Introduction to (Mathematical Models of) Concurrent Systems

Józef Winkowski

Institute of Computer Science, Polish Academy of Sciences
Warsaw, Poland, 2012.12.4

Contents

1. Motivation

Mathematical models of concurrent systems and their behaviours are necessary in order to specify and define such systems and their behaviours in a precise way, and to verify (prove) if the systems and their behaviours thus specified or defined have the required properties.

A calculus is necessary allowing one to construct a model of a complicated system and its behaviour by combining simpler models of system components and their behaviours.

P_1, P_2 - processors, R -a resource.

l_i - P_i waiting for access to R,

m_i - P_i ready to release R,

n_i - P_i ready to perform its private activity,

R_{free} - R is free, $R_{occupied}$ - R is occupied,

a_i - P_i requires R, $\overline{a_i}$ - R accepts a_i,

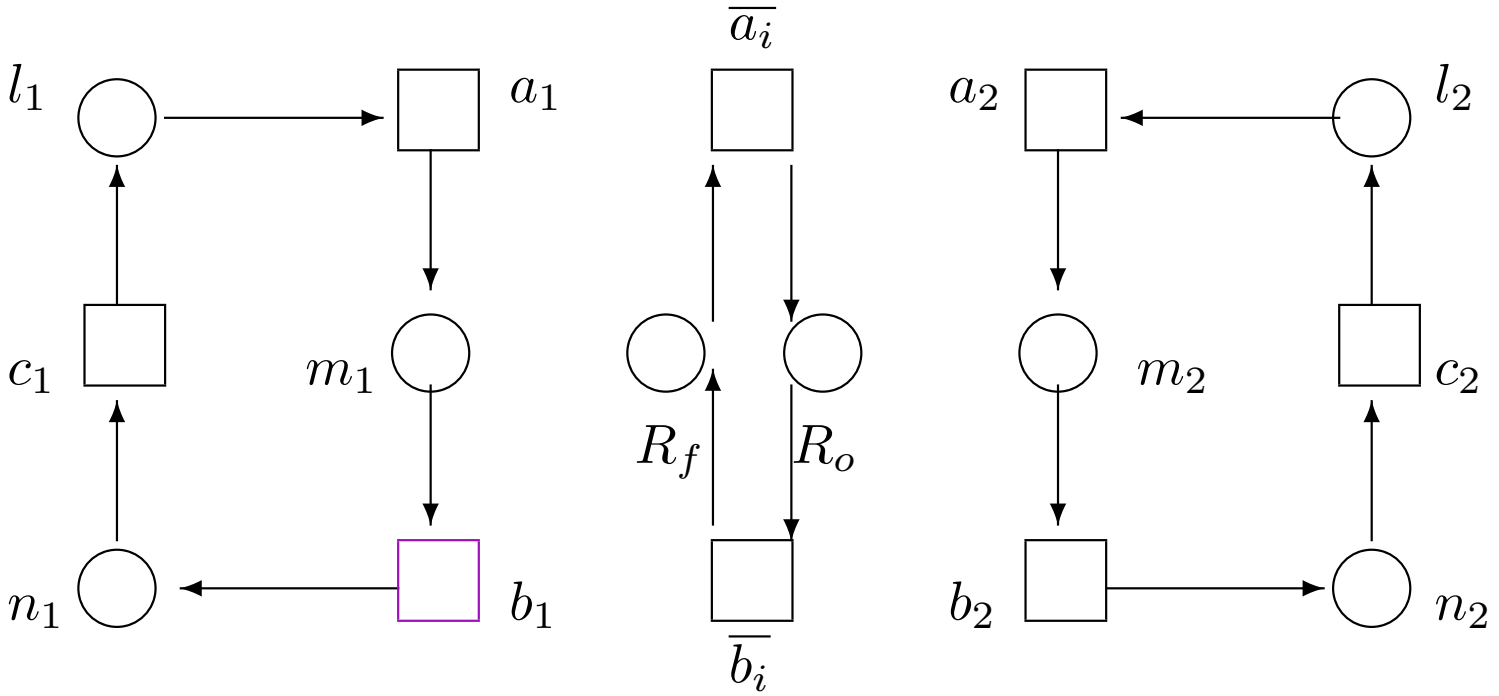
b_i - P_i releases R, $\overline{b_i}$ -R accepts b_i,

c_i - P_i performs its private activity.

$l_i = a_i m_i, \qquad m_i = b_i n_i, \qquad n_i = c_i l_i,$

$R_{free} = \overline{a_1} R_{occupied} + \overline{a_2} R_{occupied}, \qquad R_{occupied} = \overline{b_1} R_{free} + \overline{b_2} R_{free},$

$S = (l_1 \parallel R_{free} \parallel n_2) - \{a_1, \overline{a_1}, b_1, \overline{b_1}, a_2, \overline{a_2}, b_2, \overline{b_2}\}.$

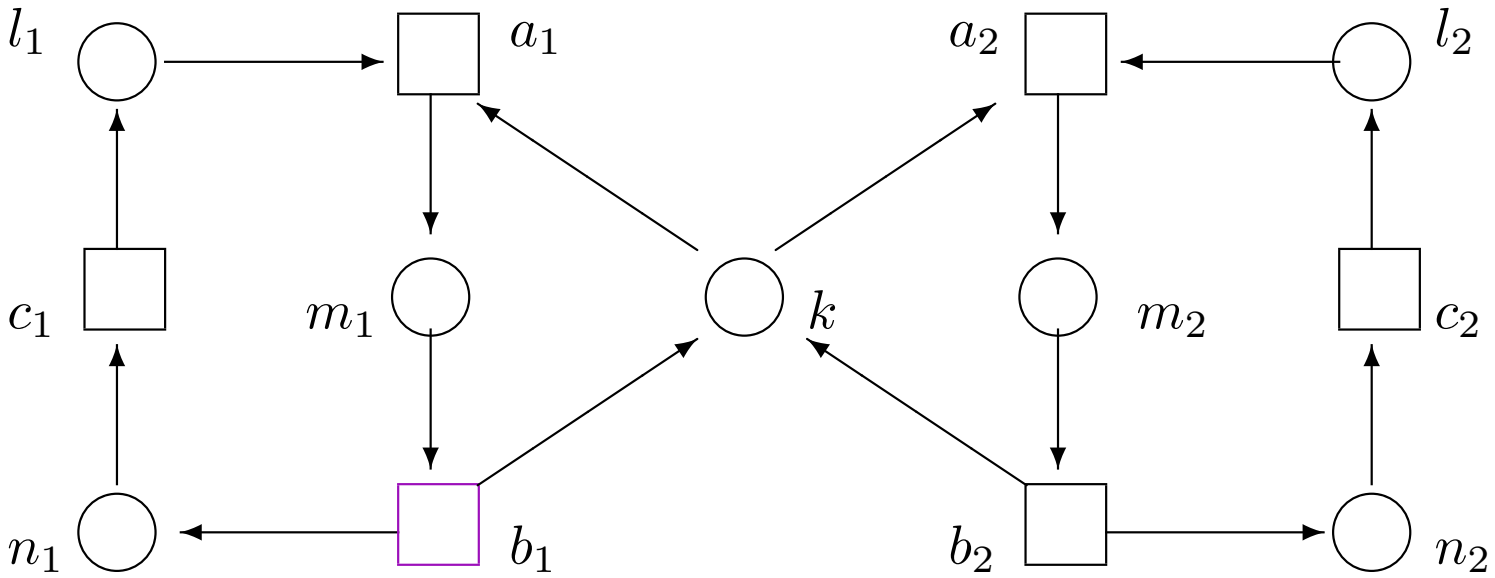

l_1
a_1
c_1
m_1
n_1
b_1
$\overline{a_i}$
R_f
R_o
$\overline{b_i}$
a_2
l_2
m_2
c_2
b_2
n_2

2. Condition/Event Petri nets

By a *Petri net*, or a *net*, we mean a triple $\mathbf{N} = (B, E, F)$ which consists of two disjoint sets B and E, and of a relation $F \subseteq (B \times E) \cup (E \times B)$. Elements of B are called *conditions*. Elements of E are called *events*. F is called the *causal dependency relation*.

By a *morphism* from $\mathbf{N}$ to a net $\mathbf{N}' = (B', E', F')$ we mean a mapping $h : B \cup E \to B' \cup E'$ such that $h(B) \subseteq B'$, $h(E) \subseteq E'$, and for each $e \in E$ the restriction of h to Fe is a bijection between Fe and $F'h(e)$ and the restriction of h to eF is a bijection between eF and $h(e)F'$.

(Reisig, W., *Petri Nets: An Introduction*, Springer-Verlag, 1985)



$$B = \{k, l_1, m_1, n_1, l_2, m_2, n_2\}$$

$$E = \{a_1, b_1, c_1, a_2, b_2, c_2\}$$

$$F = \{(k, a_1), (l_1, a_1), (a_1, m_1), (m_1, b_1), (b_1, n_1), (b_1, k), (n_1, c_1), (c_1, l_1),$$

$$(k, a_1), (l_1, a_1), (a_1, m_1), (m_1, b_1), (b_1, n_1), (b_1, k), (n_1, c_1), (c_1, l_1)\}$$

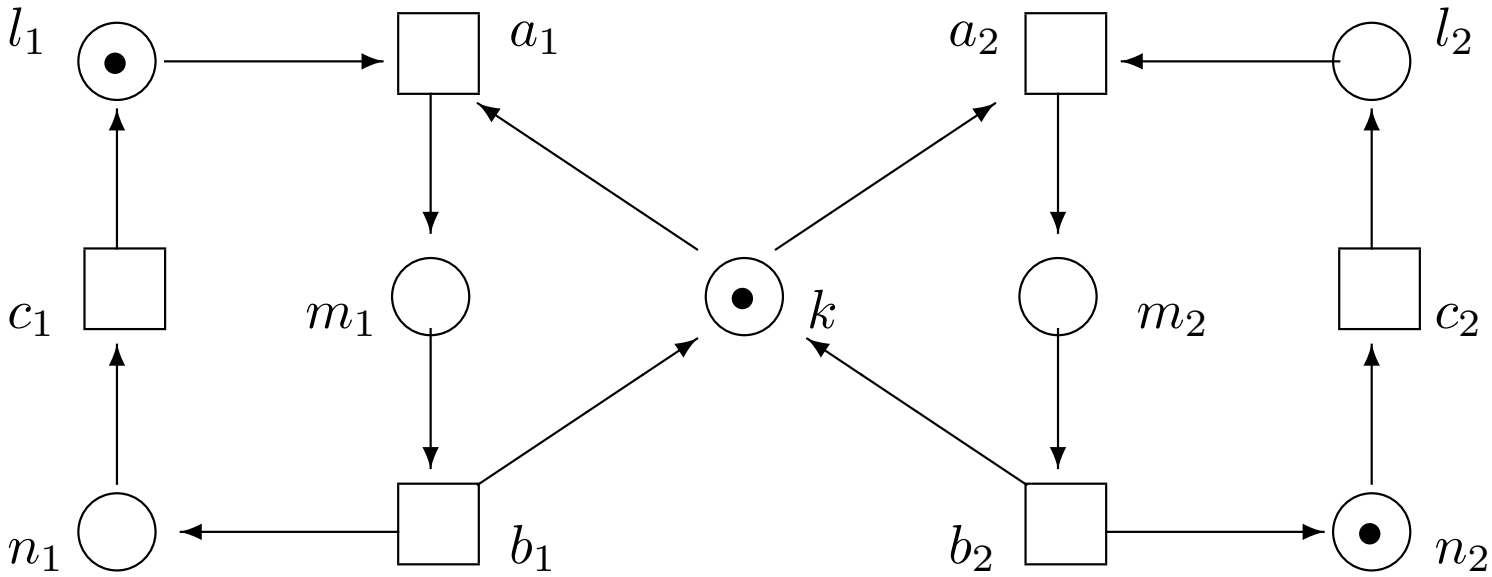
A net $\mathbf{N} = (B, E, F)$ is a structure to define *runs* of a system in a way allowing one to reflect concurrency.

Each condition is represented as a *place* which is either empty (when the condition is not satisfied) or contains a *token* (when the condition is satisfied).

The set of conditions which are satified in a system state is represented by the respective distribution of tokens in places, called a *marking*.

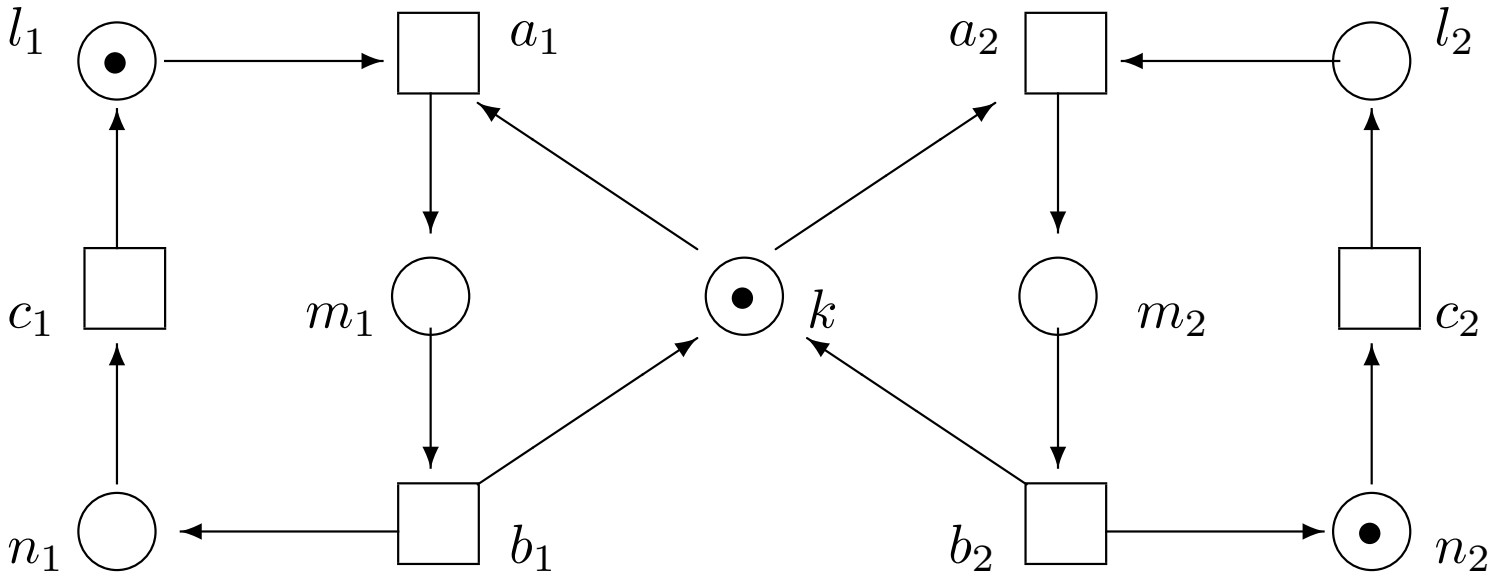
Each event is represented as a *transition* which may change the marking representing the current system state according to a rule, called the *firing rule*.

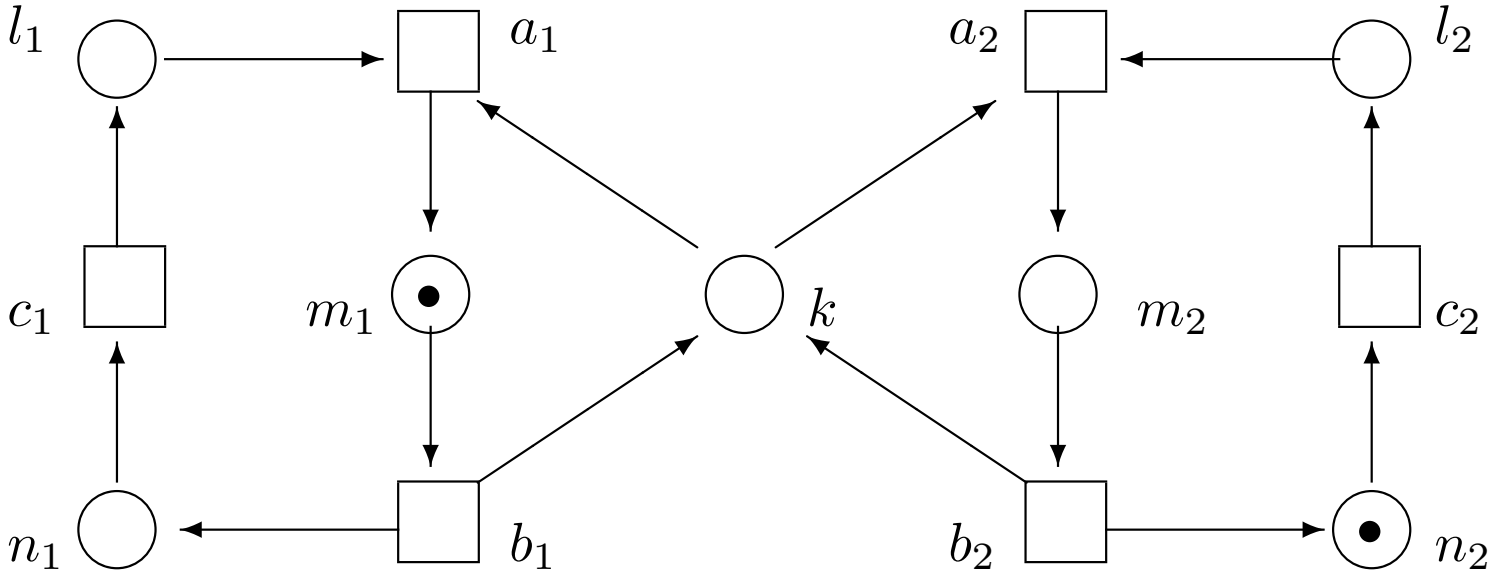
A transition t may *fire* in a particular marking M iff $Ft \subseteq M$ and $tF \cap M = \emptyset$. A firing of t leads to a new marking $M' = (M - Ft) \cup tF$.



