# Concurrency Theory

## Lecture 18: True Concurrency Semantics of Petri Nets (2)

Joost-Pieter Katoen and Thomas Noll

Lehrstuhl für Informatik 2
Software Modeling and Verification Group

http://moves.rwth-aachen.de/teaching/ws-19-20/ct

December 10, 2019

# Overview

# Overview

# Introduction

- Interleaving semantics of Petri nets = set of sequential runs
  - a sequential run is a total ordering of transition occurrences

# Introduction

- Interleaving semantics of Petri nets = set of sequential runs
  - a sequential run is a total ordering of transition occurrences
- The set of all sequential runs can be represented by a marking graph

# Introduction

- Interleaving semantics of Petri nets = set of sequential runs
  - a sequential run is a total ordering of transition occurrences
- The set of all sequential runs can be represented by a marking graph
- Partial-order semantics of Petri nets = set of distributed runs
  - a distributed run is an acyclic (causal) net which contains no choices
  - a distributed run is a partial ordering of transition occurrences

# Introduction

- Interleaving semantics of Petri nets = set of sequential runs
  - a sequential run is a total ordering of transition occurrences
- The set of all sequential runs can be represented by a marking graph
- Partial-order semantics of Petri nets = set of distributed runs
  - a distributed run is an acyclic (causal) net which contains no choices
  - a distributed run is a partial ordering of transition occurrences
- Today: the set of all distributed runs can be represented by a specific branching process, the unfolding

# Branching process: preamble

- A branching process represents a set of distributed runs

[1] In net jargon, a choice is called a conflict.

# Branching process: preamble

- A branching process represents a set of distributed runs
- It explicitly represents each possible resolution of each choice[1]

[1] In net jargon, a choice is called a conflict.

# Branching process: preamble

- A branching process represents a set of distributed runs
- It explicitly represents each possible resolution of each choice[1]
- It is an acyclic (occurrence) net containing choices.

[1] In net jargon, a choice is called a conflict.

# Branching process: preamble

- A branching process represents a set of distributed runs
- It explicitly represents each possible resolution of each choice[1]
- It is an acyclic (occurrence) net containing choices.
- It is a partial ordering with conflicts of transition occurrences.

[1] In net jargon, a choice is called a conflict.

# Branching process: preamble

- A branching process represents a set of distributed runs
- It explicitly represents each possible resolution of each choice[1]
- It is an acyclic (occurrence) net containing choices.
- It is a partial ordering with conflicts of transition occurrences.
- The true concurrency semantics of a net is a specific branching process, called unfolding.

[1] In net jargon, a choice is called a conflict.

# Branching process: preamble

- A branching process represents a set of distributed runs
- It explicitly represents each possible resolution of each choice[1]
- It is an acyclic (occurrence) net containing choices.
- It is a partial ordering with conflicts of transition occurrences.
- The true concurrency semantics of a net is a specific branching process, called unfolding.
- A net unfolding is the true concurrency counterpart of a marking graph.

[1] In net jargon, a choice is called a conflict.

# Branching process: preamble

- A branching process represents a set of distributed runs
- It explicitly represents each possible resolution of each choice[1]
- It is an acyclic (occurrence) net containing choices.
- It is a partial ordering with conflicts of transition occurrences.
- The true concurrency semantics of a net is a specific branching process, called unfolding.
- A net unfolding is the true concurrency counterpart of a marking graph.
- It is the unique maximal branching process in a complete lattice.

[1] In net jargon, a choice is called a conflict.

# Branching process: preamble

- A branching process represents a set of distributed runs
- It explicitly represents each possible resolution of each choice[1]
- It is an acyclic (occurrence) net containing choices.
- It is a partial ordering with conflicts of transition occurrences.
- The true concurrency semantics of a net is a specific branching process, called unfolding.
- A net unfolding is the true concurrency counterpart of a marking graph.
- It is the unique maximal branching process in a complete lattice.
- The reachable markings of a 1-bounded net are covered by a finite prefix of this maximal branching process.

[1] In net jargon, a choice is called a conflict.

# Overview

# Elementary system nets

## Net

An elementary net system $N$ is a tuple $(P, T, F, M_0)$ where:

- $P$ is a countable set of places
- $T$ is a countable set of transitions with $P \cap T = \varnothing$
- $F \subseteq (P \times T) \cup (T \times P)$ are the arcs satisfying:

$$\forall t \in T.\ {}^\bullet t \text{ and } t^\bullet \text{ are finite and non-empty}$$

- $M_0 : P \to \mathbb{N}$ is the initial marking.

Places and transitions are generically called nodes.

# Elementary system nets

## Net

An elementary net system $N$ is a tuple $(P, T, F, M_0)$ where:

- $P$ is a countable set of places
- $T$ is a countable set of transitions with $P \cap T = \varnothing$
- $F \subseteq (P \times T) \cup (T \times P)$ are the arcs satisfying:

$$\forall t \in T.\ {}^\bullet t \text{ and } t^\bullet \text{ are finite and non-empty}$$

- $M_0 : P \to \mathbb{N}$ is the initial marking.

Places and transitions are generically called nodes.

Assumption: (possibly) infinite elementary nets are 1-bounded. Thus any marking can be viewed as a subset of places.

# Causal nets

# Causal nets

A causal net constitutes the basis for a “distributed” run.

# Causal nets

A causal net constitutes the basis for a "distributed" run.

It is a (possibly infinite) net which satisfies:

# Causal nets

A causal net constitutes the basis for a "distributed" run.

It is a (possibly infinite) net which satisfies:

1. It has no place branches: at most one arc ends or starts in a place

# Causal nets

A causal net constitutes the basis for a "distributed" run.

It is a (possibly infinite) net which satisfies:

1. It has no place branches: at most one arc ends or starts in a place
2. It is acyclic

# Causal nets

A causal net constitutes the basis for a “distributed” run.
It is a (possibly infinite) net which satisfies:

1. It has no place branches: at most one arc ends or starts in a place
2. It is acyclic
3. Each sequences of arcs (flows) has a unique first element

# Causal nets

A causal net constitutes the basis for a "distributed" run.
It is a (possibly infinite) net which satisfies:

1. It has no place branches: at most one arc ends or starts in a place
2. It is acyclic
3. Each sequences of arcs (flows) has a unique first element
4. The initial marking contains all places without incoming arcs.

