Constructing a finite and complete prefix

Branching process (O,h) of N is complete, if it contains "as much information as $Unf(N)$".

Definition (Complete prefix):

Let $N=(S,T,W,M_0)$ and (O,h) one of its branching processes.

We say (O,h) is <u>complete</u> (a complete prefix of $Unf(N)$) if for all $M \in R(N)$ there is $C \in C_{fin}(O,h)$:

- $Mark(C) = M$ // M represented in (O,h)
- For all $t \in T$ with $M[t\rangle$ there is $\oplus e \in C_{fin}(O,h)$ with $h(e)=t$ // t represented in (O,h).

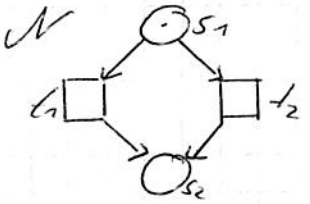
Observation:

Unfolding can be reconstructed from complete prefix
↳ ALL markings + their firings present.

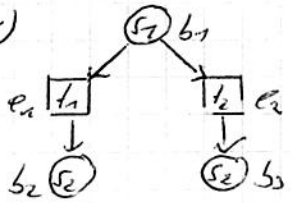
Does not hold if only reachable markings preserved.

Example:

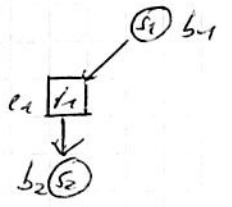
N



$Unf(N)$



complete prefix



prefix, not complete
$M[t_2\rangle$ missing.

In the following:

- N has finitely many markings (safe)
  ↳ $Unf(N)$ contains a complete prefix that is <u>finite</u>
- Compute it (by variant of algorithm above)
  ↳ identify events at which computation can be stopped without loosing information (cut-off events).

Part of branching process lying "behind a configuration"

Definition:

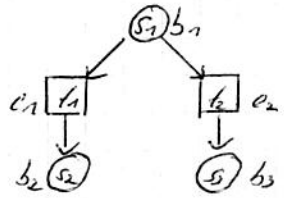
Let $C \in Pim(O,h)$ with $O = (B, E, G)$.

Then ΠC is the pair (O', h') where O' is the subnet of O with

- $B' \cup E' = \{x \in B \cup E \mid x \notin C \cup {}^\bullet C \text{ and } \neg x \# y \text{ f.a. } y \in C\}$
- $h' = h|_{O'}$

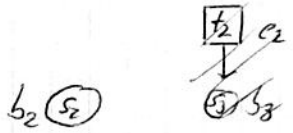
Example:

(O,h)



$C = \{e_1\}$

$\Pi C =$



conflict.

Lemma:

Let (O,h) be a branching process of (N, M_0).

Let $C \in Pim(O,h)$. Then ΠC is a branching process of $(N, Mark(C))$.

If $(O,h) = Unf(N, M_0)$, then $\Pi C =_{iso} Unf(N, Mark(C))$.

Some considerations:

Let C_1, C_2 be finite configurations with

$Mark(C_1) = M = Mark(C_2)$.

Then by lemma:

$\Pi C_1 =_{iso} Unf(N, M) =_{iso} \Pi C_2$.

By transitivity

$\Pi C_1 =_{iso} \Pi C_2$.

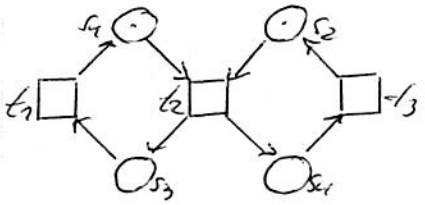
Let $I: B_1 \cup E_1 \to B_2 \cup E_2$ be the isomorphism

induces mapping from finite extensions of C_1

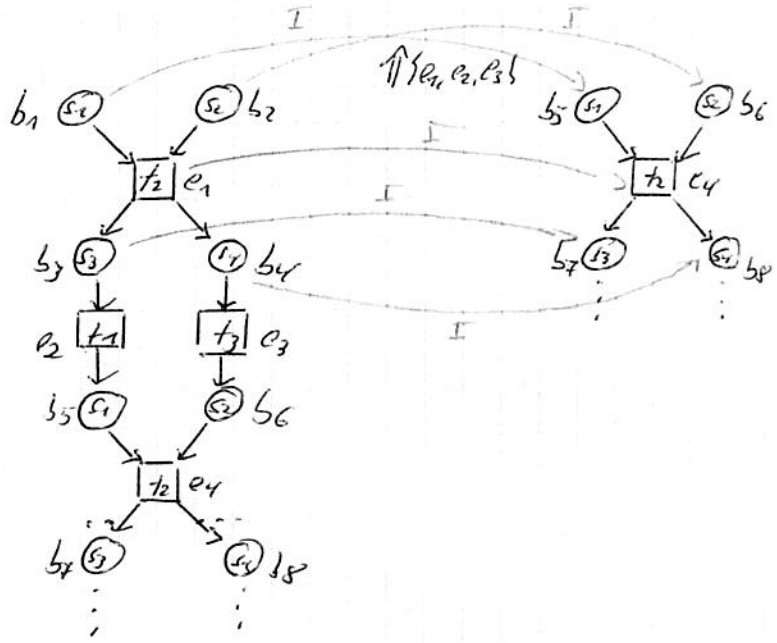
to finite extensions of C_2:

$C_1 \oplus E \mapsto C_2 \oplus I(E)$.

Example:



$\Uparrow \emptyset$



$\{e_1\} \longmapsto \overline{I}(\{e_1\}) = \{e_4\}$,

since

$\emptyset \oplus \{e_1\} \longmapsto \{e_1, e_2, e_3\} \oplus \{e_4\}$.

Idea (McMillan '92)

Attach to each event a reachable marking of the Petri net

Definition (Local configuration):

Consider a branching process (O, h) with $O = (B, E, G)$.

The Local configuration of event $e \in E$ is

$[e] := \{e' \in E \mid e' \leq e\}$.

Assume e' is added to a branching process so that

$Mark([e']) = Mark([e])$

for some event e added before.

By lemma: $\Uparrow [e] =_{iso} \Uparrow [e']$.

Intuitively: Sufficient to pursue construction of $[e]$, only.

↳ Mark $[e']$ as cut-off

Its not that easy.

Definition (Adequate order):

A strict partial order $\prec \subseteq \mathcal{C}_{fin}(Unf(N)) \times \mathcal{C}_{fin}(Unf(N))$ is called *adequate*, if

- $\prec$ is well-founded // no infinite decreasing sequence $C_1 \succ C_2 \succ C_3 \ldots$
- $C_1 \subset C_2$ implies $C_1 \prec C_2$ // $\prec$ refines $\subset$
- If $C_1 \prec C_2$ and $Mark(C_1) = Mark(C_2)$, then $C_1 \oplus E \prec C_2 \oplus I(E)$.
  f.e. finite extensions E of C_1 preserved by finite extensions.

Example (McMillan's adequate order):

$C \prec C'$: if $|C| < |C'|$ // less events

Definition (Cut-off event):

Let $(O,h) \in Unf(N)$ with $O = (B,E,G)$ and $e \in E$.
Let $\prec$ be an adequate order on the configurations of $Unf(N)$.
Event e' is a *cut-off event of* (O,h) *wrt.* $\prec$
if (O,h) contains a local configuration $[e]$ with
(i) $Mark([e]) = Mark([e'])$
(ii) $[e] \prec [e']$.

Family of algorithms parametrised by adequate order $\prec$
↳ Events added according to $\prec$
↳ Cut-offs identified and marked
↳ Terminates, when no more events can be added.

Procedure (ERV algorithm for finite and complete prefixes):

input: $N = (S,T,W,M_0)$ with $M_0 = \{s_1, \ldots, s_n\}$
$\prec$ adequate order on $\mathcal{C}_{fin}(Unf(N))$.

begin:

$Fin := \{(s_1, \emptyset), \ldots, (s_n, \emptyset)\}$.

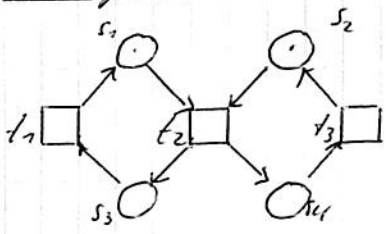
$pe := PE(Fin)$

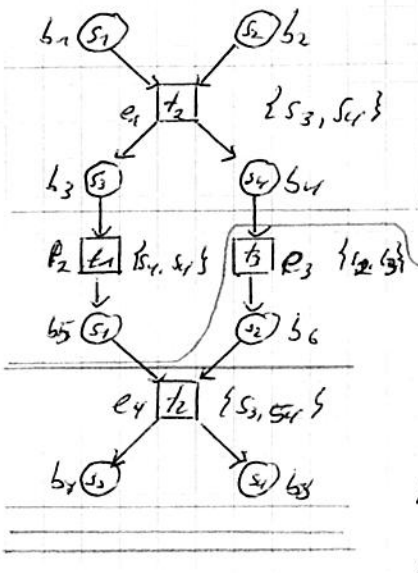
```
cut_off := ∅
while pe ≠ ∅ do
    let e = (t, X) ∈ pe with [e] ≤-minimal
    if [e] ∩ cut_off = ∅ then
        add to Fin event e
        add to Fin (s, e) f.a. s ∈ t•
        pe := PE(Fin).
        if e is cut-off of Fin then
            cut_off := cut_off ∪ {e}
        end if
    else
        pe := pe \ {e}
    end if
end while
```

end

output: Fin, a finite and complete prefix of $Unf(N)$.

