COMPUTING SCIENCE

Causality in Structured Occurrence Nets

Jetty Kleijn and Maciej Koutny

TECHNICAL REPORT SERIES

No. CS-TR-1290 November 2011

TECHNICAL REPORT SERIES

No. CS-TR-1290 November, 2011

Causality in Structured Occurrence Nets

J. Kleijn, M. Koutny

Abstract

Structured occurrence nets consist of multiple occurrence nets - each recording causality and concurrency in an execution of a component of a concurrent system. These occurrence nets are linked together by means of various types of relationships, aimed at representing dependencies between communicating and evolving sub-systems. In this paper, we investigate causality in the basic class of communication structured occurrence nets cso-nets). We start by introducing the corresponding system-level model of communication structured Place Transition Nets (cspt-nets) which extend Place Transition Nets with an explicit structuring into communicating sub-systems and process interaction based on a combination of synchronous and asynchronous communication. After that we develop a cso-net based process semantics for cspt-nets showing that causality in cso-nets is underpinned by stratified order structures extending causal partial orders with weak causality.

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

Bibliographical details

KLEIJN, J., KOUTNY, M.

Causality in Structured Occurrence Nets
[By] J. Kleijn, M. Koutny
Newcastle upon Tyne: Newcastle University: Computing Science, 2011.

(Newcastle University, Computing Science, Technical Report Series, No. CS-TR-1290)

Added entries

NEWCASTLE UNIVERSITY
Computing Science. Technical Report Series. CS-TR-1290

Abstract

Structured occurrence nets consist of multiple occurrence nets - each recording causality and concurrency in an execution of a component of a concurrent system. These occurrence nets are linked together by means of various types of relationships, aimed at representing dependencies between communicating and evolving sub-systems. In this paper, we investigate causality in the basic class of communication structured occurrence nets cso-nets). We start by introducing the corresponding system-level model of communication structured Place Transition Nets (cspt-nets) which extend Place Transition Nets with an explicit structuring into communicating sub-systems and process interaction based on a combination of synchronous and asynchronous communication. After that we develop a cso-net based process semantics for cspt-nets showing that causality in cso-nets is underpinned by stratified order structures extending causal partial orders with weak causality.

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.

Suggested keywords

CONCURRENCY
OCCURRENCE NET
STRUCTURED OCCURRENCE NET
PLACE TRANSITION NET
SEMANTICAL FRAMEWORK
CAUSALITY SEMANTICS
PROCESS SEMANTICS
SYNCHRONOUS AND ASYNCHRONOUS COMMUNICATION

Causality in Structured Occurrence Nets

Jetty Kleijn[1] and Maciej Koutny[2]

[1] LIACS, Leiden University
P.O.Box 9512, NL-2300 RA Leiden, The Netherlands
kleijn@liacs.nl
[2] School of Computing Science, Newcastle University
Newcastle upon Tyne, NE1 7RU, United Kingdom
maciej.koutny@ncl.ac.uk

Abstract. Structured occurrence nets consist of multiple occurrence nets — each recording causality and concurrency in an execution of a component of a concurrent system. These occurrence nets are linked together by means of various types of relationships, aimed at representing dependencies between communicating and evolving sub-systems. In this paper, we investigate causality in the basic class of communication structured occurrence nets (CSO-nets). We start by introducing the corresponding system-level model of communication structured Place Transition Nets (CSPT-nets) which extend Place Transition Nets with an explicit structuring into communicating sub-systems and process interaction based on a combination of synchronous and asynchronous communication. After that we develop a CSO-net based process semantics for CSPT-nets showing that causality in CSO-nets is underpinned by stratified order structures extending causal partial orders with weak causality.

Keywords: concurrency, occurrence net, structured occurrence net, place transition net, semantical framework, causality semantics, process semantics, synchronous and asynchronous communication.

1 Introduction

Occurrence nets [2] are acyclic Petri nets that can be used to record execution histories of concurrent systems, in particular, the concurrency and causality relations between events. Each occurrence net defines a partial order of its transition occurrences (representing the events) in which causally related occurrences are ordered while concurrent transition occurrences remain unordered. Occurrence nets are typically used to capture the causal semantics of standard net classes like Elementary Net Systems and Place Transition Nets [6, 13, 22].

In structured occurrence nets, invented by Brian Randell and then formally elaborated in [15, 20, 21], occurrence nets are combined by various types of relationships representing dependencies between communicating and evolving sub-systems. Thus structured occurrence nets make use of temporal and spatial abstractions that can be seen as consequences of how a system has been conceived rather than as interpretations generated by the analysis of the system. There are different ways to structure occurrence nets. In this paper, we start from communication structured occurrence nets

(CSO-nets) which are the simplest variant of structured occurrence nets. A CSO-net describes a combination of occurrence nets that proceed concurrently and communicate occasionally.

Figure 1 shows a communication structured occurrence net consisting of two occurrence nets that communicate along the thick dashed arc and edge. Note that these communication links represent a direct (causal) relationship between transitions and connect them directly unlike the usual arcs in Petri nets which can only relate places to transitions or vice versa. The communication flow represented by a thick dashed arc is unidirectional from source to target transition and indicates that the latter cannot occur before the former. In other words, in any execution of this occurrence net, either the source event of the communication precedes the target event or they are executed *synchronously* (in one step). The thick dashed edge is an abbreviation; it stands for the combination of two such arcs, one in either direction. Hence, the two transitions involved are meant to be executed synchronously.

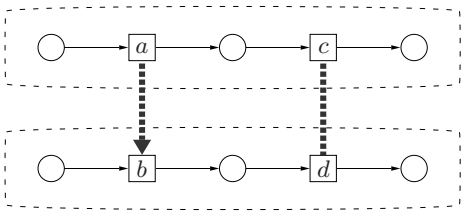

Fig. 1. A communication structured occurrence net.

In this paper, we investigate the causality structure of CSO-nets. We establish that stratified order structures [8, 11, 12], an extension of partial orders, adequately describe the relations between the transitions of a CSO-net. Moreover, we identify a system-level model with an operational semantics that fits well with the concept of CSO-nets. This model is an extension of the well-known Place Transition Nets (PT-nets) [6].