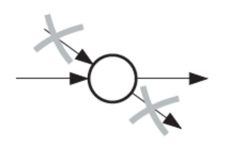
# Causal nets

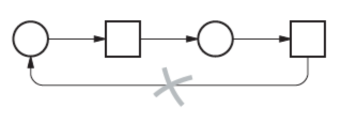
A causal net constitutes the basis for a "distributed" run.
It is a (possibly infinite) net which satisfies:

1. It has no place branches: at most one arc ends or starts in a place
2. It is acyclic
3. Each sequences of arcs (flows) has a unique first element
4. The initial marking contains all places without incoming arcs.

**Intuition**

No place branches, no sequence of arcs forms a loop, and each sequence of arcs has a first node.





# Causal nets

A causal net constitutes the basis for a distributed run.

It is a possibly infinite net which satisfies:

1. Has no place branches: at most one arc ends or starts in a place
2. Is acyclic
3. Each sequences of arcs (flows) has a first element
4. The initial marking contains all places without incoming arcs

# Causal nets

A causal net constitutes the basis for a distributed run.

It is a possibly infinite net which satisfies:

1. Has no place branches: at most one arc ends or starts in a place
2. Is acyclic
3. Each sequences of arcs (flows) has a first element
4. The initial marking contains all places without incoming arcs

**Causal net**

A (possibly infinite) net $K = (Q, V, G, M_0)$ is called a causal net iff:

# Causal nets

A causal net constitutes the basis for a distributed run.

It is a possibly infinite net which satisfies:

1. Has no place branches: at most one arc ends or starts in a place
2. Is acyclic
3. Each sequences of arcs (flows) has a first element
4. The initial marking contains all places without incoming arcs

## Causal net

A (possibly infinite) net $K = (Q, V, G, M_0)$ is called a causal net iff:

1. for each $q \in Q$, $|{}^\bullet q| \leqslant 1$ and $|q^\bullet| \leqslant 1$

# Causal nets

A causal net constitutes the basis for a distributed run.

It is a possibly infinite net which satisfies:

1. Has no place branches: at most one arc ends or starts in a place
2. Is acyclic
3. Each sequences of arcs (flows) has a first element
4. The initial marking contains all places without incoming arcs

## Causal net

A (possibly infinite) net $K = (Q, V, G, M_0)$ is called a causal net iff:

1. for each $q \in Q$, $|^\bullet q| \leqslant 1$ and $|q^\bullet| \leqslant 1$
2. the transitive closure (called causal order) $G^+$ of $G$ is irreflexive

# Causal nets

A causal net constitutes the basis for a distributed run.

It is a possibly infinite net which satisfies:

1. Has no place branches: at most one arc ends or starts in a place
2. Is acyclic
3. Each sequences of arcs (flows) has a first element
4. The initial marking contains all places without incoming arcs

## Causal net

A (possibly infinite) net $K = (Q, V, G, M_0)$ is called a causal net iff:

1. for each $q \in Q$, $|{}^\bullet q| \leqslant 1$ and $|q^\bullet| \leqslant 1$
2. the transitive closure (called causal order) $G^+$ of $G$ is irreflexive
3. for each node $x \in Q \cup V$, the set $\{ y \mid (y, x) \in G^+ \}$ is finite

# Causal nets

A causal net constitutes the basis for a distributed run.

It is a possibly infinite net which satisfies:

1. Has no place branches: at most one arc ends or starts in a place
2. Is acyclic
3. Each sequences of arcs (flows) has a first element
4. The initial marking contains all places without incoming arcs

## Causal net

A (possibly infinite) net $K = (Q, V, G, M_0)$ is called a causal net iff:

1. for each $q \in Q$, $|{}^\bullet q| \leqslant 1$ and $|q^\bullet| \leqslant 1$
2. the transitive closure (called causal order) $G^+$ of $G$ is irreflexive
3. for each node $x \in Q \cup V$, the set $\{ y \mid (y, x) \in G^+ \}$ is finite
4. $M_0$ equals the minimal set of places in $K$ under $G^+$, i.e.,

$$M_0 \; = \; {}^\circ K \; = \; \{ q \in Q \mid {}^\bullet q = \varnothing \}.$$

# Properties of causal nets

# Properties of causal nets

**Boundedness of causal nets**

Every causal net is one-bounded, i.e., in every marking every place will hold at most one token.

# Properties of causal nets

## Boundedness of causal nets

Every causal net is one-bounded, i.e., in every marking every place will hold at most one token.

## Absence of superfluous places and transitions

Every causal net has a step sequence that visits all places and fires every transition.

# What is a distributed run?

## Distributed run

A distributed run of a one-bounded elementary net system $N$ is:

1. a labeled causal net $K$
2. in which each transition $t$ (with ${}^\bullet t$ and $t^\bullet$) is an action of $N$.

# What is a distributed run?

## Distributed run

A distributed run of a one-bounded elementary net system $N$ is:

1. a labeled causal net $K$
2. in which each transition $t$ (with ${}^\bullet t$ and $t^\bullet$) is an action of $N$.

A distributed run $K$ of $N$ is complete iff (the marking) ${}^\circ K$ represents the initial marking of $N$ and (the marking) $K^\circ$ does not enable any transition.

# What is a distributed run?

## Distributed run

A distributed run of a one-bounded elementary net system $N$ is:

1. a labeled causal net $K$
2. in which each transition $t$ (with ${}^\bullet t$ and $t^\bullet$) is an action of $N$.

A distributed run $K$ of $N$ is complete iff (the marking) ${}^\circ K$ represents the initial marking of $N$ and (the marking) $K^\circ$ does not enable any transition.

Examples on the black board.

# What is a distributed run?

## Distributed run

A distributed run of a one-bounded elementary net system $N$ is:

1. a labeled causal net $K$
2. in which each transition $t$ (with ${}^\bullet t$ and $t^\bullet$) is an action of $N$.

A distributed run $K$ of $N$ is complete iff (the marking) ${}^\circ K$ represents the initial marking of $N$ and (the marking) $K^\circ$ does not enable any transition.

Examples on the black board.

Today: a characterization of distributed runs using homomorphisms.

