COMPUTING SCIENCE

Step Semantics of Boolean Nets

Jetty Kleijn, Maciej Koutny, Marta Pietkiewicz-Koutny and Grzegorz Rozenberg

TECHNICAL REPORT SERIES

No. CS-TR-1298 November 2011

TECHNICAL REPORT SERIES

No. CS-TR-1298 November, 2011

Step Semantics of Boolean Nets

J. Kleijn, M. Koutny, M. Pietkiewicz-Koutny and G. Rozenberg

Abstract

Boolean nets - such as elementary net systems - are a family of Petri net models with very simple markings which are sets of places. We investigate several classes of boolean nets distinguished by different kinds of individual connections between places and transitions, as well as different ways in which these connections are combined in order to specify the effect of executing steps of transitions. The latter aspect can be captured by connection monoids. A key advantage of using connection monoids is that by describing the step semantics of a class of Petri nets in terms of a connection monoid, one can apply results developed within a general theory of Petri net synthesis. In this paper, we provide an extensive classification of boolean nets which can be described by connection monoids. This classification is based on the realisation that the different ways of interpreting combinations of connections can be made explicit using a higher level monoid. Moreover, we demonstrate that connection monoids can capture other behavioural properties of boolean nets, such as structural conflicts between transitions.

Printed and published by Newcastle University,
Computing Science, Claremont Tower, Claremont Road,
Newcastle upon Tyne, NE1 7RU, England.

Bibliographical details

KLEIJN, J., KOUTNY, M., PIETKIEWICZ-KOUTNY, M., ROZENBERG, G.

Step Semantics of Boolean Nets
[By] J. Kleijn, M. Koutny, M. Pietkiewicz-Koutny, G. Rozenberg
Newcastle upon Tyne: Newcastle University: Computing Science, 2011.

(Newcastle University, Computing Science, Technical Report Series, No. CS-TR-1298)

Added entries

NEWCASTLE UNIVERSITY
Computing Science. Technical Report Series. CS-TR-1298

Abstract

Boolean nets - such as elementary net systems - are a family of Petri net models with very simple markings which are sets of places. We investigate several classes of boolean nets distinguished by different kinds of individual connections between places and transitions, as well as different ways in which these connections are combined in order to specify the effect of executing steps of transitions. The latter aspect can be captured by connection monoids. A key advantage of using connection monoids is that by describing the step semantics of a class of Petri nets in terms of a connection monoid, one can apply results developed within a general theory of Petri net synthesis. In this paper, we provide an extensive classification of boolean nets which can be described by connection monoids. This classification is based on the realisation that the different ways of interpreting combinations of connections can be made explicit using a higher level monoid. Moreover, we demonstrate that connection monoids can capture other behavioural properties of boolean nets, such as structural conflicts between transitions.

About the authors

Jetty Kleijn is a visiting fellow within the School of Computing Science, Newcastle University.

Maciej Koutny obtained his MSc (1982) and PhD (1984) from the Warsaw University of Technology. In 1985 he joined the then Computing Laboratory of the University of Newcastle upon Tyne to work as a Research Associate. In 1986 he became a Lecturer in Computing Science at Newcastle, and in 1994 was promoted to an established Readership at Newcastle. In 2000 he became a Professor of Computing Science.

Marta Pietkiewicz-Koutny received her M.Sc. in Applied Mathematics from the Warsaw University of Technology in 1982. In 1984 she joined the Department of Operational Research, Institute of Econometrics in the Warsaw University of Economics where she worked as a junior lecturer until 1986. In 1987 she joined the Computing Laboratory of the University of Newcastle upon Tyne first as a research associate and then as a demonstrator (1988-1997). In the period 1997-2000 she was a Ph.D. student at the Department of Computing Science of the University of Newcastle upon Tyne, and in December 2000 she was awarded her Ph.D. degree. In the period 2000-2003 she was a researcher on the EU-funded DSoS (Dependable Systems of Systems) project, and in 2003 she was appointed a lecturer in the School of Computing Science. Dr. Pietkiewicz-Koutny's research interests concentrate on modelling and validation of concurrent systems. The main topic of her research is the synthesis of Petri nets from transition systems.

Grzegorz Rozenberg is a Professor of Computer Science at Leiden University, The Netherlands, and an Adjunct Professor at the Department of Computer Science of University of Colorado at Boulder, U.S.A. He is the head of the Theoretical Computer Science group at Leiden Institute of Advanced Computer Science (LIACS), and the scientific director of Leiden Center for Natural Computing (LCNC). He has published more than 500 papers, 6 books, and is a (co-)editor of about 90 books. He is a member of an editorial board of about 20 journals and book series. He has been a member of the program committees and invited speaker for practically all major conferences in theoretical computer science in Europe. He is a Foreign Member of the Finnish Academy of Sciences and Letters, a member of Academia Europaea, and he is holder of Honorary Doctorates of the University of Turku, Finland, the Technical University of Berlin, Germany, and the University of Bologna, Italy. He has received the Distinguished Achievements Award of the European Association for Theoretical Computer Science "in recognition of his outstanding scientific contributions to theoretical computer science". He is a Highly Cited Researcher by ISI.

Suggested keywords

PETRI NET
BOOLEAN NET
ELEMENTARY NET SYSTEM
SET NET
ARC EXTENSION
STEP SEMANTICS
CONNECTION MONOID
TAU-NET
NET SYNTHESIS

Step Semantics of Boolean Nets*

Jetty Kleijn[1], Maciej Koutny[2],
Marta Pietkiewicz-Koutny[2], and Grzegorz Rozenberg[1,3]

[1] LIACS, Leiden University, 2300 RA, The Netherlands
{kleijn,rozenber}@liacs.nl
[2] School of Computing Science, Newcastle University, NE1 7RU, UK
{maciej.koutny,marta.koutny}@ncl.ac.uk
[3] Department of Computer Science, University of Colorado at Boulder
430 UCB Boulder, CO 80309-0430, U.S.A.

Abstract. Boolean nets — such as elementary net systems — are a family of Petri net models with very simple markings which are sets of places. We investigate several classes of boolean nets distinguished by different kinds of individual connections between places and transitions, as well as different ways in which these connections are combined in order to specify the effect of executing steps of transitions. The latter aspect can be captured by connection monoids. A key advantage of using connection monoids is that by describing the step semantics of a class of Petri nets in terms of a connection monoid, one can apply results developed within a general theory of Petri net synthesis. In this paper, we provide an extensive classification of boolean nets which can be described by connection monoids. This classification is based on the realisation that the different ways of interpreting combinations of connections can be made explicit using a higher level monoid. Moreover, we demonstrate that connection monoids can capture other behavioural properties of boolean nets, such as structural conflicts between transitions.
Keywords: Petri net, boolean net, elementary net system, set net, arc extension, step semantics, connection monoid, τ-net, net synthesis

1 Introduction

Boolean nets are Petri nets where markings are simply sets of places. An example of boolean nets are the elementary net systems (or EN-systems) [12] which are generally regarded as a fundamental Petri net model. There are, however, situations when EN-systems do not provide a satisfactory modelling tool. For example, the recently introduced [8] class of boolean nets, called SET-nets, provides a net based computational model matching very closely the computations exhibited by reaction systems (a framework for investigating processes carried out by biochemical reactions in living cells, see, e.g., [7]).

The main distinguishing feature of SET-nets is that there is neither a concept of token counting nor a concept of conflict between executed transitions. In this way, although SET-nets resemble EN-systems in terms of markings and static structure, the execution semantics of the two net models are completely different. Figure 1(a, b, c) shows

* This research was supported by the Pascal Chair award from Leiden University.

three examples of SET-nets, and Figure 1(d, e, f) depicts their reachability graphs with arcs labelled by sets (steps) of executed transitions. The first SET-net shows that there is no conflict between transitions a and b even though there is only one token in their shared input place. Also, executing the step $\{\mathtt{a},\mathtt{b}\}$ produces only one token in place q. This effect is further illustrated by the second SET-net which can execute $\{\mathtt{a},\mathtt{b}\}$ and still place q contains only a single token. Note that if we were to interpret the nets in Figure 1(a, b) as EN-systems, then the step $\{\mathtt{a},\mathtt{b}\}$ would not be allowed in (a), and no step at all would be enabled in (b). Note that SET-nets can have self-loops, while this is not possible in EN-systems. So the net in Figure 1(c) which has a self-loop between a and p, cannot be interpreted as an EN-system.

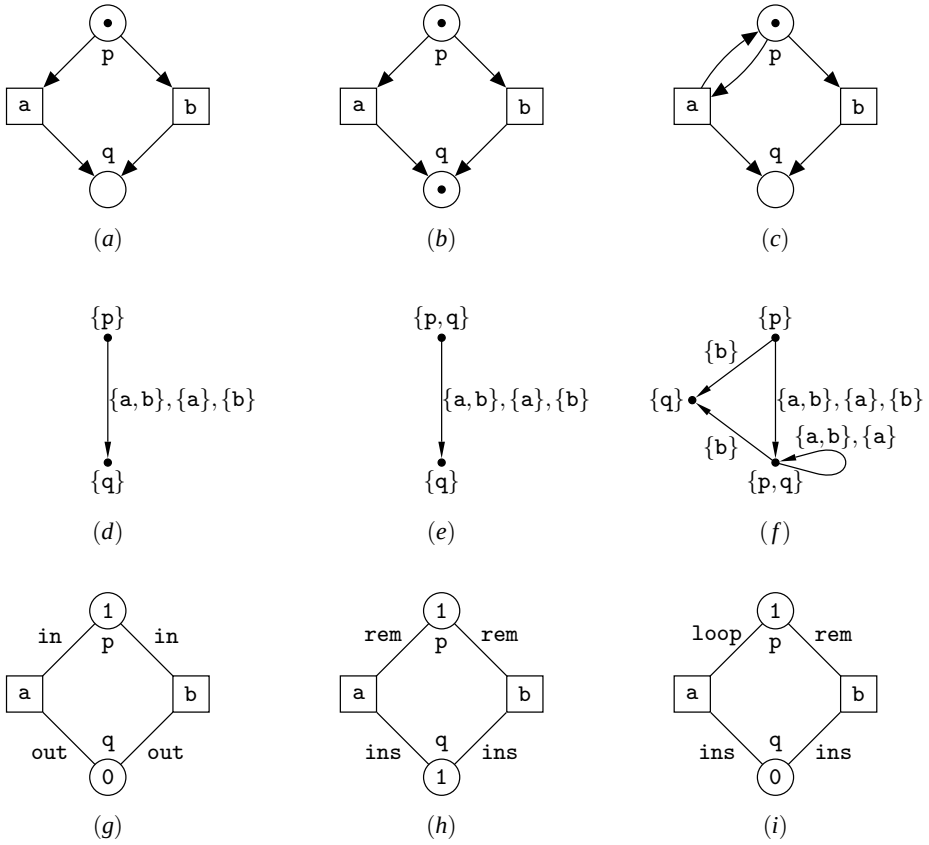


Fig. 1. Three SET-nets (a, b, c) together with their concurrent reachability graphs (d, e, f); rendering of the net in (a) in the form of τ_{EN}-nets (g); and rendering of the nets in (b, c) in the form of τ_{SN}-nets (h, i).