$k = R$ free, $a_i = P_i$ takes R, $b_i = P_i$ releases R, $c_i =$ a private activity of P_i

$M = \{k, l_1, n_2\}$


l_1
a_1
a_2
l_2
c_1
m_1
k
m_2
c_2
n_1
b_1
b_2
n_2


l_1
a_1
a_2
l_2
c_1
m_1
k
m_2
c_2
n_1
b_1
b_2
n_2

3. Net processes

A particular finite sequence of consecutive firings of transitions of a net $\mathbf{N} = (B, E, F)$ is

$\sigma = M_0 t_0 M_1 t_1 ... t_n M_{n+1}$,

where $Ft_i \subseteq M_i$, $t_i F \cap M_i = \emptyset$, and $M_{i+1} = (M_i - Ft_i) \cup t_i F$ for $0 \leq i \leq n$.

Such a sequence, called a *firing sequence*, represent a run of the net $\mathbf{N}$ from the marking M_o to the marking M_{n+1}, and M_{n+1} is said to be *reachable* from M_o.

Firing sequences $\sigma = M_0 t_0 M_1 t_1 ... t_n M_{n+1}$ and $\sigma' = M'_0 t'_0 M'_1 t'_1 ... t'_{n'} M'_{n'+1}$

such that $M_{n+1} = M'_0$ can be *concatenated* with the following result which is also a firing sequence

$\sigma\sigma' = M_0 t_0 M_1 t_1 ... t_n M_{n+1} t'_0 M'_1 t'_1 ... t'_{n'} M'_{n'+1}$.

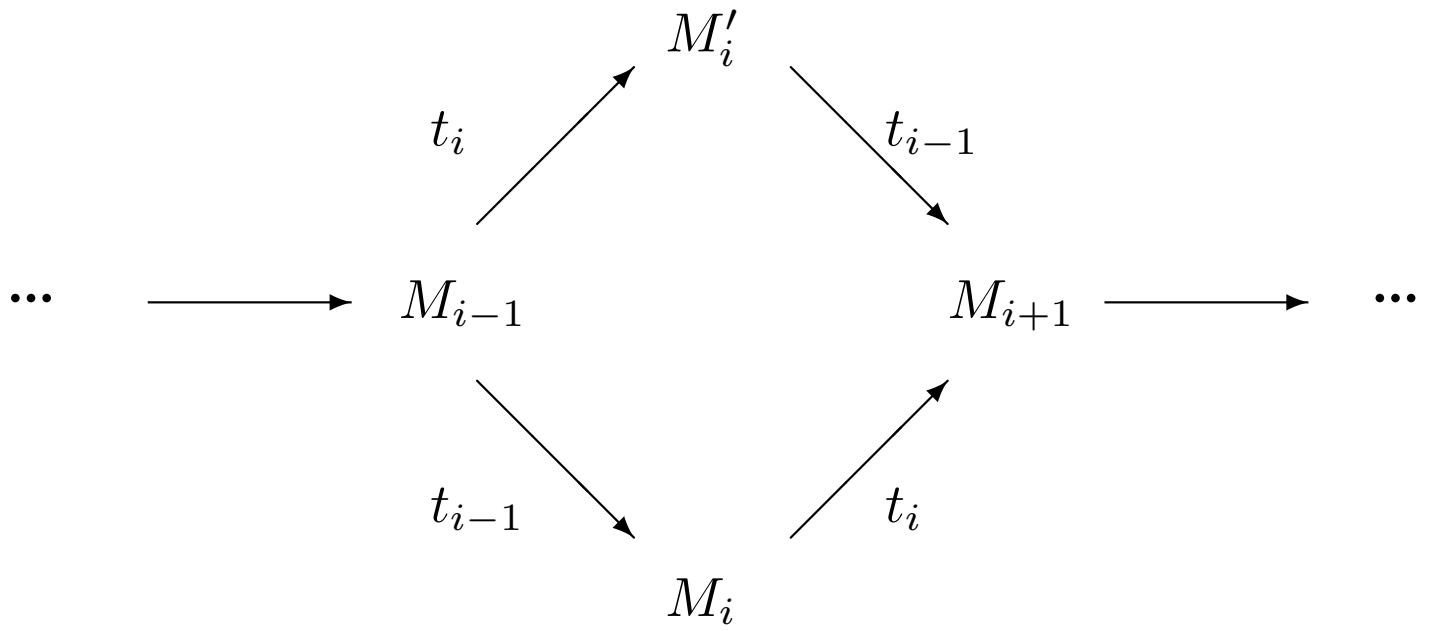
There may be many firing sequences representing the same run. Consequently, a run should be defined as an equivalence class of firing sequences with respect to the least equivalence such that $\sigma \equiv \sigma'$ whenever

$\sigma = M_0 t_0 M_1 t_1 ... t_n M_{n+1}$,

$\sigma' = M_0 t_0 M_1 t_1 ... M_{i-1} t_i M_i' t_{i-1} ... t_n M_{n+1}$

with $Ft_i \subseteq M_{i-1}$ (and hence $Ft_i \cap Ft_{i-1} = \emptyset$).

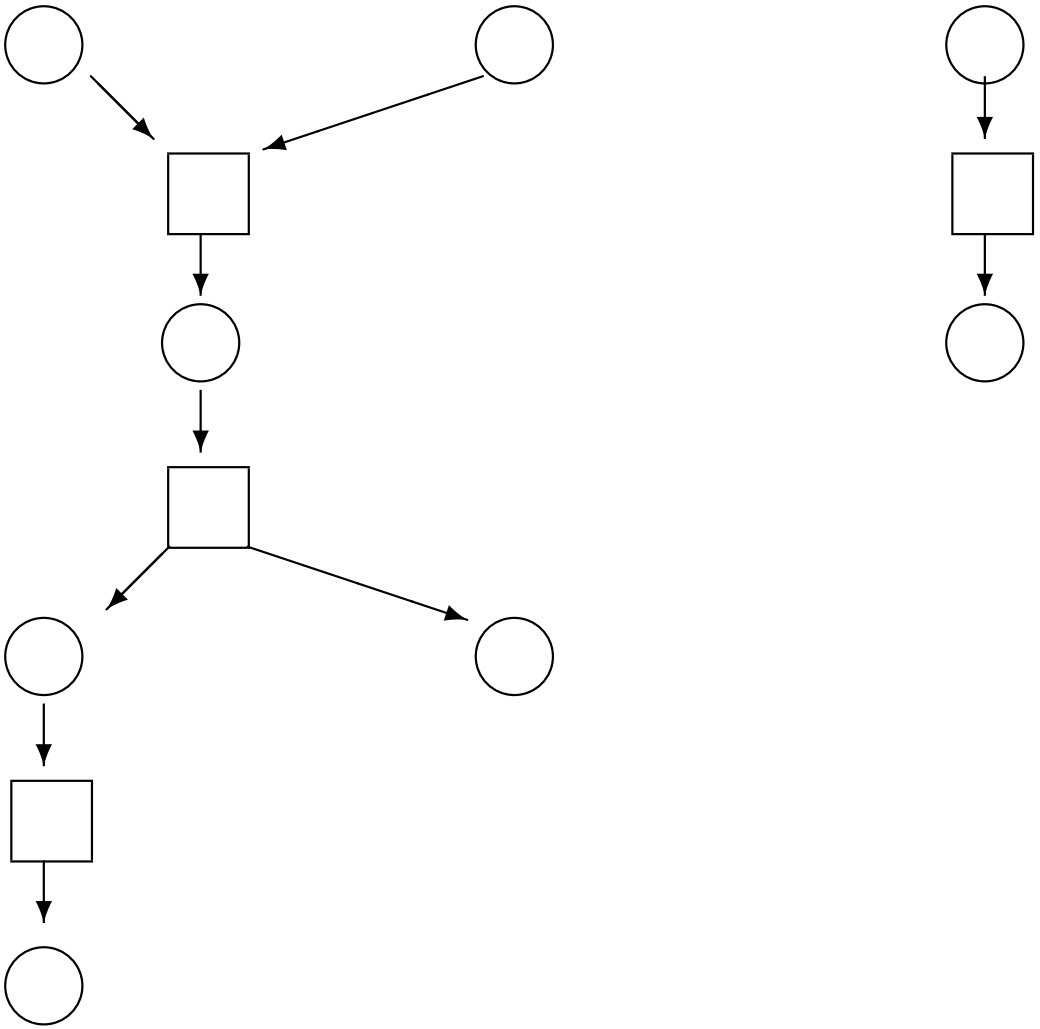
This equivalence is a *congruence* with respect to the operation of concatenating firing sequences.


M_i'
t_i
t_{i-1}
$\cdots$
M_{i-1}
M_{i+1}
$\cdots$
t_{i-1}
t_i
M_i

Another way of representing runs of a net $\mathbf{N} = (B, E, F)$ is to define them as so called *processes*, where a process is an isomorphism class of *concrete processes*, and a concrete process is a pair $P = (\mathbf{N}', h)$ consisting of a *causal net* $\mathbf{N}'$ and of a morphism h from $\mathbf{N}'$ to $\mathbf{N}$.

A *causal net* is a net $\mathbf{N} = (B, E, F)$ such that the reflexive and transitive closure F^* of F is a partial order $\leq$, each $b \in B$ has at most one $e' \in E$ such that $e'Fb$ and at most one $e'' \in E$ such that bFe'', and different conditions b' and b'' are distinguishable in the sense that $Fb' \neq Fb''$ or $b'F \neq b''F$.

Such a net is said to be *finitary* if the set $\leq e$ of predecessors of each event $e \in E$ contains at most a finite number of events.



To the equivalence class $[\sigma]$ of a firing seqence $\sigma = M_0t_0M_1t_1...t_nM_{n+1}$ of a net $\mathbf{N}$ there corresponds the finitary causal net $\mathbf{C}([\sigma])$ which can be defined as follows.

Let $\sim$ be the least equivalence such that every initial segments of σ of the form $\alpha = M_0t_0M_1t_1...t_kM_{k+1}$ and $\alpha' = M_0t_0M_1t_1...M_{i-1}t_iM_i't_{i-1}M_{i+1}...t_kM_{k+1}$ are equivalent whenever $Ft_i \cap Ft_{i-1} = \emptyset$ and every initial segments of σ of the form $\alpha = M_0t_0M_1t_1...t_kM_{k+1}$ and $\alpha = M_0t_0M_1t_1...t_{k-2}M_{k-1}t_kM'_{k+1}$ are equivalent whenever $Ft_k \cap Ft_{k-1} = \emptyset$.

Let E' be the set of equivalence classes of $\sim$ and for every such a class $e' = [M_0t_0M_1t_1...t_kM_{k+1}]_\sim$ let $\lambda(e')$ be the last transition t_k.

Let $B' = \{(e', p) : e' \in E' \wedge p \in \lambda(e')F\} \cup \{(0, p) : p \in M_0\}$.

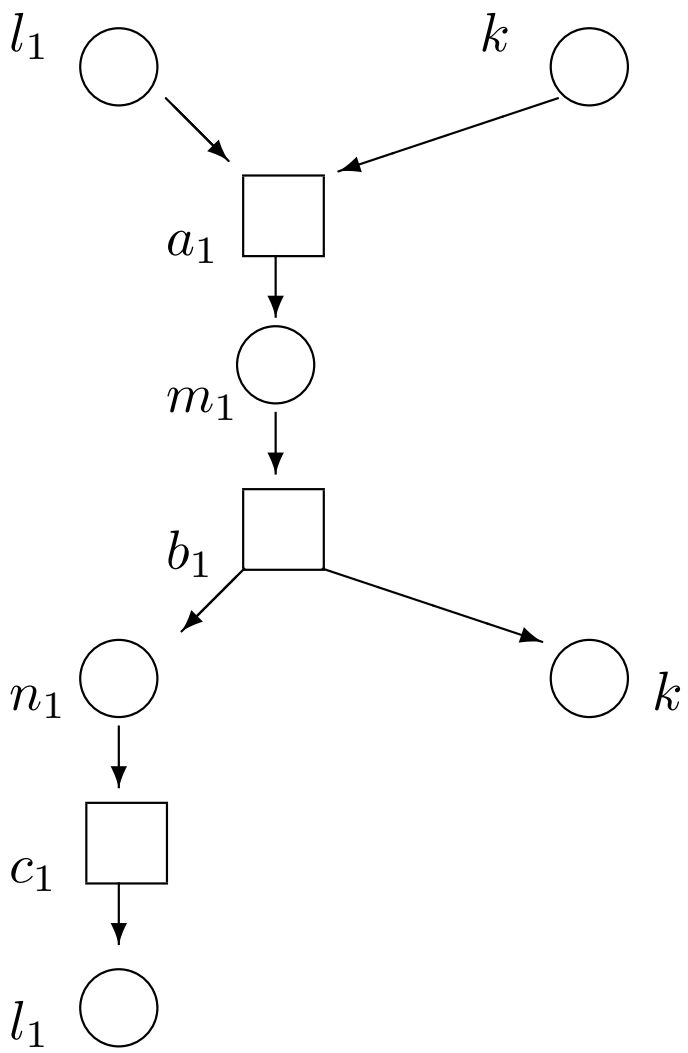
Let $(e', b) \in F'$ iff $b = (e', p)$ for some $p \in B$.

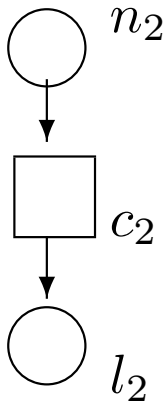
Let $(b, e') \in F'$ iff either $b = ([M_0t_0...t_{k-1}M_k]_\sim, p)$, $e' = [M_0t_0...t_{k-1}M_kt_kM_{k+1}]_\sim$ and $p \in Ft_k$ or else $b = (0, p)$, $e' = [M_0t_0M_1]_\sim$ and $p \in Ft_0$.

Define $\mathbf{C}([\sigma]) = (B', E', F')$

The correspondence $h_{[\sigma])}$ between the causal net $\mathbf{C}([\sigma]) = (B', E', F')$ just defined and the original net $\mathbf{N}$, where $h_{[\sigma]}((0, p)) = h_{[\sigma]}((e, p)) = p$ and $h_{[\sigma]}([M_0 t_0 ... t_k M_{k+1}]_\sim) = t_k$, is a net morphism. Consequently, $P([\sigma]) = (\mathbf{C}([\sigma]), h_{[\sigma]})$ is a concrete process of $\mathbf{N}$ that represents the equivalence class $[\sigma]$ of firing sequences as a process.

To the concatenation of firing sequences there corresponds a concatenation of the corresponding processes.





