that:

- Q is a finite set of states, $q_I \in Q$ is the initial state, and $q_F \in Q$ is the final state;
- $\delta : Q \to \{\mathsf{inc}, \mathsf{dec}\} \times \{1, 2\} \times Q \times Q$ is the transition function.

If the size of $\mathcal{M}$ is n, the counters of $\mathcal{M}$ are bounded by $2 \Uparrow n$. Thus, a configuration of $\mathcal{M}$ is a tuple $\langle q, v_1, v_2\rangle$ such that $q \in Q$ and $v_1, v_2 \in \{0, \ldots, (2 \Uparrow n) - 1\}$. Every configuration $\langle q, v_1, v_2\rangle$ of $\mathcal{M}$ has a unique successor which is defined as follows, where we consider only counter 1:

- if $\delta(q) = \langle \mathsf{inc}, 1, q', q'' \rangle$ and $v_1 < (2 \Uparrow n) - 1$, the successor is $\langle q', v_1 + 1, v_2 \rangle$;
- if $\delta(q) = \langle \mathsf{inc}, 1, q', q'' \rangle$ and $v_1 = (2 \Uparrow n) - 1$, the successor is $\langle q'', v_1, v_2 \rangle$;
- if $\delta(q) = \langle \mathsf{dec}, 1, q', q'' \rangle$ and $v_1 > 0$, the successor is $\langle q', v_1 - 1, v_2 \rangle$;
- if $\delta(q) = \langle \mathsf{dec}, 1, q', q'' \rangle$ and $v_1 = 0$, the successor is $\langle q'', v_1, v_2 \rangle$.

Given such a machine $\mathcal{M}$, the halting problem is to decide whether the final state (i.e., some configuration $\langle q_F, v_1, v_2 \rangle$) is reachable from the initial configuration $\langle q_I, 0, 0 \rangle$. By standard results on simulating Turing machines by counter machines, that decision problem is not elementary (cf., e.g., [18]).

For a machine $\mathcal{M}$ of size n as above, let $\mathcal{N}_\mathcal{M}$ be an unordered Petri data net which is constructed as follows. In fact, $\mathcal{N}_\mathcal{M}$ will be able to simulate operations on $2n$ counters C_k and C'_k for $k \in [n]$. C_n and C'_n are the two counters of $\mathcal{M}$, and for each $k \in [n]$, C_k and C'_k are $2 \Uparrow k$-bounded. For each $k < n$, simulations by $\mathcal{N}_\mathcal{M}$ of operations on C_{k+1} and C'_{k+1} will use operations on C_k and C'_k.

The set of places of $\mathcal{N}_\mathcal{M}$ is

$$start \uplus P_\mathcal{M} \uplus Q \uplus \{ready_k : k \in [n]\} \uplus$$
$$\{0_D, 1_D, scratch_D, lock_D, checked_D, unchecked_D : D \in \{C_k, C'_k : k \in [n]\}\}$$

where $P_\mathcal{M}$ will be defined implicitly and consists of places for controlling $\mathcal{N}_\mathcal{M}$. Let $s_{I,\mathcal{M}}$ be a marking in which there is one token at place q_I, there is one token at place $start$, and all other places are empty.

The transitions of $\mathcal{N}_\mathcal{M}$ will be constructed so that (i)–(iv) below are satisfied. We write $s \to_\surd s'$ (resp., $s \to_\times s'$) iff $s \to s'$ and place $ready_n$ is nonempty (resp., empty) in s'. Hence, $s \to^*_\times \to_\surd s'$ means that s' is reachable from s by a nonempty sequence of transitions for which $ready_n$ is empty in every intermediate marking, and that $ready_n$ is nonempty in s'. As another example, $s \not\to^\omega_\times$ means that there does not exist an infinite sequence of transitions from s for which $ready_n$ is empty in every intermediate marking.

(i) For every s reachable from $s_{I,\mathcal{M}}$, there is one token within Q, and at most one token within $\{ready_k : k \in [n]\}$.

(ii) For every s reachable from $s_{I,\mathcal{M}}$, every $k \in [n]$ such that $ready_{k'}$ is nonempty in s for some $k' \geq k$, and every $D \in \{C_k, C'_k\}$, we have that a value $v_s(D) \in \{0, \ldots, (2 \Uparrow k) - 1\}$ is encoded in s as follows. Moreover, if $k' > k$ then $v_s(D) = 0$.

- $scratch_D$, $lock_D$ and $checked_D$ are empty, and $unchecked_D$ contains exactly $2 \Uparrow (k-1)$ tokens and they carry mutually distinct data;
- for each $i \in [2 \Uparrow (k-1)]$, if the i-th bit of $v_s(D)$ is $b \in \{0, 1\}$, then for some datum d carried by a token at place $unchecked_D$, the number of tokens at b_D which carry d is i, and the number of tokens at $(1-b)_D$ which carry d is 0;
- each datum carried by a token at 0_D or 1_D is carried by some token at $unchecked_D$.

