Introduction to Model-Checking

Theory and Practice

Beihang International Summer School 2018

Petri Nets

dealing with infinite systems

Unbounded nets

- For model-checking, we have only worked with bounded nets, i.e. nets with a finite reachability graph
- Several questions:
  - is it possible to test whether a system is bounded ?
  - can we analyze the behavior of unbounded nets ?

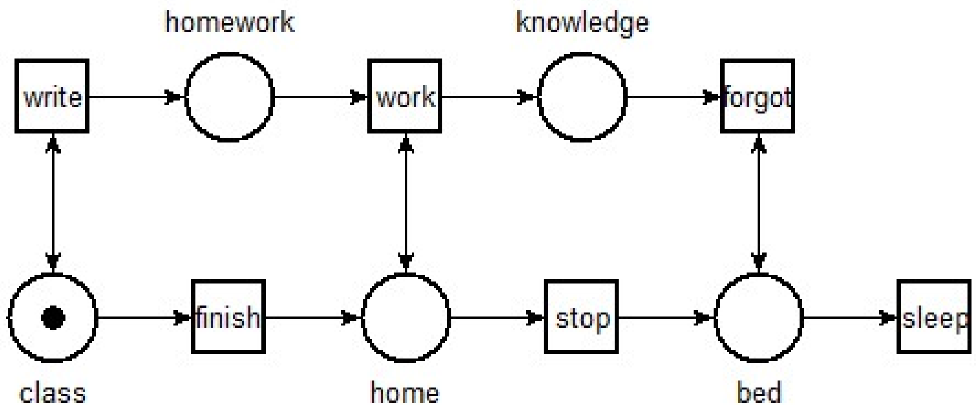
Example of an unbounded net

We model the life of a student at Beihang

1. initially the student is in class where (s)he is writing homework
2. then (s)he can finish class and go home, where (s)he can work, reread his homework and gain knowledge
3. next (s)he can stop working and go to bed where he can forget some of what (s)he learned today
4. finally, he can end the day and go to sleep

A day in the life of a student

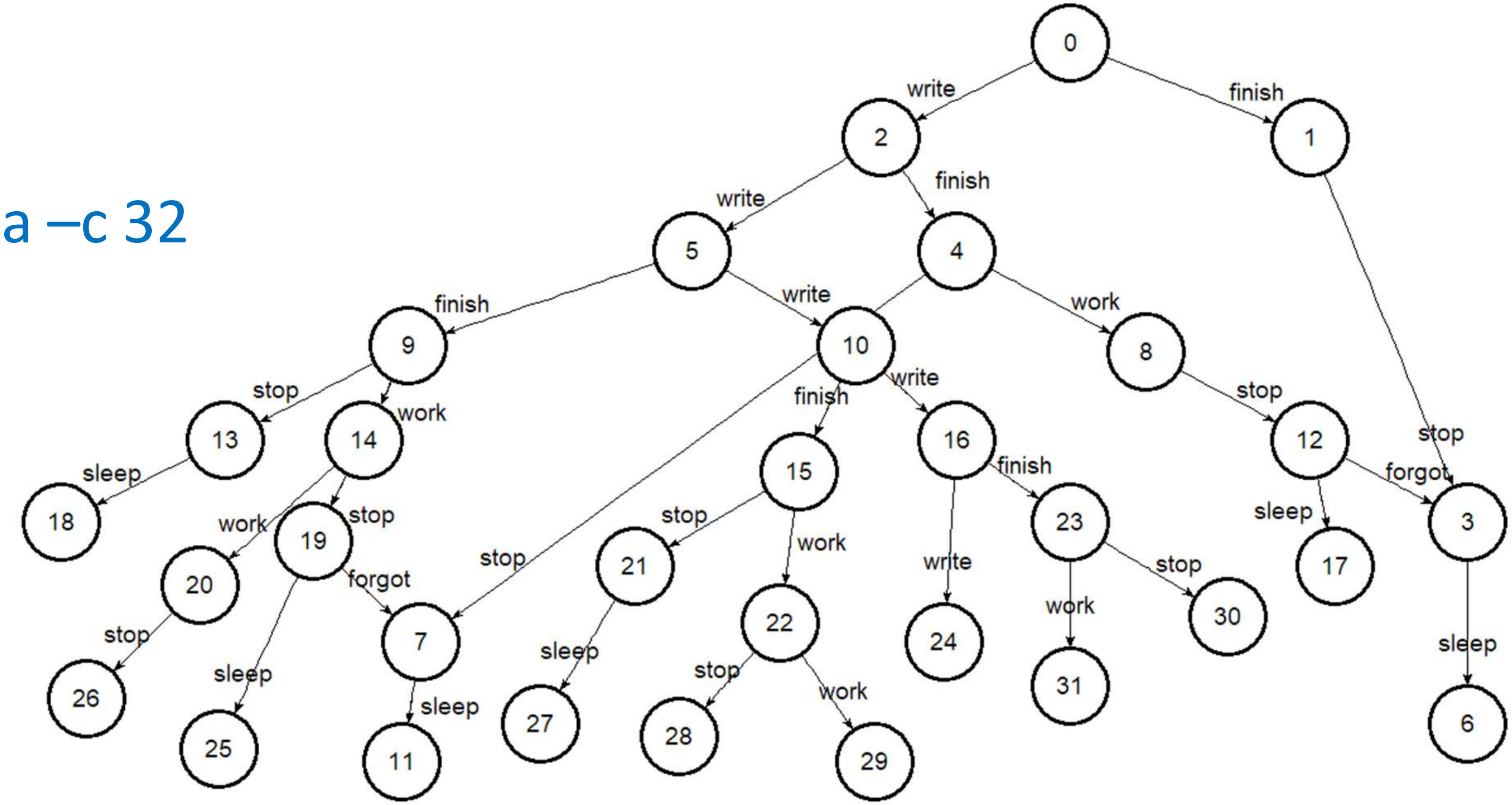
1. initially the student is in class where (s)he is writing homework
2. then (s)he can finish class and go home, where (s)he can work, reread his homework and gain knowledge
3. next (s)he can stop working and go to bed where he can forget some of what (s)he learned today
4. finally he can end the day



Partial graph

We can approximate the behavior of the net by using a bound on the number of states we can generate