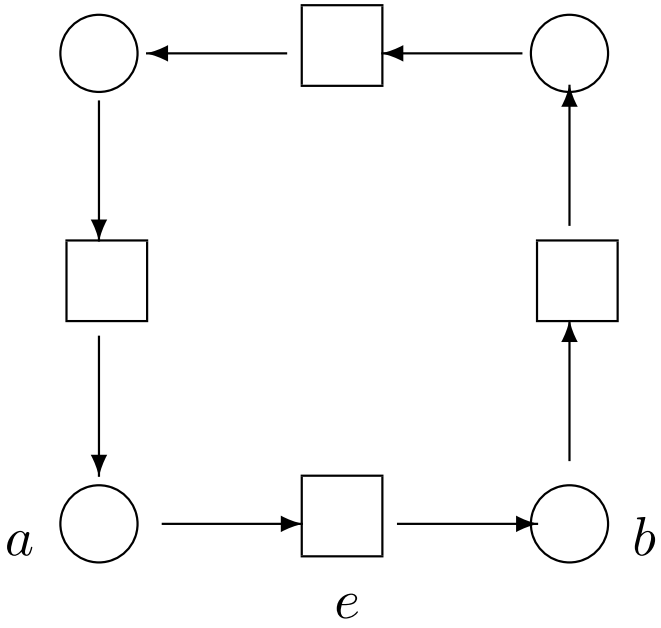
$P([\sigma])$ for $\sigma = \{l_1, k, n_2\}c_2\{l_1, k, l_2\}a_1\{m_1, l_2\}b_1\{n_1, k, l_2\}c_1\{l_1, k, l_2\}$

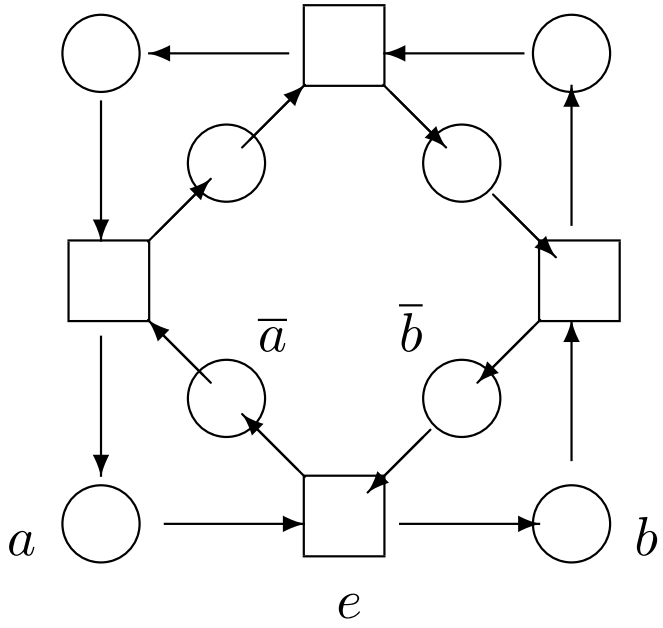
4. Condition/Event systems

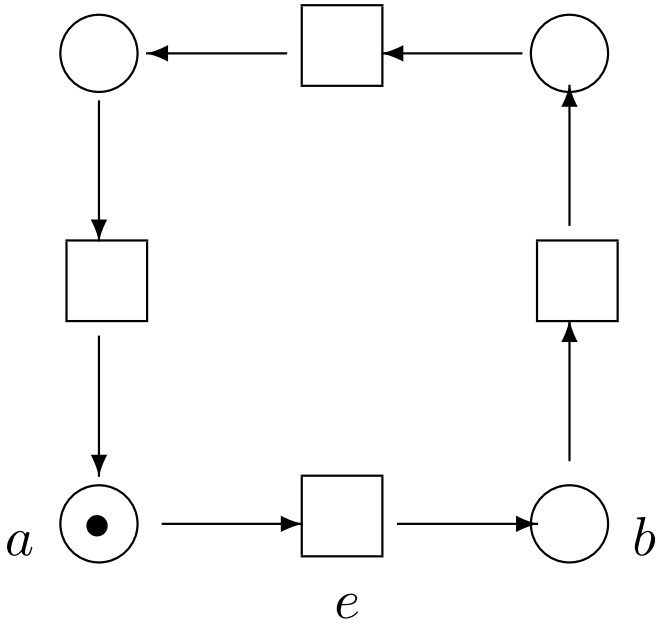
In order to represent a system starting from a given initial state, a net $\mathbf{N} = (B, E, F)$ is considered together with a distinguished *initial marking*, and the structure $\mathbf{S} = (B, E, F, M_0)$ is called a *Condition/Event system* or a *C/E system*. Without a loss of generality we may assume that such a system is *contact-free* in the sense that for any marking M reachable from M_0 and for any transition t the inclusion $Ft \subseteq M$ implies $tF \cap M = \emptyset$. This follows from the fact that to every net there corresponds a contact-free net with essentially the same behaviour.

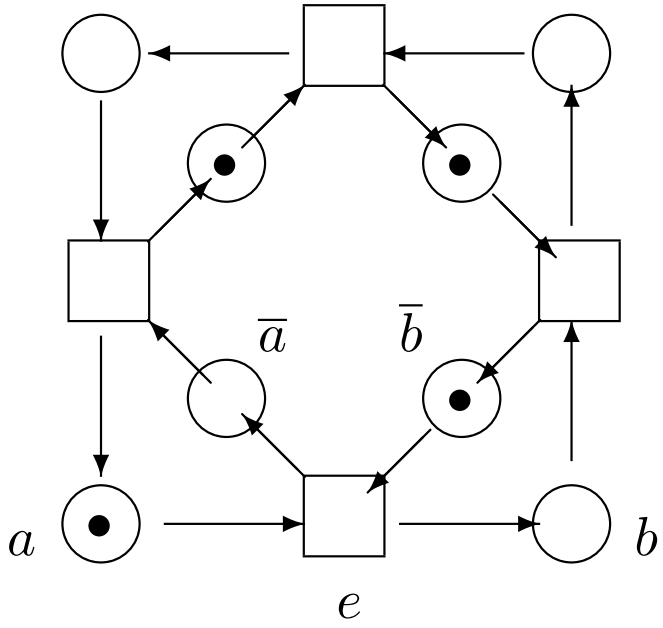
Given a net $\mathbf{N} = (B, E, F)$, there exists a *contact-free completion* of $\mathbf{N}$ defined as the net $\overline{\mathbf{N}} = (\overline{B}, \overline{E}, \overline{F})$ which consists of the set $\overline{B}$ of conditions $b \in B$ and their negations $\overline{b}$, the set $\overline{E} = E$ of events, and the causal dependency relation $\overline{F}$, where $x\overline{F}y$ iff xFy or $x = \overline{b}$ for some $b \in yF - Fy$ or $y = \overline{b}$ for some $b \in Fx - xF$.

Given $x \in \overline{B}$, by $|x|$ we denote that $b \in B$ for which $x = b$ or $x = \overline{b}$.


a
b
e


$\overline{a}$
$\overline{b}$
a
b
e


a
b
e


$\overline{a}$
$\overline{b}$
a
b
e

5. Behaviours of C/E systems

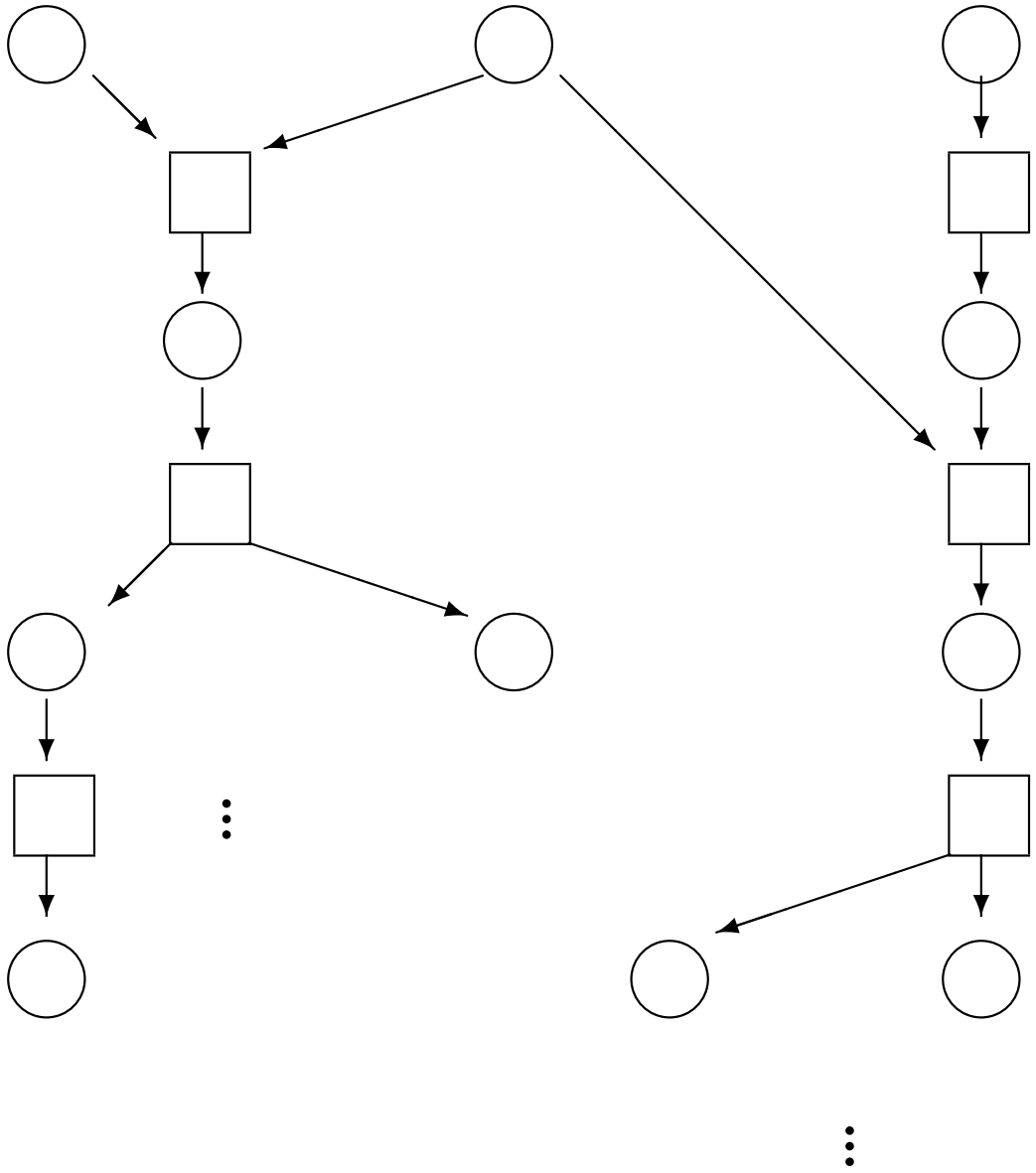
An important feature of any contact-free C/E system $\mathbf{S}$ with a net $\mathbf{N}$ is that its behaviour can be represented by a single object that contains complete information on all the possible system runs. Such an object, called the *branching unfolding* of $\mathbf{S}$, can be defined as an isomorphism class of *concrete unfoldings* of $\mathbf{S}$, where a concrete unfolding is a pair $U = (\mathbf{N}', h)$ consisting of a *finitary occurrence net* $\mathbf{N}'$ and of a morphism h from $\mathbf{N}'$ to $\mathbf{N}$.

(Nielsen, M., Plotkin, G. D., Winskel, G., *Petri nets, event structures and domains, part 1*, Theoretical Computer Science, 13(1),1981)

An *occurrence net* is a net $\mathbf{N} = (B, E, F)$ such that the reflexive and transitive closure F^* of F is a partial order $\leq$, each $b \in B$ has at most one $e' \in E$ such that $e'Fb$, different conditions b' and b'' are distinguishable in the sense that $Fb' \neq Fb''$ or $b'F \neq b''F$, and the *conflict relation* $\#$ defined as follows is irreflexive:

$a_1 \# a_2$ iff there exist $e_1, e_2 \in E$ such that $Fe_1 \cap Fe_2 \neq \emptyset$, $e_1 F^* a_1$, and $e_2 F^* a_2$.

Such a net is said to be *finitary* if the set $\leq e$ of predecessors of each event $e \in E$ contains at most a finite number of events.



The occurrence net of a branching unfolding of a C/E system $\mathbf{S} = (B, E, F, M_0)$ can be defined as follows.

Let $\sim$ be the least equivalence such that every firing sequences of the form $\alpha = M_0 t_0 M_1 t_1 ... t_k M_{k+1}$ and $\alpha' = M_0 t_0 M_1 t_1 ... M_{i-1} t_i M_i' t_{i-1} M_{i+1} ... t_k M_{k+1}$ are equivalent whenever $Ft_i \cap Ft_{i-1} = \emptyset$ and every firing sequences of the form $\alpha = M_0 t_0 M_1 t_1 ... t_k M_{k+1}$ and $\alpha = M_0 t_0 M_1 t_1 ... t_{k-2} M_{k-1} t_k M_{k+1}'$ are equivalent whenever $Ft_k \cap Ft_{k-1} = \emptyset$.

Let E' be the set of equivalence classes of $\sim$ and for every such a class $e' = [M_0 t_0 M_1 t_1 ... t_k M_{k+1}]_\sim$ let $\lambda(e')$ be the last transition t_k.

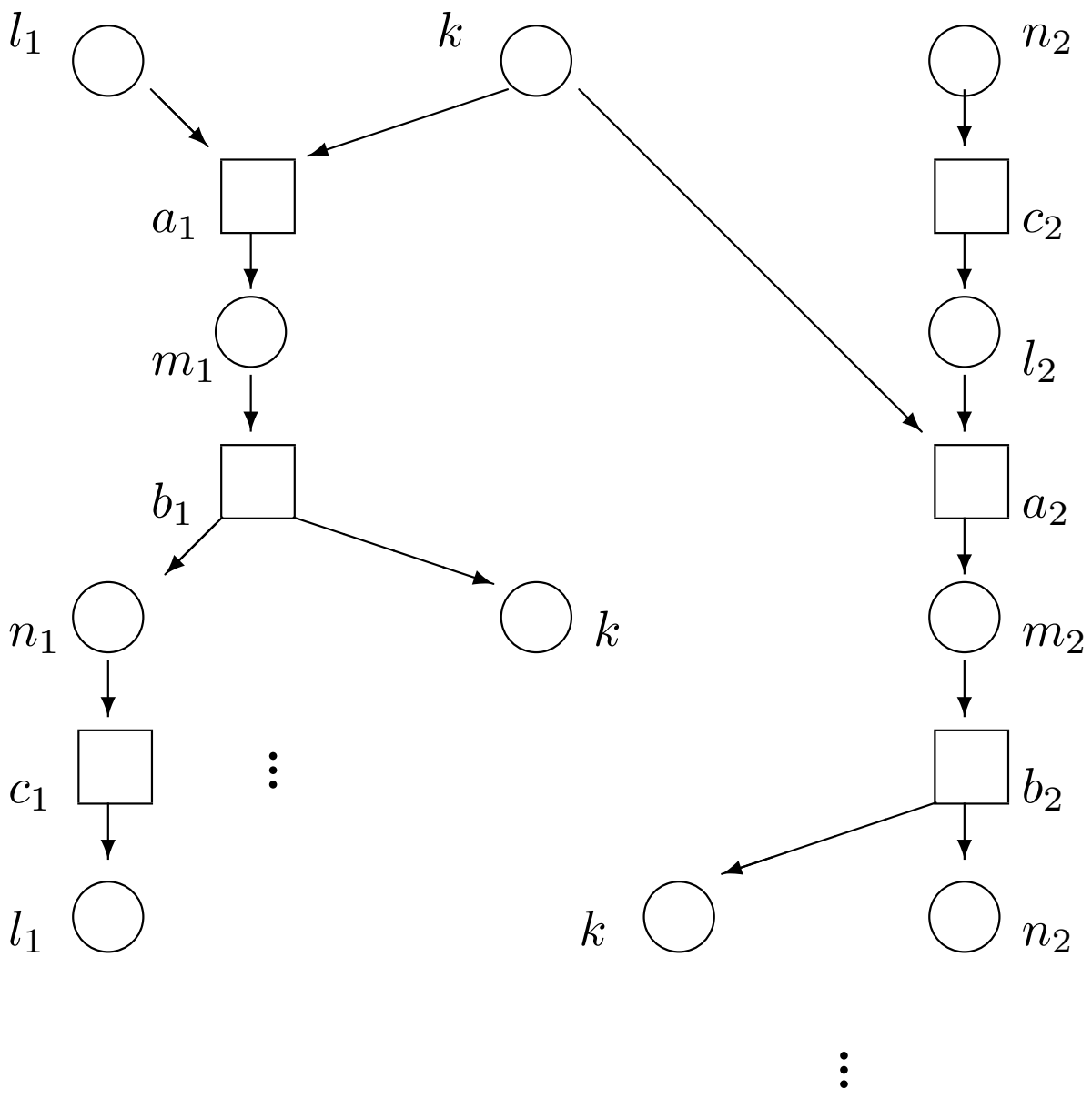
Let $B' = \{(e', p) : e' \in E' \wedge p \in \lambda(e')F\} \cup \{(0, p) : p \in M_0\}$.