In this paper, we will present and investigate several classes of boolean nets distinguished by different kinds of individual connections between places and transitions, as well as different ways in which these connections are combined in order to define the effect of the execution of steps of transitions. The latter aspect can be captured by connection monoids. A key advantage of using connection monoids is that by describing the step semantics of a class of Petri nets in terms of a connection monoid, one can apply results developed within a general theory of Petri net synthesis as outlined in [3, 5]. More precisely, we will discuss boolean nets that can be seen as instances of τ-nets [3], parameterised net classes for which the net synthesis problem has been investigated and solved using regions of transition systems [6]. We will deal with a whole variety of boolean nets with the aim of classifying them through representing connection monoids (conn-monoids). A central part of our investigation is a detailed study of conn-monoids for boolean nets which allows us to capture not only the step semantics of nets, leading to a classification of the different ways of interpreting combinations of connections can be made explicit using a higher level monoid. In this paper, the classification is explained through a distinction between 'weaker' and 'stronger' connections which are often left implicit in the step semantics. It also allows one to express structural conflicts between transitions thanks to a special 'blocking' connection. In this way, conn-monoids emerge as a single formalism (at two levels) for boolean τ-nets which can be used to deal with conflicts, concurrency and net synthesis.

The paper is organised as follows. First, we present the basics of SET-nets and EN-systems. Section 3 recalls the general setup of [3, 5] in which Petri net classes are defined using conn-monoids and τ-nets. Section 4 shows how to build conn-monoids for EN-systems and SET-nets. The observations in this section are extended in Section 5 to general boolean τ-nets. Section 6 discusses further examples taken from two main sub-classes of boolean τ-nets: those with an underlying EN-system semantics, and those with an underlying SET-net semantics. Moreover, we show there how conn-monoids can be used to capture the a-priori and a-posteriori semantics. In a concluding Section 7, we summarise the contribution of this paper and briefly discuss future work.

The paper is a revised and extended version of the workshop paper [9] presented at the Workshop on Applications of Region Theory (ART), held in Newcastle in June 2011.

2 SET-nets and EN-systems

As their underlying structure, both SET-nets and EN-systems have a *net* which is a triple (P,T,F) such that P and T are disjoint finite sets of *places* and *transitions*, respectively, and $F \subseteq (T \times P) \cup (P \times T)$ is the *flow* of the net. We use the standard dot-notation: for any place or transition x, we let ${}^\bullet x = \{y \mid (y,x) \in F\}$ be its set of input elements and $x^\bullet = \{y \mid (x,y) \in F\}$ its output elements. This extends in the usual way to sets of places and/or transitions. For EN-systems, we will have the additional structural assumption that the underlying net has no 'self-loops' i.e., ${}^\bullet t \cap t^\bullet = \varnothing$ for all $t \in T$.

A *marking* of a SET-net or EN-system is a subset of places of their underlying net. A place belonging to a given marking is said to be marked. In diagrams, places are drawn as circles and transitions as rectangles. If $(x,y) \in F$, then (x,y) is an *arc* leading from

node x to *node y*. Markings are indicated by drawing in each place belonging to a given marking, a small black dot (a 'token').

A SET-*net* is a tuple $N = (P,T,F,M_0)$ such that (P,T,F) is a net and $M_0 \subseteq P$ is its *initial* marking. An EN-*system* is a tuple $N = (P,T,F,M_0)$ such that (P,T,F) is a net without self-loops and $M_0 \subseteq P$ is its *initial* marking.

The dynamics of SET-nets is defined as follows. Let $N = (P,T,F,M_0)$ be a SET-net, and let $t \in T$. Then t is *enabled* at a marking M if ${}^\bullet t \subseteq M$. In such a case, t can be *executed*, leading to the marking $M' = (M \setminus {}^\bullet t) \cup t^\bullet$. A subset U of T, called a *step*, is *enabled* at M if ${}^\bullet U \subseteq M$. An enabled U can be *executed*, leading to the marking $M' = (M \setminus {}^\bullet U) \cup U^\bullet$.

Hence, in a SET-net, a step U is enabled whenever each of its input places belongs to the current marking, i.e., each of its elements is enabled. When U is executed, its input places lose their tokens, while all output places will be marked. If a place is both input and output for U, it is marked before and after the execution of U. Furthermore, output places of U that were marked before its execution will remain marked. It is also worthwhile to observe that there may be distinct transitions t, $u \in U$ for which ${}^\bullet t \cap {}^\bullet u \neq \varnothing$ or $t^\bullet \cap u^\bullet \neq \varnothing$. This has no effect on their participation in the execution of U.

The dynamics of EN-systems is defined in a similar way, except that the enabling condition is crucially different. Let $N = (P,T,F,M_0)$ be an EN-system and let $t \in T$. Then t is *enabled* at a marking M if ${}^\bullet t \subseteq M$ and $t^\bullet \cap M = \varnothing$. If t is enabled at M, it can be *executed* which results in the marking $M' = (M \setminus {}^\bullet t) \cup t^\bullet$. A step U of T is *enabled* at M if each $t \in U$ is enabled at M and $({}^\bullet t \cup t^\bullet) \cap ({}^\bullet u \cap u^\bullet) = \varnothing$, for any two distinct transitions $t,u \in U$. Then U can be *executed* leading to $M' = (M \setminus {}^\bullet U) \cup U^\bullet$.

Hence, in an EN-system, if a step U is enabled at marking M then each of its input places is marked and none of its output places is marked. Actually, a step can only ever be enabled if the input/output neighbourhood of the transitions in U do not overlap (i.e., if there is no *structural conflict* in U).

> *In* SET*-nets and* EN*-systems markings are sets and tokens are manipulated using set-based rather than multiset-based arithmetic. We will refer to such Petri net models as being boolean.*

To each of the above two net models we can add *inhibitor arcs* and *activator arcs* connecting places to transitions, by adding relations *Inh* and *Act* to their specification. Given the set of places P and set of transitions T of a SET-net or EN-system, $Inh, Act \subseteq P \times T$ define its set of *inhibitor* and *activator* arcs, respectively. For each transition $t \in T$, we denote ${}^\circ t = \{p \mid (p,t) \in Inh\}$ for the set of inhibitor places of t, and ${}^\blacklozenge t = \{p \mid (p,t) \in Act\}$ for its activator places. (Both notions are extended to sets of transitions.) The intuition behind these *context arcs* is that in order for a transition to be enabled at a marking, its activator places should be marked and its inhibitor places should not be marked. Thus the dynamics of these extended net classes is adapted in the following way: a step U is *enabled* at a marking when it is enabled in the underlying SET-net or EN-system and ${}^\blacklozenge U \subseteq M$ and ${}^\circ U \cap M = \varnothing$. When U is enabled at M and it is executed, then the resulting marking is defined as before (here the activator and inhibitor arcs have no effect). Note that what we just described is an *a-priori semantics* for context arcs (see, e.g., [11]).

3 Connection monoids and τ-nets

In the general setup of [3, 5] Petri net classes are defined on basis of individual connections between places and transitions. Moreover, the effect of the simultaneous execution of a *step* (a set or multiset of transitions) on a given place is calculated using a dedicated commutative monoid which returns the composite connection between that place and the step. For boolean nets, we will assume that each step is a set of transitions rather than a multiset as in [3, 5]. This simplifies the presentation and is harmless as boolean nets as we consider them here, would not allow true multiset steps anyway.

Connection monoids describe the relation between a place and a step. A *connection monoid* (or conn-monoid) is a set $\mathbb{S}$ of *connections* together with a commutative and associative binary composition operation $\oplus$, and a neutral element (identity) $\mathbf{0}$. We will use the same symbol $\mathbb{S}$ for a conn-monoid and for its underlying set of connections.

Let $\mathbb{S}$ be a conn-monoid. Then, a *net-type over* $\mathbb{S}$ is a transition system $\tau = (Q, \mathbb{S}, \Delta)$ where Q is a set of *states*, and $\Delta : Q \times \mathbb{S} \to Q$ is a partial function such that $\Delta(q, \mathbf{0}) = q$, for all $q \in Q$. For every state q, the set $enbld_\tau(q) = \{s \mid \Delta(q,s) \text{ is defined}\}$ consists of all connections from $\mathbb{S}$ that are *enabled* at q. Each net-type $\tau = (Q, \mathbb{S}, \Delta)$ defines a class of nets, the so-called τ-*nets*. The net-type specifies through Q the values that can be assigned to places; through $\mathbb{S}$ the possible connections and the result of combining connections; and through Δ the enabling conditions and newly generated values (the effect of connections).