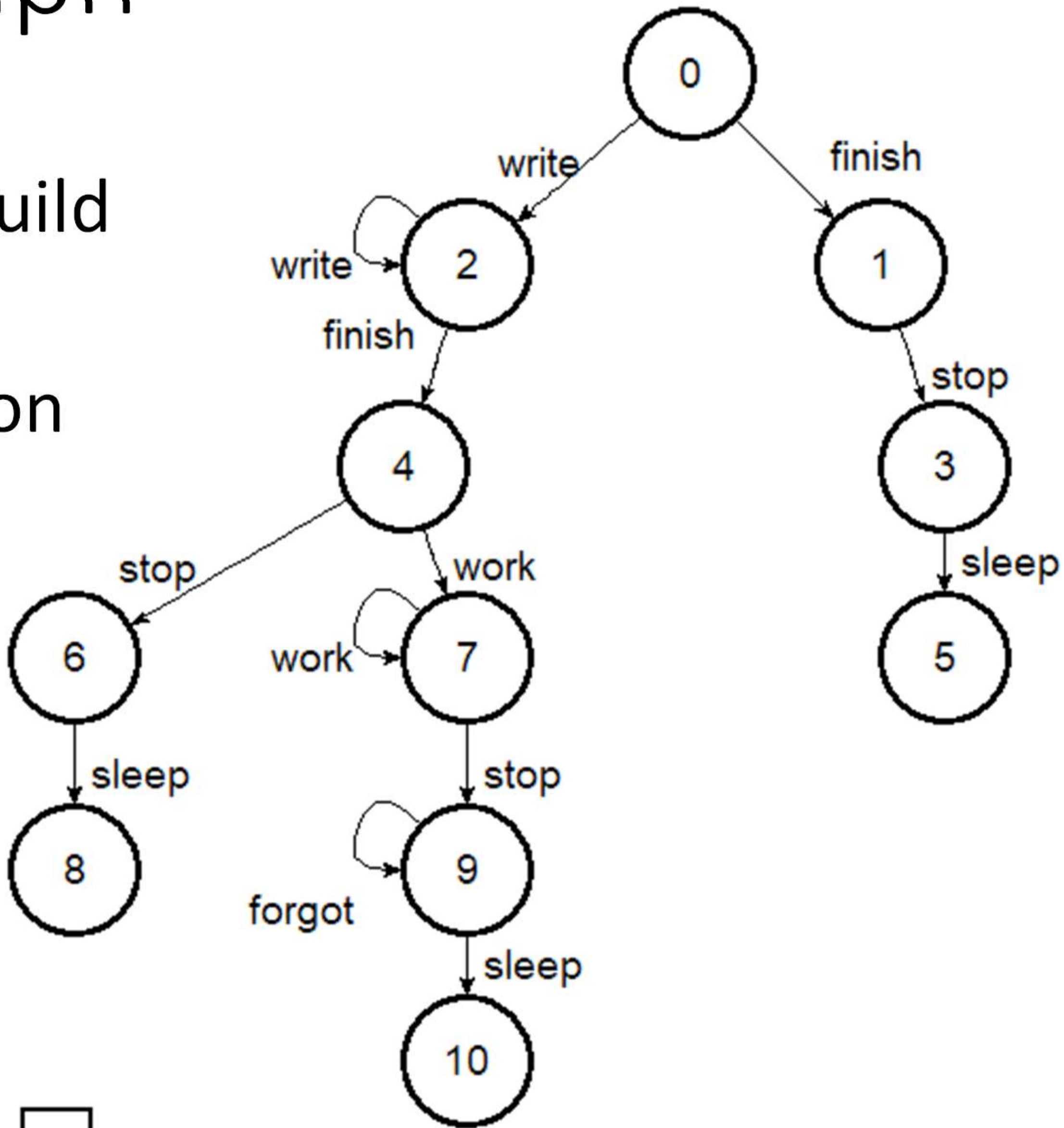
tina –c 32

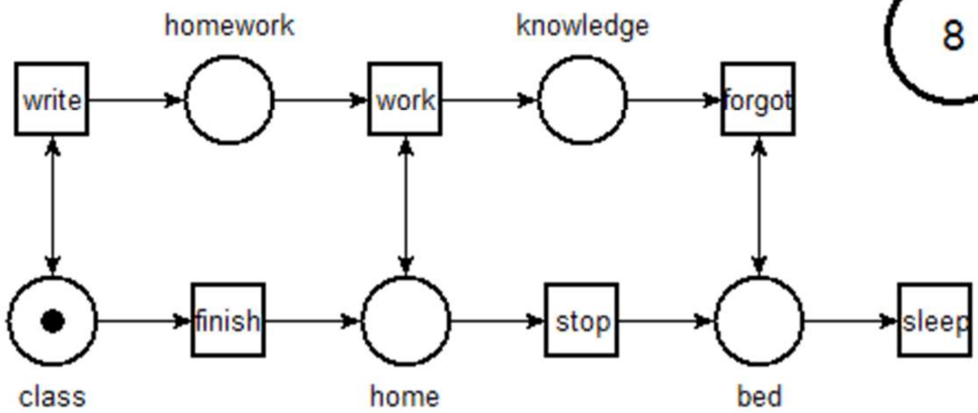


Coverability graph

Another solution is to build the coverability graph

≡ an over-approximation of the behavior





Monotonicity

- Consider a marked net (N, m_0)
- The *monotonicity* property states that:

if $m \;\rightarrow m'$ in N then necessarily, $\forall \Delta \in \mathbb{N}^P$

$$m + \Delta \;\rightarrow m' + \Delta$$

$m'_0 - m_0 = \Delta \geq 0$

- As a consequence, if $m'_0 \geq m_0$ then:

$|reach(m'_0)| \geq |reach(m_0)|$ (more markings)

$|RG(m'_0)| \geq |RG(m_0)|$ (more transitions)

Deciding whether N is k-safe

- Consider a marked net (N, m_0)

- We observe that the net has an infinite number of markings ($\equiv$ unbounded $\equiv$ not k-safe) iff there is a reachable marking m, and a vector $\Delta > \bar{0}$, such that:

(repetitive) increasing sequence

$$m_0 \;\rightarrow\; \ldots \rightarrow\; m \;\rightarrow\; \ldots \rightarrow m + \Delta$$

Dickson's lemma

- **Idea**: try to find an infinite, "non-increasing", sequence of elements of $\mathbb{N}^2$. Same with $\mathbb{N}^P$.

Proof

$\to^*$ is the "transitive closure" of $\to$

- If we have $m \ \to \ \ldots \to m + \Delta$, then we can "pump" the same sequence of transitions

$$m \ \to^* \ m + \Delta \to^* m + 2.\Delta \to^* m + 3.\Delta \to \ \ldots$$

we write: $m \ \to^* \ m + \omega.\Delta$

- Conversely, if it is impossible to find m and Δ such that $m \to^* m + \Delta$ then the reachability graph of (N, m_0) must be finite

Coverability graph

- We consider markings with values in $\mathbb{N} \cup \{\omega\}$, where ω stands for an unbounded amount of tokens

$$n + \omega = \omega$$
$$\omega + \omega = \omega$$
$$\omega - n = \omega$$
$$0 \,.\, \omega = 0$$
$$n \,.\, \omega = \omega$$

- We can define transitions on ω-markings as we do with normal markings

Coverability graph

- Build a graph by computing the successors

- If we find two markings such that $m \;\to^{*}\; m'$ and $\Delta = \; m' \; - \; m > \bar{0}$ then replace m' with