Whenever $s_{I,\mathcal{M}} \to^* \to_\surd s$ let $c(s)$ be the configuration $\langle q, v_s(C_n), v_s(C'_n) \rangle$ of $\mathcal{M}$, where q is nonempty in s.

(iii) We have $s_{I,\mathcal{M}} \not\to^\omega_\times$ and:

– there exists $s_{I,\mathcal{M}} \rightarrow^*_\times \rightarrow_\surd s$ such that $c(s) = \langle q_I, 0, 0\rangle$;
– for all $s_{I,\mathcal{M}} \rightarrow^*_\times \rightarrow_\surd s$, $c(s) = \langle q_I, 0, 0\rangle$.

(iv) Whenever $s_{I,\mathcal{M}} \rightarrow^* \rightarrow_\surd s$, we have $s \not\rightarrow^\omega_\times$ and:

– there exists $s \rightarrow^*_\times \rightarrow_\surd s'$ such that $c(s')$ is the successor of $c(s)$ in $\mathcal{M}$;
– for all $s \rightarrow^*_\times \rightarrow_\surd s'$, $c(s')$ is the successor of $c(s)$ in $\mathcal{M}$.

To satisfy (i)–(iv), it suffices to simulate the following operations on $D \in \{C_k, C'_k\}$ for $k \in [n]$:

- $\mathsf{setup}(D)$, which assumes that the places 0_D, 1_D, $scratch_D$, $lock_D$, $checked_D$ and $unchecked_D$ are empty, and sets them up so that value 0 is encoded;
- $\mathsf{inc}(D)$ (resp., $\mathsf{dec}(D)$), which increments (resp., decrements) D, and can be executed iff the resulting value is in the range $\{0, \ldots, (2 \Uparrow k) - 1\}$;
- $\mathsf{iszero}(D)$ (resp., $\mathsf{ismax}(D)$), which does not change the value of D, but can be executed iff it equals 0 (resp., $(2 \Uparrow k) - 1$).

When started from $s_{I,\mathcal{M}}$, $\mathcal{N}_\mathcal{M}$ will first perform:

$\mathsf{setup}(C_1)$; $\mathsf{setup}(C'_1)$; move a token from $start$ to $ready_1$;
$\mathsf{setup}(C_2)$; $\mathsf{setup}(C'_2)$; move a token from $ready_1$ to $ready_2$;
...
$\mathsf{setup}(C_n)$; $\mathsf{setup}(C'_n)$; move a token from $ready_{n-1}$ to $ready_n$

To present simulations of the counter operations, we employ pseudo-code as above which is straightforward to implement using the places $P_\mathcal{M}$. The simulations for C_{k+1} and C'_{k+1} may invoke only operations on C_k and C'_k, so recursion depth is bounded by n.

Since counters C_1 and C'_1 are 2-bounded, the five operations on them are trivial to simulate.

Suppose $k \in \{1, \ldots, n-1\}$ and $D \in \{C_{k+1}, C'_{k+1}\}$. Table 1 contains pseudo-code for $\mathsf{setup}(D)$. In addition to the emptiness of 0_D, 1_D, $scratch_D$, $lock_D$, $checked_D$ and $unchecked_D$, we can assume that $ready_k$ is nonempty and that C_k and C'_k have value 0. The first outer loop uses C_k to iterate through the $2 \Uparrow k$ binary digits of D. For each digit, a representation of its value 0 is set up by choosing nondeterministically a datum d, performing the first inner loop to check that d is fresh (i.e., distinct from each datum which currently is carried by a token at $unchecked_D$), and performing the second inner loop to restore the contents of $unchecked_D$ and to put the correct number of tokens carrying d onto 0_D. The second outer loop ensures that, at the end of $\mathsf{setup}(D)$, C_k and C'_k have value 0.

Among the remaining operations, simulating $\mathsf{inc}(D)$ and $\mathsf{dec}(D)$ is harder than simulating $\mathsf{iszero}(D)$ and $\mathsf{ismax}(D)$. By symmetry, we consider only $\mathsf{inc}(D)$. The first outer loop in Table 2 uses C_k to iterate through the binary digits of D. It can only terminate by choosing nondeterministically a digit with value 0. The latter is altered to 1, but values of all previous digits are unchanged. The second outer loop iterates through the remaining digits of D, it checks that their values are 1, and alters them to 0. The inner loops ensure that if the first two outer loops have terminated, then for each $i \in [2 \Uparrow k]$, it must have been the i-th binary digit of D which was processed during the i-th outer iteration. The third outer loop completes a representation of the new value of D by transferring the contents of place $checked_D$ to place $unchecked_D$.