Given $\tau = (Q, \mathbb{S}, \Delta)$, a τ-*net* is a tuple $N = (P, T, F, M_0)$, where P and T are, respectively, disjoint finite sets of places and transitions, $F : (P \times T) \to \mathbb{S}$ is a *connection mapping*, and M_0 is the *initial marking* of N (in general, a marking is a mapping from P to Q). For a place p of N and a step $U = \{t_1, \ldots, t_n\}$ of transitions, we define the composite connection $F(p,U)$ between U and p by $F(p,t_1) \oplus \cdots \oplus F(p,t_n)$. Moreover, $F(p, \varnothing) = \mathbf{0}$.

A step U is *enabled* at a marking M if $F(p,U) \in enbld_\tau(M(p))$, for every place $p \in P$. The *execution* of such a step produces the marking M' such that $M'(p) = \Delta(M(p), F(p,U))$, for every place $p \in P$. The *concurrent reachability graph* $CRG(N)$ of N is formed by executing inductively from M_0 all possible enabled steps of N.

4 Connection monoids for EN-systems and SET-nets

Starting from EN-systems, we will now present a number of specific classes of boolean nets defined on basis of their place-transition connections. In what follows we describe the structure of the connection monoids by a so-called Cayley table displaying the outcome of all possible combinations of connections.

4.1 EN-systems

In EN-systems, there are three basic connections between places and transitions:

– $F(p,t) = \top$ p and t are disconnected (independent) 

– $F(p,t) = \mathtt{in}$ there is an arc from p to t 

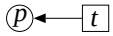
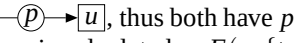
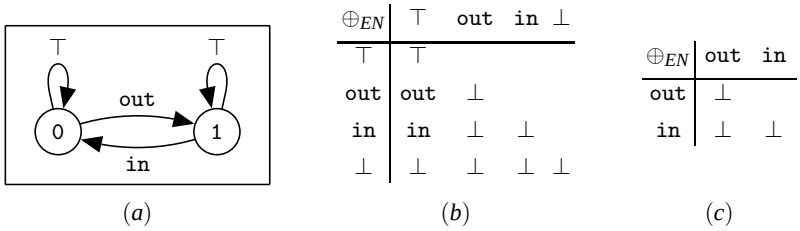
– $F(p,t) = \mathtt{out}$ there is an arc from t to p 

Figure 2(a) depicts τ_{EN}, the net-type showing how the connections between a place and a transition in an EN-system determine the enabledness of the transition w.r.t. that place and the resulting marking if it is executed. In the diagram, $\mathtt{0}$ and $\mathtt{1}$ mean that the place is respectively *empty* (i.e., not marked) and *full* (marked). Thus, if the place p is marked and there is an arc from p to the transition t ($F(p,t) = \mathtt{in}$), then t may be executed as far as p is concerned and the effect will be that p is empty after the execution of t. If p and t are not connected, t may always be executed from the point of view of p and its execution has no effect on the marking of p. There is no explicit reference to $F(p,t) = \mathtt{in}$ for the case that p is empty nor to $F(p,t) = \mathtt{out}$ when p is full. In these cases the marking of p prohibits the enabledness of t. In addition to the three standard types of connections, τ_{EN} has a special 'blocking' connection $\bot$ which does not label any arc (is never enabled), hence $\bot \notin \mathit{enbld}_{\tau_{EN}}(\mathtt{0}) \cup \mathit{enbld}_{\tau_{EN}}(\mathtt{1})$. The connection $\bot$ is also used to capture *structural conflict* between transitions. As such it is a convenient device to capture precisely those steps which are not allowed, because of the internal conflicting relations between their transitions w.r.t. a place. Figure 1(g) shows the way in which the net in Figure 1(a) can be represented as a τ_{EN}-net.

The conn-monoid $\mathbb{S}_{EN} = (\{\top, \mathtt{out}, \mathtt{in}, \bot\}, \oplus_{EN}, \top)$ is defined through the Cayley table in Figure 2(b). Here $\mathtt{out} \oplus_{EN} \mathtt{out} = \mathtt{out} \oplus_{EN} \mathtt{in} = \mathtt{in} \oplus_{EN} \mathtt{out} = \mathtt{in} \oplus_{EN} \mathtt{in} = \bot$ corresponds directly to the requirement that the neighbourhoods of transitions in a step must be disjoint for it to be executed.

For example, if we have two transitions, t and u, both removing tokens from place p,  , thus both have p as an input place, then the connection of the step $\{t,u\}$ w.r.t. p is calculated as $F(p,\{t,u\}) = F(p,t) \oplus_{EN} F(p,u) = \mathtt{in} \oplus_{EN} \mathtt{in} = \bot$ implying that on account of p, t and u can never be executed together in a step $\{t,u\}$.



(a) (b) (c)

Fig. 2. net-type τ_{EN}, and the Cayley table of $\mathbb{S}_{EN}$.

In all conn-monoids, $\top$ will from now on denote the identity element (**0**) and — if present — $\bot$ is the absorbing element. The monoid $\mathbb{S}_{EN}$ is the most restrictive monoid over $\top$, $\mathtt{in}$, $\mathtt{out}$, and $\bot$, because its operation does not yield any non-$\bot$ results except when $\top$ is involved. This is clearly seen in Figure 2(c) which depicts the non-trivial part

of the Cayley table from Figure 2(*b*), while omitting the values implicitly due to commutativity. In what follows we will present conn-monoids using a minimal presentation of their Cayley table similar to the one in Figure 2(*c*). Note that here and for the other conn-monoids presented in this section, we do not explicitly prove associativity of the monoid operation. The reader is invited to check, but for all cases associativity will also follow from more general results in Section 5.

A *basis* of a conn-monoid is any irreducible subset of its non-$\bot$ connections such that all non-$\bot$ connections can be derived from it.

Proposition 1. $\{\top, \mathtt{in}, \mathtt{out}\}$ *is the only basis of* $\mathbb{S}_{EN}$.

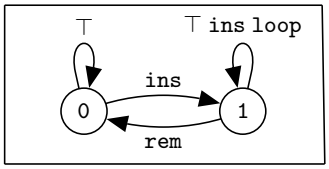
Proof. Follows directly from the table in Figure 2(*b*). □

4.2 SET-nets

Now there are four possible connections between places and transitions:

- $F(p,t) = \top$ — p and t are disconnected (independent)
- $F(p,t) = \mathtt{rem}$ — there is an arc from p to t
- $F(p,t) = \mathtt{ins}$ — there is an arc from t to p
- $F(p,t) = \mathtt{loop}$ — there is an arc from t to p, and from p to t

Figure 3(*a*) depicts τ_{SN}. Comparing Figure 3(*a*) and Figure 2(*a*) brings to light the important difference between the meaning of an arc from a transition to a place in an EN-system (connection `out`) and the meaning of such an arc in a SET-nets (connection `ins`): Figure 4 shows the only `out`-labelled arc in τ_{EN} and the `ins`-labelled arcs in τ_{SN}. Figure 1(*h*,*i*) shows the way in which the SET-nets in Figure 1(*b*,*c*) can be represented as τ_{SN}-nets.



(*a*)

$\oplus_{SN}$	ins	rem	loop
ins	ins		
rem	loop	rem	
loop	loop	loop	loop

(*b*)

Fig. 3. net-type τ_{SN}, and the simplified table of $\mathbb{S}_{SN} = (\{\top, \mathtt{ins}, \mathtt{rem}, \mathtt{loop}\}, \oplus_{SN}, \top)$.

The simplified Cayley table of the conn-monoid $\mathbb{S}_{SN}$ is shown in Figure 3(*b*). From the table we see, e.g., that if p is an output place of a transition t and input place to u, $\boxed{t} \dashrightarrow \textcircled{p} \dashrightarrow \boxed{u}$, then the connection of the step $\{t,u\}$ w.r.t. p is given by $F(p,\{t,u\}) = F(p,t) \oplus_{SN} F(p,u) = \mathtt{ins} \oplus_{SN} \mathtt{rem} = \mathtt{loop}$ and so, as far as p is concerned, $\{t,u\}$ can be executed if p contains a token; moreover, p will also have a token after the execution of $\{t,u\}$.

Fig. 4. Difference between arcs from transitions to places in EN-systems and SET-nets.

Another important property of the $\mathbb{S}_{SN}$ monoid is the idempotence of its operation (see the diagonal of the Cayley table of $\mathbb{S}_{SN}$). This reflects one of the main features of SET-nets, namely that since resources are not quantified, they can be used by many transitions with the same connectivity in tandem, as though they were just one such transition. Note furthermore, that since SET-nets know no structural conflict, $\bot$ is not introduced through $\oplus_{SN}$. Consequently, $\bot$ is not necessary in the case of τ_{SN} and $\mathbb{S}_{SN}$.

Proposition 2. $\{\top, \mathtt{ins}, \mathtt{rem}\}$ *is the only basis of* $\mathbb{S}_{SN}$.

Proof. Follows directly from the table in Figure 3(*b*). □

This insight forms a formal justification of the way in which the arcs in τ_{SN}-nets are drawn: direct arrows are used for `ins` and `rem`, but `loop` as a 'compound' connection can be depicted by the 'compound' representation for `ins` and `rem`. Note that in the $\bot$-less version of $\mathbb{S}_{SN}$, `loop` is the absorbing element, but this will change when we add inhibitor arcs. First however, we add inhibitor arcs to EN-systems.

4.3 EN-systems with inhibitor arcs

In comparison with EN-systems, in the case of ENI-systems (elementary net systems with inhibitor arcs), we now have one more connection to take into account:

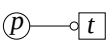
– $F(p,t) = \mathtt{inh}$ there is an inhibitor arc from p to t 

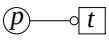
Figure 5 shows the net-type τ_{ENI}, and the simplified Cayley table of the conn-monoid $\mathbb{S}_{ENI} = (\{\top, \mathtt{out}, \mathtt{in}, \mathtt{inh}, \bot\}, \oplus_{ENI}, \top)$. From this we see that the monoid $\mathbb{S}_{ENI}$ has $\mathbb{S}_{EN}$ as a submonoid and captures an additional type of structural conflict: $\mathtt{in} \oplus_{ENI} \mathtt{inh} = \mathtt{inh} \oplus_{ENI} \mathtt{in} = \bot$. That $\mathtt{out} \oplus_{ENI} \mathtt{inh} = \mathtt{inh} \oplus_{ENI} \mathtt{out} = \mathtt{out}$ is a consequence of the a-priori semantics.