Like most Petri net models, PT-nets are an essentially asynchronous concurrent model with a sequential (firing sequence) semantics and a step semantics based on multisets of transitions that may occur simultaneously when enough resources are available for such a combined occurrence. Consequently, whenever a step occurs each of its transitions (or more general, each of its sub-multisets) could also have occurred (the so-called sub-step property). The PT-net model has no (structural) possibility to express that an enabled transition has to (wait in order to) synchronise with another one. On the other hand, it is not difficult to make an otherwise enabled transition wait for the occurrence of a second one by using a message (in the form of a token left by the second one in a special input place of the first transition). These considerations motivate the introduction of *channel places* in this paper. These channels will be used to implement the causality expressed through the communication arcs (the thick dashed edges) in the original CSO-nets.

Figure 2 shows a PT-net with three channel places corresponding to the CSO-net of Figure 1 in its default initial state. Intuitively, with the channel connecting transition a

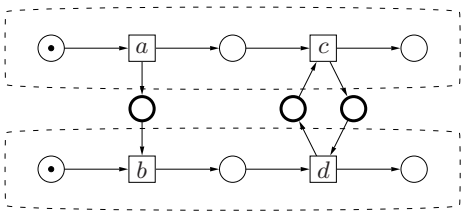

Fig. 2. A PT-net with explicit channel places implementing the desired communication protocols.

(its input) to transition b (its output), we have the following operational semantics: If a occurs it adds a 'message' (a token) to the channel; this message may either remain there to serve later as input to b (the usual asynchronous communication of PT-nets), or be directly picked up by b in the *same* step (synchronous communication). Usually, synchronous communication implies that a sender waits for the receipt of the message before proceeding. In the (new) Petri net interpretation: synchronous communication entails an instantaneous receipt of the token by b. The channel place moreover allows asynchronous communication. Therefore, we will refer to these channel places as *a/sync* channels. The communication connection provided by a/sync channels can be compared to a telephone connection with an answering machine: either the caller waits for the callee to answer the phone (and then they communicate synchronously), or the caller leaves a message on the answering machine to be listened to later by the callee.
Note that in the initial marking of Figure 2, the step $\{a, b\}$ can be executed as well as a followed by b, but b cannot be executed before a. Moreover, if there are two (initially empty) channels connecting c to d and d to c, as in Figure 2, then c and d can only occur synchronously as a step $\{c, d\}$.

The paper is organised as follows. In the next section we provide basic definitions concerning stratified order structures which are the causality structures used in this paper. Section 3 introduces communication structured PT-nets (CSPT-nets) which extend PT-nets with an explicit structuring into sub-systems communicating through a/sync channels. After that, in Sections 4 and 5, we develop a cso-nets based process semantics for CSPT-nets, following a generic approach (semantical framework) which has been used successfully to define processes of PT-nets. In particular, we conclude that causality in cso-nets is underpinned by stratified order structures. As full proofs of various results presented in this paper are omitted, Section 6 outlines the way in which the semantical framework of [13] can support their efficient development. Section 7 contains remarks on related work, and Section 8 concludes the paper.

2 Causality structures

Causality structures, such as causal partial orders, can be seen as instances of more general relational structures, where a *relational structure* is a tuple $R = (X, Q_1, \ldots, Q_n)$ with X being a finite *domain*, and the Q_i's binary relations on X. For relational structures with the same domain and arity, R and R', we write $R \subseteq R'$ if the subset inclusion holds component-wise. The intersection $\bigcap \mathcal{R}$ of a non-empty set $\mathcal{R}$ of relational structures with the same arity and domain is also defined component-wise.

To capture causal relationships between events occurring in a concurrent system history, one can use a suitable *ordering relation*. In its basic form, such a relation is a partial order, generated from local causalities (reflecting the generally accepted view that causality is transitive and acyclic). A *partially ordered set* (or poset) is a relational structure $po = (X, \prec)$ consisting of a finite set X and a transitive and irreflexive relation $\prec$ on X. Two distinct elements a, b of X are *unordered*, $a \frown b$, if neither $a \prec b$ nor $b \prec a$ holds. Note that if a poset is interpreted as capturing causal relationships in a run or history of a concurrent system then, for two distinct events a and b, $a \prec b$ means that a was a causal predecessor of b, while $a \frown b$ means that a and b were independently executed events. Intuitively, $\prec$ represents the 'earlier than' relationship in X shared by all observations of the history represented by po.

Although causal partial orders have found several applications in semantics and analyses of concurrent systems, for systems with a complex structure, partial orders may need to be extended to more expressive order structures which support additional relations between events, as described next. A *stratified order structure* (or SO-structure, see [8, 11, 12]) $sos = (X, \prec, \sqsubset)$ comprises two binary relations, $\prec$ (*causality*) and $\sqsubset$ (*weak causality*) on a finite set X such that, for all $x, y, z \in X$:

$$S1: \ x \not\sqsubset x \qquad S3: \ x \sqsubset y \sqsubset z \ \wedge \ x \neq z \implies x \sqsubset z$$
$$S2: \ x \prec y \implies x \sqsubset y \qquad S4: \ x \sqsubset y \prec z \ \vee \ x \prec y \sqsubset z \implies x \prec z \ .$$

Intuitively, $\prec$ represents the 'earlier than' relationship in X, and $\sqsubset$ the 'not later than' relationship. Accordingly, $\prec$ is a partial order, and $x \prec y$ implies $y \not\sqsubset x$.

