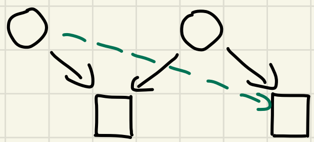
Tutorial 8 27.11

Free-choice Petri nets – short theoretical introduction:

Three equivalent conditions:



- $\forall_{p,q \in P} \; p^\bullet \cap q^\bullet = \emptyset$ or $p^\bullet = q^\bullet$
- $\forall_{t,u \in T} \; {}^\bullet t \cap {}^\bullet u = \emptyset$ or ${}^\bullet t = {}^\bullet u$
- $\forall_{p \in P, t \in T} \; (p,t) \in F \Rightarrow {}^\bullet t \times p^\bullet \subseteq F$

Def. $R \subseteq P$ is a **trap** if $R^\bullet \subseteq {}^\bullet R$

$S \subseteq P$ is a **siphon** if ${}^\bullet S \subseteq S^\bullet$

Fact. Siphons and traps are closed under union.

Commoner's Theorem

a free-choice net is live $\Leftrightarrow$ every proper siphon includes an initially marked trap

1. Show that the reachability problem for free-choice Petri nets is not easier than for the general nets.

Which equivalent representation of Petri nets would be the easiest to work with?

Counter automata as the arcs have weight one

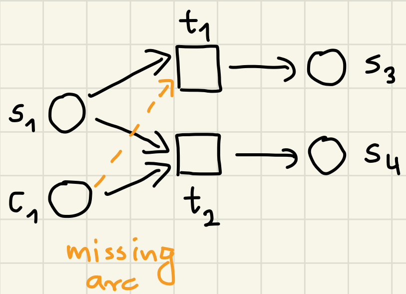
Idea: represent counter automaton without zero tests using free-choice Petri nets

Attempt 1

assume that we have k counters

* straightforward representation
* places represent states s_i and counters c_i of the automaton
* transition moves a token from s_i to s_j and increment / decrement at most one counter place (representing automaton action)

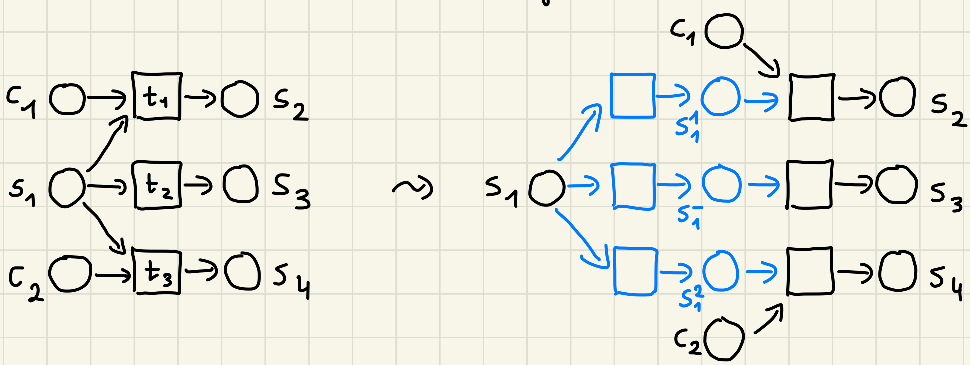
Problem: it may not be a free-choice net



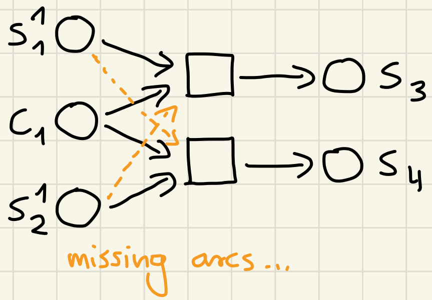
from state s_1 we can go to s_3 or s_4; however, if only one transition decrements some counter, it is not a free-choice net