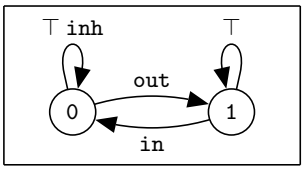
Proposition 3. $\{\top, \mathtt{in}, \mathtt{out}, \mathtt{inh}\}$ *is the only basis of* $\mathbb{S}_{ENI}$.

Proof. Follows directly from the table in Figure 5(*b*). □

4.4 SET-nets with inhibitor arcs

Like when adding inhibitor arcs to elementary nets systems, we have to cater for one additional connection for SNI-nets (set-nets with inhibitor arcs):

– $F(p,t) = \mathtt{inh}$ there is an inhibitor arc from p to t 



(*a*)

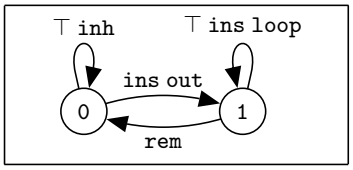
$\oplus_{ENI}$	out	in	inh
out	$\bot$		
in	$\bot$	$\bot$	
inh	out	$\bot$	inh

(*b*)

Fig. 5. net-type τ_{ENI} (*a*), and the simplified Cayley table of $\mathbb{S}_{ENI}$ (*b*).

Figure 6 shows the net-type τ_{SNI}, and the simplified Cayley table of the conn-monoid $\mathbb{S}_{SNI} = (\{\top, \mathtt{ins}, \mathtt{rem}, \mathtt{loop}, \mathtt{inh}, \mathtt{out}, \bot\}, \oplus_{SNI}, \top)$. Note that $\mathbb{S}_{SN}$ is a submonoid of $\mathbb{S}_{SNI}$. In contrast to the net-type τ_{SN}, we do need $\bot$ since structural conflicts occur when inhibitors are combined with a token-consuming arc (exactly as in EN-systems). Thus $\mathbb{S}_{SNI}$ captures conflicts: $\mathtt{rem} \oplus_{SNI} \mathtt{inh} = \mathtt{inh} \oplus_{SNI} \mathtt{rem} = \bot$ and $\mathtt{loop} \oplus_{SNI} \mathtt{inh} = \mathtt{inh} \oplus_{SNI} \mathtt{loop} = \bot$.

Furthermore, the monoid must be closed w.r.t. its operation and so out had to be added as a new connection to describe $\mathtt{ins} \oplus_{SNI} \mathtt{inh}$ since SNI-nets are considered here under the a-priori semantics. Notice that although out on its own has the same meaning here as in EN-systems and ENI-systems, it is now understood differently when combined with other connections. An example is $\mathtt{out} \oplus_{SNI} \mathtt{out} = \mathtt{out}$ rather than $\mathtt{out} \oplus_{ENI} \mathtt{out} = \bot$ since the step semantics of SNI-nets is different from that of ENI-systems.



(*a*)

$\oplus_{SNI}$	ins	rem	inh	loop	out
ins	ins				
rem	loop	rem			
inh	out	$\bot$	inh		
loop	loop	loop	$\bot$	loop	
out	out	$\bot$	out	$\bot$	out

(*b*)

Fig. 6. net-type τ_{SNI} (*a*), and the simplified Cayley table of $\mathbb{S}_{SNI}$ (*b*).

Proposition 4. $\{\top, \mathtt{ins}, \mathtt{rem}, \mathtt{inh}\}$ *is the only basis of* $\mathbb{S}_{SNI}$.

Proof. Follows directly from the table in Figure 6(*b*). □

Like for $\mathbb{S}_{SN}$, also the operation of $\mathbb{S}_{SNI}$ is idempotent. Even stronger:

Proposition 5. *Let $U \neq \varnothing$ be a step of transitions and $\mathbb{F} = \{F(p,t) \mid t \in U\}$.*

$$F(p,U) = \begin{cases} \top & \textit{if } \mathbb{F} = \{\top\} \\ \mathtt{ins} & \textit{if } \mathbb{F} \subseteq \{\mathtt{ins}, \top\} & \wedge\ \mathtt{ins} \in \mathbb{F} \\ \mathtt{rem} & \textit{if } \mathbb{F} \subseteq \{\mathtt{rem}, \top\} & \wedge\ \mathtt{rem} \in \mathbb{F} \\ \mathtt{inh} & \textit{if } \mathbb{F} \subseteq \{\mathtt{inh}, \top\} & \wedge\ \mathtt{inh} \in \mathbb{F} \\ \mathtt{out} & \textit{if } \mathbb{F} \subseteq \{\mathtt{inh}, \mathtt{ins}, \mathtt{out}, \top\} & \wedge\ (\mathtt{out} \in \mathbb{F} \ \vee \{\mathtt{inh}, \mathtt{ins}\} \subseteq \mathbb{F}) \\ \mathtt{loop} & \textit{if } \mathbb{F} \subseteq \{\mathtt{ins}, \mathtt{rem}, \mathtt{loop}, \top\} & \wedge\ (\mathtt{loop} \in \mathbb{F} \vee \{\mathtt{rem}, \mathtt{ins}\} \subseteq \mathbb{F}) \\ \bot & \textit{otherwise}\,. \end{cases}$$

Proof. Follows directly from the table in Figure 6(*b*). □

4.5 en-systems with inhibitor and activator arcs

Finally we add activator arcs to ENI-systems, leading to EN-systems with context arcs, or ENC-systems. Thus the last connection we consider is:

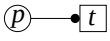
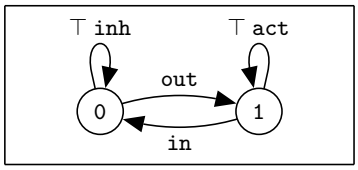
– $F(p,t) = \mathtt{act}$ there is an activator arc from p to t 

Figure 7 shows the net-type τ_{ENC}, and the simplified Cayley table of the conn-monoid $\mathbb{S}_{ENC}$ for ENC-systems (with the a-priori step semantics). Note that $\mathbb{S}_{ENI}$ is a submonoid of $\mathbb{S}_{ENC}$.



(*a*)

$\oplus_{ENC}$	out	in	inh	act
out	$\bot$			
in	$\bot$	$\bot$		
inh	out	$\bot$	inh	
act	$\bot$	in	$\bot$	act

(*b*)

Fig. 7. net-type τ_{ENC} (*a*), and the simplified Cayley table of $\mathbb{S}_{ENC}$ (*b*).

Proposition 6. $\{\top, \mathtt{out}, \mathtt{in}, \mathtt{inh}, \mathtt{act}\}$ *is the only basis of* $\mathbb{S}_{ENC}$.

Proof. Follows directly from the table in Figure 7(*b*). □

In the simplified Cayley table of $\mathbb{S}_{ENC}$, we see that $\mathtt{inh} \oplus_{ENC} \mathtt{out} = \mathtt{out} \oplus_{ENC} \mathtt{inh} = \mathtt{out}$ and $\mathtt{act} \oplus_{ENC} \mathtt{in} = \mathtt{in} \oplus_{ENC} \mathtt{act} = \mathtt{in}$. These pairs of connections reflect that while the transitions involved are enabled with respect to the given place — which should be empty in the first case (for the inh and out connections) and marked in the second case (for the act and in connections) — they affect it in a different way. The connection designated for testing (inh and act, respectively) is 'weaker' than the connection that induces a state change (out and in, respectively) in the sense that the 'stronger' connection decides the final result.

4.6 SET-nets with inhibitor and activator arcs

Adding activator arcs to SNI-systems yields SET-nets with context arcs (SNC-systems). Again, we have an activator connection to consider:

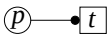
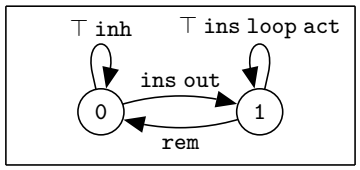
– $F(p,t) = \texttt{act}$ there is an activator arc from p to t 

Figure 8 shows the net-type τ_{SNC}, and the simplified Cayley table of the conn-monoid $\mathbb{S}_{SNC} = (\{\top, \texttt{ins}, \texttt{rem}, \texttt{inh}, \texttt{act}, \texttt{loop}, \texttt{out}, \bot\}, \oplus_{SNC}, \top)$ for SET-nets with inhibitor and activator arcs (under the a-priori step semantics). Note that $\mathbb{S}_{SNI}$ is a submonoid of $\mathbb{S}_{SNC}$.



(*a*)

$\oplus_{SNC}$	ins	rem	inh	act	loop	out
ins	ins					
rem	loop	rem				
inh	out	$\bot$	inh			
act	loop	rem	$\bot$	act		
loop	loop	loop	$\bot$	loop	loop	
out	out	$\bot$	out	$\bot$	$\bot$	out

(*b*)

Fig. 8. net-type τ_{SNC} (*a*), and the simplified Cayley table of $\mathbb{S}_{SNC}$ (*b*).

Proposition 7. $\{\top, \texttt{ins}, \texttt{rem}, \texttt{inh}, \texttt{act}\}$ *is the only basis of* $\mathbb{S}_{SNI}$.

Proof. Follows directly from the table in Figure 8(*b*). □

4.7 Connection patterns

As may have become clear from the above, there are 9 different possible connection patterns: from 0 (unmarked) and from 1 (marked), either to 0 or to 1, or undefined. They are all depicted in Figure 9. Until now we have discussed only 7 of them as occurring in an actual net-type; the remaining two (depicted in the two topmost positions of the middle column in Figure 9) will be discussed in Section 6. Observe that the connections in and rem have the same topological pattern, but a different intended semantics as reflected by their algebraic properties (the first one is not idempotent, but the second one is). Note furthermore that nevertheless, similar to act and in Figure 7(*b*), in the table in Figure 8(*b*) composing act and rem yields rem.