# Net homomorphisms

[2] Here $h(X)$ for set $X$ of nodes is defined by $h(X) = \bigcup_{x \in X} h(x)$.
[3] Due to the 1-boundedness, a marking $M$ is a subset of the set $P$ of places.

# Net homomorphisms

## Homomorphism

A homomorphism from $N_1 = (P_1, T_1, F_1, M_{0,1})$ to $N_2 = (P_2, T_2, F_2, M_{0,2})$ is a mapping $h : P_1 \cup T_1 \to P_2 \cup T_2$ such that:

[2] Here $h(X)$ for set $X$ of nodes is defined by $h(X) = \bigcup_{x \in X} h(x)$.

[3] Due to the 1-boundedness, a marking $M$ is a subset of the set $P$ of places.

# Net homomorphisms

## Homomorphism

A homomorphism from $N_1 = (P_1, T_1, F_1, M_{0,1})$ to $N_2 = (P_2, T_2, F_2, M_{0,2})$ is a mapping $h : P_1 \cup T_1 \rightarrow P_2 \cup T_2$ such that: [2]

1. $h(P_1) \subseteq P_2$ and $h(T_1) \subseteq T_2$, and

[2] Here $h(X)$ for set $X$ of nodes is defined by $h(X) = \bigcup_{x \in X} h(x)$.

[3] Due to the 1-boundedness, a marking $M$ is a subset of the set $P$ of places.

# Net homomorphisms

## Homomorphism

A homomorphism from $N_1 = (P_1, T_1, F_1, M_{0,1})$ to $N_2 = (P_2, T_2, F_2, M_{0,2})$ is a mapping $h : P_1 \cup T_1 \to P_2 \cup T_2$ such that: [2]

1. $h(P_1) \subseteq P_2$ and $h(T_1) \subseteq T_2$, and
2. $\forall t \in T_1$, the restriction of $h$ to ${}^\bullet t$ is a bijection between ${}^\bullet t$ (in $N_1$) and ${}^\bullet h(t)$ (in $N_2$), and similarly for $t^\bullet$ and $h(t)^\bullet$, and

[2] Here $h(X)$ for set $X$ of nodes is defined by $h(X) = \bigcup_{x \in X} h(x)$.

[3] Due to the 1-boundedness, a marking $M$ is a subset of the set $P$ of places.

# Net homomorphisms

## Homomorphism

A homomorphism from $N_1 = (P_1, T_1, F_1, M_{0,1})$ to $N_2 = (P_2, T_2, F_2, M_{0,2})$ is a mapping $h : P_1 \cup T_1 \rightarrow P_2 \cup T_2$ such that: [2]

1. $h(P_1) \subseteq P_2$ and $h(T_1) \subseteq T_2$, and
2. $\forall t \in T_1$, the restriction of $h$ to ${}^\bullet t$ is a bijection between ${}^\bullet t$ (in $N_1$) and ${}^\bullet h(t)$ (in $N_2$), and similarly for $t^\bullet$ and $h(t)^\bullet$, and
3. the restriction of $h$ to $M_{0,1}$ is a bijection between $M_{0,1}$ and $M_{0,2}$.[3]

[2] Here $h(X)$ for set $X$ of nodes is defined by $h(X) = \bigcup_{x \in X} h(x)$.

[3] Due to the 1-boundedness, a marking $M$ is a subset of the set $P$ of places.

# Net homomorphisms

## Homomorphism

A homomorphism from $N_1 = (P_1, T_1, F_1, M_{0,1})$ to $N_2 = (P_2, T_2, F_2, M_{0,2})$ is a mapping $h : P_1 \cup T_1 \to P_2 \cup T_2$ such that: [2]

1. $h(P_1) \subseteq P_2$ and $h(T_1) \subseteq T_2$, and
2. $\forall t \in T_1$, the restriction of $h$ to ${}^\bullet t$ is a bijection between ${}^\bullet t$ (in $N_1$) and ${}^\bullet h(t)$ (in $N_2$), and similarly for $t^\bullet$ and $h(t)^\bullet$, and
3. the restriction of $h$ to $M_{0,1}$ is a bijection between $M_{0,1}$ and $M_{0,2}$.[3]

## Intuition

A homomorphism is a mapping between nets that preserves the nature of nodes and the environment of nodes. A homomorphism from $N_1$ to $N_2$ means that $N_1$ can be folded onto a part of $N_2$, or in other words, that $N_1$ can be obtained by partially unfolding a part of $N_2$.

[2] Here $h(X)$ for set $X$ of nodes is defined by $h(X) = \bigcup_{x \in X} h(x)$.

[3] Due to the 1-boundedness, a marking $M$ is a subset of the set $P$ of places.

# Distributed run using homomorphisms

**Distributed run** **[Best and Fernandez, 1988]**

A distributed run of an elementary net system $N$ is a pair $(K, h)$ where $K$ is a causal net and $h$ is a homomorphism from $K$ to $N$.[4]

[4] Best and Fernandez called this a process of a net.

[5] In the previous lecture, the labeling $h$ was explicitly given as $\ell$.

# Distributed run using homomorphisms

**Distributed run** **[Best and Fernandez, 1988]**

A distributed run of an elementary net system $N$ is a pair $(K, h)$ where $K$ is a causal net and $h$ is a homomorphism from $K$ to $N$.[4]

**Intuition**

A distributed run $(K, h)$ of $N$ may be viewed as a net $K$ of which the places and transitions are labeled by places and transitions of $N$

[4] Best and Fernandez called this a process of a net.

[5] In the previous lecture, the labeling $h$ was explicitly given as $\ell$.

# Distributed run using homomorphisms

## Distributed run [Best and Fernandez, 1988]

A distributed run of an elementary net system $N$ is a pair $(K, h)$ where $K$ is a causal net and $h$ is a homomorphism from $K$ to $N$.[4]

## Intuition

A distributed run $(K, h)$ of $N$ may be viewed as a net $K$ of which the places and transitions are labeled by places and transitions of $N$ such that the labeling $h$ forms a net homomorphism from $K$ to $N$.[5]

[4] Best and Fernandez called this a process of a net.

[5] In the previous lecture, the labeling $h$ was explicitly given as $\ell$.

# Examples

# Overview

# Conflicts

A distributed run is based on a causal net. A branching process on an occurrence net. Main difference: the presence of conflicts (aka: choices).