Let $(e', b) \in F'$ iff $b = (e', p)$ for some $p \in B$.

Let $(b, e') \in F'$ iff either $b = ([M_0 t_0 ... t_{k-1} M_k]_\sim, p)$, $e' = [M_0 t_0 ... t_{k-1} M_k t_k M_{k+1}]_\sim$ and $p \in Ft_k$ or else $b = (0, p)$, $e' = [M_0 t_0 M_1]_\sim$ and $p \in Ft_0$.

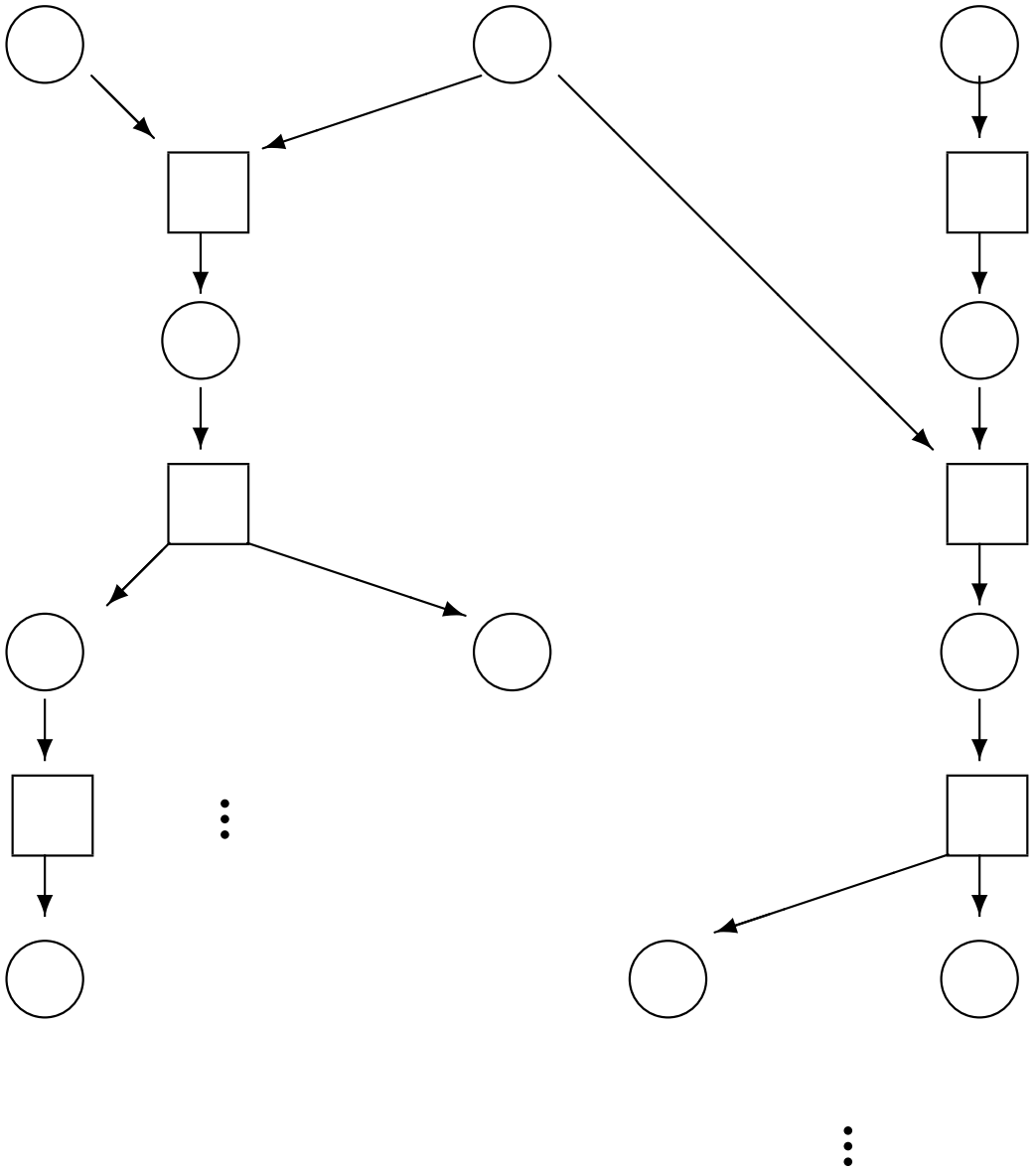
Define $\mathbf{OC}(\mathbf{S}) = (B', E', F')$.

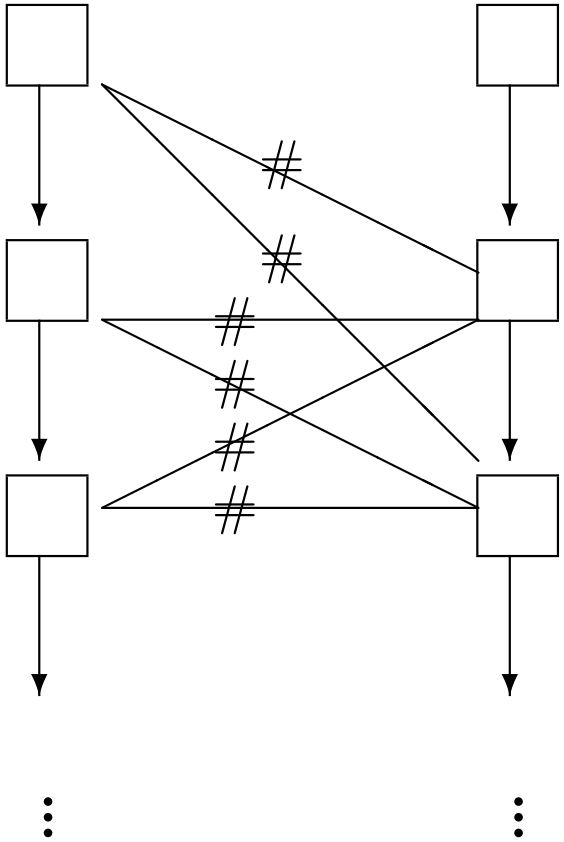
The correspondence h between the finitary occurrence net $\mathbf{OC}(\mathcal{S}) = (B', E', F')$ just defined and the system net $\mathbf{N} = (B, E, F)$, where $h((0, p)) = h((e, p)) = p$ and $h([M_0 t_0 ... t_k M_{k+1}]_\sim) = t_k$ is a net morphism. Consequently,
$U(\mathbf{S}) = (\mathbf{OC}(\mathbf{S}), h)$ is a branching unfolding of $\mathbf{S}$.


l_1
k
n_2
a_1
c_2
m_1
l_2
b_1
a_2
n_1
k
m_2
c_1
b_2
l_1
k
n_2

6. Event structures

By omitting in a finitary occurrence net $\mathbf{N} = (B, E, F)$ conditions, by restricting the relation F^* to E, and by specifying which elements of E are in a *conflict relation* $\#$ in the sense that they cannot occur in the same run, we obtain a structure $\mathbf{E}(\mathbf{N}) = (E, \leq, \#)$ called an *event structure*.





Formally, an event structure is $\mathbf{E} = (E, \leq, \#)$ such that E is a set, $\leq$ is a partial order on E such that the sets $\leq e$ are finite for all $e \in E$, and $\#$ is a symmetrical and irreflexive binary relation on E such that $e_1 \# e_3$ whenever $e_3 \# e_2$ for some e_2 such that $e_2 \leq e_1$.

Elements of E are called *events*. The partial order $\leq$ is called the *causal dependency relation*. The relation $\#$ is called the *conflict relation*. A subset $X \subseteq E$ that does not contain e_1 and e_2 such that $e_1 \# e_2$ and is left-closed in the sense that $e_1 \in X$ whenever $e_1 \leq e_2$ for some $e_2 \in X$ is called a *configuration* of $\mathbf{E}$. By $\mathbf{conf}(\mathbf{E})$ we denote the set of configurations of $\mathbf{E}$.

Given an event structure $\mathbf{E} = (E, \leq, \#)$, the set $F = \mathbf{conf}(\mathbf{E})$ of its configurations is partially ordered by inclusion and this order is *directed complete* in the sense that every $X \subseteq E$ that is *directed* (i.e., every $x, y \in X$ are contained in some $z \in X$) has the least upper bound $\bigsqcup X \in F$.

Consequently, in F there is the *Scott topology*, i.e. the topology consisting of *Scott open* sets, where a Scott open set is an upper closed set that does not contain the least upper bounds of directed subsets of its complement.

Consequently, every probability measure on the Borel σ-algebra generated by Scott open subsets of F is determined uniquely by a normalized continuous valuation ν, where ν assigns to every Scott open subset U a number $\nu(U)$ such that: (1) $\nu(\emptyset) = 0$ (strictness), (2) $U_1 \subseteq U_2$ implies $\nu(U_1) \leq \nu(U_2)$ (monotonicity), (3) $\nu(U_1 \cup U_2) + \nu(U_1 \cap U_2) = \nu(U_1) + \nu(U_2)$ (modularity), (4) $\nu(\bigcup D) = \bigsqcup_{U \in D} \nu(U)$ for any directed set D of open subsets of F (continuity).

This is essential for modelling random behaviours of concurrent systems.

Given an event structure $\mathbf{E} = (E, \leq, \#)$, the set $F = \mathbf{conf}(\mathbf{E})$ of its configurations is:

coherent: $\bigcup X \in F$ for every $X \subseteq F$ such that every $x, y \in X$ have an upper bound $z \in F$,

stable: $\bigcap X \in F$ for every nonempty $X \subseteq F$ with an upper bound $z \in F$,

coincidence-free: in every $x \in F$ every different elements e and e' can be separated by some $y \subseteq x$ from F in the sense that either $e \in y$ and $e' \notin y$ or $e \notin y$ and $e' \in y$,

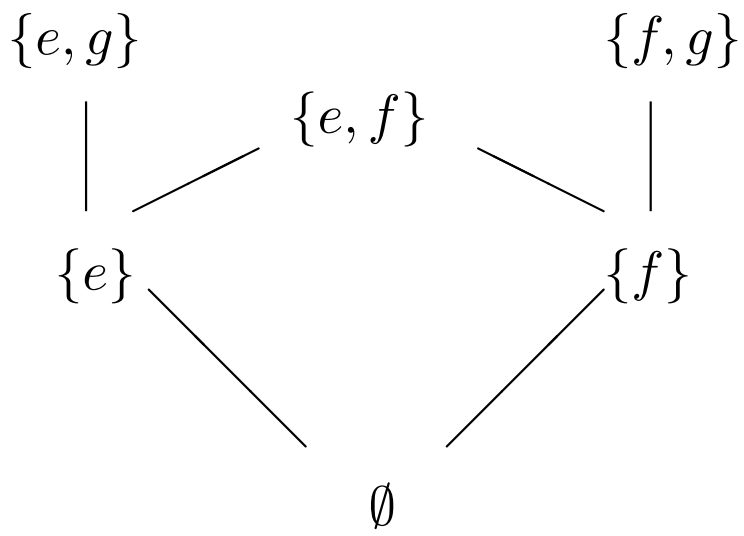
finitary: in every $x \in F$ every element e belongs to a finite $y \subseteq x$ from F,

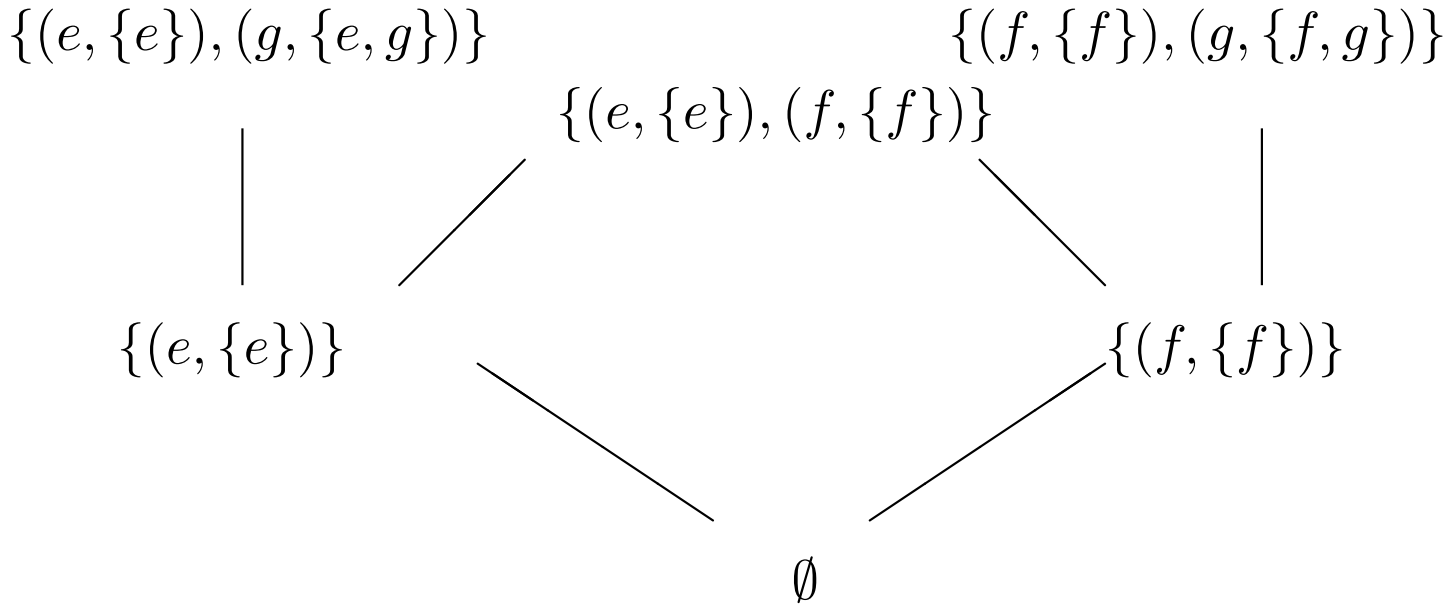
full: $\bigcup F = E$,

prime: for every $x, y \in F$ and every $e \in x \cap y$,
$\bigcap(z \in F : e \in z \subseteq x) = \bigcap(z \in F : e \in z \subseteq y)$.

A pair $\mathbf{C} = (E, F)$, where F is a coherent, stable, coincidence-free, finitary, and full family of subsets of E, will be called a *configuration structure*. The members of F are called *configurations* of $\mathbf{C}$. If F is prime then $\mathbf{C}$ will be called a *prime* configuration structure. If $F = \mathbf{conf}(\mathbf{E})$ for an event structure $\mathbf{E} = (E, \leq, \#)$ then $\mathbf{C}$ will be written as $\mathbf{C}(\mathbf{E})$.

Each configuration structure can be made prime by providing each element of each configuration with minimal subconfigurations containing this element.