Also the connections loop and act have the same topological pattern. Note furthermore that they appear together — as different connections — in τ_{SNC}. Moreover, the effect of combining act and loop is loop and not act. This is in accordance with the step semantics of SET-nets, by whichadding tokens happens after removing or testing. So, again, loop as a connection that induces a change of the state is 'stronger' than the testing connection act.

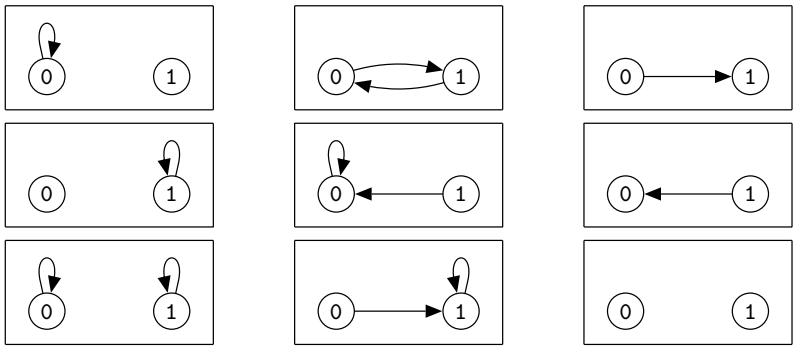


Fig. 9. Possible connection patterns in boolean τ-nets.

Thus we arrive at the crucial point in our considerations where it becomes clear that the sophisticated (and sometimes surprising) nature of different connections necessarily involves algebraic properties in addition to topological ones.

5 General boolean τ-nets

Based on the experience gained in the previous section, a general approach to describe boolean τ-nets using conn-monoids is proposed. Though the choices made here are examples and not absolute, the result illustrates how such a general set-up facilitates the formal reasoning about connections between places and transitions and their composition. We will define two super-monoids, $\mathbb{S}_{boolS}$ (for boolean nets with a SET-net interpretation) and $\mathbb{S}_{boolE}$ (for boolean nets with an underlying standard EN-systems semantics).

First, we propose a general approach for the classification of the possible connections in boolean τ-nets with SET-net inspired semantics. Here we take the view that each arc in a net-type defines an *enabling* condition and a *result*, and has a *strength* (is 'weak' or 'strong'). Weak connections impose constraints only when it comes to enabling, but, unlike strong connections, they do not impose constraints on the state resulting from transition execution. So, when transitions with weak connections are combined with transitions with stronger connections, the latter decide the resulting state (note that the notion being of weak/strong connection is a relative rather than absolute property). For example, `loop` is strong in $\mathbb{S}_{SN}$ as executing its transition finishes by putting a token in the place. This is supported by the algebraic property of absorption of `loop` (see the table in Figure 3). In other words, when combined with other connections, `loop` always decides the effect of a step on a place. On the other hand, `rem` is weak in $\mathbb{S}_{SN}$ as with this connection the enabling conditions are important, but the state of the net place after a step (with a transition connected to the place by `rem`) may be decided by another transition in that step; for example, $\texttt{rem} \oplus_{SN} \texttt{ins} = \texttt{loop}$. Hence, although `rem` removes a token from a place (is a state changing connection), and strong when compared with `act`, when combined with `ins`, it is weaker and the place will still have a token after the step (with transitions connected by `rem` and `loop`) was executed.

As another example, although ins does not satisfy the absorption property in $\mathbb{S}_{SNC}$, it is considered strong. The result of the execution of a step on a place with a transition connected to that place by ins is always as the effect of ins acting alone. In other words, ins is considered strong, because it dictates the final result of a step.

Testing connections, like inh and act, are always considered weak. They do not affect the resulting state and only enabling conditions are important for them (under the a-priori semantics, see the discussion in Section 6.3). In $\mathbb{S}_{SNC}$ we have $\mathtt{act} \oplus_{SNC} \mathtt{loop} = \mathtt{loop}$ and $\mathtt{act} \oplus_{SNC} \mathtt{ins} = \mathtt{loop}$ justified by the observation that both pairs contain an 'active' connection (loop and ins, respectively).

Given such distinction between weak and strong, we have now 25 different connections ∂_{xy} with $x,y \in \{\mathtt{w}, \mathtt{s}, \overline{\mathtt{w}}, \overline{\mathtt{s}}, \mathtt{n}\}$ and x referring to arrows outgoing from 0 and y to arrows outgoing from 1, where we assume that n means non-enabledness, $\overline{(.)}$ implies changing the state, w a weak arrow, and s a strong arrow,

The intuition behind, for example, $\partial_{\mathtt{ss}}$ is that the connection makes a transition always enabled (whethrer the place is marked or not) and the execution of the transition keeps the state of the place unchanged. Thus we may consider for example encodings of the connections, encountered until now including those of $\mathbb{S}_{SNC}$:

$\bot$	$\top$	in	out	ins	rem	loop	inh	act
$\partial_{\mathtt{nn}}$	$\partial_{\mathtt{ww}}$	$\partial_{\mathtt{n}\overline{\mathtt{s}}}$	$\partial_{\overline{\mathtt{s}}\mathtt{n}}$	$\partial_{\overline{\mathtt{s}}\mathtt{s}}$	$\partial_{\mathtt{n}\overline{\mathtt{w}}}$	$\partial_{\mathtt{ns}}$	$\partial_{\mathtt{wn}}$	$\partial_{\mathtt{nw}}$

In the above, the weak connections are: $\top$, inh, act and rem. They are either not affecting the resulting state like inh or act, or like rem have no final say about the resulting state. The topological patterns corresponding to these connections are shown in Figure 10 with the weak arcs now indicated by dashed lines.

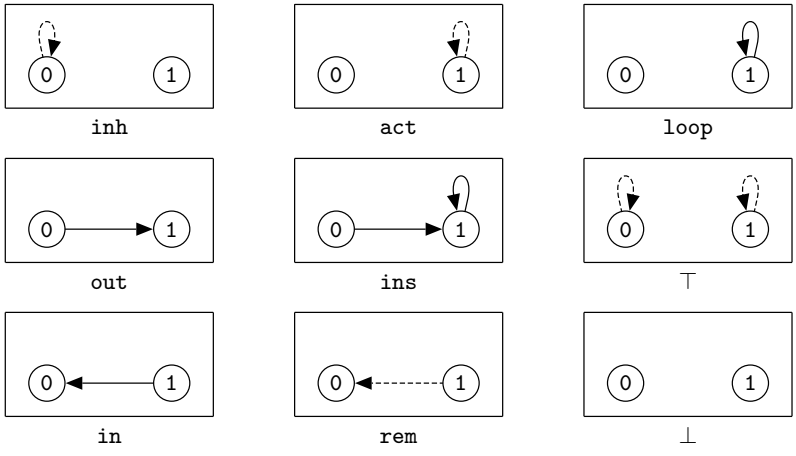


Fig. 10. Connections in boolean τ-nets considered in Section 4 with weak arcs indicated by dashed lines.

The possible algebraic operations for the 25 connections can be described in a general, abstract way. It is still our aim here to have an outcome consistent with the semantics of SNC-systems. Moreover, we try to follow the idea where possible that stronger arcs dominate weaker ones. The central concept here is the associative and commutative operation $\odot_S$ given by the following table:

$\odot_S$	w	$\overline{\text{w}}$	s	$\overline{\text{s}}$	n
w	w				
$\overline{\text{w}}$	$\overline{\text{w}}$	$\overline{\text{w}}$			
s	s	s	s		
$\overline{\text{s}}$	$\overline{\text{s}}$	$\overline{\text{s}}$	n	$\overline{\text{s}}$	
n	n	n	n	n	n

Proposition 8. $\mathbb{S}_{connS} = (\{\text{w}, \text{s}, \overline{\text{w}}, \overline{\text{s}}, \text{n}\}, \odot_S, \text{w})$ *is a commutative monoid.*

Proof. We only need to show that $(a \odot_S b) \odot_S c = a \odot_S (b \odot_S c)$ for all $a, b, c \in \{\text{w}, \text{s}, \overline{\text{w}}, \overline{\text{s}}, \text{n}\}$. To start with, if $\text{n} \in \{a, b, c\}$ then $(a \odot_S b) \odot_S c = \text{n} = a \odot_S (b \odot_S c)$. Otherwise, we observe that the following hold:

- If $\text{s} \in \{a, b, c\}$ and $\overline{\text{s}} \notin \{a, b, c\}$ then $(a \odot_S b) \odot_S c = \text{s} = a \odot_S (b \odot_S c)$.
- If $\overline{\text{s}} \in \{a, b, c\}$ and $\text{s} \notin \{a, b, c\}$ then $(a \odot_S b) \odot_S c = \overline{\text{s}} = a \odot_S (b \odot_S c)$.
- If $\overline{\text{s}} \in \{a, b, c\}$ and $\text{s} \in \{a, b, c\}$ then $(a \odot_S b) \odot_S c = \text{n} = a \odot_S (b \odot_S c)$.
- If $a, b, c \in \{\text{w}, \overline{\text{w}}\}$ and $\overline{\text{w}} \in \{a, b, c\}$ then $(a \odot_S b) \odot_S c = \overline{\text{w}} = a \odot_S (b \odot_S c)$.
- If $a = b = c = \text{w}$ then $(a \odot_S b) \odot_S c = \text{w} = a \odot_S (b \odot_S c)$. □

With this operation we can now define the operation on the (encoded) connections by $\partial_{xy} \oplus_S \partial_{x'y'} = \partial_{x \odot_S x' \; y \odot_S y'}$ for the super-monoid $\mathbb{S}_{boolS}$.