# Conflicts

A distributed run is based on a causal net. A branching process on an occurrence net. Main difference: the presence of conflicts (aka: choices).

## Conflict

Let $N = (P, T, F, M)$ be a net. Nodes $x_1$ and $x_2$ are in conflict, denoted $x_1 \# x_2$, if there exist distinct transitions $t_1, t_2 \in T$ such that $^\bullet t_1 \cap pre t_2 \neq \varnothing$ and $(t_1, x_1) \in F^*$ and $(t_2, x_2) \in F^*$.

Node $x$ is in self-conflict whenever $x \# x$.

# Conflicts

A distributed run is based on a causal net. A branching process on an occurrence net. Main difference: the presence of conflicts (aka: choices).

## Conflict

Let $N = (P, T, F, M)$ be a net. Nodes $x_1$ and $x_2$ are in conflict, denoted $x_1 \# x_2$, if there exist distinct transitions $t_1, t_2 \in T$ such that $^\bullet t_1 \cap \mathit{pre} t_2 \neq \varnothing$ and $(t_1, x_1) \in F^*$ and $(t_2, x_2) \in F^*$.

Node $x$ is in self-conflict whenever $x \# x$.

## Examples

On the black board.

# Conflicts

A distributed run is based on a causal net. A branching process on an occurrence net. Main difference: the presence of conflicts (aka: choices).

## Conflict

Let $N = (P, T, F, M)$ be a net. Nodes $x_1$ and $x_2$ are in conflict, denoted $x_1 \# x_2$, if there exist distinct transitions $t_1, t_2 \in T$ such that $^\bullet t_1 \cap \, pret_2 \neq \varnothing$ and $(t_1, x_1) \in F^*$ and $(t_2, x_2) \in F^*$.

Node $x$ is in self-conflict whenever $x \# x$.

## Examples

On the black board.

Note that in a causal net $\# = \varnothing$ as $^\bullet t_1 \cap {}^\bullet t_2 = \varnothing$ for any two distinct transitions $t_1$ and $t_2$.

# Occurrence net

## Occurrence net

A net $K = (Q, V, G, M)$ is an occurrence net iff:

1. for each $q \in Q$, $|{}^\bullet q| \leqslant 1$
2. the transitive closure $G^+$ of $G$ is irreflexive
3. for each node $x \in Q \cup V$ we have $\{ y \mid (y, x) \in G^+ \}$ is finite
4. no transition $v \in V$ is in self-conflict
5. $M_0 = {}^\circ K \ = \ \{ q \in Q \mid \ {}^\bullet q = \varnothing \}$.

# Occurrence net

## Occurrence net

A net $K = (Q, V, G, M)$ is an occurrence net iff:

1. for each $q \in Q$, $|{}^\bullet q| \leqslant 1$
2. the transitive closure $G^+$ of $G$ is irreflexive
3. for each node $x \in Q \cup V$ we have $\{\, y \mid (y, x) \in G^+ \,\}$ is finite
4. no transition $v \in V$ is in self-conflict
5. $M_0 = {}^\circ K \; = \; \{\, q \in Q \mid \; {}^\bullet q = \varnothing \,\}$.

## Remark

Since $\# = \varnothing$ in a causal net, and each causal net fulfils the remaining conditions, every causal net is an occurrence net.

# Example

# Branching process

**Branching process** **[Engelfriet 1991]**

A branching process of net $N$ is a pair $(K, h)$ where $K = (Q, V, G, M)$ is an occurrence net and $h$ a net homomorphism from $K$ to $N$ such that:

$$\forall v, v' \in Q. \ ({}^\bullet v = {}^\bullet v' \text{ and } h(v) = h(v') \quad \text{implies} \quad v = v') \, .$$

# Branching process

**Branching process** **[Engelfriet 1991]**

A branching process of net $N$ is a pair $(K, h)$ where $K = (Q, V, G, M)$ is an occurrence net and $h$ a net homomorphism from $K$ to $N$ such that:

$$\forall v, v' \in Q.\ ({}^\bullet v = {}^\bullet v' \text{ and } h(v) = h(v') \quad \text{implies} \quad v = v')\,.$$

Every distributed run is a branching process. The reverse is not true.

# Branching process

**Branching process** **[Engelfriet 1991]**

A branching process of net $N$ is a pair $(K, h)$ where $K = (Q, V, G, M)$ is an occurrence net and $h$ a net homomorphism from $K$ to $N$ such that:

$$\forall v, v' \in Q.\ ({}^\bullet v = {}^\bullet v' \text{ and } h(v) = h(v') \quad \text{implies} \quad v = v')\,.$$

Every distributed run is a branching process. The reverse is not true.

**Examples**

On the black board.

# Examples

# Properties of branching processes

Let $K$ be a branching process of net $N$. Then:

1. A place of $K$ can get marked at most once, and an event (aka: transition) of $K$ can occur at most once in any step sequence of $K$
2. For $Q' \subseteq Q$: $K$ has some reachable marking $M$ with $Q' \subseteq M$ if and only if all places in $Q'$ are pairwise concurrent.

Nodes $x, y$ are concurrent if neither $(x, y) \in G^+$, nor $(y, x) \in G^+$ nor $x \# y$.

# Overview

# Relating branching processes

## Homomorphisms and isomorphisms between branching processes

Let $B_1 = (K_1, h_1)$ and $B_2 = (K_2, h_2)$ be two branching processes of net $N$. A homomorphism from $B_1$ to $B_2$ is a homomorphism $h$ from $K_1$ to $K_2$ such that $h_2 \circ h = h_1$.[6]

[6] The composition of two net homomorphisms is a net homomorphism.

# Relating branching processes

## Homomorphisms and isomorphisms between branching processes

Let $B_1 = (K_1, h_1)$ and $B_2 = (K_2, h_2)$ be two branching processes of net $N$. A homomorphism from $B_1$ to $B_2$ is a homomorphism $h$ from $K_1$ to $K_2$ such that $h_2 \circ h = h_1$.[6]

An isomorphism is a bijective homomorphism. $B_1$ and $B_2$ are isomorphic if there is an isomorphism from $B_1$ to $B_2$.

[6] The composition of two net homomorphisms is a net homomorphism.