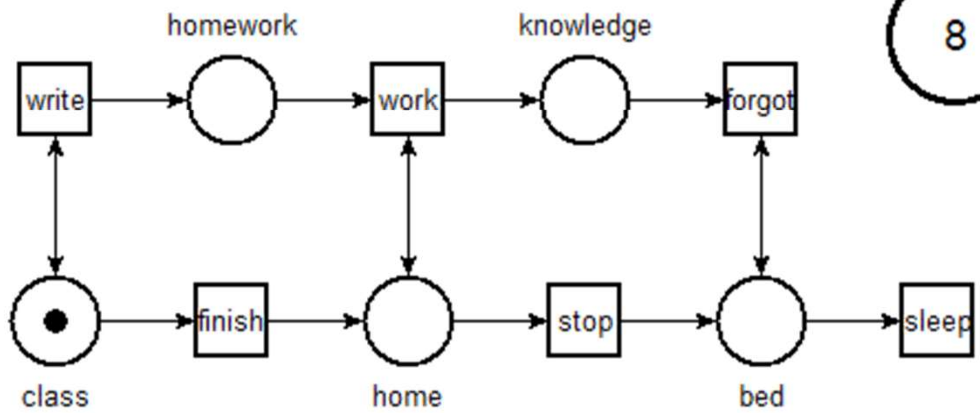
$$m + \Delta\,.\,\omega$$

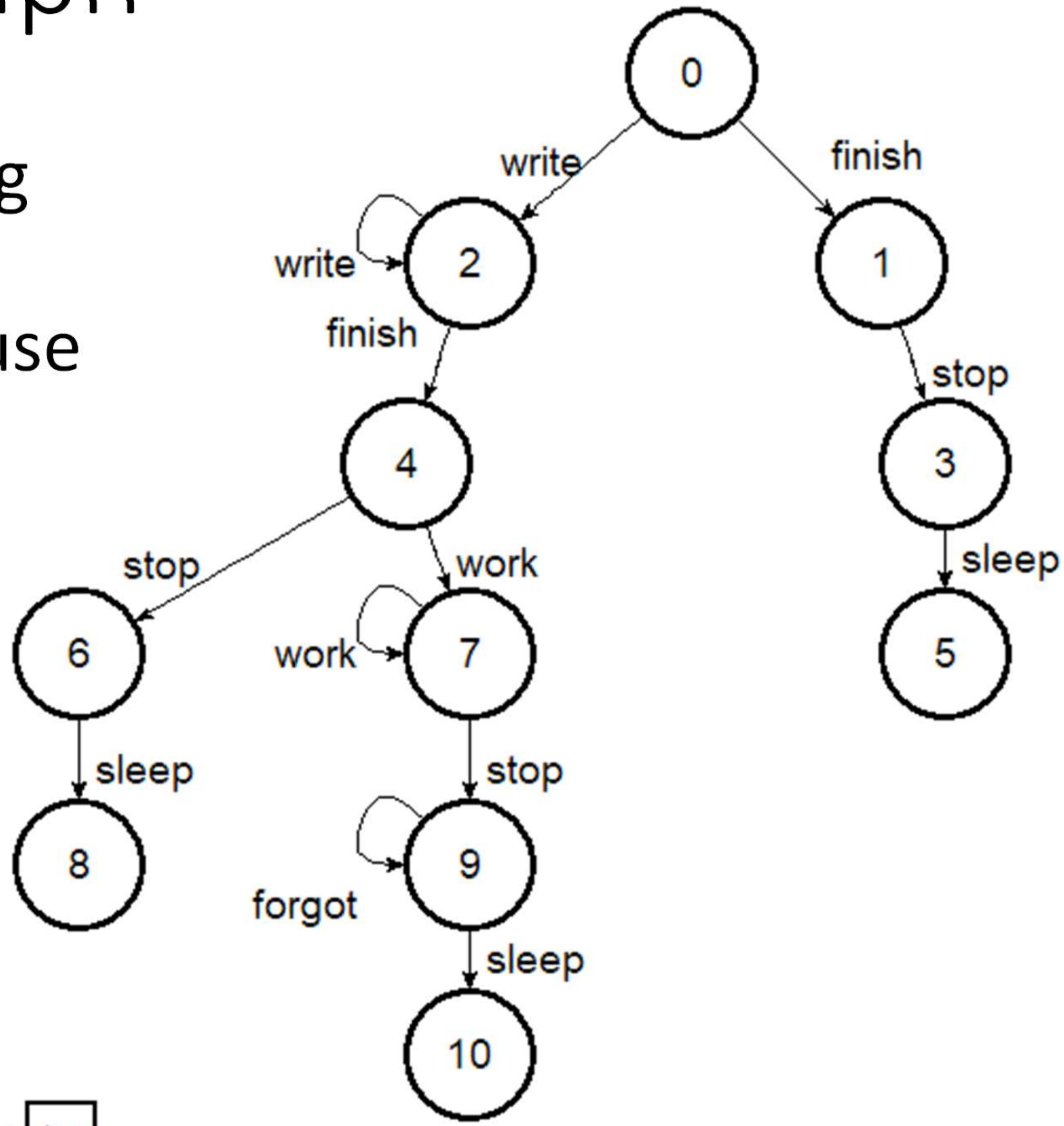
- Stop when you have enumerated all possibilities

Coverability graph

we put together increasing markings, e.g. we have a “loop” $(2, write, 2)$ because only place $homework$ increases.

$$2 = (1, \omega, 0, 0, 0)$$





Comparison R.G. vs C.G.

- Consider a marked net (N, m_0)
- By construction, we have:

$\Leftrightarrow$ marking m is in $reach(m_0)$ if and only if m appears as a vertex (node) in the Reachability Graph

- By construction, we have:

$\Rightarrow$ marking m is in $reach(m_0)$ implies m is covered by some vertex m_ω of the Coverability Graph: $m \leq m_\omega$

What you need to remember

- It is possible to reason about unbounded nets, but we only have an over-approximation in this case

- Coverability tree/graph ≡ Karp and Miller algorithm

Structural Analysis

where linear algebra meets P/T nets

Analyzing without exhaustive search

- All our methods so far have relied on constructing the R. G. of a net
- Even the coverability graph is $\approx$ an exhaustive enumeration of the states
- When finite, the size of the R. G. can be exponential in size
- Next $\rightarrow$ a method to prove properties on nets based only on a mathematical analysis of their "topology"

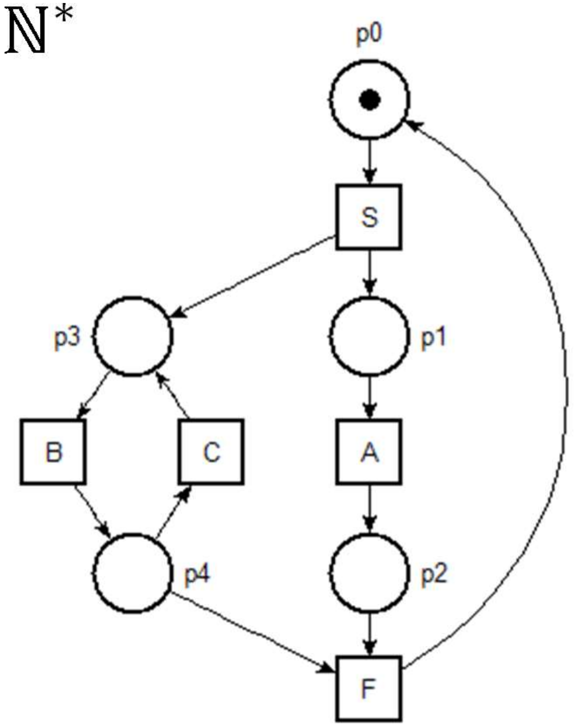
Reminder

A P/T net is a tuple $N = \langle P, T, F, W \rangle$ where

- P is a finite set of places
- T is a distinct finite set of transitions ($P \cap T = \emptyset$)
- F is the flow relation: $F \subseteq (P \times T) \cup (T \times P)$
- W are the weight of the arcs: $W : F \rightarrow \mathbb{N}^*$