Theorem 1. $\mathbb{S}_{boolS} = (\{\partial_{xy} \mid x, y \in \{\text{w}, \text{s}, \overline{\text{w}}, \overline{\text{s}}, \text{n}\}\}, \oplus_S, \partial_{\text{ww}})$ *is a conn-monoid.*

Proof. Follows from Proposition 8 and $\partial_{xy} \oplus_S \partial_{x'y'} = \partial_{x \odot_S x' \; y \odot_S y'}$. □

We have now obtained a general systematisation for boolean nets with an underlying set-nets step semantics. It should be noted though that some entries in the scheme are based on sometimes perhaps somewhat arbitrary assumptions for cases where the intended operational meaning was not clear. In particular, defining $\text{w} \odot_S \overline{\text{w}} = \overline{\text{w}}$ gives priority to the change of state. Also, $\text{s} \odot_S \overline{\text{s}} = \text{n}$ reflects that s and $\overline{\text{s}}$ are both strong and 'uncompromising', with one insisting on changing the state whereas the other wants to preserve it, a conflict that cannot be reconciled. Clearly, assumptions of this kind are not cast in stone. With different application motivated models in mind, one may freely modify them, study, and appreciate the differences.

Since, as intended, $\mathbb{S}_{SN}$, $\mathbb{S}_{SNI}$ and $\mathbb{S}_{SNC}$ are all submonoids of $\mathbb{S}_{boolS}$, we do not need to prove their associativity since it follows from Theorem 1.

To generalise $\mathbb{S}_{EN}$, $\mathbb{S}_{ENI}$ and $\mathbb{S}_{ENC}$ in a similar manner, we need to modify slightly the operation $\odot_S$, and the new version, denoted by $\odot_E$, is given in the following table:

$\odot_E$	w	$\overline{\text{w}}$	s	$\overline{\text{s}}$	n
w	w				
$\overline{\text{w}}$	$\overline{\text{w}}$	$\overline{\text{w}}$			
s	s	s	s		
$\overline{\text{s}}$	$\overline{\text{s}}$	$\overline{\text{s}}$	n	n	
n	n	n	n	n	n

The only difference between $\odot_S$ and $\odot_E$ is when we combine two strong, state changing, connections. In nets with an underlying EN-system semantics, connections of this kind are not idempotent (in fact, they cannot be combined at all). The reason is that in EN-systems, we cannot compose two transitions that are trying to remove or deposit tokens in the same place at the same time. Then, similarly as for nets with underlying SET-net semantics, we define $\partial_{xy} \oplus_E \partial_{x'y'} = \partial_{x \odot_E x'\, y \odot_E y'}$.

Proposition 9. $\mathbb{S}_{connE} = (\{\mathtt{w}, \mathtt{s}, \overline{\mathtt{w}}, \overline{\mathtt{s}}, \mathtt{n}\}, \odot_E, \mathtt{w})$ *is a commutative monoid.*

Proof. We only need to show that $(a \odot_E b) \odot_E c = a \odot_E (b \odot_E c)$ for all $a,b,c \in \{\mathtt{w}, \mathtt{s}, \overline{\mathtt{w}}, \overline{\mathtt{s}}, \mathtt{n}\}$. To start with, if $\mathtt{n} \in \{a,b,c\}$ then $(a \odot_E b) \odot_E c = \mathtt{n} = a \odot_E (b \odot_E c)$. Otherwise, we observe that the following hold:

- If $\mathtt{s} \in \{a,b,c\}$ and $\overline{\mathtt{s}} \notin \{a,b,c\}$ then $(a \odot_E b) \odot_E c = \mathtt{s} = a \odot_E (b \odot_E c)$
- If $\overline{\mathtt{s}} \in \{a,b,c\}$ and $\mathtt{s} \notin \{a,b,c\}$ then
  $(a \odot_E b) \odot_E c = \overline{\mathtt{s}} = a \odot_E (b \odot_E c)$, if only one of a,b,c is $\overline{\mathtt{s}}$;
  $(a \odot_E b) \odot_E c = \mathtt{n} = a \odot_E (b \odot_E c)$, if two of them are $\overline{\mathtt{s}}$;
  $(a \odot_E b) \odot_E c = \mathtt{n} = a \odot_E (b \odot_E c)$, if all of them are $\overline{\mathtt{s}}$;
- If $\overline{\mathtt{s}} \in \{a,b,c\}$ and $\mathtt{s} \in \{a,b,c\}$ then $(a \odot_E b) \odot_E c = \mathtt{n} = a \odot_E (b \odot_E c)$.
- If $a,b,c \in \{\mathtt{w}, \overline{\mathtt{w}}\}$ and $\overline{\mathtt{w}} \in \{a,b,c\}$ then $(a \odot_E b) \odot_E c = \overline{\mathtt{w}} = a \odot_E (b \odot_E c)$.
- If $a = b = c = \mathtt{w}$ then $(a \odot_E b) \odot_E c = \mathtt{w} = a \odot_E (b \odot_E c)$. □

Theorem 2. $\mathbb{S}_{boolE} = (\{\partial_{xy} \mid x,y \in \{\mathtt{w}, \mathtt{s}, \overline{\mathtt{w}}, \overline{\mathtt{s}}, \mathtt{n}\}\}, \oplus_E, \partial_{\mathtt{ww}})$ *is a conn-monoid.*

Proof. Follows from Proposition 9 and $\partial_{xy} \oplus_E \partial_{x'y'} = \partial_{x \odot_E x'\, y \odot_E y'}$. □

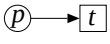
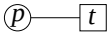
As $\mathbb{S}_{EN}$, $\mathbb{S}_{ENI}$ and $\mathbb{S}_{ENC}$ are all submonoids of $\mathbb{S}_{boolE}$, their associativity follows from Theorem 2.

6 Case studies

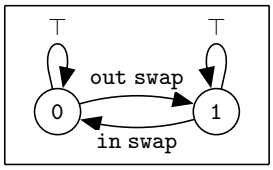
In this section, we discuss two further examples of submonoids of $\mathbb{S}_{boolE}$ and $\mathbb{S}_{boolS}$ corresponding to two classes of boolean nets, viz. Flip-Flop nets (for $\mathbb{S}_{boolE}$) and Switching SET-nets (for $\mathbb{S}_{boolS}$). Also, we look how conn-monoids and the encoding of connections can help to reason about a-priori and a-posteriori step semantics.

6.1 Flip-Flop nets

First we discuss a class of boolean nets for which the synthesis problem was studied in [13]. There are four possible connections between places and transitions in Flip-Flop nets (FF-nets):

- $F(p,t) = \top$ p and t are disconnected (or independent) 
- $F(p,t) = \mathtt{in}$ there is an arc from p to t 
- $F(p,t) = \mathtt{out}$ there is an arc from t to p 
- $F(p,t) = \mathtt{swap}$ there is a swap arc from t to p 

The first three connections work in the same way as those in EN-systems. A transition t is never disabled by a `swap` connection with a place p since p may be always switched from `0` to `1` and vice versa. Moreover, FF-nets are a conservative extension of EN-systems. Figure 11(a) depicts the net-type τ_{FFN}. The Cayley table is defined under the assumption (reflecting the EN-systems semantics) that transitions connected to places by `swap` will be in conflict with transitions connected to the same places by `in`, `out` or `swap` connections. Its encoded version is given in Figure 11(c) with `swap` encoded as $\partial_{\overline{\mathtt{ss}}}$. Note that $\mathbb{S}_{FFN}$ is a submonoid of $\mathbb{S}_{boolE}$.



(a)

$\oplus_{FFN}$	`out`	`in`	`swap`
`out`	$\bot$		
`in`	$\bot$	$\bot$	
`swap`	$\bot$	$\bot$	$\bot$

(b)

$\oplus_{FFN}$	$\partial_{\mathtt{sn}}$	$\partial_{\mathtt{n\overline{s}}}$	$\partial_{\overline{\mathtt{ss}}}$
$\partial_{\overline{\mathtt{sn}}}$	$\partial_{\mathtt{nn}}$		
$\partial_{\mathtt{n\overline{s}}}$	$\partial_{\mathtt{nn}}$	$\partial_{\mathtt{nn}}$	
$\partial_{\overline{\mathtt{ss}}}$	$\partial_{\mathtt{nn}}$	$\partial_{\mathtt{nn}}$	$\partial_{\mathtt{nn}}$

(c)

Fig. 11. net-type τ_{FFN} (a), the Cayley table of $\mathbb{S}_{FFN}$ (b), and its encoded version (c).

As the step semantics was never defined for FF-nets, instead of using an EN-systems-like semantics, we can also consider a more permissive treatment of combining connections for this class of nets with $\mathtt{swap} \oplus_{FFN'} \mathtt{out} = \mathtt{out}$, $\mathtt{swap} \oplus_{FFN'} \mathtt{in} = \mathtt{in}$ and $\mathtt{swap} \oplus_{FFN'} \mathtt{swap} = \mathtt{swap}$. The intuition behind this semantics is that if two transitions change the state of a place in the same way, they are not in conflict and both can be executed; moreover, `swap` lets the other connections decide the result of the execution of a step. Therefore, unlike in Figure 11(c) where `swap` was encoded as $\partial_{\overline{\mathtt{ss}}}$, it will now be encoded as $\partial_{\overline{\mathtt{ww}}}$ and considered a weak connection. Using this kind of step semantics, we obtain the Cayley table in Figure 12.

$\oplus_{FFN'}$	`out`	`in`	`swap`
`out`	$\bot$		
`in`	$\bot$	$\bot$	
`swap`	`out`	`in`	`swap`

(a)

$\oplus_{FFN'}$	$\partial_{\mathtt{sn}}$	$\partial_{\mathtt{n\overline{s}}}$	$\partial_{\overline{\mathtt{ww}}}$
$\partial_{\overline{\mathtt{sn}}}$	$\partial_{\mathtt{nn}}$		
$\partial_{\mathtt{n\overline{s}}}$	$\partial_{\mathtt{nn}}$	$\partial_{\mathtt{nn}}$	
$\partial_{\overline{\mathtt{ww}}}$	$\partial_{\mathtt{sn}}$	$\partial_{\mathtt{n\overline{s}}}$	$\partial_{\overline{\mathtt{ww}}}$

(b)

Fig. 12. Cayley table of $\mathbb{S}_{FFN'}$ (a) and its encoded version (b).