Individual observations of a concurrent systems are often represented by sequences of groups of simultaneously occurring events (step sequences). Hence one can consider *(singular) step sequences* which are sequences of mutually disjoint non-empty sets $\chi = X_1 \ldots X_k$ ($k \geq 0$). Singular step sequences correspond in a natural way to a special class of posets. A poset $spo = (X, \prec)$ is *stratified* if $a \frown b \frown c$ implies $a \frown c$, for all distinct a, b, c in X. Note that if a poset is interpreted as an observation of concurrent system behaviour, then $a \prec b$ means that a was observed before b, while $a \frown b$ means that a and b were observed as simultaneous. Now, given a singular step sequence $\chi = X_1 \ldots X_k$, we have that $\mathsf{spo}(\chi) = (\bigcup_i X_i, \bigcup_{i<j} X_i \times X_j)$ is a stratified poset. Conversely, each stratified poset spo induces a unique singular step sequence $\mathsf{steps}(spo))$ satisfying $spo = \mathsf{spo}(\mathsf{steps}(spo))$. We may therefore identify each stratified poset spo with $\mathsf{steps}(spo)$ or, equivalently, each singular step sequence χ with $\mathsf{spo}(\chi)$.

Finally, we relate SO-structures with their step sequence (or stratified poset) observations. First, it is easy to see that if $spo = (X, \prec)$ is a stratified poset, then $\mathsf{sos}(spo) = (X, \prec, \prec \cup \frown)$ is an SO-structure. One can then identify executions corresponding to (or consistent with) a given SO-structure sos: a stratified poset spo is an *extension* of sos if $sos \subseteq \mathsf{sos}(spo)$. We denote this by $spo \in \mathsf{ext}(sos)$.

Fact 1 *For every* SO*-structure* sos, $\mathsf{ext}(sos) \neq \varnothing$ *and* $sos = \bigcap \mathsf{sos}(\mathsf{ext}(sos))$.

In other words, one can recover an SO-structure by intersecting its stratified order extensions, in a similar way as one recovers a poset from its linearisations.

One can also generate SO-structures from local relationships between events, in the same way as partial orders are generated from acyclic relations. A *pre-*SO*-structure* is a relational structure $\varrho = (X, \prec, \sqsubset)$ such that the relation $\gamma \circ \prec \circ \gamma$ is irreflexive, where $\gamma = (\prec \cup \sqsubset)^*$. Then the *so-closure* is $\varrho^{\mathsf{so}} = (X, \gamma \circ \prec \circ \gamma, \gamma \setminus id_X)$. Note that in a pre-SO-structure ϱ there are no $x_0, x_1, \ldots, x_n = x_0$ such that $x_0 \prec x_1$ and, for all $0 < i < n$, $x_i \prec x_{i+1}$ or $x_i \sqsubset x_{i+1}$. This can be regarded as a kind of acyclicity.

Fact 2 *For every pre-*SO*-structure ϱ, ϱ^{so} is an* SO*-structure.*

3 PT-nets and CSPT-nets

In this section, we first recall the standard definitions concerning PT-nets, including their concurrency semantics based on step sequences (multisets of transitions executed simultaneously). We then extend the concept of PT-net by composing several nets into a single system by letting them communicate via a/sync channel places.

Recall that a multiset over a set X is a function $\mu : X \to \{0, 1, 2, \ldots\}$. In this paper, a multiset may be represented by listing its elements with repetitions, e.g., $\mu = \{y, y, z\}$ is a multiset such that $\mu(y) = 2$, $\mu(z) = 1$, and $\mu(x) = 0$ otherwise. We treat sets as multisets without repetitions, and applying a labelling function ℓ to a set $Z = \{z_1, \ldots, z_k\} \subseteq X$ yields a multiset $\ell(Z) = \{\ell(z_1), \ldots, \ell(z_k)\}$.

A *Place Transition Net* (or PT-net) is a tuple $PT = (P, T, F, M_{init})$ such that P and T are disjoint finite sets of nodes, called respectively *places* and *transitions*, $F \subseteq (T \times P) \cup (P \times T)$ is the *flow* relation, and M_{init} is the *initial* marking, where a *marking* is any multiset of places. The *inputs* and *outputs* of a node x are the sets ${}^\bullet x = \{y \mid (y, x) \in F\}$ and $x^\bullet = \{y \mid (x, y) \in F\}$. It is assumed that the inputs and outputs of any transition are non-empty.

In diagrams, places are represented by circles, transitions by rectangles, the flow relation by directed arcs, and a marking (global state) by *tokens* (small black dots) drawn inside places. Figure $3(a)$ depicts a PT-net representing a producer, an unbounded asynchronous buffer (the middle place b_0), and two consumers. The producer can execute: m (making an item), a (adding a new item to the buffer), and f (failing to add an item). Each of the two consumers represented by the tokens in place p_3 can cyclically execute: g (getting an item), and u (using the item). Initially, the system is in the marking $M_{init} = \{b_0, p_1, p_3, p_3\}$.

The operational behaviour of PT can be captured by its step sequences. A *step* U is a multiset of transitions. It is *enabled* at a marking M if $M(p) \geq \sum_{t \in p^\bullet} U(t)$, for every place p. In such a case, the *execution* of U leads to the marking M' given by:

$$M'(p) = M(p) - \sum_{t \in p^\bullet} U(t) + \sum_{t \in {}^\bullet p} U(t) \ ,$$

for every place p. We denote this by $M[U\rangle M'$. Then a *step sequence* of PT is a sequence $\chi = U_1 \ldots U_n$ $(n \geq 0)$ of steps such that there are markings $M_1, \ldots, M_n$ satisfying:

$$M_{init}[U_1\rangle M_1, \ldots, M_{n-1}[U_n\rangle M_n \ .$$

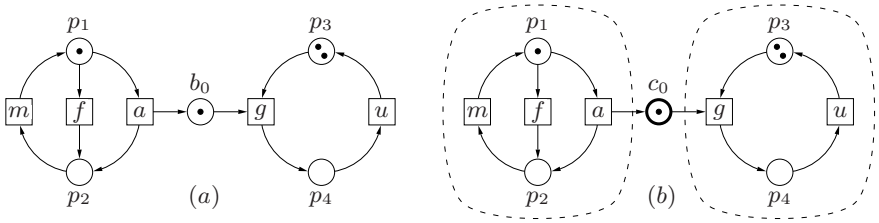

Fig. 3. Two models of a 1-producer/2-consumers system: (a) PT-net PT_0 (with *asynchronous* buffer place b_0), and (b) CSPT-net $CSPT_0$ (with *a/sync* channel place c_0).