Attempt 2

* we make $k+1$ copies of each state place s_i: a new place s_i^j for each counter c_j and one additional place s_i^-
* then we create $k+1$ tranzitions - one for each of the new places; each of these transitions takes a token from s_i and puts it on the selected new place

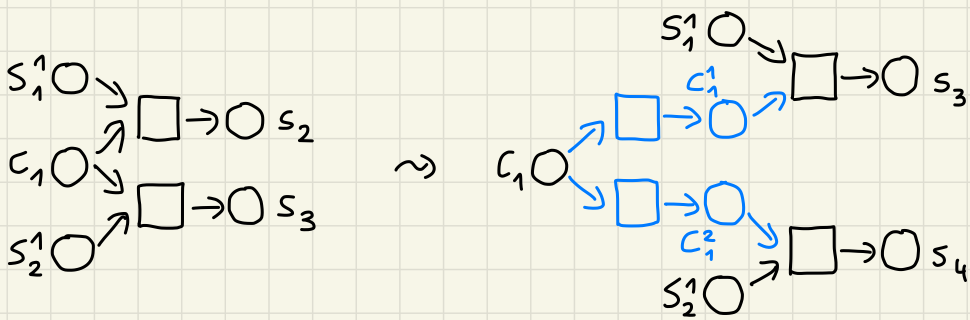


Problem: the new transitions satisfy free-choice conditions but still there is a problem with counter places decremented from different states:



Attempt 3 - final solution

* use the same approach as before but for counter places - a new one for each state



* it is easy to check that the new net is a free-choice net

2. How to compute a maximal trap in a given set S of places in a polynomial time?

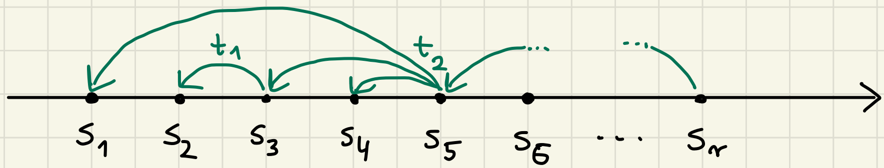
* we know that traps are closed under union, hence, a maximal trap is well-defined
* recall that $R \subseteq P$ is a trap if $R^\bullet \subseteq {}^\bullet R$
* we define the following algorithm:

  initialize $Q := S$

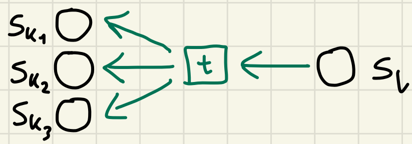
  while there exists $s \in Q$ and $t \in s^\bullet$ such that $t \notin {}^\bullet Q$
  do $Q := Q \setminus \{s\}$

  return Q

* of course the final (returned) Q is a trap
* now we'll prove that all the places that we deleted from S cannot belong to any trap $\subseteq S$
* first, we design the following structure:
  let $s_1, s_2, \dots, s_r$ be the sequence of all places deleted from Q ordered chronologically



* next, we add a directed edge from s_k to s_l for $l < k$ any time we deleted s_l because of some $t \in s_l^\bullet$ such that $s_k \in t^\bullet$, i.e.



intuition: for s_l to be a part of a trap one of s_{k_i} has to belong to trap

* let s_i be a place that we deleted
  - if there are no outgoing edges there exists $t \in s_i^\bullet$ such that $t \notin {}^\bullet S$
  - if there are some outgoing edges, in order to add s_i to the trap, we have to add at least one end of these edges too; we continiue the process and finally reach some place s_j s.t. some $t \in s_j^\bullet$ and $t \notin {}^\bullet S$
* thus, we have found the maximal trap

3. Prove that liveness in free-choice nets is co-NP complete.

4.4 The non-liveness problem is NP-complete

Commoner's Theorem leads to the following nondeterministic algorithm for deciding if a free choice system is *not* live:

(1) guess a set of places R;

(2) check if R is a siphon;

(3) if R is a siphon, compute the maximal trap Q included in R;

(4) if $M_0(Q) = 0$, then answer "non-live".