# Relating branching processes

## Homomorphisms and isomorphisms between branching processes

Let $B_1 = (K_1, h_1)$ and $B_2 = (K_2, h_2)$ be two branching processes of net $N$. A homomorphism from $B_1$ to $B_2$ is a homomorphism $h$ from $K_1$ to $K_2$ such that $h_2 \circ h = h_1$.[6]

An isomorphism is a bijective homomorphism. $B_1$ and $B_2$ are isomorphic if there is an isomorphism from $B_1$ to $B_2$.

Being isomorphic is an equivalence relation. Its equivalence classes are called isomorphism classes.

[6] The composition of two net homomorphisms is a net homomorphism.

# Approximation of branching processes

## Approximation

Let $B_1$ and $B_2$ be two branching processes of net $N$. $B_1$ approximates $B_2$, denoted $B_1 \sqsubseteq B_2$, if there exists an injective homomorphism from $B_1$ to $B_2$.

# Approximation of branching processes

## Approximation

Let $B_1$ and $B_2$ be two branching processes of net $N$. $B_1$ approximates $B_2$, denoted $B_1 \sqsubseteq B_2$, if there exists an injective homomorphism from $B_1$ to $B_2$.

## Intuition

$B_1$ approximates $B_2$ whenever every (partial) distributed run in $B_1$ is also contained in $B_2$. In other words, $B_1$ is isomorphic to an initial part of $B_2$. Being an approximation on branching processes is the analogue of being a prefix on sequences.

# Approximation of branching processes

## Approximation

Let $B_1$ and $B_2$ be two branching processes of net $N$. $B_1$ approximates $B_2$, denoted $B_1 \sqsubseteq B_2$, if there exists an injective homomorphism from $B_1$ to $B_2$.

## Intuition

$B_1$ approximates $B_2$ whenever every (partial) distributed run in $B_1$ is also contained in $B_2$. In other words, $B_1$ is isomorphic to an initial part of $B_2$. Being an approximation on branching processes is the analogue of being a prefix on sequences.

## Examples

On the black board. Obviously, $\sqsubseteq$ is a partial order on branching processes.

# Examples

# Approximation of branching processes

# Approximation of branching processes

## Approximation

Let $B_1$ and $B_2$ be two branching processes of net $N$. $B_1$ approximates $B_2$, denoted $B_1 \sqsubseteq B_2$, if there exists an injective homomorphism from $B_1$ to $B_2$.

# Approximation of branching processes

## Approximation

Let $B_1$ and $B_2$ be two branching processes of net $N$. $B_1$ approximates $B_2$, denoted $B_1 \sqsubseteq B_2$, if there exists an injective homomorphism from $B_1$ to $B_2$.

## Lemma

Approximation is preserved by isomorphism: if $B_i'$ is isomorphic to $B_i$ (for $i = 1, 2$), then $B_1 \sqsubseteq B_2$ implies $B_1' \sqsubseteq B_2'$. Thus, $\sqsubseteq$ can be extended to a partial order on isomorphism classes (of branching processes).

# Approximation of branching processes

## Approximation

Let $B_1$ and $B_2$ be two branching processes of net $N$. $B_1$ approximates $B_2$, denoted $B_1 \sqsubseteq B_2$, if there exists an injective homomorphism from $B_1$ to $B_2$.

## Lemma

Approximation is preserved by isomorphism: if $B_i'$ is isomorphic to $B_i$ (for $i = 1, 2$), then $B_1 \sqsubseteq B_2$ implies $B_1' \sqsubseteq B_2'$. Thus, $\sqsubseteq$ can be extended to a partial order on isomorphism classes (of branching processes).

## Proof.

Home exercise. Basically juggling with homomorphisms. $\square$

# Engelfriet's theorem

# Engelfriet's theorem

## Engelfriet's branching process theorem

The set of isomorphism classes of branching processes of net $N$ is a complete lattice with respect to the approximation relation $\sqsubseteq$. Formally, $(\mathbb{B}, \sqsubseteq)$ is a complete partial order, where $\mathbb{B}$ is the set of isomorphism classes of branching processes.

# Engelfriet's theorem

## Engelfriet's branching process theorem

The set of isomorphism classes of branching processes of net $N$ is a complete lattice with respect to the approximation relation $\sqsubseteq$. Formally, $(\mathbb{B}, \sqsubseteq)$ is a complete partial order, where $\mathbb{B}$ is the set of isomorphism classes of branching processes.

## Complete lattice

Recall that a complete lattice is a partial order $(\mathbb{B}, \sqsubseteq)$ such that all subsets of $\mathbb{B}$ have LUBs and GLBs.

# The true concurrency semantics of a net

## Corollary: the unfolding of a net

Every one-bounded net has a unique maximal (with respect to $\sqsubseteq$) branching process up to isomorphism. This is called the unfolding or true concurrency semantics of net $N$.

# The true concurrency semantics of a net

**Corollary: the unfolding of a net**

Every one-bounded net has a unique maximal (with respect to $\sqsubseteq$) branching process up to isomorphism. This is called the unfolding or true concurrency semantics of net $N$.

We denote by $B_{\text{max}} = ((P_{\text{max}}, T_{\text{max}}, F_{\text{max}}), h_{\text{max}})$ a representative of the isomorphism class of the maximal branching process of $N$.

# The true concurrency semantics of a net

## Corollary: the unfolding of a net

Every one-bounded net has a unique maximal (with respect to $\sqsubseteq$) branching process up to isomorphism. This is called the unfolding or true concurrency semantics of net $N$.

We denote by $B_{\text{max}} = ((P_{\text{max}}, T_{\text{max}}, F_{\text{max}}), h_{\text{max}})$ a representative of the isomorphism class of the maximal branching process of $N$.

## Example

On the black board.

# The true concurrency semantics of Petri nets

The true concurrency semantics of a Petri net is given by its unfolding.

Recall: The interleaving semantics of a Petri net is given by its marking graph.

# Overview

# Finite prefix

The maximal branching process under $\sqsubseteq$ may be infinite.

# Finite prefix

The maximal branching process under $\sqsubseteq$ may be infinite.

**Prefix of maximal branching process**

Branching process $B = (P, T, F, M_0)$ is a prefix of $B_{\max}$ if $B \sqsubseteq B_{\max}$ and $P \subseteq P_{\max}$ and $T \subseteq T_{\max}$. $B$ is finite whenever $P$ and $T$ are finite.