Example:





$pe = \{(t_2, \{b_1, b_2\})\}$, $cut\_off = \emptyset$

$pe = \{(t_1, \{b_3\}), (t_3, \{b_4\})\}$, $cut\_off = \emptyset$

$pe = \{(t_3, \{b_4, s_1\})\}$, $cut\_off = \emptyset$

$pe = \{(t_2, \{b_5, b_6\})\}$, $cut\_off = \emptyset$

$pe = \{(t_1, \{b_7\}), (t_3, \{b_8\})\}$, $cut\_off = \{e_4\}$

$pe = \{(t_3, \{b_8\})\}$, $cut\_off = \{e_4\}$

$pe = \emptyset$, $cut\_off = \{e_4\}$.

Proposition

Fin is finite.

Proof:

Let $Fin = (O, h)$ with $O = (B, E, G)$. be the output with $N = (S, T, W, M_0)$ as input

The depth $d(e) \in \mathbb{N}$ of an event $e \in E$ is the length of the longest causal chain leading to e,

$$e_1 < e_2 < \ldots < e_{n-1} < e \Rightarrow d(e) = n$$

Argue in three steps:

1) $d(e) \leq |R(N)| + 1$

Cuts correspond to reachable markings.

Hence, every sequence

$$e_1 < e_2 < \ldots < e_{|R(N)|} < e_{|R(N)|+1}$$

of events in Fin contains $i < j$ with

$$Mark([e_i]) = Mark([e_j]).$$

Furthermore,

$[e_i] \subset [e_j]$ and thus $[e_i] \lneq [e_j]$.

Hence, $[e_j]$ is a cut-off event of Fin.

2) For every event $e \in E$, the sets $^\bullet e$ and $e^\bullet$ are finite.

By definition of h, there is a bijection between $^\bullet e$ and $^\bullet h(e)$ as well as $e^\bullet$ and $h(e)^\bullet$.

Finiteness follows from finiteness of N.

3) For all $k \in \mathbb{N}$, there are only finitely many $e \in E$ so that $d(e) \leq k$.

Proceed by induction on k.

$k=0$ $\quad E_0 := \{e \in E \mid d(e) \leq 0\} = \emptyset$.

$k \to k+1$: Assume $E_k = \{e \in E \mid d(e) \leq k\}$ is finite.

Consider E_{k+1}.

We have

$$E_{k+1} \subseteq E_k^{\bullet} \cup Min(O).$$

By the hypothesis, E_k is finite and by (7) $E_k^{\bullet}$ is finite as well.

Since by definition of h

$$h(Min(O)) = M_0,$$

$Min(O)$ is finite as well.

By non-redundancy of branching processes, E_{k+1} has to be finite.

By (1) + (3), Fin has finitely many events, by (2) finitely many conditions. □

Proposition

Fin is complete.

Proof:

Consider $N = (S, T, W, M_0)$, and $Fin = (O, h)$ with $O = (B, E, G)$.

(1) For all $M \in R(N)$ there is $C \in C_{fin}(Fin)$ with $M = Mark(C)$.

Consider $M \in R(N)$.

The unfolding and the Petri net have the same reachable markings. Hence, there is

$$C_1 \in C_{fin}(Unf(N))$$

with $M = Mark(C_1)$.

If $C_1 \notin C_{fin}(Fin)$, then there is a cut-off event e_1 with

$$C_1 = [e_1] \oplus E_1$$

By definition of cut-offs, there is a local configuration $[e_2]$ with $[e_2] < [e_1]$ and

$$Mark([e_1]) = Mark([e_2]).$$

Consider

$C_2 = [e_2] \oplus I(E_1)$.

Since $\prec$ preserved by finite extensions,

$C_2 \prec C_1$.

Moreover, $Mark(C_2) = Mark(C_1)$.

If $C_2 \in Cfg(F_{in})$, we iterate the procedure to find $C_3 \prec C_2 \prec C_1$.
This iteration eventually terminates as $\prec$ is well founded.
The last configuration is in $Cfg(F_{in})$.