We denote this by $\chi \in \mathsf{steps}(PT)$, and call M_n a *reachable* marking. Note that singular step sequences can be considered as a special kind of step sequences with steps being disjoint sets.

PT-nets are a fundamental class of Petri nets, and we now introduce a derived fundamental class of structured Petri nets capable of generating structured occurrence nets. The key idea is to replace the asynchronous interprocess communication like that in Figure $3(a)$ by the more flexible a/sync communication with component PT-nets being linked through *channel* places.

A *communication structured place transition net* (or CSPT-net) is a tuple:

$$CSPT = (PT_1, \ldots, PT_k, P_0, F_0, M_0) \quad (k \geq 1)$$

such that each $PT_i = (P_i, T_i, F_i, M_i)$ is a component PT-net, P_0 is a set of *(a/sync) channel* places, M_0 is a multiset of channel places, and $F_0 \subseteq (T \times P_0) \cup (P_0 \times T)$, where $T = \bigcup_{i \geq 1} T_i$. It is assumed that the nodes of the PT_i's and P_0 are disjoint and, for every channel place c, ${}^\bullet c = \{c_{in}\}$ and $c^\bullet = \{c_{out}\}$, where c_{in} and c_{out} are transitions belonging to two distinct PT_i's. The initial marking M_{init} of $CSPT$ is the sum of all the M_i's, including M_0. The semantics of $CSPT$ is defined as before except that a step of transitions U is *enabled* at a marking M if $M(p) \geq \sum_{t \in p^\bullet} U(t)$, for every non-channel place p, and $M(c) + U(c_{in}) \geq U(c_{out})$, for every channel place c. Thus, in contrast to the usual approaches to step semantics, steps of transitions executed in CSPT-nets do not necessarily consist of accumulated (allowed combinations of) enabled single transitions.

The dot-notation and drawing conventions are as before except that the channel places are drawn with thick border lines, and each component PT-net is enclosed inside a dashed box. Figure $3(b)$ depicts a CSPT-net derived from the PT-net of Figure $3(a)$ modelling the 1-producer/2-consumers system by replacing the standard buffer place b_0 with channel place c_0. One can easily check that for the nets in Figure 3, we have $\mathsf{steps}(PT_0) \subset \mathsf{steps}(CSPT_0)$ as, for instance:

$$\{g\}\{a,u\}\{m\}\{a,g,g\} \in \mathsf{steps}(CSPT_0) \setminus \mathsf{steps}(PT_0) \ .$$

This exemplifies a fundamental difference between asynchronous communication via the buffer place b_0 and a/sync communication via the channel place c_0. Intuitively, the execution of step $\{a,g,g\}$ combines a *synchronous* communication involving a and

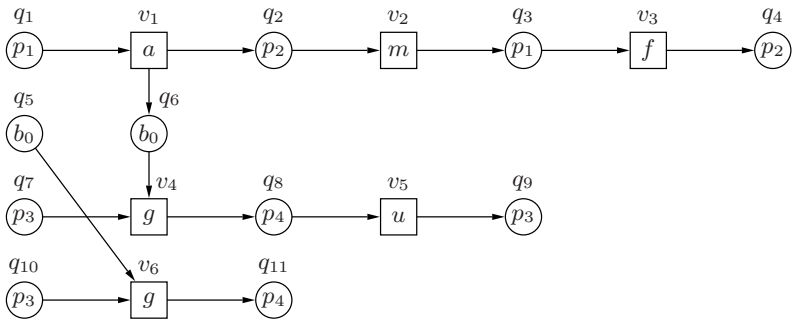

Fig. 4. An occurrence net ON_0 (labels are shown inside the nodes).

one of the g's with an *asynchronous* communication involving the other g and the token inserted into c_0 during the execution of step $\{a, u\}$.

4 Occurrence nets and structured occurrence nets

Having described two system-level classes of Petri nets, viz. PT-nets and CSPT-nets, we now proceed to present occurrence nets and communication structured occurrence nets, two similarly related classes of (behaviour-level) Petri nets used to represent concurrent histories.

An *occurrence net* is a tuple $ON = (P', T', F', \ell)$ such that P', T' and F' are places, transitions and flow relation as before, and ℓ is a labelling for $P' \cup T'$. It is assumed that $|{}^\bullet p| \leq 1$ and $|p^\bullet| \leq 1$, for every place p, and that F' is acyclic. The rule for executing steps is the same as in the case of PT-nets. The default *initial* marking M_{init}^{ON} and the *final* M_{fin}^{ON} marking of ON are sets respectively consisting of all places without inputs and all places without outputs. With this notion of the initial and final markings, the behaviour of ON is captured by the set $\mathsf{steps}(ON)$ comprising all step sequences χ satisfying $M_{init}^{ON}[\chi\rangle M_{fin}^{ON}$. For each step sequence $\chi = U_1 \ldots U_n$ belonging to $\mathsf{steps}(ON)$, we will denote by $\phi(\chi)$ the sequence of multisets $\ell(U_1) \ldots \ell(U_n)$.

Due to the acyclicity of the flow relation, and the lack of multiple inputs (or outputs) of places, each transition in T' appears exactly *once* in any step sequence χ belonging to $\mathsf{steps}(ON)$. Hence χ is a singular step sequence, and $\mathsf{spo}(\chi)$ is a well-defined stratified poset. Figure 4 shows an occurrence net with the labels coming from the PT-net shown in Figure 3(a). We observe that $M_{init}^{ON_0} = \{q_1, q_5, q_7, q_{10}\}$ and $M_{fin}^{ON_0} = \{q_4, q_9, q_{11}\}$ as well as:

$$M_{init}^{ON_0}\Big[\{v_1, v_6\}\{v_2, v_4\}\{v_3, v_5\}\Big\rangle M_{fin}^{ON_0}\ .$$

Note that $\phi(\{v_1, v_6\}\{v_2, v_4\}\{v_3, v_5\}) = \{a, g\}\{m, g\}\{f, u\}$ is a valid step sequence of PT-net of Figure 3(a).