A marking m defines a distribution of tokens to places $m : P \rightarrow \mathbb{N}$

A marked P/T net (N, m_0) is a net with initial marking m_0

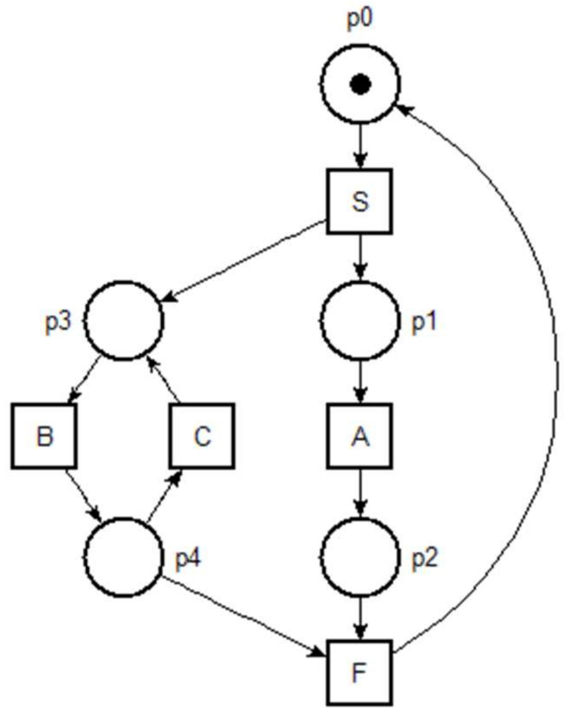


Reminder

$Pre_t(p) = W(p, t)$ if $p \in Pre(t)$
and $Pre_t(p) = 0$ otherwise

$Post_t(p) = W(t, p)$ if $p \in Post(t)$
and $Post_t(p) = 0$ otherwise

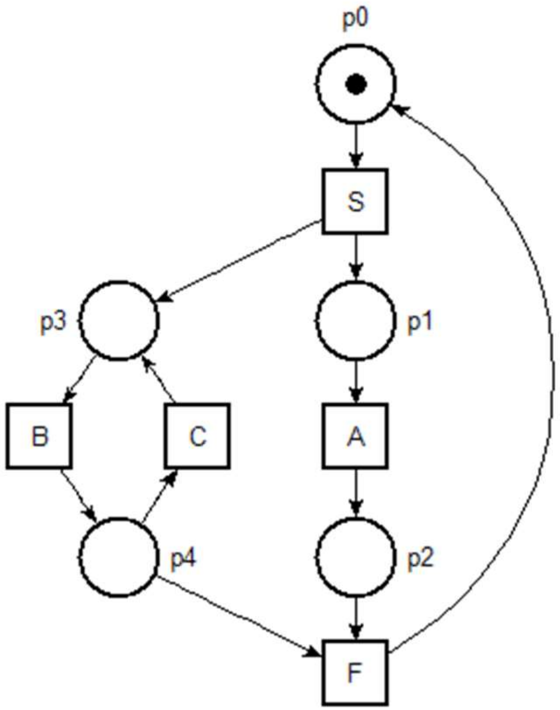
$$\begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix} \qquad Pre_F = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} \qquad Post_F = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$



Reminder

Pre	S	A	B	C	F
p_0	1	0	0	0	0
p_1	0	1	0	0	0
p_2	0	0	0	0	1
p_3	0	0	1	0	0
p_4	0	0	0	1	1

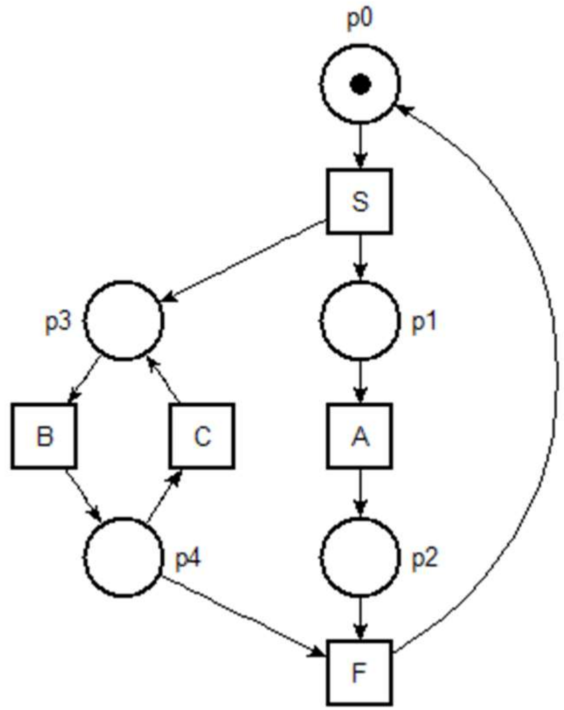
Post	S	A	B	C	F
p_0	0	0	0	0	1
p_1	1	0	0	0	0
p_2	0	1	0	0	1
p_3	1	0	0	1	0
p_4	0	0	1	0	0



Reminder

Pre	S	A	B	C	F
p_0	1	0	0	0	0
p_1	0	1	0	0	0
p_2	0	0	0	0	1
p_3	0	0	1	0	0
p_4	0	0	0	1	1

Post	S	A	B	C	F
p_0	0	0	0	0	1
p_1	1	0	0	0	0
p_2	0	1	0	0	0
p_3	1	0	0	1	0
p_4	0	0	1	0	0

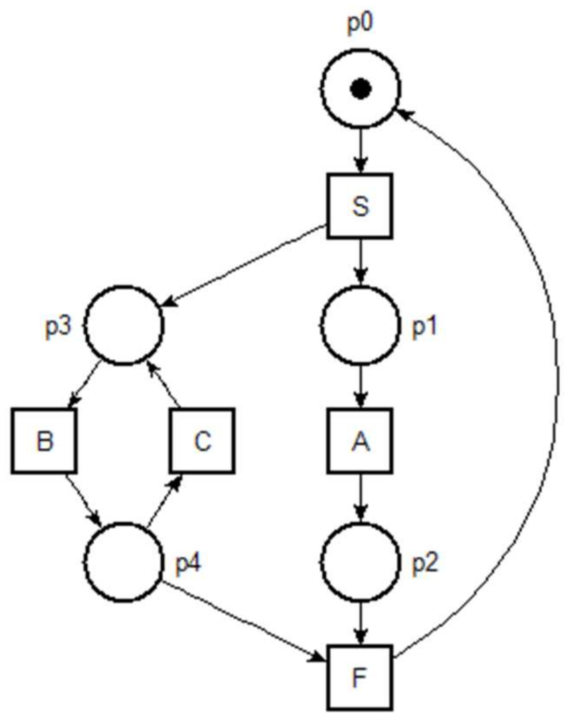


Incidence matrix $|P| \times |T|$

$$C = Post\ - Pre$$

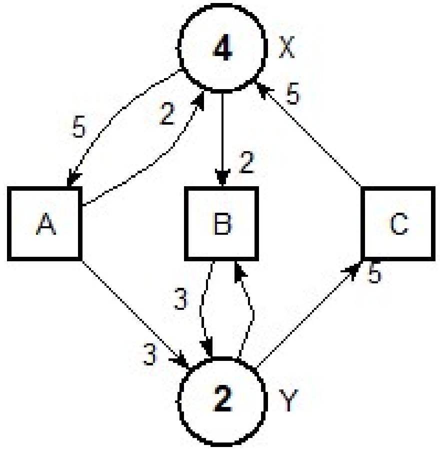
$$C(p_i, t_j) = W(t_j, p_i) - \ W(p_i, t_j)$$