$\{e, g\}$
$\{f, g\}$
$\{e, f\}$
$\{e\}$
$\{f\}$
$\emptyset$


$\{(e, \{e\}), (g, \{e, g\})\}$
$\{(f, \{f\}), (g, \{f, g\})\}$
$\{(e, \{e\}), (f, \{f\})\}$
$\{(e, \{e\})\}$
$\{(f, \{f\})\}$
$\emptyset$

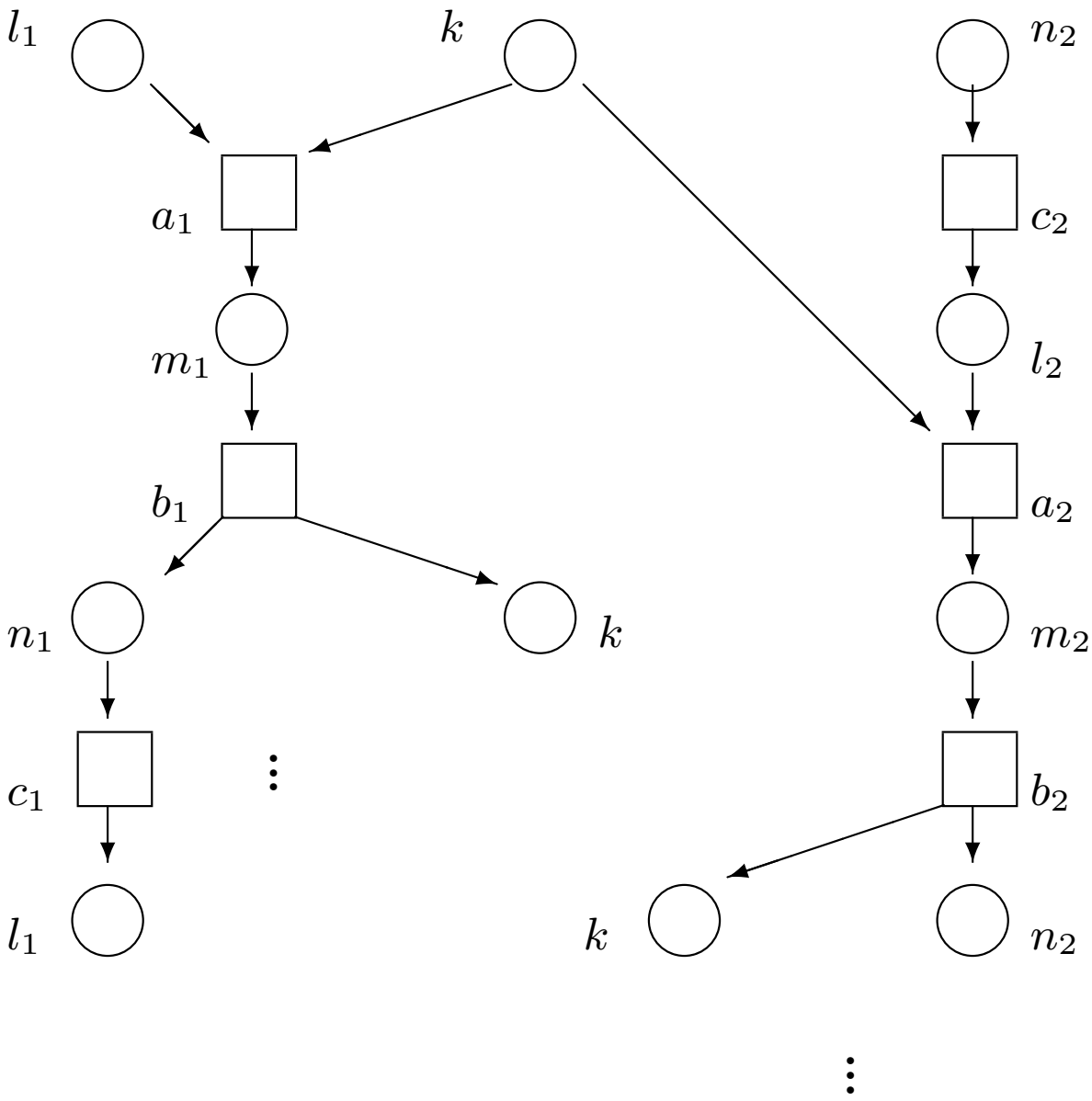
Given a prime configuration structure $\mathbf{C} = (E, F)$, the triple $\mathbf{E}(\mathbf{C}) = (E, \leq, \#)$, where

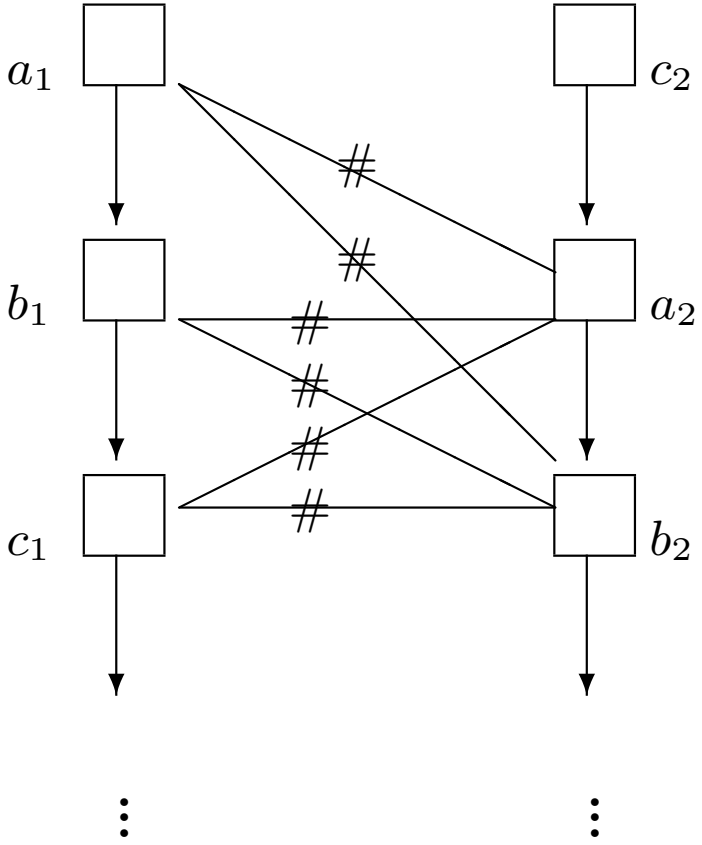
$e' \leq e$ iff for all $x \in F \quad e \in x$ implies $e' \in x$,

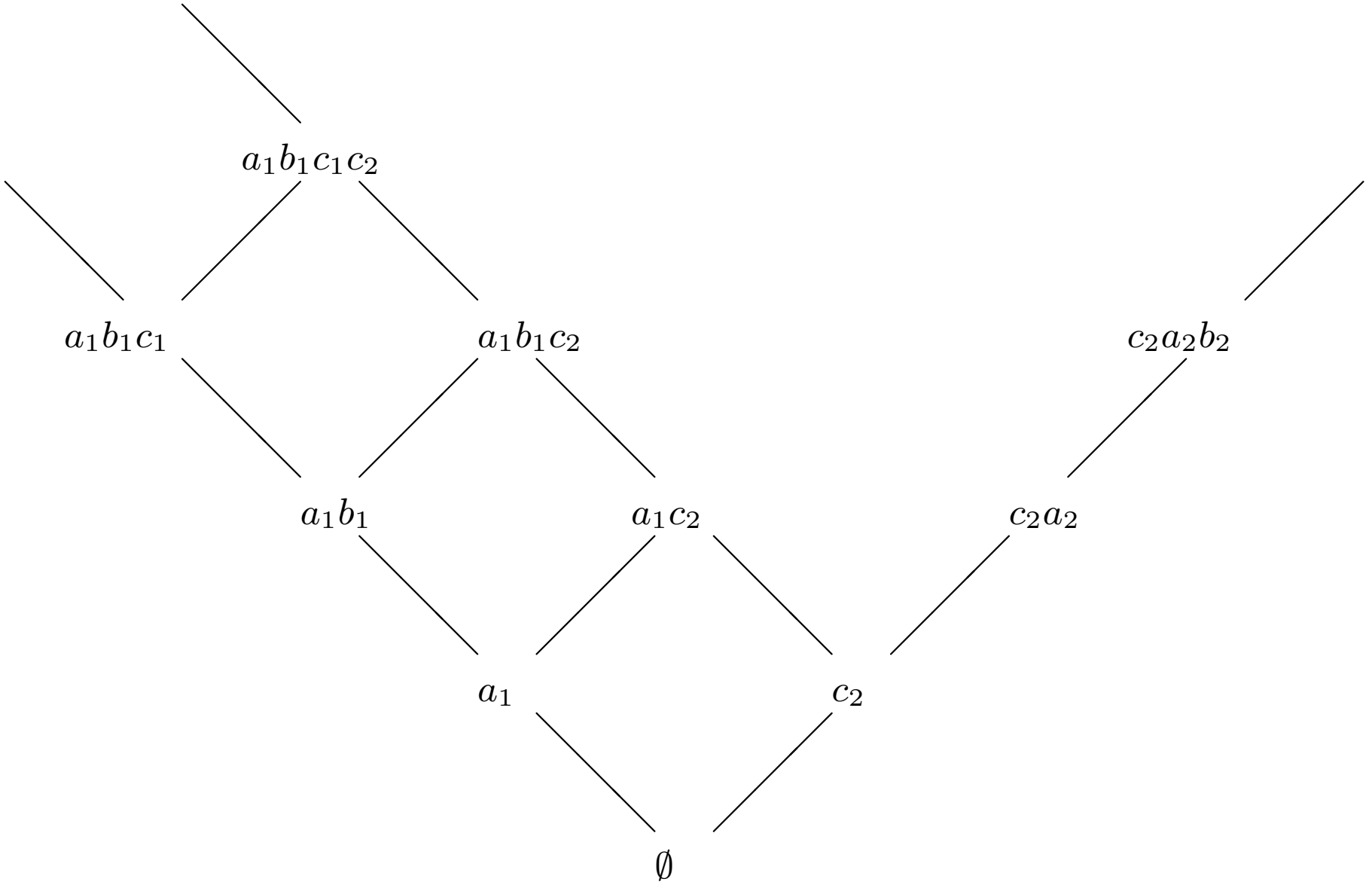
$e \# e'$ iff for all $x \in F \quad e \in x$ implies $e' \notin x$,

is an event structure.

By replacing in the unfolding $U(\mathbf{S}) = (\mathbf{OC}(\mathbf{S}), h)$ of a C/E system $\mathbf{S} = (B, E, F, M_0)$ with the net $\mathbf{N} = (B, E, F)$ the occurrence net $\mathbf{OC}(\mathbf{S})$ by the corresponding event structure $\mathbf{E}(\mathbf{OC}(\mathbf{S})) = (B', E', F')$ and by endowing this structure with the restriction of the mapping h to E' we obtain a *labelled event structure*.


l_1
k
n_2
a_1
c_2
m_1
l_2
b_1
a_2
n_1
k
m_2
c_1
b_2
l_1
k
n_2


a_1
c_2
b_1
a_2
c_1
b_2


$a_1b_1c_1c_2$
$a_1b_1c_1$
$a_1b_1c_2$
$c_2a_2b_2$
a_1b_1
a_1c_2
c_2a_2
a_1
c_2
$\emptyset$

Given a set A of *actions*, by a *labelled event structure* over A we mean $\mathbf{L} = (E, \leq, \#, l)$, where $(E, \leq, \#, l)$ is an event structure and $l : E \to A$ is a mapping (the *labelling*). By a *labelled configuration structure* over $\mathbf{A}$ we mean $\mathbf{D} = (E, F, l)$, where (E, F) is a configuration structure and $l : E \to A$ is a mapping (the *labelling*).

By $\mathbf{les}(A)$ we denote the class of labelled event structures over A.

By $\mathbf{lcs}(A)$ we denote the class of labelled configuration structures over A.

Labelled event structures over A are partially ordered by the following relation $\sqsubseteq$.

Given labelled event structures $\mathbf{L} = (E, \leq, \#, l) \in \mathbf{les}(A)$ and $\mathbf{L}' = (E', \leq', \#', l') \in \mathbf{les}(A)$, we say that $\mathbf{L}$ is an *initial segment* of $\mathbf{L}'$ and we write $\mathbf{L} \sqsubseteq \mathbf{L}'$ iff

$E \subseteq E'$, $\#$ is the restriction of $\#'$ to E, and $\leq e = \leq' e$ for every $e \in E$.

The relation $\sqsubseteq$ is a *chain complete partial order*, i.e., every chain of labelled event structures has the least upper bound. The labelled event structure $\mathbf{nil} = (\emptyset, \emptyset, \emptyset, \emptyset)$ is the least element of $\mathbf{les}(A)$ with respect to this order.

The least upper bound $\bigsqcup C$ of a chain $C = (\mathbf{L}_i : i \in I)$ of labelled event structures $\mathbf{L}_i = (E_i, \leq_i, \#_i, l_i)$ can be defined as the labelled event structure $\mathbf{L} = (E, \leq, \#, l)$, where $E = \bigcup(E_i : i \in I)$, $\leq = \bigcup(\leq_i : i \in I)$, $\# = \bigcup(\#_i : i \in I)$, $l = \bigcup(l_i : i \in I)$.

Labelled configuration structures over A are partially ordered by the following relation $\sqsubseteq$.

Given labelled configuration structures $\mathbf{D} = (E, F, l) \in \mathbf{lcs}(A)$ and $\mathbf{D}' = (E', F', l') \in \mathbf{lcs}(A)$, we say that $\mathbf{D}$ is an *initial segment* of $\mathbf{D}'$ and we write $\mathbf{D} \sqsubseteq \mathbf{D}'$ iff

$E \subseteq E'$, $F \subseteq F'$, every $x \subseteq E$ such that $x \in F'$ belongs to F, and l is the restriction of l' to E.

The relation $\sqsubseteq$ is a *chain complete partial order*, i.e., every chain of labelled configuration structures has the least upper bound. The labelled configuration structure $\mathbf{nilcs} = (\emptyset, \emptyset, \emptyset,)$ is the least element of $\mathbf{lcs}(A)$ with respect to this order.

The least upper bound $\bigsqcup C$ of a chain $C = (\mathbf{D}_i : i \in \omega)$ of labelled event structures $\mathbf{D}_i = (E_i, F_i, l_i)$ can be defined as the labelled event structure $\mathbf{D} = (E, F, l)$, where $E = \bigcup(E_i : i \in \omega)$, $l = \bigcup(l_i : i \in \omega)$, and F is the set of $x \subseteq E$ such that $x = \bigcup(x_i : \imath \in \omega)$ for $x_i = \bigcup(z \in F_i : z \subseteq x)$.

7 . Operations on event structures

A labelled event structure may be used to represent the behaviour of an *agent* which may communicate with its environment.

Concentrating on communication aspect we may use labelled event structures over A, where A consists of actions α, β,... of sending messages, of the corresponding *complementary* actions $\overline{\alpha}$, $\overline{\beta}$,... of receiving messages, and of *internal* actions, all represented by a distinguished element τ.

Moreover, by introducing suitable operations on labelled event structures over A we may represent the behaviours of complex agents by combining behaviours of their components.

As basic operations on labelled event structures we choose *parallel composition*, *summation*, *lifting*, *restriction*, and *relabelling*, which are defined as follows, and the constant **nil**.

(Milner, R., *A Calculus of Communicating Systems*, Springer LNCS 92, 1980)

(Winskel, G., *Event Structure Semantics for CCS and Related Languages*, Springer LNCS 140, 1982)

Parallel composition

Possible communications between two agents set in parallel manifest in the property that some pairs of events of executing complementary actions are combined to form internal events of the resulting agent. The behaviour of this agent can be defined as the parallel composition of the behaviours of its communicating components.

Formally, the parallel composition of labelled event structures $\mathbf{L}_1 = (E_1, \leq_1, \#_1, l_1) \in \mathbf{les}(A)$ and $\mathbf{L}_2 = (E_2, \leq_2, \#_2, l_2) \in \mathbf{les}(A)$ can be defined as follows (cf. Frits W. Vaandrager, Report CS-R8903 of Centre for Mathematics and Computer Science, Amsterdam).

Let $*$ stands for "undefined". Let

$E_1 \times_* E_2 = \{(e, *) : e \in E_1\} \cup \{(*, e) : e \in E_2\}$

$\cup\{(e, e') : e \in E_1, e' \in E_2, l_1(e) \text{ and } l_2(e') \text{ complementary}\}$

with the projections $\pi_i : E_1 \times_* E_2 \to E_i$ be given by $\pi_i((e_1, e_2)) = e_i$ for $i = 0, 1$.

Call a subset $X \subseteq E_1 \times_* E_2$ a *preconfiguration* iff its projections on E_1 and E_2 are finite configurations of the respective event structures, every $e, e' \in X$ such that $\pi_0(e) = \pi_0(e') \neq *$ or $\pi_1(e) = \pi_1(e') \neq *$ are identical, and the reflexive and transitive closure of the following relation is a partial order $\leq_X$ on X:

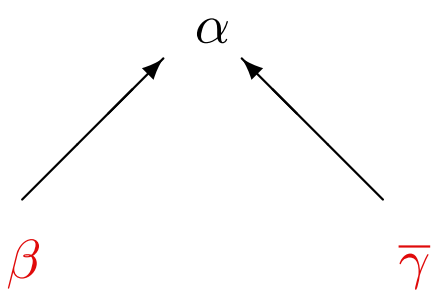
eRe' iff $\pi_1(e) \leq_1 \pi_1(e')$ or $\pi_2(e) \leq_2 (e')$.

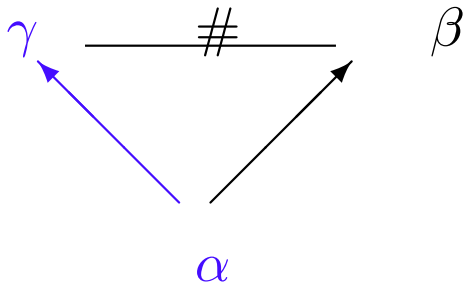
Call a preconfiguration X a *complete prime* and define $l(X)$ as $l_1(e)$, or as $l_2(e')$, or as τ, if X has a unique maximal element $(e, *)$, or $(*, e')$, or (e, e'), respectively.