# Finite prefix

The maximal branching process under $\sqsubseteq$ may be infinite.

**Prefix of maximal branching process**

Branching process $B = (P, T, F, M_0)$ is a prefix of $B_{\max}$ if $B \sqsubseteq B_{\max}$ and $P \subseteq P_{\max}$ and $T \subseteq T_{\max}$. $B$ is finite whenever $P$ and $T$ are finite.

**Finite prefix existence theorem** **[McMillan, 1992]**

For every finite one-bounded net $N$, there exists a finite prefix $B_{\text{fin}}$ of $B_{\max}$ that covers all reachable markings of $N$. The size of the finite prefix can maximally be exponential in the size of $N$.

# Finite prefix

The maximal branching process under $\sqsubseteq$ may be infinite.

## Prefix of maximal branching process

Branching process $B = (P, T, F, M_0)$ is a prefix of $B_{\max}$ if $B \sqsubseteq B_{\max}$ and $P \subseteq P_{\max}$ and $T \subseteq T_{\max}$. $B$ is finite whenever $P$ and $T$ are finite.

## Finite prefix existence theorem [McMillan, 1992]

For every finite one-bounded net $N$, there exists a finite prefix $B_{\text{fin}}$ of $B_{\max}$ that covers all reachable markings of $N$. The size of the finite prefix can maximally be exponential in the size of $N$.

## Proof.

Follows directly from two facts:

1. Every reachable marking is represented by some cut of $B_{\max}$, and
2. The set of reachable markings of a finite one-bounded net is finite.

□

# Configurations

# Configurations

## Configurations

Let $K = (Q, V, G, M_0)$ be an occurrence net, $\prec = G^+$ and $\preceq = G^*$.

# Configurations

## Configurations

Let $K = (Q, V, G, M_0)$ be an occurrence net, $\prec = G^+$ and $\preceq = G^*$.
The set $C \subseteq V$ is a configuration of $K$ whenever:

1. $x \in C$ implies $y \in C$, for all $y \preceq x$ (downward-closed wrt. $\preceq$)
2. $\forall x, y \in C. \neg(x \# y)$ (conflict-free)

# Configurations

## Configurations

Let $K = (Q, V, G, M_0)$ be an occurrence net, $\prec = G^+$ and $\preceq = G^*$.
The set $C \subseteq V$ is a configuration of $K$ whenever:

1. $x \in C$ implies $y \in C$, for all $y \preceq x$ (downward-closed wrt. $\preceq$)
2. $\forall x, y \in C. \neg(x \# y)$ (conflict-free)

## Intuition and examples

A configuration can be seen as the set of transitions that have occurred so far in a distributed run.

# Configurations

## Configurations

Let $K = (Q, V, G, M_0)$ be an occurrence net, $\prec = G^+$ and $\preceq = G^*$.
The set $C \subseteq V$ is a configuration of $K$ whenever:

1. $x \in C$ implies $y \in C$, for all $y \preceq x$ (downward-closed wrt. $\preceq$)
2. $\forall x, y \in C. \neg(x \# y)$ (conflict-free)

## Intuition and examples

A configuration can be seen as the set of transitions that have occurred so far in a distributed run. Examples on the black board.

# Configurations

## Configurations

Let $K = (Q, V, G, M_0)$ be an occurrence net, $\prec = G^+$ and $\preceq = G^*$.
The set $C \subseteq V$ is a configuration of $K$ whenever:

1. $x \in C$ implies $y \in C$, for all $y \preceq x$ (downward-closed wrt. $\preceq$)
2. $\forall x, y \in C. \neg(x \# y)$ (conflict-free)

## Intuition and examples

A configuration can be seen as the set of transitions that have occurred so far in a distributed run. Examples on the black board.

## Fact

For configuration $C$ of $B_{\max}$ (of net $N$), and $x_1 \ldots x_n$ a linearisation of the transitions in $C$ (respecting $\preceq$), the sequence $h_{\max}(x_1) \ldots h_{\max}(t_n)$ is a sequential run of the original net $N$.

# Cuts

# Cuts

## Cuts

Let $C$ be a finite configuration of a branching process $B = (K, h)$.

# Cuts

## Cuts

Let $C$ be a finite configuration of a branching process $B = (K, h)$. Then:

$$Cut(C) \;=\; ({}^{\circ}K \cup C^{\bullet}) \setminus {}^{\bullet}C.$$

# Cuts

## Cuts

Let $C$ be a finite configuration of a branching process $B = (K, h)$. Then:

$$Cut(C) \; = \; ({}^{\circ}K \cup C^{\bullet}) \setminus {}^{\bullet}C.$$

If $B$ is a branching process of $N$, then $h(Cut(C))$ is a reachable marking of net $N$. We denote $h(Cut(C))$ by $M(C)$, the marking of configuration $C$.

# Cuts

## Cuts

Let $C$ be a finite configuration of a branching process $B = (K, h)$. Then:

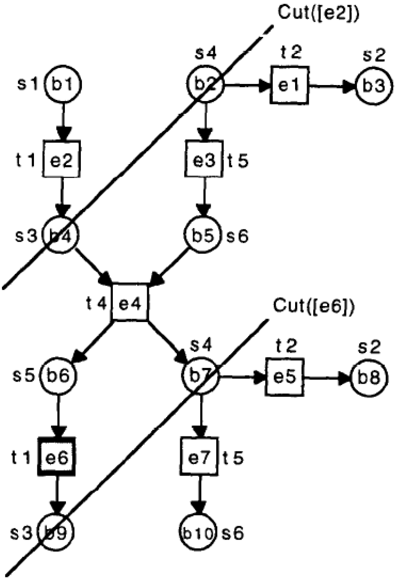
$$Cut(C) \;=\; ({}^{\circ}K \cup C^{\bullet}) \setminus {}^{\bullet}C.$$

If $B$ is a branching process of $N$, then $h(Cut(C))$ is a reachable marking of net $N$. We denote $h(Cut(C))$ by $M(C)$, the marking of configuration $C$.

## Intuition

Cuts correspond to markings reached by firing all transitions in a given finite configuration.