The resulting step semantics is less 'blocking'. Still, also $\mathbb{S}_{FFN'}$ is a submonoid of $\mathbb{S}_{boolE}$. Notice, that the change of encoding for `swap` was necessary. Not only because it follows from the behavioural analysis of this connection in the new step semantics, but also because of the calculations of relative 'strengths' in `0` and `1` (when connections

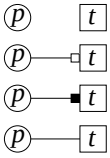
are combined) according to $\odot_E$. With encoding $\partial_{\overline{\mathtt{ss}}}$ for `swap`, the table in Figure 12(*b*) would be inconsistent: the first two rows would imply that combining $\overline{\mathtt{s}}$ with $\overline{\mathtt{s}}$ should give `n`, whereas the third one that it should give $\overline{\mathtt{s}}$.

The shared part of Cayley tables of $\mathbb{S}_{FFN}$ and $\mathbb{S}_{FFN'}$ (containing only EN-system connections) coincides with the Cayley table of $\mathbb{S}_{EN}$ and so FF-nets are a conservative extension of EN-systems under both interpretations of `swap`.

6.2 Switching SET-nets

We now turn to boolean τ-nets operating under the rules of the monoid $\mathbb{S}_{boolS}$. Suppose that we were given the task of designing a class of nets, called Switching SET-nets (SSET-nets) which should offer four types of connections that enable transitions to maintain the state of a place (do nothing), change the state to the opposite one (swap), reset the state to `0`, and reset the state to `1`. In other words, SSET-nets should support the following kinds of connections:

- $F(p,t) = \top$ — *p* and *t* are disconnected (or independent)
- $F(p,t) = \mathtt{set0}$ — *t* resets the state of *p* to `0`
- $F(p,t) = \mathtt{set1}$ — *t* resets the state of *p* to `1`
- $F(p,t) = \mathtt{swap}$ — there is a swap arc from *t* to *p*



To encode these connections, we have in principle 25 different options for each of them. This number can however immediately be reduced by noting that $\top$ should be $\partial_{\mathtt{ww}}$, and that the remaining three connections should adhere to the patterns shown in Figure 13. The patterns do not show which of the arrows are weak and which are strong and, in fact, the design task is basically to decide what strength to assign to them.

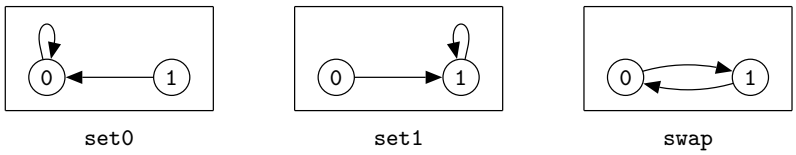


Fig. 13. Possible connection patterns for SSET-nets.

In all, there are $4^3 = 64$ different ways in which the connections `set0`, `set1` and `swap` could be interpreted, each potentially leading to a different variant of SSET-nets. To narrow down the choice, it is reasonable to require some further semantical properties like changing the state of a place is always strong. Then connection `set0` must be interpreted as $\partial_{\mathtt{w}\overline{\mathtt{s}}}$ or $\partial_{\mathtt{s}\overline{\mathtt{s}}}$, `set1` as $\partial_{\overline{\mathtt{s}}\mathtt{w}}$ or $\partial_{\overline{\mathtt{s}}\mathtt{s}}$, and `swap` as $\partial_{\overline{\mathtt{ss}}}$. Hence now we only need to consider 4 potential variants of SSET-nets. As a matter of fact, we can delete 3 of them after observing that in $\mathbb{S}_{boolS}$ we have $\partial_{\mathtt{s}\overline{\mathtt{s}}} \odot_S \partial_{\overline{\mathtt{ss}}} = \partial_{\mathtt{n}\overline{\mathtt{s}}}$ (`in`) and $\partial_{\overline{\mathtt{s}}\mathtt{s}} \odot_S \partial_{\overline{\mathtt{ss}}} = \partial_{\overline{\mathtt{s}}\mathtt{n}}$ (`out`). This, in turn, would mean that SSET-nets would also have to include the `in` and

out connections, respectively. However, according to our specific design task, SSET-nets should have four types of connections and so this extension is not allowed. As a consequence, there is only one possibility left with connection set0 interpreted as $\partial_{\mathtt{w}\overline{\mathtt{s}}}$, and set1 as $\partial_{\overline{\mathtt{s}}\mathtt{w}}$. The step semantics in such a case can be calculated and presented as the Cayley table of a submonoid of $\mathbb{S}_{boolS}$ as shown in Figure 14(a). Knowing the encoding for all the SSET-net connections we can create its decoded version as shown in Figure 14(b). It is also worth observing that $\oplus_{SSN}$ is a non-blocking operation.

$\oplus_{SSN}$	$\partial_{\mathtt{w}\overline{\mathtt{s}}}$	$\partial_{\overline{\mathtt{s}}\mathtt{w}}$	$\partial_{\overline{\mathtt{s}}\overline{\mathtt{s}}}$
$\partial_{\mathtt{w}\overline{\mathtt{s}}}$	$\partial_{\mathtt{w}\overline{\mathtt{s}}}$		
$\partial_{\overline{\mathtt{s}}\mathtt{w}}$	$\partial_{\overline{\mathtt{s}}\overline{\mathtt{s}}}$	$\partial_{\overline{\mathtt{s}}\mathtt{w}}$	
$\partial_{\overline{\mathtt{s}}\overline{\mathtt{s}}}$	$\partial_{\overline{\mathtt{s}}\overline{\mathtt{s}}}$	$\partial_{\overline{\mathtt{s}}\overline{\mathtt{s}}}$	$\partial_{\overline{\mathtt{s}}\overline{\mathtt{s}}}$

(a)

$\oplus_{SSN}$	set0	set1	swap
set0	set0		
set1	swap	set1	
swap	swap	swap	swap

(b)

Fig. 14. The encoded (a) and decoded (b) versions of the Cayley table of $\mathbb{S}_{SSN}$.

Suppose now that extending SSET-nets with additional connections is allowed. Then we could re-consider the 3 variants of SSET-nets we have just rejected, and see what is the minimal number of additional connections to make things work. Consider, in particular, set0 interpreted as $\partial_{\mathtt{s}\overline{\mathtt{s}}}$, and set1 as $\partial_{\overline{\mathtt{s}}\mathtt{s}}$. Then, as we have already seen, one needs to add in and out in order to construct a submonoid of $\mathbb{S}_{boolS}$. In fact, these are the only connections which are needed, as demonstrated by the Cayley table in Figure 15.

$\oplus_{SSN'}$	$\partial_{\mathtt{s}\overline{\mathtt{s}}}$	$\partial_{\overline{\mathtt{s}}\mathtt{s}}$	$\partial_{\overline{\mathtt{s}}\overline{\mathtt{s}}}$	$\partial_{\mathtt{n}\overline{\mathtt{s}}}$	$\partial_{\overline{\mathtt{s}}\mathtt{n}}$
$\partial_{\mathtt{s}\overline{\mathtt{s}}}$	$\partial_{\mathtt{s}\overline{\mathtt{s}}}$				
$\partial_{\overline{\mathtt{s}}\mathtt{s}}$	$\partial_{\mathtt{n}\mathtt{n}}$	$\partial_{\overline{\mathtt{s}}\mathtt{s}}$			
$\partial_{\overline{\mathtt{s}}\overline{\mathtt{s}}}$	$\partial_{\mathtt{n}\overline{\mathtt{s}}}$	$\partial_{\overline{\mathtt{s}}\mathtt{n}}$	$\partial_{\overline{\mathtt{s}}\overline{\mathtt{s}}}$		
$\partial_{\mathtt{n}\overline{\mathtt{s}}}$	$\partial_{\mathtt{n}\overline{\mathtt{s}}}$	$\partial_{\mathtt{n}\mathtt{n}}$	$\partial_{\mathtt{n}\overline{\mathtt{s}}}$	$\partial_{\mathtt{n}\overline{\mathtt{s}}}$	
$\partial_{\overline{\mathtt{s}}\mathtt{n}}$	$\partial_{\mathtt{n}\mathtt{n}}$	$\partial_{\overline{\mathtt{s}}\mathtt{n}}$	$\partial_{\overline{\mathtt{s}}\mathtt{n}}$	$\partial_{\mathtt{n}\mathtt{n}}$	$\partial_{\overline{\mathtt{s}}\mathtt{n}}$

(a)

$\oplus_{SSN'}$	set0	set1	swap	in	out
set0	set0				
set1	$\perp$	set1			
swap	in	out	swap		
in	in	$\perp$	in	in	
out	$\perp$	out	out	$\perp$	out

(b)

Fig. 15. The encoded (a) and decoded (b) versions of the Cayley table of $\mathbb{S}_{SSN'}$.

Looking closer at Figures 14 and 15, one can observe that although there is no conflict on acquiring resources, there is a different kind of conflict between connections, viz. a conflict of *intentions*. There is no problem with combining e.g., swap with set0 when the current state is 1, but when the current state is 0, one connection wants to reset the state to 1 and the other wants to keep it as 0. This conflict is solved differently in the two step semantics: According to $\oplus_{SSN}$, the connection that is 'strong' in 0 will dominate and the combined effect in 0 (of w and $\overline{\mathtt{s}}$) will be that of $\overline{\mathtt{s}}$, hence swap. On

the other hand, when $\oplus_{SSN'}$ is used, the conflict of intentions (between $\mathtt{s}$ and $\overline{\mathtt{s}}$) cannot be resolved and the combined result in $\mathtt{0}$ is $\mathtt{n}$.