Similar to the way CSPT-nets were derived from PT-nets, we extend occurrence nets with a/sync channel places.

A *communication structured occurrence net* (or CSO-net) is a tuple

$$CSON = (ON_1, \ldots, ON_k, P'_0, F'_0, \ell'_0) \quad (k \geq 1)$$

such that each $ON_i = (P'_i, T'_i, F'_i, \ell'_i)$ is an occurrence net, P'_0 is a set of channel places with a labelling ℓ'_0, and $F'_0 \subseteq (T' \times P'_0) \cup (P'_0 \times T')$, where $T' = \bigcup_{i \geq 1} T'_i$. It is further assumed that:

- the nodes of the ON_i's are disjoint.
- for every channel place c, $|{}^\bullet c| \leq 1$ and $|c^\bullet| \leq 1$, and the input and output transitions of c belong to distinct ON_i's.
- the relation $\varrho_{CSON} = (T', \bigcup_{i \geq 1}(F'_i \circ F'_i)|_{T'_i \times T'_i}, (F'_0 \circ F'_0)|_{T' \times T'})$ is a pre-SO-structure. This, in particular, means that the relation:

$$\mathsf{sos}(CSON) = \varrho^{\mathsf{so}}_{CSON}$$

is a well-defined SO-structure *generated* by $CSON$ (see Fact 2).

The default *initial* M^{CSON}_{init} marking of $CSON$ and the *final* M^{CSON}_{fin} marking of $CSON$ are the sum of the default initial markings of the ON_i's, together with all the channel places with no inputs, respectively the sum of the default final markings of the ON_i's, together with all the channel places with no outputs. The set $\mathsf{steps}(CSON)$ of step sequences executed by $CSON$ is then defined as for occurrence nets, assuming that channel places are treated as in the case of CSPT-nets. As before, also for a step sequence $\chi = U_1 \ldots U_n$ belonging to $\mathsf{steps}(CSON)$, we will denote by $\phi(\chi)$ the sequence of multisets $\ell(U_1) \ldots \ell(U_n)$. It can easily be seen that step sequences belonging to $\mathsf{steps}(CSON)$ are singular. Moreover, $\mathsf{steps}(CSON)$ is non-empty.

Intuitively, $\mathsf{sos}(CSON)$ is a causal structure underpinning $CSON$ which can be justified by the fact that the executions of a CSO-net are fully consistent with the underlying causal structure:

Theorem 1. $\mathsf{steps}(CSON) = \mathsf{ext}(\mathsf{sos}(CSON))$.

Moreover, the underlying causal structure can be obtained by intersecting all the orderings induced by the step sequences of $CSON$:

Theorem 2. $\mathsf{sos}(CSON) = \bigcap \mathsf{spo}(\mathsf{steps}(CSON))$.

Figure 5 shows a CSO-net $CSON_0$ with the labels coming from the CSPT-net of Figure 3(b). We observe that $M^{CSON_0}_{init} = \{q_1, q_5, q_7, q_{10}\}$ and $M^{CSON_0}_{fin} = \{q_4, q_9, q_{11}\}$ as well as:

$$M^{CSON_0}_{init} \Big[\{v_1, v_4, v_6\}\{v_2, v_5\}\{v_3\}\Big\rangle M^{CSON_0}_{fin} \ .$$

Note that $\phi(\{v_1, v_4, v_6\}\{v_2, v_5\}\{v_3\}) = \{a, g, g\}\{m, u\}\{f\}$ is a valid step sequence of the CSPT-net of Figure 3(b). Moreover, the underpinning causal structure is given by the SO-closure of the following relation:

$$\varrho_{CSON_0} = \Big(\big\{v_1, v_2, v_3, v_4, v_5, v_6\big\}\,,\ \big\{(v_1, v_2), (v_2, v_3), (v_4, v_5)\big\}\,,\ \big\{(v_1, v_4)\big\}\Big)\ .$$

Hence there is a weak causal dependency between v_1 and v_4, meaning that v_4 cannot be executed before v_1, only after v_1 or simultaneously with v_1.

It is also interesting to re-visit the CSO-net $CSON$ of Figure 2 which contains a cycle. This is not a problem as it simply indicates that the transitions labelled by c and d are synchronised. Formally, we have $\phi(\mathsf{steps}(CSON)) = \big\{\{a, b\}\{c, d\}, \{a\}\{b\}\{c, d\}\big\}$.

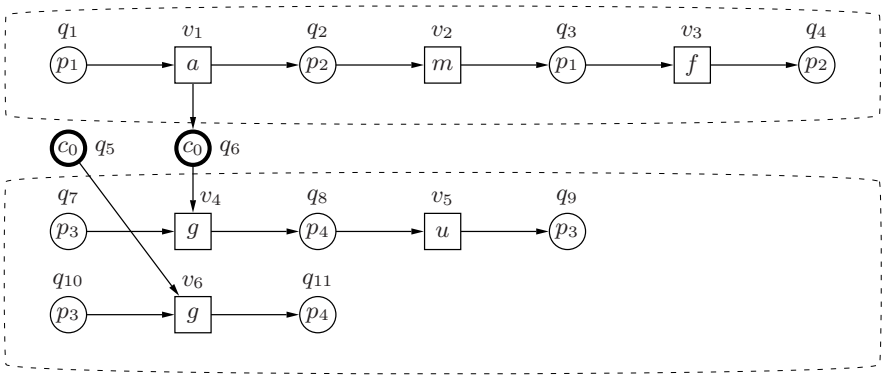

Fig. 5. A communication structured occurrence net $CSON_0$.

5 cso-net semantics of cspt-nets

We have described two new classes of Petri nets, CSPT-nets and CSO-nets, based on explicit structuring and a/sync communication between component nets. CSPT-nets are specifications of systems' designs, whereas the role of CSO-nets is to capture behaviours of such systems. We have also argued that CSO-nets are underpinned by SO-structures which are causality structures extending causal partial orders, and so they in turn might provide a causality semantics of CSPT-nets. Our next goal is to clarify how CSO-nets can be derived from CSPT-nets in a way which is consistent with their operational semantics.