# Example


Cut([e2])
s4
t2
s2
s1
b1
b2
e1
b3
t1
e2
e3
t5
s3
b4
b5
s6
t4
e4
Cut([e6])
s4
t2
s2
s5
b6
b7
e5
b8
t1
e6
e7
t5
s3
b9
b10
s6

# Transition causes

## Transition causes

Let $K = (Q, V, G)$ be an occurrence net and $v \in V$. The set $[v]$ of causes of $v$ is defined by:

$$[v] \;=\; \{\, v' \in V \mid v' \preceq v \,\}.$$

(Recall that $\preceq$ denotes $G^*$, the reflexive and transitive closure of $G$.)

# Transition causes

## Transition causes

Let $K = (Q, V, G)$ be an occurrence net and $v \in V$. The set $[v]$ of causes of $v$ is defined by:

$$[v] \; = \; \{\, v' \in V \mid v' \preceq v \,\}.$$

(Recall that $\preceq$ denotes $G^*$, the reflexive and transitive closure of $G$.)

## Example

On the black board

# Transition causes

## Transition causes

Let $K = (Q, V, G)$ be an occurrence net and $v \in V$. The set $[v]$ of causes of $v$ is defined by:

$$[v] \;=\; \{\, v' \in V \mid v' \preceq v \,\}.$$

(Recall that $\preceq$ denotes $G^*$, the reflexive and transitive closure of $G$.)

## Example

On the black board

## Facts

1. For each $v$, $[v]$ is a finite configuration.
2. For every configuration $C$ of $K$, either $v \notin C$ or $[v] \subseteq C$.

# Cut-off event

# Cut-off event

## Cut-off event

Let $B_{\max} = ((P_{\max}, T_{\max}, G_{\max}), h_{\max})$. Transition $t \in T_{\max}$ is a cut-off transition if there exists a transition $t' \in T_{\max} \cup \{\perp\}$ such that:

$$\big|[t']\big| < \big|[t]\big| \text{ and } M([t]) = M([t']).$$

# Cut-off event

## Cut-off event

Let $B_{\max} = ((P_{\max}, T_{\max}, G_{\max}), h_{\max})$. Transition $t \in T_{\max}$ is a cut-off transition if there exists a transition $t' \in T_{\max} \cup \{ \perp \}$ such that:

$$\big|[t']\big| < \big|[t]\big| \text{ and } M([t]) = M([t']).$$

## Dummy transition

Remark: $\perp$ is a dummy transition having no input places and ${}^\circ B_{\max}$ as output places, for which we let $[\perp] = \varnothing$. This yields that if $M([t]) = M_0$, then $t$ is a cut-off transition.

# Cut-off event

## Cut-off event

Let $B_{\max} = ((P_{\max}, T_{\max}, G_{\max}), h_{\max})$. Transition $t \in T_{\max}$ is a cut-off transition if there exists a transition $t' \in T_{\max} \cup \{ \perp \}$ such that:

$$\big|[t']\big| < \big|[t]\big| \text{ and } M([t]) = M([t']).$$

## Dummy transition

Remark: $\perp$ is a dummy transition having no input places and $^\circ B_{\max}$ as output places, for which we let $[\perp] = \varnothing$. This yields that if $M([t]) = M_0$, then $t$ is a cut-off transition.

## Fact

If $\big|[t']\big| < \big|[t]\big|$ and $M([t]) = M([t'])$, then the "continuations" of $B_{\max}$ from $Cut([t])$ and $Cut([t'])$ are isomorphic.

# McMillan prefix

# McMillan prefix

## McMillan prefix

The McMillan prefix of one-bounded net $N$ is the branching process $B_{\text{fin}}$, the unique prefix of $B_{\text{max}}$ having $T_{\text{fin}}$ as set of transitions satisfying for each $t \in T_{\text{max}}$::

$$t \in T_{\text{fin}} \quad \text{iff} \quad \text{no transition } t' \prec t \text{ is a cut-off transition.}$$

# Computing the McMillan prefix

## Algorithm

1. Start with the empty branching process

# Computing the McMillan prefix

## Algorithm

1. Start with the empty branching process
2. Add transitions one at the time, in order of increasing size of their sets of causes

# Computing the McMillan prefix

## Algorithm

1. Start with the empty branching process
2. Add transitions one at the time, in order of increasing size of their sets of causes
3. On adding $t$, compare $M([t])$ with $M([t'])$ for each $t'$ that was added before $t$

# Computing the McMillan prefix

## Algorithm

1. Start with the empty branching process
2. Add transitions one at the time, in order of increasing size of their sets of causes
3. On adding $t$, compare $M([t])$ with $M([t'])$ for each $t'$ that was added before $t$
4. If $M([t]) = M([t'])$, then $t$ is a cut-off transition, and its successors are not explored

# Computing the McMillan prefix

## Algorithm

1. Start with the empty branching process
2. Add transitions one at the time, in order of increasing size of their sets of causes
3. On adding $t$, compare $M([t])$ with $M([t'])$ for each $t'$ that was added before $t$
4. If $M([t]) = M([t'])$, then $t$ is a cut-off transition, and its successors are not explored
5. Terminate when no further transitions can be added.

# Computing the McMillan prefix

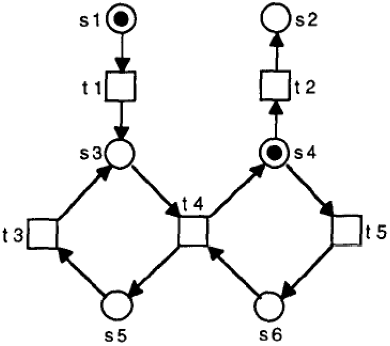
## Algorithm

1. Start with the empty branching process
2. Add transitions one at the time, in order of increasing size of their sets of causes
3. On adding $t$, compare $M([t])$ with $M([t'])$ for each $t'$ that was added before $t$
4. If $M([t]) = M([t'])$, then $t$ is a cut-off transition, and its successors are not explored
5. Terminate when no further transitions can be added.

## Remark

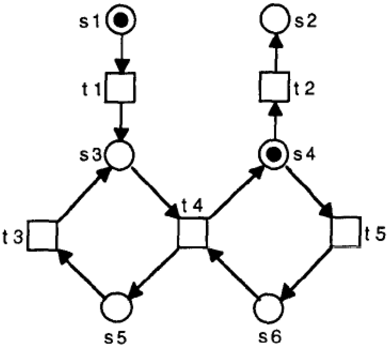
Termination is ensured by the finiteness of the number of reachable markings on $N$, as $N$ is one-bounded.