C	S	A	B	C	F
p_0	-1	0	0	0	1
p_1	1	-1	0	0	0
p_2	0	1	0	0	1
p_3	1	0	-1	1	0
p_4	0	0	1	-1	-1



Another example

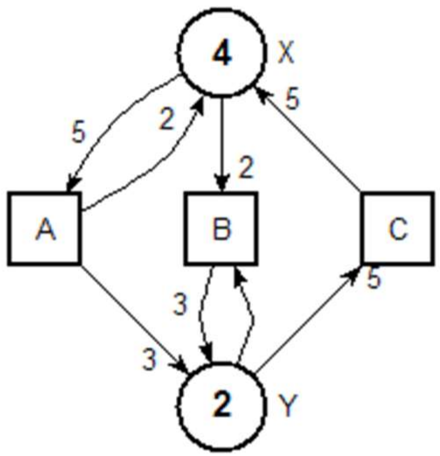
C	A	B	C
X	-3	-2	5
Y	3	2	-5



A net is pure if $\forall p, t\,.\, W(t,p) \times W(p,t) = 0$. (no test arcs!)
When a net is pure, we can reconstruct the flow function from the incidence matrix.
Compared to the previous example, this net is not *pure.*

Commutative image of a trace

- A trace σ is a sequence of transitions, e.g. $A.A.B.C.\ldots$
- Let $\#_t(\sigma)$ be the number of occurrences of t in the trace σ, the Parikh's number of t



The commutative image of σ, also called its Parikh's image, is the vector:

$$\#(\sigma) = \left(\#_{t_1}(\sigma), \ldots, \#_{t_n}(\sigma)\right) \in \mathbb{N}^T$$

e.g. $\#(A.A.B.C) = (2, 1, 1)$

Fundamental equation

- We already saw that we can write markings as vectors in $\mathbb{N}^P$
- We can also write traces as vectors in $\mathbb{N}^T$ (but we forget in which order transitions are fired).

Fundamental equation:

if $m_1 \rightarrow^{\sigma} \ m_2$ then we have that

$$m_2 = m_1 + C.\#(\sigma)$$

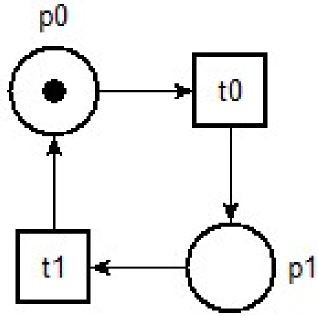
Corollary

Reachability:

If a marking m is reachable from (N, m_0) then there is a vector $X \in \mathbb{N}^T$ such that $m = C.X + m_0$

- We can use this property to show that some markings are not reachable

Example

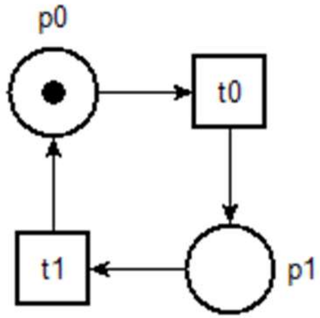


- Initial marking $m_0 = (1\ 0)$
- Incidence matrix $\begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix}$
- Is it possible to reach marking $(1\ 1)$ from m_0?
- *Answer*: of course not, because the following equation has no solutions:

$$(1\ 1) = (1\ 0) + \begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix} . (x_1\ x_2)$$

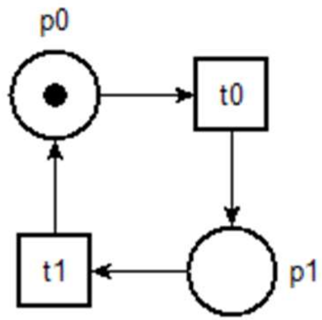
or equivalently: $x_2 - x_1 = 0$ and $x_1 - x_2 = 1$

T-invariants



- If we have a vector T such that $C.T = \bar{0}$ then we have that $m = m + C.T$
- Firing a trace with commutative image T gets you back to the state where you came from. So they indicate possible cycles (loops) in the reachability graph
- T is called a *transition invariant* or *T-invariants*
- in our example, $T = (1\ 1)$ is an invariant.

P-invariants

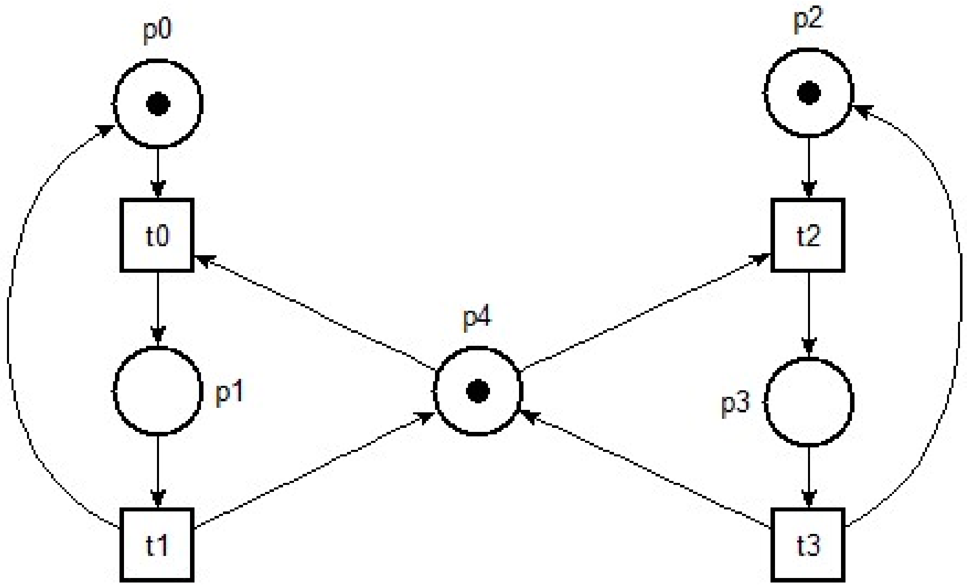


- If we have a vector F such that $F.C = \bar{0}$ then we have that $F.m = F.(m + C.X)$ for all $X \in \mathbb{N}^T$
- When we fire transitions from a marking m_1 and reach marking m_2 then $F.m_1 = F.m_2$. That is, the value $F.m$ is constant among all reachable states.
- F is called a *place invariant* or *P-invariants*