In this section we formalise two key definitions. The first one provides a full characterisation of CSO-nets corresponding to the behaviours of a given CSPT-net.

A *process* of CSPT-net $CSPT = (PT_1, \ldots, PT_k, P_0, F_0, M_0)$ is a CSO-net:

$$CSON = (ON_1, \ldots, ON_k, P_0', F_0', \ell_0')$$

with the overall labelling ℓ (determining its components' labeling and ℓ_0') such that it:

- labels places of ON_i with places of PT_i, for each i.
- labels transitions of ON_i with transitions of PT_i, for each i.
- labels channel places of $CSON$ with channel places of $CSPT$.
- yields $\ell(M_{init}^{CSON}) = M_{init}$.
- is injective on ${}^\bullet t$ and $t^\bullet$ and, moreover $\ell({}^\bullet t) = {}^\bullet \ell(t)$ and $\ell(t^\bullet) = \ell(t)^\bullet$, for all transitions t of $CSON$.

We denote this by $CSON \in \mathsf{proc}(CSPT)$. For example, $CSON_0 \in \mathsf{proc}(CSPT_0)$, where $CSPT_0$ and $CSON_0$ are respectively the nets in Figures 3(b) and 5.

The soundness of the process definition can be justified by showing that the step sequences of CSO-processes provide an exact representation of step sequences of the original CSPT-net:

Theorem 3. $\mathsf{steps}(CSPT) = \phi(\mathsf{steps}(\mathsf{proc}(CSPT)))$.

Moreover, one can see that a similar property holds also for reachable markings.

The above notion of process is an 'axiomatic' characterisation and provides no clues as to how to generate processes of CSPT-nets in practice. This issue is addressed by the second definition based on so-called net *unfolding*, itself a prominent technique behind model checking tools [7, 17, 18].

A CSO-net *generated* by a step sequence $\chi = U_1 \ldots U_n$ of $CSPT$ is the last element in the sequence $CSON_0, \ldots, CSON_n$ where each

$$CSON_j = (\hat{ON}_1^j, \ldots, \hat{ON}_k^j, \hat{P}_0^j, \hat{F}_0^j, \hat{\ell}^j)$$

is a CSO-net constructed in the following way (below the label of a node x^y is x):

Step 0: We set $\hat{P}_0^0 = \{c^m \mid c \in P_0 \wedge 1 \leq m \leq M_{init}(c)\}$ and $\hat{F}_0^0 = \varnothing$. Moreover, for every $i = 1, \ldots, k$:

$$\hat{P}_i^0 = \{p^m \mid p \in P_i \wedge 1 \leq m \leq M_{init}(p)\} \text{ and } \hat{T}_i^0 = \hat{F}_i^0 = \varnothing \ .$$

Step j: Given $CSON_{j-1}$ we extend the sets of nodes and arcs as follows (below $i = 1, \ldots, k$ and $\triangle x$ denotes the number of the nodes of $CSON_{j-1}$ labelled by x):

$$\begin{aligned}
\hat{P}_0^j &= \hat{P}_0^{j-1} \cup \{c^{m+\triangle c} \mid c \in P_0 \wedge 1 \leq m \leq \textstyle\sum_{t \in {}^\bullet c} U_j(t)\} \\
\hat{P}_i^j &= \hat{P}_i^{j-1} \cup \{p^{m+\triangle p} \mid p \in P_i \wedge 1 \leq m \leq \textstyle\sum_{t \in {}^\bullet p} U_j(t)\} \\
\hat{T}_i^j &= \hat{T}_i^{j-1} \cup \{t^{m+\triangle t} \mid t \in T_i \wedge 1 \leq m \leq U_j(t)\} \ .
\end{aligned}$$

Then, for every new transition $v = t^m$, we *choose*[3] two sets of places:

$$In_v \subseteq \{p \in \bigcup_{i \geq 0} \hat{P}_i^{j-1} \mid p^\bullet = \varnothing\} \cup (\hat{P}_0^j \setminus \hat{P}_0^{j-1}) \text{ and } Out_v \subseteq \bigcup_{i \geq 0} (\hat{P}_i^j \setminus \hat{P}_i^{j-1})$$

in such a way that:

- $|{}^\bullet t| = |In_v|$ and $|t^\bullet| = |Out_v|$.
- In_v comprises a place labelled q for each $q \in {}^\bullet t$.
- Out_v comprises a place labelled r for each $r \in t^\bullet$.
- the sets $In_v \cup Out_v$ and $In_w \cup Out_w$ are disjoint for distinct transitions v and w.

Finally, we add to $\hat{F}_i^{j-1}$ all arc sets $In_v \times \{v\}$ and $\{v\} \times Out_v$ obtaining $\hat{F}_i^j$.

The resulting net $CSON_n$ is said to belong to $\mathsf{proc}_{CSPT}(\chi)$. The construction is illustrated in Figure 6 for the CSPT-net of Figure 3(b). The net is isomorphic to $CSON_0$ of Figure 5 which, as we already noted, is a process of $CSPT_0$. This is not a mere coincidence, as we have the following general result:

Theorem 4. $\mathsf{proc}(CSPT) = \mathsf{proc}_{CSPT}(\mathsf{steps}(CSPT))$.

In other words, axiomatically and operationally defined processes of a CSPT-net are the same (up to net isomorphism).

[3] This means that, in general, more than one process can be constructed for a given χ.