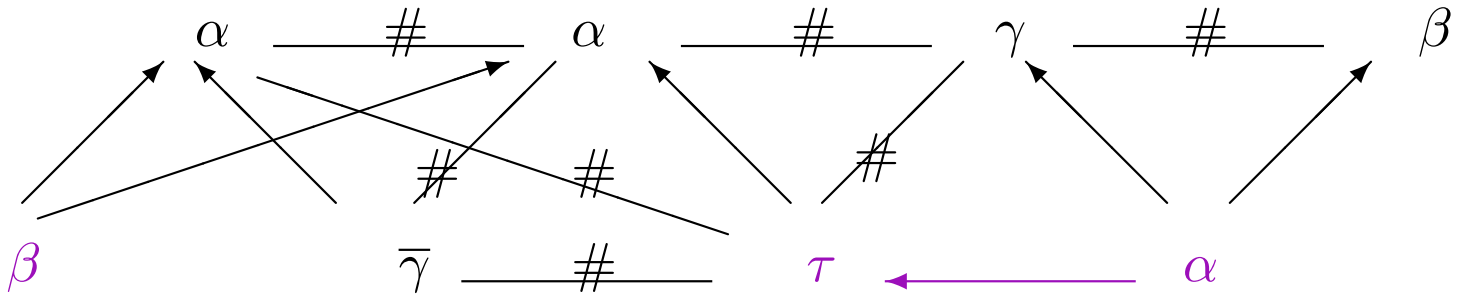
Define $\mathbf{L}$ as $(E, \leq, \#, l)$, where $E = \{X : X \text{ is a complete prime}\}$, $X \leq Y$ iff $X \subseteq Y$, and $X \# Y$ iff $X \cup Y$ is not a preconfiguration.

Then $\mathbf{L}$, the result of the parallel composition of $\mathbf{L}_1$ and $\mathcal{L}_2$, is a labelled event structure, written as $\mathbf{L}_1 \parallel \mathbf{L}_2$.

In order to prove that the result $\mathbf{L}$ of the parallel composition of labelled event structures $\mathbf{L}_1$ and $\mathbf{L}_2$ is a labelled event structure it suffices to consider $X\#Y \leq Z$ and show that $X\#Z$. To this end, it suffices to prove that $X \cup Z$ can not be a preconfiguration. However, if $X \cup Z$ is a preconfiguration then $X \cup Y$ must be a preconfiguration too since $\pi_1(X \cup Y) \subseteq \pi_1(X \cup Z)$ is a finite configuration of $\mathbf{L}_1$ and, similarly, $\pi_2(X \cup Y)$ is a finite configuration of $\mathbf{L}_2$, and $\leq_{X \cup Y}$ is a partial order.


α
β
$\overline{\gamma}$


γ
$\#$
β
α


α
$\#$
α
$\#$
γ
$\#$
β
$\#$
$\#$
$\#$
β
$\overline{\gamma}$
$\#$
τ
α

Similarly, the parallel composition of labelled configuration structures $\mathbf{D}_1 = (E_1, F_1, l_1) \in \mathbf{lcs}(A)$ and $\mathbf{D}_2 = (E_2, F_2, l_2) \in \mathbf{lcs}(A)$ can be defined as follows (cf. Winskel, G., Event structure semantics for CCS and related languages, Springer LNCS 140, 1982, pp. 561-576).

Let $*$ stands for "undefined". Let

$E_1 \times_* E_2 = \{(e, *) : e \in E_1\} \cup \{(*, e) : e \in E_2\}$

$\cup\{(e, e') : e \in E_1, e' \in E_2, l_1(e) \text{ and } l_2(e') \text{ complementary}\}$

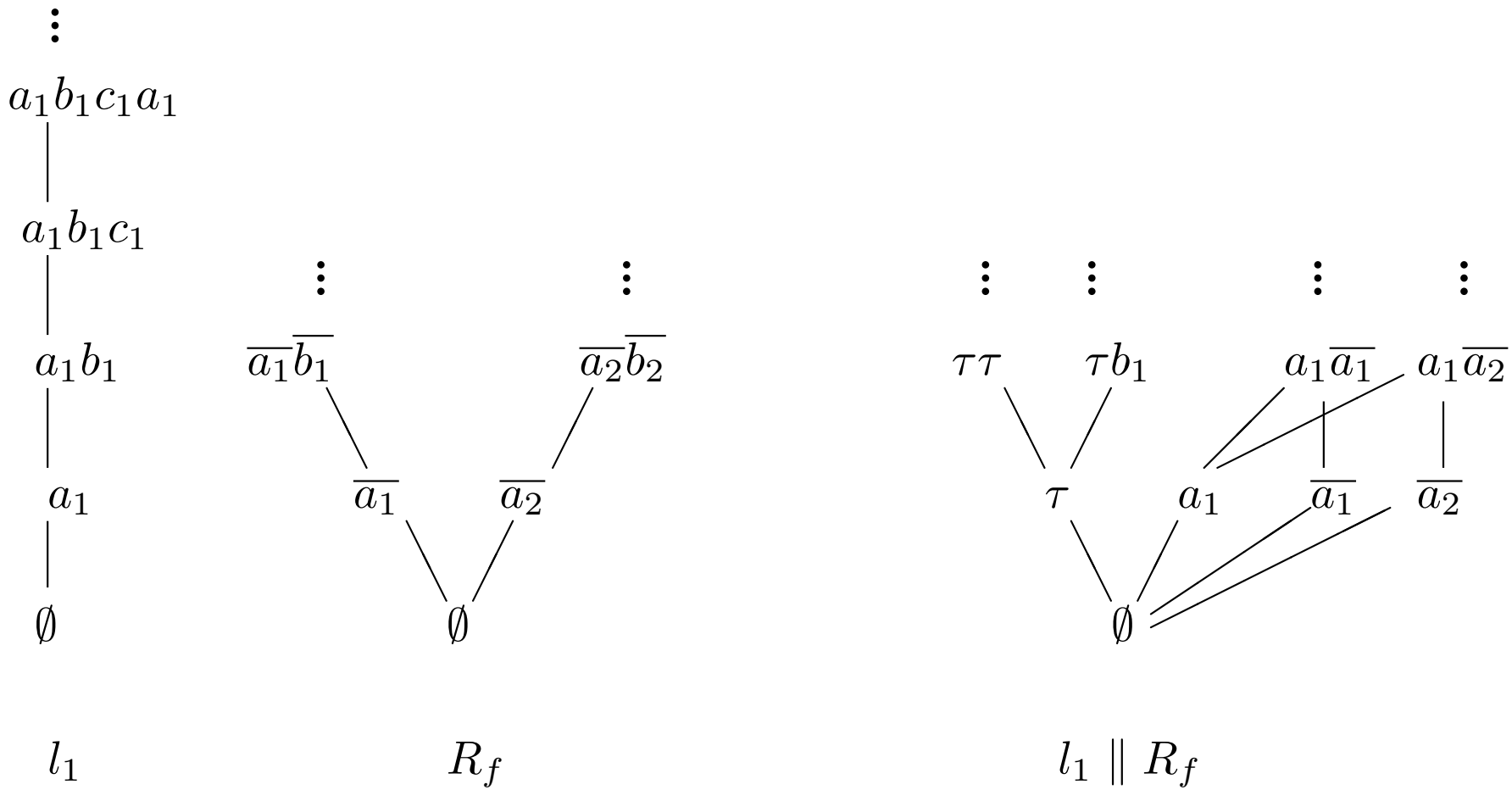
with the projections $\pi_i : E_1 \times_* E_2 \to E_i$ be given by $\pi_i((e_1, e_2)) = e_i$ for $i = 0, 1$.

Let F be the set of subsets $x \subseteq E_1 \times_* E_2$ such that the projections of x on E_1 and E_2 are respectively configurations from F_1 and F_2, every $e, e' \in X$ such that $\pi_0(e) = \pi_0(e') \neq *$ or $\pi_1(e) = \pi_1(e') \neq *$ are identical, for every different e, e' there exists $y \subseteq x$ such that the projections of y on E_1 and E_2 are respectively configurations from F_1 and F_2 and e and e' can be separated by y in the sense that either $e \in y$ and $e' \notin y$ or $e \notin y$ and $e' \in y$, and or every $e \in x$ there exists a finite $y \subseteq x$ such that the projections of y on E_1 and E_2 are respectively configurations from F_1 and F_2 and $e \in y$.

Let $l(e)$ be defined as $l_1(e_1)$, or as $l_2(e_2)$, or as τ, if $e = (e_1, *)$, or $e = (*, e_2)$, or $e = (e_1, e_2)$, respectively.

Define $\mathbf{D}$ as (E, F, l).

Then $\mathbf{D}$, the result of the parallel composition of $\mathbf{D}_1$ and $\mathbf{D}_2$, is a labelled configuration structure, written as $\mathbf{D}_1 \parallel \mathbf{D}_2$.



l_1 R_f $l_1 \parallel R_f$

Summation

The behaviour which proceeds as one indeterministically chosen of two given behaviours can be defined as the sum of these behaviours.

Formally, the sum of labelled event structures $\mathbf{L}_1 = (E_1, \leq_1, \#_1, l_1) \in \mathbf{les}(A)$ and $\mathbf{L}_2 = (E_2, \leq_2, \#_2, l_2) \in \mathbf{les}(A)$ can be defined as the labelled event structure $\mathbf{L} = (E, \leq, \#, l)$, where

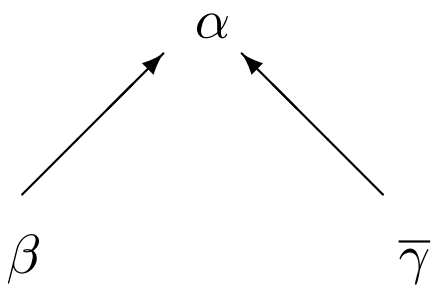
$E = \{(1, e_1) : e_1 \in E_1\} \cup \{(2, e_2) : e_2 \in E_2\}$,

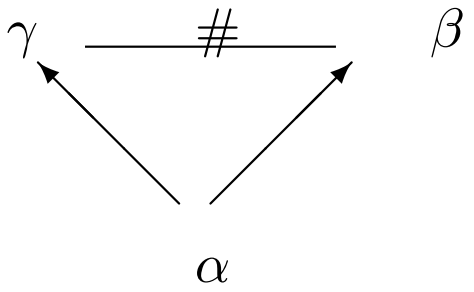
$e \leq e'$ iff $e = (i, e_i)$ and $e' = (i, e'_i)$ and $e_i \leq_i e'_i$ for $i = 1$ or $i = 2$,

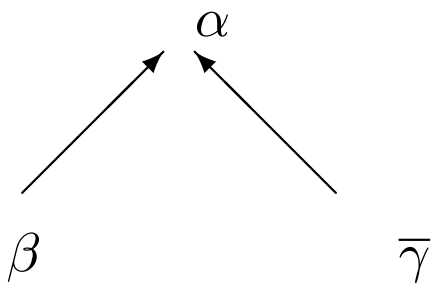
$e \# e'$ iff $e = (i, e_i)$ and $e' = (j, e'_j)$ for $i \neq j$, or $e = (i, e_i)$ and $e' = (i, e'_i)$ and $e_i \#_i e'_i$ for $i = 1$ or $i = 2$,

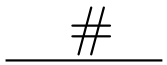
$l(e) = l_i(e_i)$ for $e = (i, e_i)$ and $i = 1, 2$.

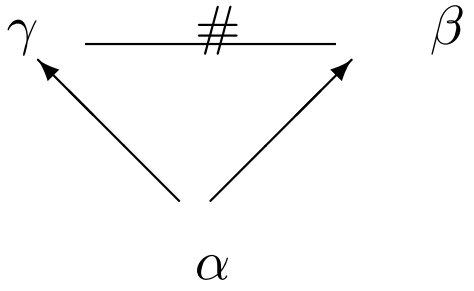
$\mathbf{L}$ is written as $\mathbf{L}_1 + \mathbf{L}_2$.


α
β
$\overline{\gamma}$


γ
$\#$
β
α


α
β
$\overline{\gamma}$


$\#$


γ
$\#$
β
α

Similarly, the sum of labelled configuration structures $\mathbf{D}_1 = (E_1, F_1, l_1) \in \mathbf{lcs}(A)$ and $\mathbf{D}_2 = (E_2, F_2, l_2) \in \mathbf{lcs}(A)$ can be defined as the labelled configuration structure $\mathbf{D} = (E, F, l)$, where

$E = \{(1, e_1) : e_1 \in E_1\} \cup \{(2, e_2) : e_2 \in E_2\}$,

F is the set of sets $\{i\} \times x$ with $x \in F_i, \quad i = 1, 2$,

$l(e) = l_i(e_i)$ for $e = (i, e_i)$ and $i = 1, 2$.

$\mathbf{D}$ is written as $\mathbf{D}_1 + \mathbf{D}_2$.

Lifting

A behaviour preceded by execution of an action can be defined as the result of applying operation called lifting.

Formally, the result of lifting of a labelled event structure $\mathbf{L} = (E, \leq, \#, l)$ by an action $\lambda \in A$ can be defined as the labelled event structure $\mathbf{L}' = (E, \leq', \#', l')$, where

$E' = \{(1, \lambda)\} \cup \{(2, e_2) : e_2 \in E\}$,

$e \leq' e'$ iff $e = (1, \lambda)$ and $e' = (2, e_2')$, or $e = (2, e_2)$ and $e' = (2, e_2')$ and $e_2 \leq e_2'$,

$e \#' e'$ iff $e = (2, e_2)$ and $e' = (2, e_2')$ and $e_2 \# e_2'$

$l'((1, \lambda)) = \lambda$ and $l'(e) = l(e_2)$ for $e = (2, e_2)$.

$\mathbf{L}'$ is written as $\lambda\mathbf{L}$.

Similarly, the result of lifting of a labelled configuration structure $\mathbf{D} = (E, F, l)$ by an action $\lambda \in A$ can be defined as the labelled configuration structure $\mathbf{D}' = (E, F', l')$, where

$E' = \{(1, \lambda)\} \cup \{(2, e_2) : e_2 \in E\}$,

F' is the set of sets $\emptyset$ and $\{(1, \lambda)\} \cup \{2\} \times x$ with $x \in F$,

$l'((1, \lambda)) = \lambda$ and $l'(e) = l(e_2)$ for $e = (2, e_2)$.

$\mathbf{D}'$ is written as $\lambda\mathbf{D}$.

Restriction

Forbiding in a behaviour an action $\alpha \neq \tau$ results in a behaviour without executions of α and events preceded by executions of α. The respective operation is called the restriction.

Formally, the result of restricting an action $\alpha \neq \tau$ in a labelled event structure $\mathbf{L} = (E, \leq, \#, l)$ can be defined as the labelled event structure $\mathbf{L}' = (E', \leq', \#', l')$, where

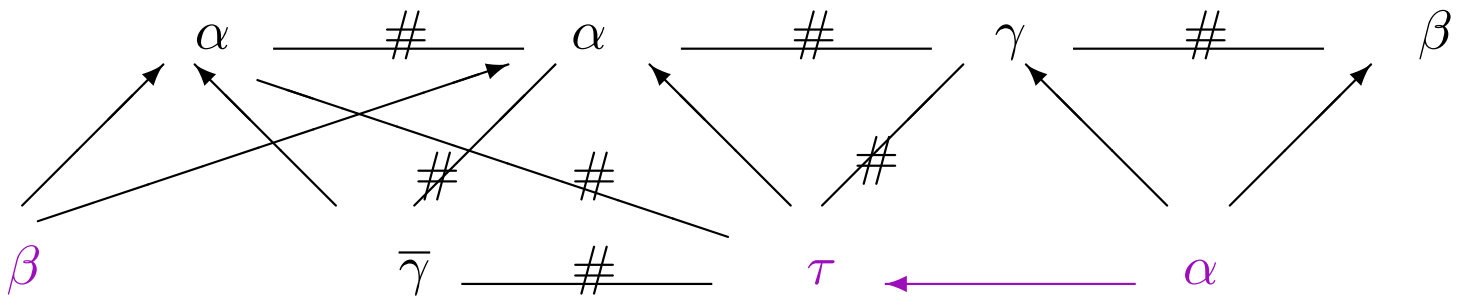
$E' = \{e \in E : l(e') \neq \alpha \text{ for all } e' \leq e\}$,

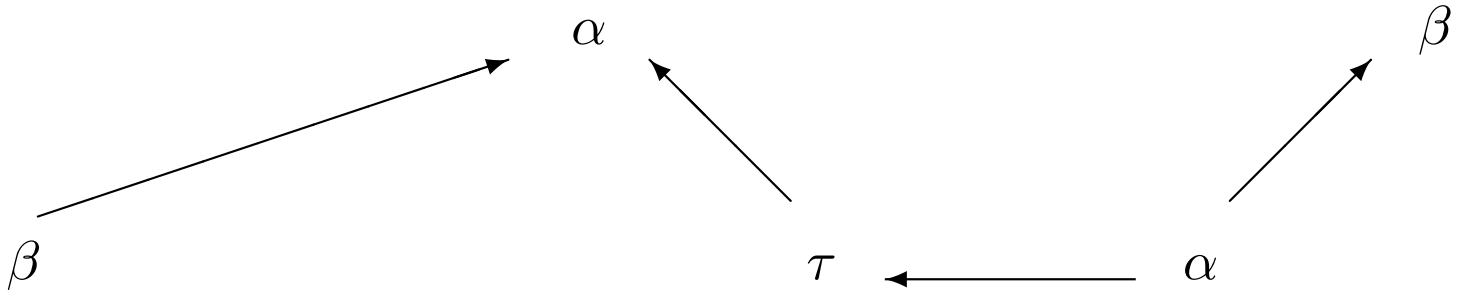
$e \leq' e'$ iff $e, e' \in E'$ and $e \leq e'$,

$e \#' e'$ iff $e, e' \in E'$ and $e \# e'$,

$l' = l|E'$.