# Example net and one of its branching processes

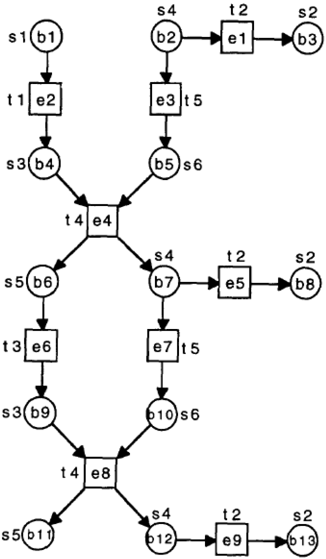



A sample one-bounded elementary system net

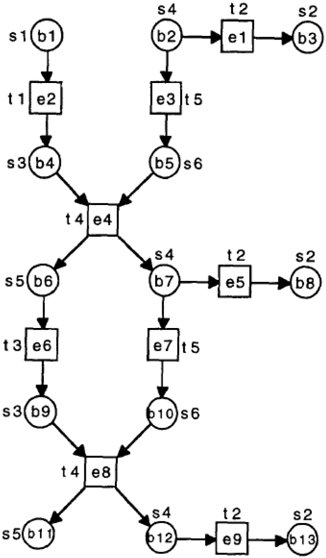
# Example net and one of its branching processes



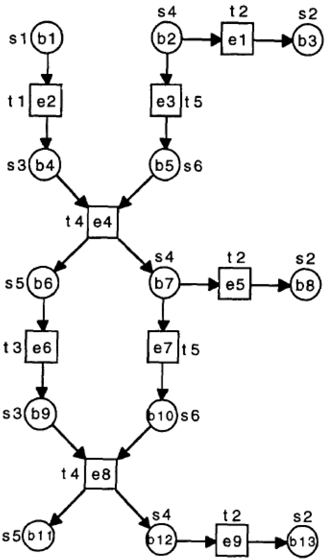

A sample one-bounded elementary system net

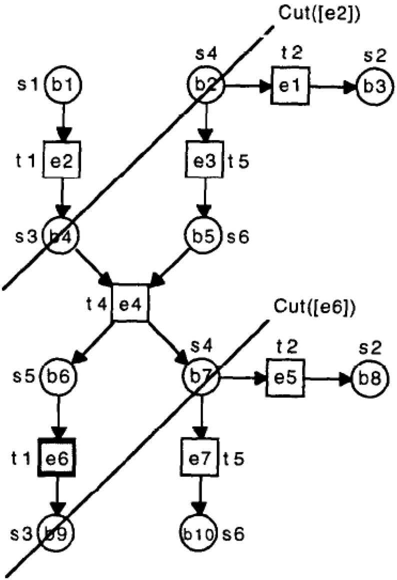


# Its McMillan prefix

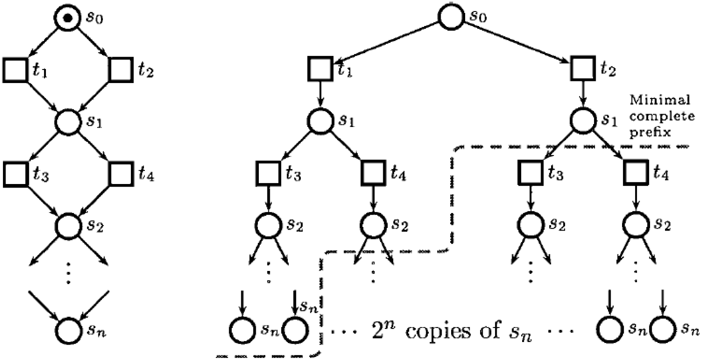


# Its McMillan prefix



# An exponentially-sized McMillan prefix




For every marking $M$ all the configurations $[t]$ satisfying $M([t]) = M$ have the same size, and therefore there exist no cut-off events [Kishinevsky and Taubin]

# Overview

# Summary

- A branching process captures several distributed runs of $N$

# Summary

- A branching process captures several distributed runs of $N$
- It is represented by a relaxed notion of causal net, the occurrence net

# Summary

- A branching process captures several distributed runs of $N$
- It is represented by a relaxed notion of causal net, the occurrence net
- Branching processes are mapped to $N$ via homomorphisms

# Summary

- A branching process captures several distributed runs of $N$
- It is represented by a relaxed notion of causal net, the occurrence net
- Branching processes are mapped to $N$ via homomorphisms
- A homomorphism is a structure-preserving mapping between two nets

# Summary

- A branching process captures several distributed runs of $N$
- It is represented by a relaxed notion of causal net, the occurrence net
- Branching processes are mapped to $N$ via homomorphisms
- A homomorphism is a structure-preserving mapping between two nets
- Approximation (denoted $\sqsubseteq$) is a partial-order on branching processes

# Summary

- A branching process captures several distributed runs of $N$
- It is represented by a relaxed notion of causal net, the occurrence net
- Branching processes are mapped to $N$ via homomorphisms
- A homomorphism is a structure-preserving mapping between two nets
- Approximation (denoted $\sqsubseteq$) is a partial-order on branching processes
- Isomorphic branching process with $\sqsubseteq$ are a complete lattice

# Summary

- A branching process captures several distributed runs of $N$
- It is represented by a relaxed notion of causal net, the occurrence net
- Branching processes are mapped to $N$ via homomorphisms
- A homomorphism is a structure-preserving mapping between two nets
- Approximation (denoted $\sqsubseteq$) is a partial-order on branching processes
- Isomorphic branching process with $\sqsubseteq$ are a complete lattice
- True concurrency semantics of $N$ = the maximal element (under $\sqsubseteq$)

# Summary

- A branching process captures several distributed runs of $N$
- It is represented by a relaxed notion of causal net, the occurrence net
- Branching processes are mapped to $N$ via homomorphisms
- A homomorphism is a structure-preserving mapping between two nets
- Approximation (denoted $\sqsubseteq$) is a partial-order on branching processes
- Isomorphic branching process with $\sqsubseteq$ are a complete lattice
- True concurrency semantics of $N$ = the maximal element (under $\sqsubseteq$)
- For 1-bounded nets, the McMillan prefix covers all reachable markings