2) Assume $M \in \mathcal{M}$.
We already know that is $C' \in Cfg(F_{in})$
with $Mark(C') = M$.
Hence, there is a $\prec$-minimal such configuration $C \in Cfg(F_{in})$
with $Mark(C) = M$.
Assume C contains a cut-off event.
By (1), we would find a $\prec$ smaller configuration C_{ne}
with $Mark(C_{ne}) = M$ and $C_{ne} \prec C$.
This contradicts minimality of C.
Hence, C contains no cut-off events, and

$C \otimes Ref$

is also a configuration of $Cfg(F_{in})$, where e labelled by t.

□

Note:

- If $\prec$ is an adequate and total order,
the number of non-cut-off events in F_{in} does not exceed $|RCM|$.
- There are total adequate orders for safe Petri nets.

SAT-based Verification:

Consider Petri net $N = (S, T, W, M_0)$.

Let (O, h) be finite and complete prefix with $O = (B, E, G)$.

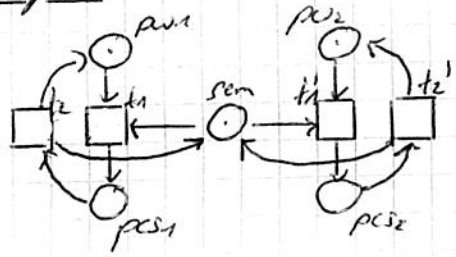
Interested in <u>safety properties</u> of N

↳ Violations expressible in terms of reachability (in N)

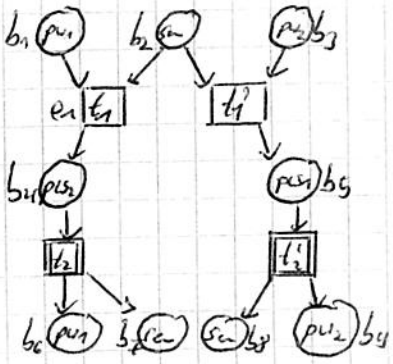
↳ Rephrase violations on configurations of (O, h).

<u>Example:</u>

N



(O, h)



Safety properties:

<u>Mutual exclusion:</u>

violated iff $M(pcs_1) = 1 = M(pcs_2)$ for some $M \in R(N)$.

iff $M(b_4) = 1 = M(b_5)$ for some $M \in R(O)$

<u>Approach:</u>

In a branching process, every $e \in E$ can be executed at most once

↳ Map events $e \in E$ to Boolean variables x_e.

Translate finite and complete prefix + safety property into Boolean satisfiability problem (SAT).

$C \wedge \neg \varphi$

that is

unsatisfiable iff property of interest holds.

Every satisfying assignment yields configuration C in (O, h) (and hence transition sequence in N) where final marking violates property.

Construction of $\mathcal{C}$:

Formula so that assignment to variables yields configuration

$$C = \{e \in E \mid x_e = 1\}.$$

Idea: Encode causality as implication.

$$\mathcal{C} = \underbrace{\bigwedge_{e \in E} \bigwedge_{f \in {}^\bullet e} (\neg x_e \vee x_f)}_{\text{causally closed } (e \Rightarrow f)} \wedge \underbrace{\bigwedge_{e \in E} \bigwedge_{f \in (e)^\bullet \setminus \{e\}} (\neg x_e \vee \neg x_f)}_{\text{conflict free } e \Rightarrow \neg f,\ f \Rightarrow \neg e}$$

Example:

$$\underbrace{(\neg x_{e_3} \vee x_{e_4})}_{e_3} \wedge \underbrace{(\neg x_{e_4} \vee x_{e_3})}_{e_4} \wedge \underbrace{(\neg x_{e_1} \vee \neg x_{e_2})}_{e_1} \wedge \underbrace{(\neg x_{e_2} \vee \neg x_{e_1})}_{e_2}$$

Note:

- CNF, as required by SAT solvers
- Property independent
- Extra variables can be introduced for places (often unnecessary, treat markings as fired markings of configurations)
- Quadratic in O, codings → down to linear

Construction of $\mathcal{V}$:

- Property dependent: Express violation of property.
- If configuration satisfies $\mathcal{V}$, property does not hold.
- C gives hint to bad behaviour (counterexample).

Mutex:

$$\gamma = \underbrace{(x_{e_4} \wedge \neg x_{e_3})}_{\text{by marked}} \wedge \underbrace{(x_{e_2} \wedge \neg x_{e_4})}_{\text{by marked.}}$$

Similar formula for all marking descriptions.