- in our example, $P = (1\ 1)$ is also a P-invariant, meaning that $p_0 + p_1$ is an invariant $(= 1)$

Example: mutual exclusion

We want to ensure mutual exclusion between places p_1 and p_3



Therefore we want to prove $p_1 + p_3 \leq 1$

We can find some P-invariants for the net

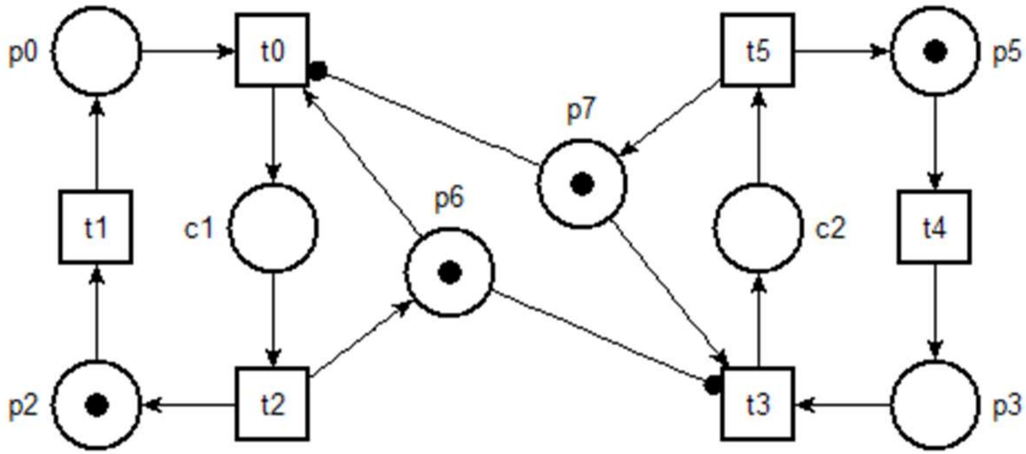
$$p_0 + p_1 = 1 \qquad\qquad p_2 + p_3 = 1$$

$$p_1 + p_4 + p_3 = 1$$

So it is obvious !

Example: mutual exclusion

We want to ensure mutual exclusion between places c_1 and c_2



Therefore we want to prove $c_1 + c_2 \leq 1$

We can find some P-invariants for the net

$$p_0 + p_2 + c_1 = 1 \qquad p_5 + p_3 + c_2 = 1$$

$$c_1 + p_6 = 1 \qquad c_2 + p_7 = 1$$

Try to find the right combination !

Siphon and Traps

using the net topology

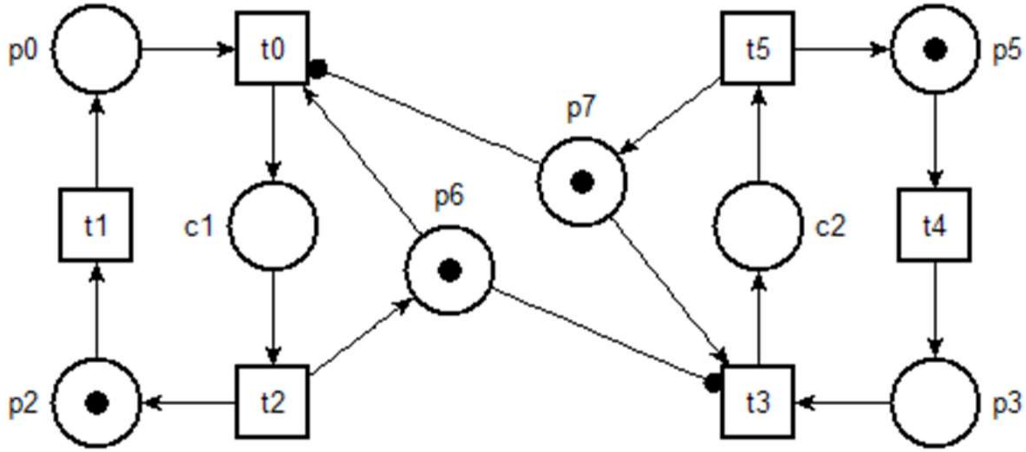
Trap

- A trap is a set of places $S \subseteq P$ such that
$$Post(S) \subseteq Pre(S)$$
- Therefore, when you take some tokens from S (you fire a transition in $Post(S)$), you should also leave at least one token in $S \Rightarrow$ once there is a token in S, there will always be a token in S
- We say that trap S is marked in m if $m(p) \geq 1$ for at least one place $p \in S$

Traps $\equiv$ once marked, marked forever

Mutual Exclusion

In this example, the set $\{p_6, p_7\}$ is a trap. Therefore there is always a token in at least one of them.



Therefore we have $p_6 + p_7 \geq 1$

If we add the invariants computed previously, we obtain: $c_1 + p_6 = 1$ and $c_2 + p_7 = 1$.

Hence $c_1 + p_6 + c_2 + p_7 = 2$ and $p_6 + p_7 \geq 1$, which gives: $c_1 + c_2 \leq 1$ as needed

Siphon

- A siphon is a set of places $S \subseteq P$ such that
$$Pre(S) \subseteq Post(S)$$
- Therefore, when you put tokens in S (you fire a transition in $Pre(S)$), you should also take tokens from $S \Rightarrow$ once there are no tokens in S, then you cannot fire transition in $Pre(S)$

Siphon $\equiv$ once empty, empty forever

Siphons and Traps

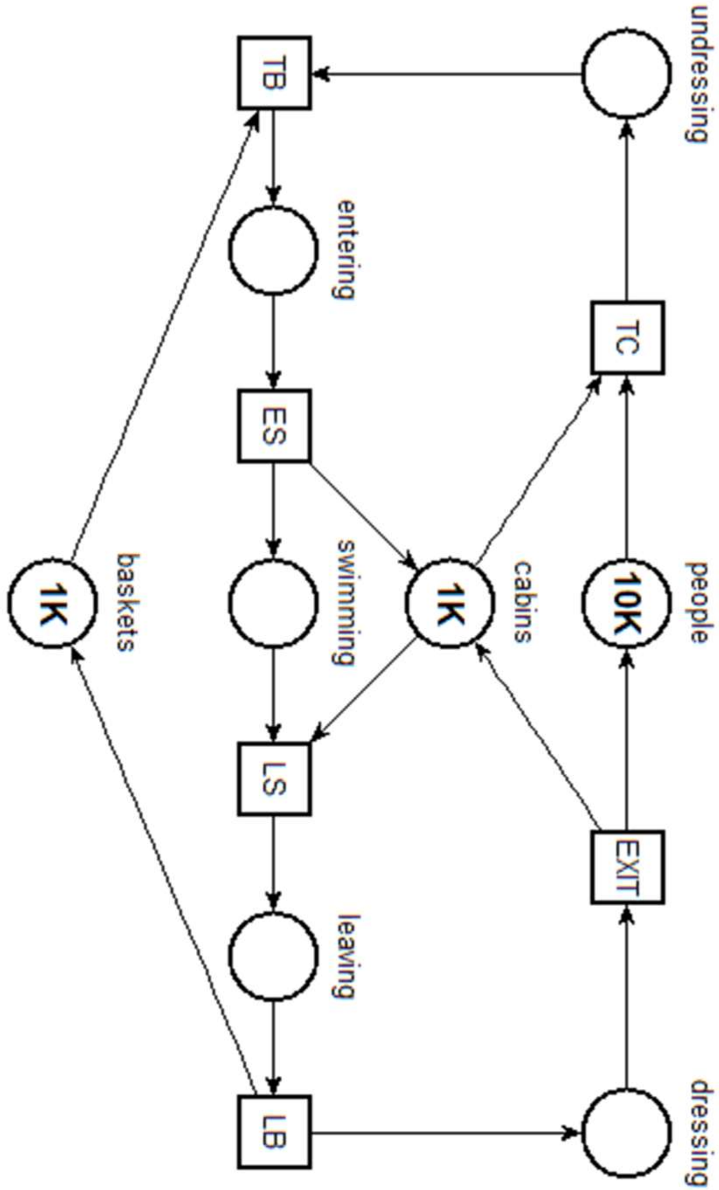
If every siphon contains a marked trap then the net is deadlock free