Steps (2) and (4) can be performed in polynomial time in the size of the system. Exercise 4.5 gives an algorithm for step (3); the reader can prove its correctness and show that its complexity is polynomial as well. It follows from these results that the non-liveness problem for free-choice systems is in NP.

The obvious corresponding deterministic algorithm consists of an exhaustive search through all subsets of places. However, since the number of these subsets is 2^n for a net with n places, the algorithm has exponential complexity.

We now show that the non-liveness problem is NP-complete. As a consequence, no polynomial algorithm to decide liveness of a free-choice system exists unless P=NP.

Theorem 4.28 *Complexity of the non-liveness problem of free-choice systems*

The following problem is NP-complete:

Given a free-choice system, to decide if it is not live.

Proof:

Commoner's Theorem shows that the problem is in NP. The hardness is proved by a reduction from the satisfiability problem for propositional formulas in conjunctive normal form (CNF-SAT).

A formula ϕ is a conjunction of clauses $C_1, \ldots, C_m$ over variables $x_1, \ldots, x_n$. A literal l_i is either a variable x_i or its negation $\overline{x_i}$. The negation of l_i is denoted by $\overline{l_i}$. A clause is a disjunction of literals.

Let ϕ be a formula. We construct a free-choice system (N, M_0) in several stages, and show that ϕ is satisfiable iff (N, M_0) is not live.

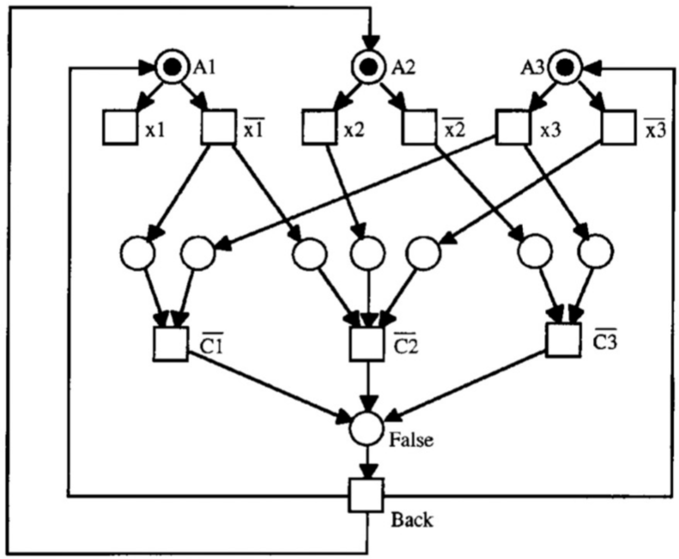


Fig. 4.7 The free-choice system corresponding to ϕ

- For every variable x_i, define a place A_i, two transitions x_i and $\overline{x}_i$ and arcs (A_i, x_i) and $(A_i, \overline{x}_i)$. Let A denote the set $\{A_1, \ldots, A_n\}$.

- For every clause C_j, define a transition $\overline{C_j}$. For every clause C_j and for every literal l_i appearing in C_j define a place $(\overline{l_i}, \overline{C_j})$, an arc leading from the transition $\overline{l_i}$ to the place $(\overline{l_i}, \overline{C_j})$, and an arc leading from $(\overline{l_i}, \overline{C_j})$ to the transition $\overline{C_j}$.

- Define a place *False* and, for every clause C_j, an arc $(\overline{C_j}, \mathit{False})$. Define a transition *Back*, an arc $(\mathit{False}, \mathit{Back})$ and, for every variable x_i, an arc (Back, A_i).

- Define M_0 as the marking that puts one token in all and only the places of A.