$\mathbf{L}'$ is written as $\mathbf{L} - \alpha$.


α
#
α
#
γ
#
β
#
#
#
β
γ
#
τ
α


α
β
β
τ
α

Similarly, the result of restricting an action $\alpha \neq \tau$ in a labelled configuration structure $\mathbf{D} = (E, F, l)$ can be defined as the labelled configuration structure $\mathbf{D}' = (E', F', l')$, where

F' is the set of those $x \in F$ which do not contain any e with $l(e) = \alpha$,

$E' = \bigcup F'$,

$l' = l|E'$.

$\mathbf{D}'$ is written as $\mathbf{D} - \alpha$.

Relabelling

Given a mapping $S : A \to A$ such that $S(\tau) = \tau$ and $S(\overline{\alpha}) = \overline{S(\alpha)}$, replacing in a behaviour every action α by $S(\alpha)$ results in a behaviour.

The respective operation is called the relabelling.

Formally, the result of relabelling of a labelled event structure $\mathbf{L} = (E, \leq, \#, l)$ can be defined as the labelled event structure $\mathbf{L}' = (E', \leq', \#', l')$, where

$E' = E$, $\leq' = \leq$, $\#' = \#$, and $l'(e) = S(l(e))$ for every $e \in E$.

$\mathbf{L}'$ is written as $S(\mathbf{L})$.

Similarly, the result of relabelling of a labelled configuration structure $\mathbf{D} = (E, F, l)$ according to a mapping $S : A \to A$ such that $S(\tau) = \tau$ and $S(\overline{\alpha}) = \overline{S(\alpha)}$ can be defined as the labelled configuration structure $\mathbf{D}' = (E', F', l')$, where

$E' = E$, $F' = F$, $l'(e) = S(l(e))$ for every $e \in E$.

$\mathbf{D}'$ is written as $S(\mathbf{D})$.

The basic operations on labelled event (resp.: configuration) structures are *continuous* w.r. to the prefix order in the sense that they preserve the suprema of countable chains in the respective cartesian powers of $\mathbf{les}(A)$ (resp.: $\mathbf{lcs}(\mathrm{A})$).

Let $F : (\mathbf{les}(A))^{m+n} \to (\mathbf{les}(A))^m$ be a continuous mapping which transforms each pair $(\mathbf{L}, \mathbf{L}')$ with $\mathbf{L} \in (\mathbf{les}(A))^m$ and $\mathbf{L}' \in (\mathbf{les}(A))^n$ into some $\mathbf{L}'' = F(\mathbf{L}, \mathbf{L}') \in (\mathbf{les}(A))^n$. Then we have:

(1) the fixed-point equation $\mathbf{x} = F(\mathbf{x}, \mathbf{y})$ has a least solution, $\mathbf{fix}_{\mathbf{x}} F(\mathbf{x}, \mathbf{y})$,

(2) this solution is given by $\sqcup(\mathbf{x}_i : i \in \omega)$, where $\mathbf{x}_0 = \mathbf{nil}^m$ and $\mathbf{x}_{i+1} = F(\mathbf{x}_i, \mathbf{y})$ for $i \in \omega$,

(3) the correspondence $\mathbf{y} \mapsto \mathbf{fix}_{\mathbf{x}} F(\mathbf{x}, \mathbf{y})$ is a continuous mapping from $(\mathbf{les}(A)^n$ to $(\mathbf{les}(A)^m$.

We call the correspondence between F and $\mathbf{y} \mapsto \mathbf{fix}_{\mathbf{x}} F(\mathbf{x}, \mathbf{y})$ a *fixed-point operator*.

$$F : (l_1, m_1, n_1) \mapsto (a_1 m_1, b_1 n_1, c_1 l_1)$$

$$x = (l_1, m_1, n_1)$$

$$F(x) = (a_1 m_1, b_1 n_1, c_1 l_1)$$

$$x = F(x)$$

$$x_0 = (\mathbf{nil}, \mathbf{nil}, \mathbf{nil})$$

$$x_1 = F(x_0) = (a_1, b_1, c_1)$$

$$x_2 = F(x_1) = (a_1 b_1, b_1 c_1, c_1 a_1)$$

$$...$$

$$x = (a_1 b_1 c_1 a_1 b_1 c_1 ..., b_1 c_1 a_1 b_1 a_1 c_1 ..., c_1 a_1 b_1 c_1 a_1 b_1 ...)$$

$$l_1 = a_1 b_1 c_1 a_1 b_1 c_1 a_1 b_1 c_1 a_1 b_1 c_1 ...$$

$$...$$

$$F : (R_f, R_o) \mapsto (\overline{a_1} R_o + \overline{a_2} R_o, \overline{b_1} R_f + \overline{b_2} R_f)$$

$$x = (R_f, R_o)$$

$$F(x) = (\overline{a_1} R_o + \overline{a_2} R_o, \overline{b_1} R_f + \overline{b_2} R_f)$$

$$x = F(x)$$

$$x_0 = (\mathbf{nil}, \mathbf{nil})$$

$$x_1 = F(x_0) = (\overline{a_1} + \overline{a_2}, \overline{b_1} + \overline{b_2})$$

$$x_2 = F(x_1) = (\overline{a_1}(\overline{b_1} + \overline{b_2}) + \overline{a_2}(\overline{b_1} + \overline{b_2}), \overline{b_1}(\overline{a_1} + \overline{a_2}) + \overline{b_2}(\overline{a_1} + \overline{a_2}))$$

$$...$$

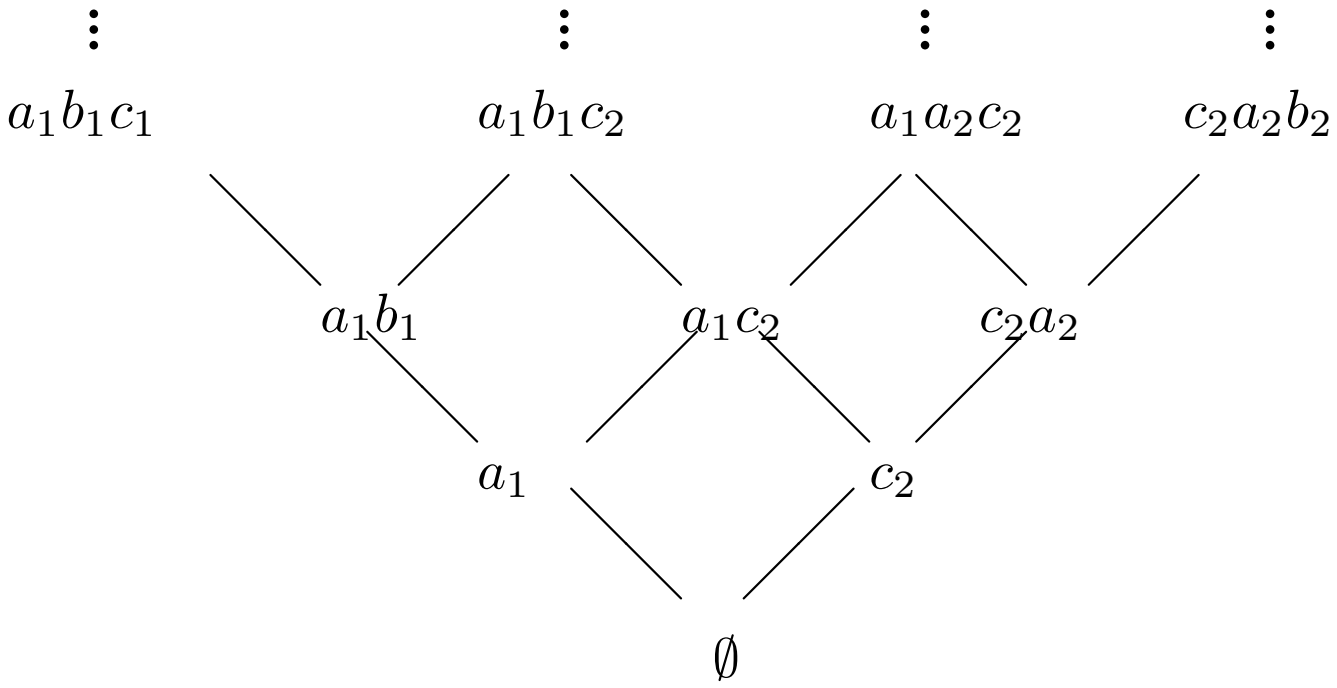
$$R_f = \overline{a_1}(\overline{b_1} + \overline{b_2}) + \overline{a_2}(\overline{b_1} + \overline{b_2})...$$

$$...$$

The fact that superpositions of continuous operations remain continuous allow us to obtain a broad class of continuous operations.

The operations in $\mathbf{les}(A)$ which can be obtained by combining constant labelled event structures and basic operations with the aid of superpositions and fixed-point operators are continuous. We call them *definable* operations.

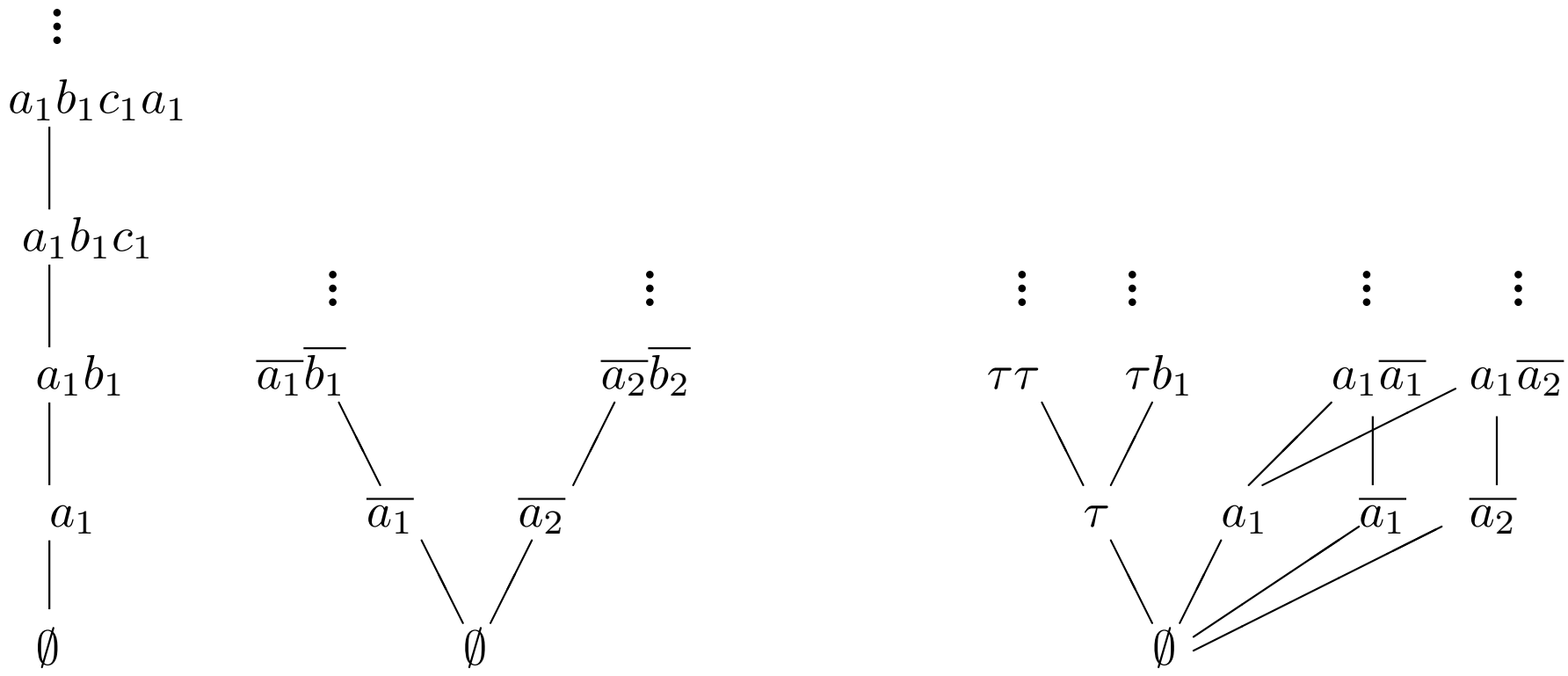
For example, the operation $\mathbf{L} \mapsto \mathbf{L}_0$ with a fixed $\mathbf{L}_0 \in \mathbf{les}(A)$ is definable, the operation $\mathbf{L} \mapsto \mathbf{fix}_{\mathbf{x}}(\mathbf{x} \parallel \mathbf{L})$ is definable, etc.



$$l_1 \parallel n_2$$

$$l_i = a_i m_i \quad m_i = b_i n_i \quad n_i = c_i l_i$$

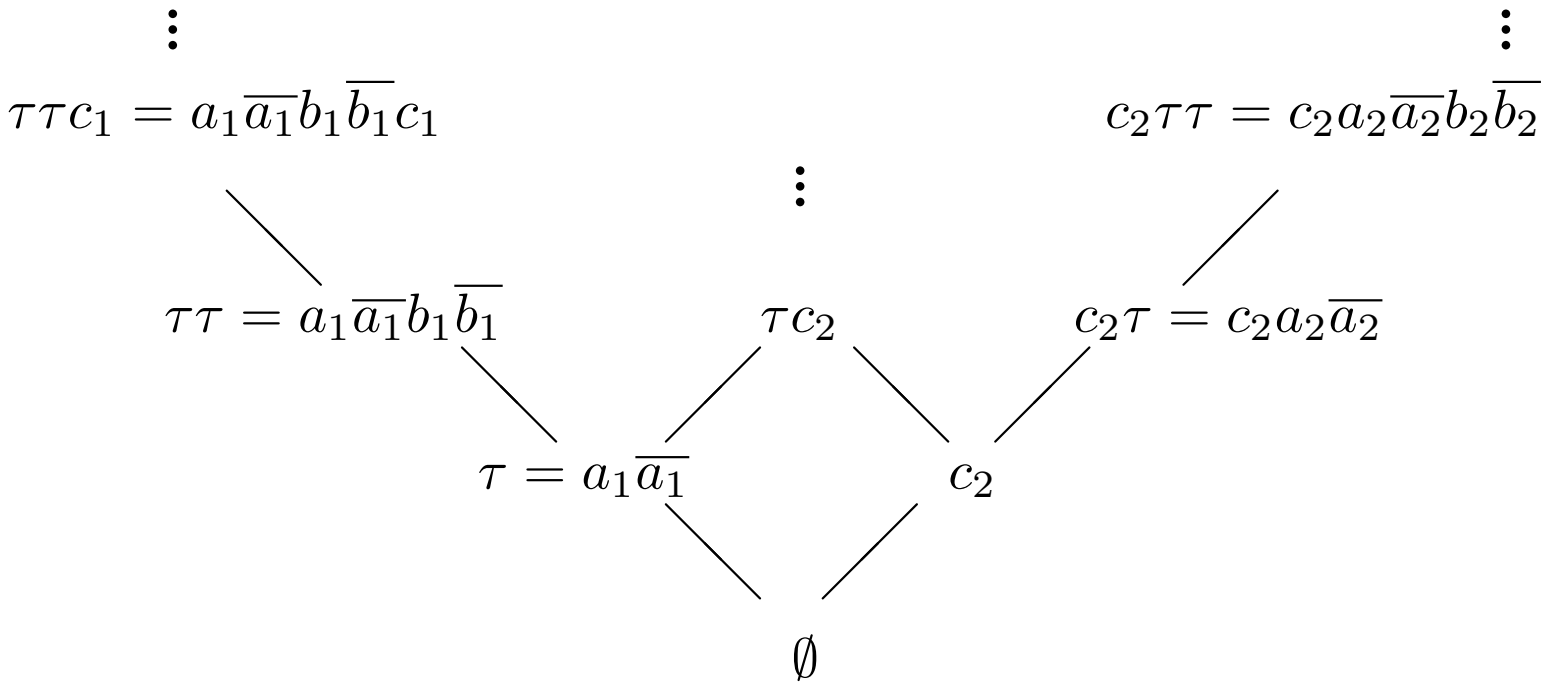
$$R_f = \overline{a_1} R_o + \overline{a_2} R_o \qquad R_o = \overline{b_1} R_f + \overline{b_2} R_f$$



l_1 $\qquad$ R_f $\qquad$ $l_1 \parallel R_f$

$$l_i = a_i m_i \quad m_i = b_i n_i \quad n_i = c_i l_i$$

$$R_f = \overline{a_1} R_o + \overline{a_2} R_o \qquad R_o = \overline{b_1} R_f + \overline{b_2} R_f$$



$$(l_1 \parallel R_f \parallel n_2) - \{a_1, a_2, b_1, b_2, \overline{a_1}, \overline{a_2}, \overline{b_1}, \overline{b_2}\}$$

$$l_i = a_i m_i \quad m_i = b_i n_i \quad n_i = c_i l_i$$

$$R_f = \overline{a_1}R_o + \overline{a_2}R_o \qquad R_o = \overline{b_1}R_f + \overline{b_2}R_f$$

8. Equivalence

As usual in mathematics, we consider labelled event structures up to isomorphisms, where isomorphisms are defined as follows.