Stubborn Sets

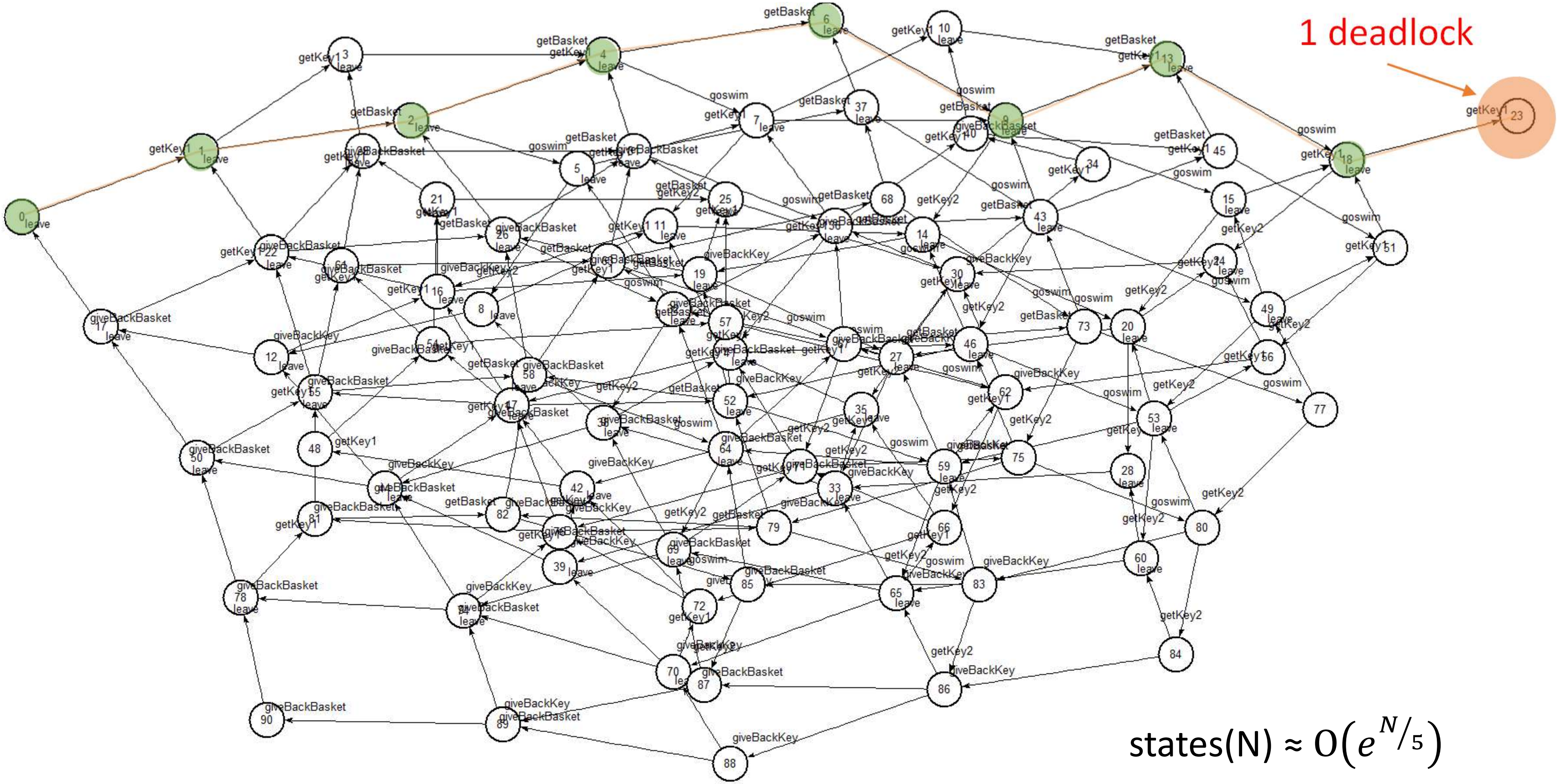
Doing better than exhaustive search

Swimming Pool again

- 60 taks need to enter a critical section ; two type of resources (30 of each)
- We want to test *deadlocks* ; starvation ; reversibility ; …



A practical example “size 2” (91 states only)