Table 1. Simulating $\mathsf{setup}(D)$

repeat
{ guess a datum d and put a token carrying d onto $lock_D$;
while not $\mathsf{iszero}(C_k)$ **do**
{ $\mathsf{dec}(C_k)$; $\mathsf{inc}(C'_k)$;
move a token carrying some $d' \neq d$ from $unchecked_D$ to $checked_D$ };
while not $\mathsf{iszero}(C'_k)$ **do**
{ $\mathsf{dec}(C'_k)$; $\mathsf{inc}(C_k)$;
move a token from $checked_D$ to $unchecked_D$;
put a token carrying d onto 0_D };
put a token carrying d onto 0_D;
move the token from $lock_D$ to $unchecked_D$;
if $\mathsf{ismax}(C_k)$ **then** exit **else** $\mathsf{inc}(C_k)$ };
while not $\mathsf{iszero}(C_k)$ **do** $\mathsf{dec}(C_k)$

Table 2. Simulating $\mathsf{inc}(D)$

move a token from $ready_{k+1}$ to $ready_k$;
repeat
{ move a token carrying some d from $unchecked_D$ to $lock_D$, and guess $b \in \{0,1\}$;
if $b = 0$ **then** guess $b' \in \{0,1\}$ **else** let $b' = b$;
while not $\mathsf{iszero}(C_k)$ **do**
{ $\mathsf{dec}(C_k)$; $\mathsf{inc}(C'_k)$; move a token carrying d from b_D to $scratch_D$ };
move a token carrying d from b_D to $scratch_D$;
while not $\mathsf{iszero}(C'_k)$ **do**
{ $\mathsf{dec}(C'_k)$; $\mathsf{inc}(C_k)$; move a token carrying d from $scratch_D$ to b'_D };
move a token carrying d from $scratch_D$ to b'_D;
move the token from $lock_D$ to $checked_D$;
if $b = 0$ and $b' = 1$ **then** exit **else** $\mathsf{inc}(C_k)$ };
while not $\mathsf{ismax}(C_k)$ **do**
{ $\mathsf{inc}(C_k)$; move a token carrying some d from $unchecked_D$ to $lock_D$;
while not $\mathsf{iszero}(C_k)$ **do**
{ $\mathsf{dec}(C_k)$; $\mathsf{inc}(C'_k)$; move a token carrying d from 1_D to $scratch_D$ };
move a token carrying d from 1_D to $scratch_D$;
while not $\mathsf{iszero}(C'_k)$ **do**
{ $\mathsf{dec}(C'_k)$; $\mathsf{inc}(C_k)$; move a token carrying d from $scratch_D$ to 0_D };
move a token carrying d from $scratch_D$ to 0_D;
move the token from $lock_D$ to $checked_D$ };
repeat
{ move a token from $checked_D$ to $unchecked_D$;
if $\mathsf{iszero}(C_k)$ **then** exit **else** $\mathsf{dec}(C_k)$ };
move a token from $ready_k$ to $ready_{k+1}$

Now, by (i)–(iv), that $\mathcal{M}$ halts is equivalent to $\mathcal{N}_\mathcal{M}$ being able to cover from $s_{I,\mathcal{M}}$ a marking in which there is a token at place q_F and a token at place $ready_n$. By ensuring that $\mathcal{N}_\mathcal{M}$ has no transitions from markings in which q_F and $ready_n$ are nonempty, that $\mathcal{M}$ halts becomes equivalent to $\mathcal{N}_\mathcal{M}$ terminating. To reduce the halting problem for $\mathcal{M}$ to the boundedness problem for $\mathcal{N}_\mathcal{N}$, it suffices to modify further the construction of the latter by adding a place whose number of tokens increases with each transition.

It remains to observe that $\mathcal{N}_\mathcal{M}$ can be constructed in space which is logarithmic in n (i.e., the size of $\mathcal{M}$). That is because recursion depth in the simulations above is bounded by n and transition arities in $\mathcal{N}_\mathcal{M}$ are at most 2, so that the number of places, the number of transitions and the sizes of transitions in $\mathcal{N}_\mathcal{M}$ are polynomial in n. □

6. Concluding remarks

We have answered questions (1) and (2) posed in Section 1. As far as we are aware, Section 5 contains the first nontrivial lower bounds on complexity of decidable problems for extensions of Petri nets by infinite data domains.

The results obtained and their proofs show that data nets are a succinct unifying formalism which is close to the underlying semantic structures, and thus a useful platform for theoretical investigations.