6.3 a-posteriori vs. a-priori step semantics

In ENC-systems, the a-posteriori enabling condition for a step U is like the a-priori condition adopted throughout the paper, but with the additional requirement of the form ${}^\bullet U \cap {}^\blacklozenge U = U^\bullet \cap {}^\circ U = \varnothing$. In other words, in the a-posteriori semantics (of a net with context arcs) in order for a step to be enabled at a marking, the activator places of the transitions forming the step should keep their token and similarly, the inhibitor places of a step should still be empty after it has occurred [10]. As we shall argue in this subsection, conn-monoids can distinguish between a-priori and a-posteriori enabling of steps. For ENC-systems, the distinction between the two semantics is explicitly captured through the tables in Figure 16(a,b) which define submonoids of $\mathbb{S}_{boolE}$. To analyse these tables and verify their soundness, we rewrite them using the encoded versions of the connections as shown in Figure 16(c,d).

$\oplus_{ENC}$	out	in	inh	act
out	$\perp$			
in	$\perp$	$\perp$		
inh	out	$\perp$	inh	
act	$\perp$	in	$\perp$	act

(a)

$\oplus_{ENC}$	out	in	inh	act
out	$\perp$			
in	$\perp$	$\perp$		
inh	$\perp$	$\perp$	inh	
act	$\perp$	$\perp$	$\perp$	act

(b)

$\oplus_{ENC}$	$\partial_{\overline{\mathtt{s}}\mathtt{n}}$	$\partial_{\mathtt{n}\overline{\mathtt{s}}}$	$\partial_{\mathtt{w}\mathtt{n}}$	$\partial_{\mathtt{n}\mathtt{w}}$
$\partial_{\overline{\mathtt{s}}\mathtt{n}}$	$\partial_{\mathtt{n}\mathtt{n}}$			
$\partial_{\mathtt{n}\overline{\mathtt{s}}}$	$\partial_{\mathtt{n}\mathtt{n}}$	$\partial_{\mathtt{n}\mathtt{n}}$		
$\partial_{\mathtt{w}\mathtt{n}}$	$\partial_{\overline{\mathtt{s}}\mathtt{n}}$	$\partial_{\mathtt{n}\mathtt{n}}$	$\partial_{\mathtt{w}\mathtt{n}}$	
$\partial_{\mathtt{n}\mathtt{w}}$	$\partial_{\mathtt{n}\mathtt{n}}$	$\partial_{\mathtt{n}\overline{\mathtt{s}}}$	$\partial_{\mathtt{n}\mathtt{n}}$	$\partial_{\mathtt{n}\mathtt{w}}$

(c)

$\oplus_{ENC}$	$\partial_{\overline{\mathtt{s}}\mathtt{n}}$	$\partial_{\mathtt{n}\overline{\mathtt{s}}}$	$\partial_{\mathtt{s}\mathtt{n}}$	$\partial_{\mathtt{n}\mathtt{s}}$
$\partial_{\overline{\mathtt{s}}\mathtt{n}}$	$\partial_{\mathtt{n}\mathtt{n}}$			
$\partial_{\mathtt{n}\overline{\mathtt{s}}}$	$\partial_{\mathtt{n}\mathtt{n}}$	$\partial_{\mathtt{n}\mathtt{n}}$		
$\partial_{\mathtt{s}\mathtt{n}}$	$\partial_{\mathtt{n}\mathtt{n}}$	$\partial_{\mathtt{n}\mathtt{n}}$	$\partial_{\mathtt{s}\mathtt{n}}$	
$\partial_{\mathtt{n}\mathtt{s}}$	$\partial_{\mathtt{n}\mathtt{n}}$	$\partial_{\mathtt{n}\mathtt{n}}$	$\partial_{\mathtt{n}\mathtt{n}}$	$\partial_{\mathtt{n}\mathtt{s}}$

(d)

Fig. 16. Tables for the a-priori (a), and a-posteriori (b) semantics of ENC-systems, and their respective encoded versions, (c) and (d).

The $\mathtt{inh}$ and $\mathtt{act}$ connections have different encoding in the a-priori and a-posteriori semantics. In the a-priori semantics, they are respectively $\partial_{\mathtt{w}\mathtt{n}}$ and $\partial_{\mathtt{n}\mathtt{w}}$. In the a-posteriori semantics, both of these testing connections are 'strong', and encoded as $\partial_{\mathtt{s}\mathtt{n}}$ and $\partial_{\mathtt{n}\mathtt{s}}$, respectively. In the tables of Figure 16(a,b) this distinction is not visible, because the connections are written in both as $\mathtt{inh}$ and $\mathtt{act}$. This is a bit misleading as they have different interpretations in the two semantics. That these two testing connections are weak under the a-priori semantics and strong in the a-posteriori semantics should not come as a surprise. Intuitively, the a-priori semantics means that transitions executed in

a step take some time to complete, while in the a-posteriori semantics it is assumed that transitions take zero time to complete. As a consequence, for the a-posteriori semantics, both the state before and the state after executing a transition are taken into account when deciding its enabledness. Recall now that strong connections are those for which both the enabling conditions and the state after the execution are equally important. Hence the a-posteriori semantics makes the `inh` and `act` connections strong. In the a-priori semantics, on the other hand, only the state prior to the transition execution is taken into account.

7 Conclusions

In the past two decades, the problem of synthesising Petri nets from (step) transition systems describing their intended (concurrent) behaviour has received considerable attention. Assuming that the target class of Petri nets are τ-nets for some net-type (transition system) τ and using the theory of regions of transition systems, it has been possible to characterise in a general way all the solutions of the synthesis problem.

In this paper, we have investigated connection monoids — a crucial ingredient of τ-nets — for various classes of boolean nets. In our investigations, we have aimed at capturing a wide range of boolean nets which can be treated as τ-nets, starting from the fundamental class of EN-systems and the recently introduced SET-nets. In the process, we have considered both possible topological descriptions of connections in boolean nets, and the way in which such connections can be combined. This has resulted in a general scheme which allows one to consider different ways of combining connections which give rise to τ-nets, i.e., net classes for which the synthesis problem can be tackled using the theory of regions. We proposed to use a higher level monoidal structure based on the relative strength of different connections, leading to a more abstract encoding of specific connection monoids. We have provided examples of special (weak and strong) interpretations of individual connections, making it possible to capture subtle interactions between connections. We also investigated the extent to which they could be used to capture other behavioural features, such as conflict between transitions. The notions and results introduced and discussed in this paper can be seen as an attempt to provide a general approach to the 'classification' of boolean nets and their step semantics. It should also be stressed that our scheme of weak and strong connections is an example of how connections could be interpreted in steps (e.g., one could consider multi-level strength of connections as the reader might have already realised).

In our future work, we plan to investigate effective synthesis algorithms for different classes of boolean τ-nets. There already exist solutions for some specific net classes, e.g., it has been shown in [13] that FF-nets can be synthesised in polynomial time. On the other hand, the synthesis problem for EN-systems is NP-complete [2]. For safe Place/Transition nets (closely related to EN-systems), an efficient synthesis algorithm was developed and implemented in the PETRIFY tool [4]. Recently, a new algorithm was proposed for the synthesis of EN-systems [1], and it turns out that it can be extended to deal also with SET-nets [1]. We feel that the techniques introduced in this paper, such as looking at the behaviours at `0` and `1` separately, may help in the development of algorithms for synthesising boolean τ-nets from step transition systems.

Acknowledgements

This research was supported by the Pascal Chair award from the Leiden Institute of Advanced Computer Science (LIACS) of Leiden University.

References

1. Badouel, E., Private communication (2011)
2. Badouel, E., Bernardinello, L., Darondeau, Ph., The Synthesis Problem for Elementary Net Systems is NP-complete, Theoretical Computer Science, **186**, 107–134 (1997)
3. Badouel, E., Darondeau, Ph., Theory of Regions, In: Reisig, W., Rozenberg, G. (eds.), Lectures on Petri Nets I: Basic Models, Advances in Petri Nets, Lecture Notes in Computer Science, **1491**, Springer-Verlag, Berlin Heidelberg New York, 529–586 (1998)
4. Cortadella, J., Kishinevsky, M., Kondratyev, A., Lavagno, L., Yakovlev, A., Logic Synthesis of Asynchronous Controllers and Interfaces, Springer (2002)
5. Darondeau, P., Koutny, M., Pietkiewicz-Koutny, M., Yakovlev, A., Synthesis of Nets with Step Firing Policies, Fundamenta Informaticae, **94**, 275–303 (2009)
6. Ehrenfeucht, A., Rozenberg, G., Partial 2-structures; Part I: Basic Notions and Representation Problem, and Part II: State Spaces of Concurrent Systems, Acta Informatica, **27**, 315–368 (1990)
7. Ehrenfeucht, A., Rozenberg, G., Reaction Systems, Fundamenta Informaticae, **76**, 1–18 (2006)
8. Kleijn, J., Koutny, M., Rozenberg, G., Modelling Reaction Systems with Petri Nets, CEUR Workshop Proceedings, **724**, 36–52 (2011)
9. Kleijn, J., Koutny, M., Pietkiewicz-Koutny, M., Rozenberg, G., Classifying Boolean Nets for Region-based Synthesis. CEUR Workshop Proceedings, **725**, 5-21 (2011)
10. Montanari, U., Rossi, F., Contextual Nets. Acta Informatica **32**, 545–596 (1995)
11. Pietkiewicz-Koutny, M., The Synthesis Problem for Elementary Net Systems with Inhibitor Arcs, Fundamenta Informaticae, **40**, 251–283 (1999)
12. Rozenberg, G., Engelfriet, J., Elementary Net Systems, Lectures on Petri Nets I: Basic Models, W.Reisig and G.Rozenberg (eds.), Lecture Notes in Computer Science, **1491**, Springer-Verlag, Berlin Heidelberg New York, 173–187 (2006)
13. Schmitt, V., Flip-Flop Nets, In: Puech, C., Reischuk, H. (eds.), STACS 1996, Lecture Notes in Computer Science, **1046**, Springer-Verlag, Berlin Heidelberg New York, 517–528 (1996)