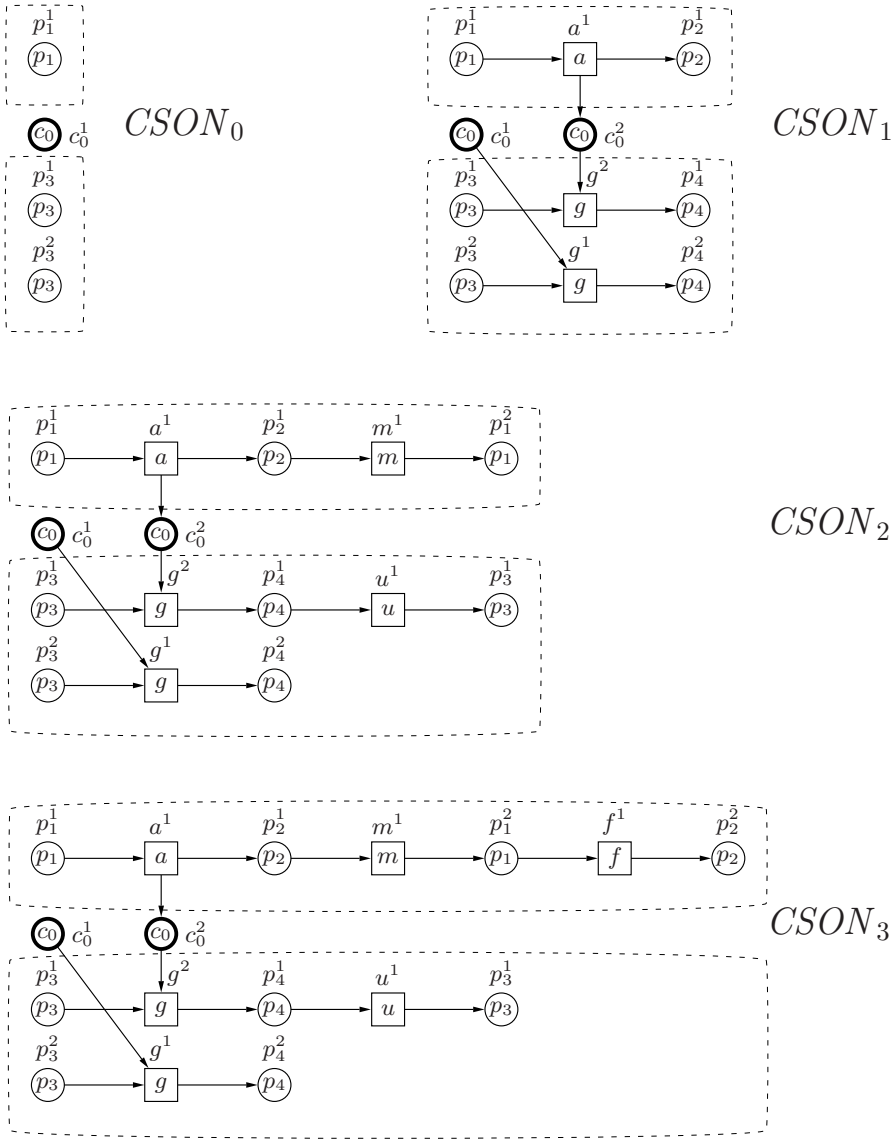

Fig. 6. Process generated for $CSPT_0$ and its step sequence $\chi = \{a, g, g\}\{m, u\}\{f\}$.

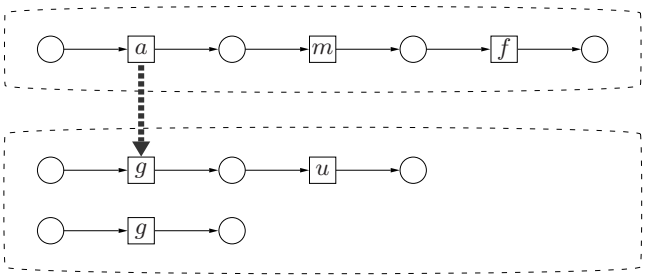

Fig. 7. Another way of representing the CSO-net $CSON_0$.

6 Discussion

The results formulated in the preceding sections can be proven by taking advantage of a general scheme introduced in [13] aimed at handling the operational and causality semantics of various kinds of Petri nets. This general scheme identifies a number of specific Properties which, when satisfied, validate, among others, Theorems 1-4 (called semantical Aims in [13]).

Some Properties concern the basic characteristics of various semantical mappings, for example, that $\mathsf{steps}(CSON) \neq \varnothing$. This is, perhaps surprisingly, a non-trivial result which relies heavily on Fact 1 which holds for all SO-structures. Another Property requires that, for every process $CSON$ generated by a step sequence χ, it is the case that χ corresponds to one of the step sequences of $CSON$. That such a property holds can be verified by re-tracing the procedure through which $CSON$ has been constructed. Still another Property states that each process can be re-generated from any of its step sequences. The proof details of these Properties follow to a large extent those developed in the past for other classes of Petri nets, yet there are several important technical lemmata which need to address specific complexities associated with a/sync channel places and net structuring (note that structured nets were not previously treated within the semantical framework of [13]).

As already mentioned, in [15, 20, 21], CSO-nets were formalised in a somewhat different way. For example, the CSO-net $CSON_0$ shown in Figure 5 would be represented as in Figure 7. Basically, the representation of [15, 20, 21] uses direct weak causality to link transitions which in the context of this paper are joined by a/sync channel places. This change of notation is sound since the sets of step sequences of the two representations are exactly the same. We have adopted here the more concrete representation since all weak causal links between executed transitions are derived through channel places. It should be stressed, however, that Theorems 1 and 2 also hold for the CSO-nets defined in [15, 20, 21], irrespective of the way in which they have been generated.

7 Related work

Motivated by the idea to identify a Petri net model fitting the CSO-nets from [15, 20, 21], we set out with the aim to identify a suitable causality semantics for CSO-nets

and in relation to this a fitting system (i.e., Petri net) model with CSO-nets unfoldings. In particular, we studied how to implement the communication semantics described in CSO-nets in PT-nets. This has led to the introduction of the — as far as we are aware — new idea of channel places with an a/sync semantics.

As already observed in [5] the concept of (synchronous) communication channels is not an existing (primitive) concept for Petri net models. To model synchronous communication, one needs additional places and transitions which may lead to complicated structures. Hence, [5] proposes to extend the Coloured Petri Net model to support communication through channels inspired by the synchronisation operators of CCS [19] and CSP [10], and communication constructs in high level programming languages. Channel communication is seen as a strong description primitive in its own right (see also [16]), and a valuable concept for structuring net models.