The proof of Theorem 4.1 does not provide precise upper bounds on complexity. It should be investigated whether upper bounds which match the lower bounds in the proofs of Theorems 5.1 and 5.2 are obtainable. In particular, are coverability, termination and boundedness for unordered Petri data nets primitive recursive?

Let us say that a data net is l, m-*safe* iff each place other than some l places never contains more than m tokens. It is not difficult to tighten the proofs of Theorems 5.1 and 5.2 to obtain that coverability, termination and boundedness are not primitive recursive for $1, 1$-safe Petri data nets, and not elementary for $2, 1$-safe unordered Petri data nets. That leaves open whether we have non-elementarity for $1, 1$-safe unordered Petri data nets. That class suffices for expressing polymorphic systems with one array of type $\langle X, = \rangle \to \langle Y, = \rangle$ without whole-array operations [16, 15].

We are grateful to Alain Finkel for a helpful discussion.

References

[1] Abdulla, P. A., Delzanno, G.: Constrained Multiset Rewriting, *AVIS '06*, ENTCS, to appear.

[2] Abdulla, P. A., Jonsson, B.: Verifying Programs with Unreliable Channels, *Inf. Comput.*, **127**(2), 1996, 91–101.

[3] Abdulla, P. A., Jonsson, B.: Model checking of systems with many identical timed processes, *Theor. Comput. Sci.*, **290**(1), 2003, 241–264.

[4] Abdulla, P. A., Nylén, A.: Timed Petri Nets and BQOs, *ICATPN*, number 2075 in LNCS, Springer, 2001.

[5] Bozzano, M., Delzanno, G.: Automatic Verification of Invalidation-based Protocols, *CAV*, number 2404 in LNCS, Springer, 2002.

[6] Bozzano, M., Delzanno, G.: Beyond Parameterized Verification, *TACAS*, number 2280 in LNCS, Springer, 2002.

[7] Delzanno, G.: An Assertional Language for Systems Parametric in Several Dimensions, *VEPAS*, number 50 in ENTCS, 2001.

[8] Delzanno, G.: *Constraint Multiset Rewriting*, Technical Report DISI-TR-05-08, Università di Genova, 2005, Extends [7, 6, 5].

[9] Dufourd, C., Finkel, A., Schnoebelen, P.: Reset Nets Between Decidability and Undecidability, *ICALP*, number 1443 in LNCS, Springer, 1998.

[10] Esparza, J., Nielsen, M.: Decidability Issues for Petri Nets – a survey, *Bull. EATCS*, **52**, 1994, 244–262.

[11] Finkel, A., McKenzie, P., Picaronny, C.: A well-structured framework for analysing Petri net extensions, *Inf. Comput.*, **195**(1–2), 2004, 1–29.

[12] Finkel, A., Schnoebelen, P.: Well-structured transition systems everywhere!, *Theor. Comput. Sci.*, **256**(1–2), 2001, 63–92.

[13] Girault, C., Valk, R., Eds.: *Petri Nets for Systems Engineering*, Springer, 2003.

[14] Higman, G.: Ordering by divisibility in abstract algebras, *Proc. London Math. Soc. (3)*, **2**(7), 1952, 326–336.

[15] Lazić, R.: Decidability of Reachability for Polymorphic Systems with Arrays: A Complete Classification, *Infinity '04*, number 138 in ENTCS, 2005.

[16] Lazić, R., Newcomb, T. C., Roscoe, A. W.: Polymorphic Systems with Arrays, 2-Counter Machines and Multiset Rewriting, *Infinity '04*, number 138 in ENTCS, 2005.

[17] Lipton, R. J.: *The Reachability Problem Requires Exponential Space*, Technical Report 62, Yale University, 1976.

[18] Meyer, A. R.: Weak monadic second-order theory of successor is not elementary-recursive, *Logic colloquium '72–73*, number 453 in Lect. Not. Math., Springer, 1975.

[19] Odifreddi, P.: *Classical Recursion Theory II*, Elsevier, 1999.

[20] Rackoff, C.: The Covering and Boundedness Problems for Vector Addition Systems, *Theor. Comput. Sci.*, **6**, 1978, 223–231.

[21] Reisig, W.: *Petri Nets: An Introduction*, Springer, 1985.

[22] Rosa Velardo, F., de Frutos Escrig, D., Marroquín Alonso, O.: On the expressiveness of Mobile Synchronizing Petri nets, *SecCo '05*, number 180 in ENTCS, 2007.

[23] Schnoebelen, P.: Verifying lossy channel systems has nonprimitive recursive complexity, *Inf. Proc. Lett.*, **83**(5), 2002, 251–261.

[24] Valk, R., Jantzen, M.: The Residue of Vector Sets with Applications to Decidability Problems in Petri Nets, *Acta Inf.*, **21**, 1985, 643–674.