An *isomorphism* between two labelled event structures $\mathbf{L} = (E, \leq, \#, l)$ and $\mathbf{L}' = (E', \leq', \#', l')$ is a bijective correspondence $f : E \to E'$ such that $e_1 \leq e_2$ iff $f(e_1) \leq' f(e_2)$, $e_1 \# e_2$ iff $f(e_1) \#' f(e_2)$, and $l'(f(x)) = l(x)$, for all $e_1, e_2, e \in E$. If such an isomorphism exists then we say that $\mathbf{L}$ and $\mathbf{L}'$ are *isomorphic* and write $\mathbf{L} \approx \mathbf{L}'$.

For labelled event structures we have a number of equivalences which reflect some similarities of the represented behaviours. Some of them are congruences for the definable operations on labelled event structures. We adapt one of such congruences - a variant of the so called history preserving equivalence (cf. A. RABINOVICH, B. A. TRAKHTENBROT, Behaviour structure and nets, *Fundamenta Informaticae* **11**(4), 1988, pp.357-404). Our definition of the history preserving equivalence is based on a concept of simulation.

A (*history preserving*) *simulation* of a labelled event structure $\mathbf{L} = (E, \leq, \#, l)$ in a labelled event structure $\mathbf{L}' = (E', \leq', \#', l')$ is a family $\varrho = (\varrho_{pq} : p \in \mathbf{conf}(\mathbf{L}), q \in \mathbf{conf}(\mathbf{L}'))$ where:

(1) each ϱ_{pq} is a set of isomorphisms $\varphi : \mathbf{L}|p \to \mathbf{L}''|\mathbf{q}$,

(2) $\varrho_{\emptyset\emptyset}$ is the one-element set consisting of the empty isomorphism, $\emptyset : \mathbf{nil} \to \mathbf{nil}$,

(3) for each $\varphi \in \varrho_{pq}$ and each $p' \in \mathbf{conf}(\mathbf{L})$ such that $p \subseteq p'$ there exists $q' \in \mathbf{conf}(\mathbf{L}')$ such that $q \subseteq q'$ and φ has an extension φ' in $\varrho_{p'q'}$.

We write such a simulation as $\varrho : \mathbf{L} \to \mathbf{L}'$ or $\mathbf{L} \xrightarrow{\varrho} \mathbf{L}'$. If $\hat{\varrho} = (\hat{\varrho}_{qp} : q \in \mathbf{conf}(\mathbf{L}'), p \in \mathbf{conf}(\mathbf{L}))$, where $\hat{\varrho}_{qp} = \varrho_{pq}$, is also a simulation then we say that $\varrho : \mathbf{L} \to \mathbf{L}'$ is a *bisimulation*.

Each isomorphism f from a labelled event structure $\mathbf{L}$ to a labelled event structure $\mathbf{L}'$ defines a bisimulation $\bar{f} : P \to Q$, where $\bar{f}_{pq}$ is the one-element set consisting of the restriction of f to $\mathbf{L}|p$ for $q = f(p)$ and the empty set for $q \neq f(p)$.

For every two labelled event structures $\mathbf{L} = (E, \leq, \#, l)$ and $\mathbf{L}' = (E', \leq', \#', l')$ such that $\mathbf{L} \sqsubseteq \mathbf{L}'$ there exists a simulation $\mathbf{in} : \mathbf{L} \to \mathbf{L}'$, where each $\mathbf{in}_{pq}$ is the one-element set consisting of the identity mapping on p for $p = q$ and the empty set for $p \neq q$.

The following property of simulations allows one to compose them.

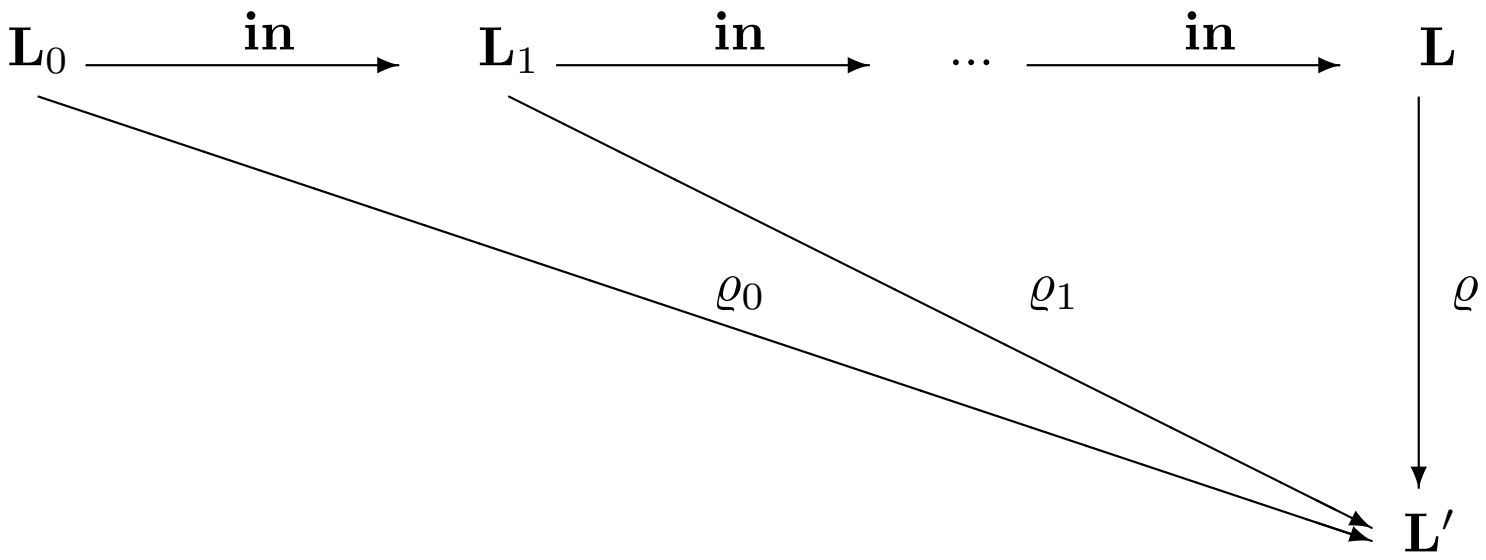
If $\varrho = (\varrho_{pq} : p \in \mathbf{conf}(\mathbf{L}), q \in \mathbf{conf}(\mathbf{L}'))$ is a simulation of $\mathbf{L} = (E, \leq, \#, l)$ in $\mathbf{L}' = (E', \leq', \#', l')$ and $\sigma = (\sigma_{qr} :\in \mathbf{conf}(\mathbf{L}'), r \in \mathbf{conf}(\mathbf{L}''))$ is a simulation of $\mathbf{L}''$ in $\mathbf{L}'' = (E'', \leq'', \#'', l'')$ then
$\varrho\sigma = (\bigcup(\varrho_{pq}\sigma_{qr} : q \in \mathbf{conf}(\mathbf{L}')) : p \in \mathbf{conf}(\mathbf{L}), r \in \mathbf{conf}(\mathbf{L}''),$ where
$\varrho_{pq}\sigma_{qr} = \{\varphi\psi : \varphi \in \varrho_{pq}, \psi \in \sigma_{qr}\}$ and $\varphi\psi$ is defined by $\varphi\psi(x) = \psi(\varphi(x))$, is a simulation of $\mathbf{L}$ in $\mathbf{L}''$. If ϱ and σ are bisimulations then so is $\varrho\sigma$.

The following relation between labelled event structures is an equivalence:

$$\mathbf{L} \sim \mathbf{L}' \text{ iff there exists a bisimulation } \varrho : \mathbf{L} \to \mathbf{L}'$$

We call it the *history preserving equivalence*.

The labelled event structures over a universe A of actions and their simulations constitute a category $\mathbf{LES}(A)$. The category $\mathbf{LES}(A)$ has colimits of countable chains. The colimit of each chain $\mathbf{L}_0 \sqsubseteq \mathbf{L}_1 \sqsubseteq ...$ coincides with its supremum $\mathbf{L} = \sqcup(\mathbf{L}_i : i \in \omega) = \bigcup(\mathbf{L}_i : i \in \omega)$. Similarly for the cartesian powers of $\mathbf{LES}(A)$.



For each commutative diagram as above with a unique simulation $\varrho : \mathbf{L} \to \mathbf{L}'$ resulting from the universal properties of colimits, this unique simulation is determined by the property: $\varrho_{pq} = (\varrho_i)_{pq}$ for all $i \in \omega$ such that $p \in \mathbf{conf}(\mathbf{L}_i)$.

The relation between the history preserving equivalence and the definable operations on labelled event structures can be studied with the aid of special functors between cartesian powers of the category $\mathbf{LES}(A)$.

Let $F : (\mathbf{LES}(A))^m \to (\mathbf{LES}(A))^n$ be a functor. We say that F in *continuous* if it preserves colimits. We say that F *preserves the prefix order* if, for all $\mathbf{L}$, $\mathbf{L}'$, $\mathbf{L} \sqsubseteq \mathbf{L}'$ implies $F(\mathbf{L}) \sqsubseteq F(\mathbf{L}')$ and the coincidence of $F(\mathbf{in}) : F(\mathbf{L}) \to F(\mathbf{L}')$ with $\mathbf{in} : F(\mathbf{L}) \to F(\mathbf{L}')$. Finally, we say that F *preserves bisimulations* if $F(\varrho) : F(\mathbf{L}) \to F(\mathbf{L}')$ is a bisimulation whenever $\varrho : \mathbf{L} \to \mathbf{L}'$ is a bisimulation.

The following property of definable operations on labelled event structures is crucial for their effect on the history preserving equivalence.

Each definable operation on labelled event structures from $\mathbf{les}(A)$ can be extended in a canonical way to a continuous functor which preserves the prefix order and bisimulations.

Consequently, the history preserving equivalence is a congruence for all definable operations on labelled event structures.

Functors $F : (\mathbf{LES}(A))^m \to (\mathbf{LES}(A))^n$ and $G : (\mathbf{LES}(A))^m \to (\mathbf{LES}(A))^n$ are said to be *equivalent* (with respect to the history preserving equivalence) if there exists a natural transformation $\varrho : F \to G$ which consists of bisimulations, that is a family $\varrho = (\varrho(\mathbf{L}) : F(\mathbf{L}) \to G(\mathbf{L}) : \mathbf{L} \in (\mathbf{les}(A))^m)$ of bisimulations such that, for each simulation $\sigma : \mathbf{L} \to \mathbf{L}'$, we have $\varrho(\mathbf{L})G(\sigma) = F(\sigma)\varrho(\mathbf{L}')$. Two definable operations on labelled configuration structures are said to be *equivalent* if their canonical extensions to functors which preserve the prefix order and bisimulations are equivalent.

The operations $P \mapsto P$ and $P \mapsto P + P$ are equivalent with the equivalence given for $P = (E, \leq, \#, l)$ by the family $\beta = (\beta(P) : P \to P + P : P \in \mathbf{les}(A)))$ of bisimulations $(\beta(P))_{pq}$,

where $(\beta(P))_{pq}$ is the one-element set consisting of the bijection $\varphi : p \to q$ defined by $\varphi(x) = ((0, x), l(x))$ for all $p \in \mathbf{conf}(P)$ and $q \in \mathbf{conf}(P + P)$ such that $q = \varphi(p)$,

where $(\beta(P))_{pq}$ is the one-element set consisting of the bijection $\psi : p \to q$ defined by $\psi(x) = ((1, y), l(y))$ for all $p \in \mathbf{conf}(P)$ and $q \in \mathbf{conf}(P + P)$ such that $q = \psi(p)$,

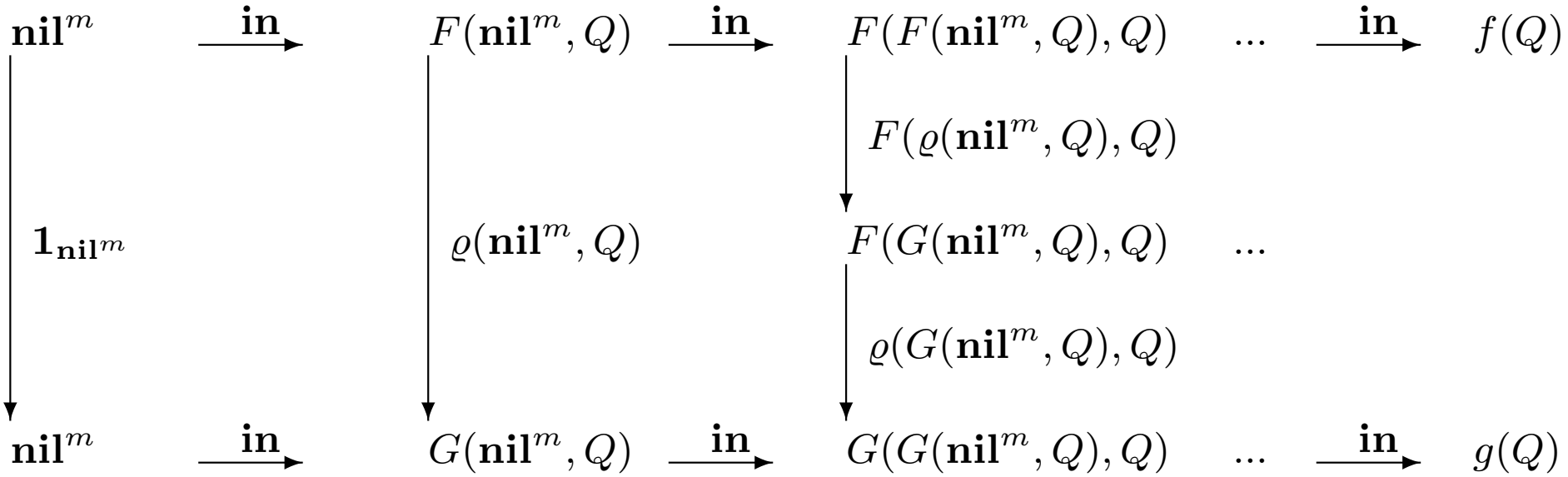
and where $(\beta(P))_{pq}$ is the empty set in all the remaining cases.

Similarly, the operations $(P, Q) \mapsto P + Q$ and $(P, Q) \mapsto Q + P$ are equivalent, the operations $(P, Q, R) \mapsto (P + Q) + R$ and $(P, Q, R) \mapsto P + (Q + R)$ are equivalent, the operations $(P, Q) \mapsto (P \parallel Q)$ and $(P, Q) \mapsto Q \parallel P$ are equivalent, and the operations $(P, Q, R) \mapsto (P \parallel Q) \parallel R$ and $(P, Q, R) \mapsto P \parallel (Q \parallel R)$ are equivalent.

For the equivalence of definable operations on labelled event structures we have the following result.

If two definable operations on labelled event structures are constructed in the same manner from equivalent definable operations then they are equivalent.

The only nontrivial part of the proof is that about operations defined by fixed-point equations.



The effect of fixed-point operators on the equivalence of operations

End