Again with the motivation that the basic net model does not offer synchronisation mechanisms for transitions (useful, e.g., for modular translations of concurrent languages and to define synchronised composition of programs), [3, 4] introduce zero-safe nets which are Petri nets with additional so-called zero places. This allows one to consider transactions, i.e., sequential executions of individual transitions (or firing sequences) leading from one stable marking (a marking in which all zero places are empty) to the next without affecting ordinary places on the way. A zero-safe net can be viewed as an ordinary PT-net with every transaction (up to ordering of concurrent transitions) as a single transition, but (again) the zero-safe version can be much smaller. An extension of zero-safe nets to model protocols in which one partner can be ahead of an other one was investigated in [14].

Both approaches are concerned with synchronous execution of transitions in an otherwise sequential setting (Petri nets with a firing sequence semantics). In REO [1], a channel-based model for exogenous coordination of (software) components, it is possible to define different types of channels in a calculus for constructing complex connectors from simpler ones. This includes synchronous and asynchronous channels. In [9] it is discussed how systems that communicate through and are coordinated by REO channels can be modeled as Petri nets, using a composition function for Petri nets to combine the nets representing the components. Again, this leads to relatively complex net structures.

8 Conclusions

We have studied the causality in communication structured occurrence nets (CSO-nets) — a basic class of structured occurrence nets introduced in [15, 20, 21]. First we have extended the standard PT-nets model with an explicit structuring into communicating sub-systems, and the new concept of a/sync channel places — combining synchronous and asynchronous communication — to facilitate interaction between subsystems. Whereas Petri nets (and, in particular, PT-nets) have an intrinsic semantical concept for simultaneity (steps), synchronising specific transitions requires an additional abstraction (a macro) to which we have added the interpretation of executing one transition ‘not before’ another transition.

After that we have developed a process semantics for the resulting CSPT-nets based on CSO-nets, following the generic semantical framework from [13] aimed at the systematic development of a causality semantics of Petri nets. This has led to the conclusion that the causality in CSO-nets is underpinned by stratified order structures extending causal partial orders with weak causality.

For the future we plan to investigate how (communication) structured nets might be used for an enhanced version of the model checking techniques based on net unfoldings [7, 17, 18]. We hope that the structuring will make it possible to analyse more complex systems than would be feasible with the existing techniques. An interesting and practically important extension of CSPT-nets would be to allow more transitions to output to and input from a given a/sync channel place. Also, allowing more than two transition to synchronise would support broadcast-like communication. Finally, we intend to investigate causality in other types of structured occurrence nets defined in [15, 20, 21].

Acknowledgement

This research was supported by the Pascal Chair award from Leiden University, the EPSRC VERDAD project, and NSFC Grants 60910004 and 2010CB328102.

References

1. F.Arbab: A Channel-based Coordination Model for Component Composition. Mathematical Structures in Computer Science **14** (2004) 1-38
2. E.Best and R.Devillers: Sequential and Concurrent Behaviour in Petri Net Theory. Theoretical Computer Science **55** (1988) 87–136
3. R.Bruni and U.Montanari: Zero-safe nets: Comparing the Collective and Individual Token Approaches. Information and Computation **156** (2000) 46-89
4. R.Bruni and U.Montanari: Zero-Safe Nets, or Transition Synchronization Made Simple. Electronic Notes in Theoretical Computer Science **7** (1997) 55–74
5. S.Christensen and D.Hansen: Coloured Petri Nets Extended with Channels for Synchronous Communication. Lecture Notes in Computer Science **815** (1994) 159–178
6. J.Desel and W.Reisig: Place/Transition Petri Nets. Lecture Notes in Computer Science **1491** (1998) 122–173
7. J.Esparza and K.Heljanko: *Unfoldings: A Partial-Order Approach to Model Checking*. Springer (2008)
8. H.Gaifman and V.R.Pratt: Partial Order Models of Concurrency and the Computation of Functions. In: LICS, IEEE Computer Society (1987) 72–85
9. J.Guillen-Scholten, F.Arbab, F.de Boer and M.Bonsangue: Modeling the Exogenous Coordination of Mobile Channel-based Systems with Petri Nets. Electronic Notes in Theoretical Computer Science **154** (2006) 121–138
10. C.A.R.Hoare: *Communicating Sequential Processes*. Prentice Hall (1985)
11. R.Janicki and M.Koutny: Invariants and Paradigms of Concurrency Theory. Lecture Notes in Computer Science **506** (1991) 59–74
12. R.Janicki and M.Koutny: Semantics of Inhibitor Nets. Information and Computation **123** (1995) 1–16

13. H.C.M.Kleijn and M.Koutny: Process Semantics of General Inhibitor Nets. Information and Computation **190** (2004) 18-69
14. M.Köhler-Bußmeier and M.Kudlek: Linear Properties of Zero-Safe Nets with Debit Tokens. Fundamenta Informaticae **85** (2008) 329-342
15. M.Koutny and B.Randell: Structured Occurrence Nets: A Formalism for Aiding System Failure Prevention and Analysis Techniques. Fundamenta Informaticae **97** (2009) 41–91
16. O.Kummer: A Petri Net View on Synchronous Channels. Petri Net Newsletter **56** (1999) 7–11
17. K.L.McMillan: A Technique of State Space Search Based on Unfoldings. Formal Methods in System Design **6** (1995) 45–65
18. R.Meyer, V.Khomenko and T.Strazny: A Practical Approach to Verification of Mobile Systems Using Net Unfoldings. Fundamenta Informaticae **94** (2009) 439–471
19. R.Milner: *Communication and Concurrency*. Prentice Hall (1989)
20. B.Randell: Occurrence Nets Then and Now: The Path to Structured Occurrence Nets. Lecture Notes in Computer Science **6709** (2011)
21. B.Randell and M.Koutny: Failures: Their Definition, Modelling and Analysis. Lecture Notes in Computer Science **4711** (2007) 260–274
22. G.Rozenberg and J.Engelfriet: Elementary Net Systems. Lecture Notes in Computer Science **1491** (1998) 12–121