It is easy to see that N is a connected free-choice net, and hence (N, M_0) a free-choice system. Moreover, (N, M_0) can be constructed in polynomial time in the length of ϕ. Figure 4.7 shows the system obtained from the formula

$$\phi = (x_1 \vee \overline{x_3}) \wedge (x_1 \vee \overline{x_2} \vee x_3) \wedge (x_2 \vee \overline{x_3}).$$

We can freely choose at every place A_i between letting the transition x_i or $\overline{x_i}$ occur. The occurrences of the selected transitions correspond to the choice of a truth assignment. After these occurrences, a transition $\overline{C_j}$ is enabled if and only if the truth assignment does not satisfy the clause C_j. If $\overline{C_j}$ is enabled, then it can occur and put a token in the place *False*, which corresponds to the fact that, since the clause C_j is false under this assignment, the whole formula ϕ is false. We now prove:

($\Rightarrow$) If ϕ is satisfiable, then (N, M_0) is not live.

Let f be a truth assignment satisfying ϕ, and let $l_1, \ldots, l_n$ be the literals mapped to *true* by f. Let $\sigma_f = l_1 \ldots l_n$.

By the construction of (N, M_0), σ_f is an occurrence sequence (in our example, we can take $\sigma_f = x_1\, x_2\, x_3$). Let $M_0 \xrightarrow{\sigma_f} M$.

We show that no transition of N is enabled at M, which proves the result. By the construction of (N, M_0) and σ_f, only $\overline{C_j}$ transitions can be enabled at M. So it suffices to prove that no transition $\overline{C_j}$ is enabled at M.

Consider a clause C_j. Since f satisfies ϕ, there exists a literal l_i in C_j such that $f(l_i) = \mathit{true}$. By the definition of σ_f, we have $l_i \in \sigma_f$ and $\overline{l_i} \notin \sigma_f$. Since $\overline{l_i} \notin \sigma_f$, the place $(\overline{l_i}, \overline{C_j})$ is not marked at M. By the construction of N, $(\overline{l_i}, \overline{C_j})$ is an input place of $\overline{C_j}$. So $\overline{C_j}$ is not enabled at M.

($\Leftarrow$) If (N, M_0) is not live, then ϕ is satisfiable. ← homework

We start with the following observation: if a transition x_i has an output place $(x_i, \overline{C_j})$ and $\overline{x_i}$ has an output place $(\overline{x_i}, \overline{C_k})$, then the set

$$Q = \{\mathit{False}, A_i, (x_i, \overline{C_j}), (\overline{x_i}, \overline{C_k})\}$$

is a trap. Moreover, Q is initially marked because $M_0(A_i) = 1$.

Now, assume that (N, M_0) is not live. By Commoner's Theorem, there exists a proper siphon R of N which includes no initially marked trap. By the construction of N, R contains *False* and at least one place A_i of A. Moreover, R contains either no place of $x_i^\bullet$ or no place of $\overline{x_i}^\bullet$; otherwise we would have $Q \subseteq R$ for the initially marked trap Q defined above.

This last property of R allows us to construct a truth assignment f satisfying the following for every place $A_i \in R$: if $x_i^\bullet \cap R \neq \emptyset$ then $f(\overline{x_i}) = \mathit{true}$ and if $\overline{x_i}^\bullet \cap R \neq \emptyset$ then $f(x_i) = \mathit{true}$.

We show that f satisfies ϕ. Let C_j be an arbitrary clause of ϕ. Since *False* is a place of R, the set R contains some input place $(\overline{l_i}, \overline{C_j})$ of $\overline{C_j}$ and hence it also contains the place A_i, which belongs to ${}^\bullet\overline{l_i}$. So $\overline{l_i}^\bullet \cap R \neq \emptyset$.

By the definition of f, we have $f(l_i) = \mathit{true}$. Since, by construction of N, l_i is a literal of C_j, the assignment f satisfies C_j. Finally, f satisfies ϕ because C_j was arbitrarily chosen. □

Source: Jorg Desel, Javier Esparza. Free Choice Petri Nets

1.[D] Construct two counterexamples proving that the Commoner's theorem does not hold for general Petri nets (weights = 1, connected).

2.[D] Prove that reachability is reducible to non-liveness in general Petri nets.

homework (not obligatory)