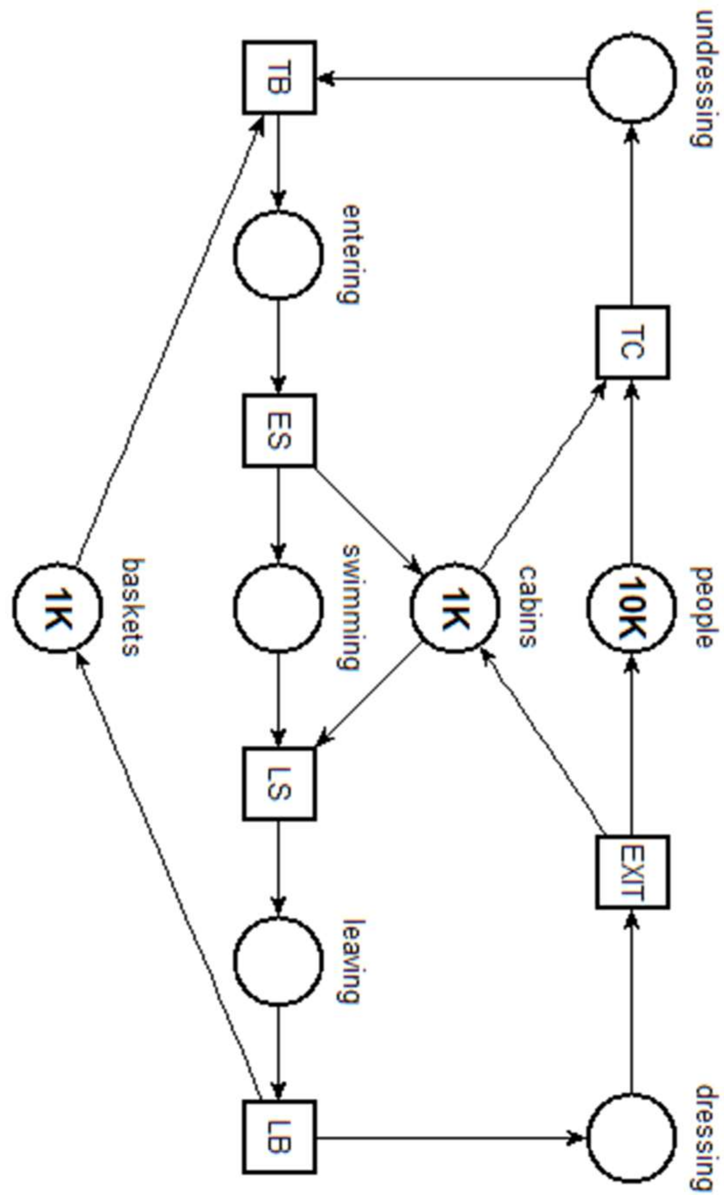


states(N) ≈ $O\left(e^{N/5}\right)$

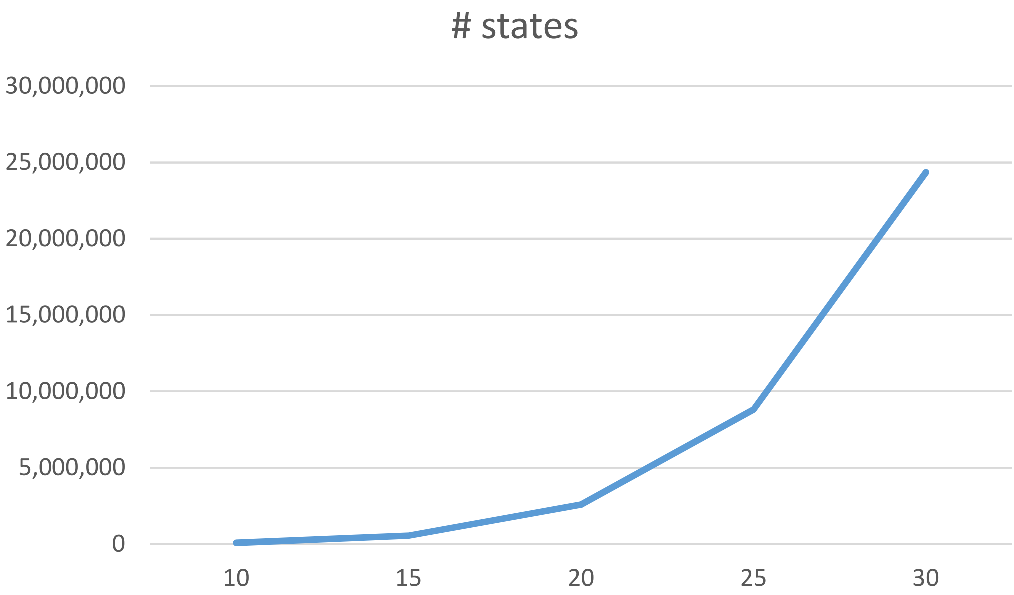
A practical example

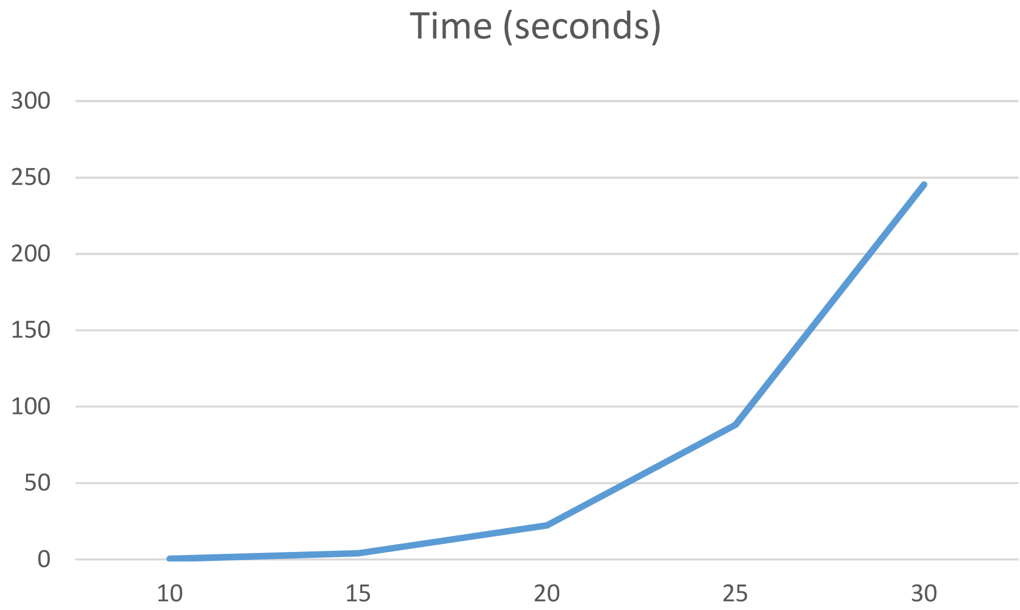
System ‘30’ has 23e6 states, exploration takes 4 minutes

With size ‘15’ it has ½ a million states and it takes 4 seconds.



Problem with exhaustive search





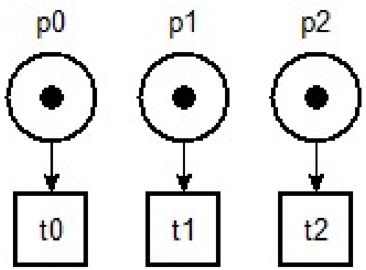
- We can do better by choosing a “clever” order and by stopping when a problem is found

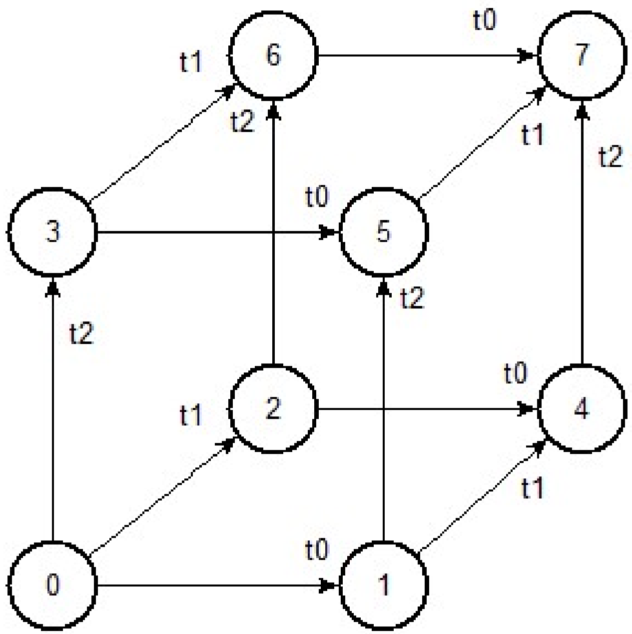
  Counterexample has size 120 !

  But breadth-first search ⇒ still 4.6e6 states and 46s

Partial-Order reduction

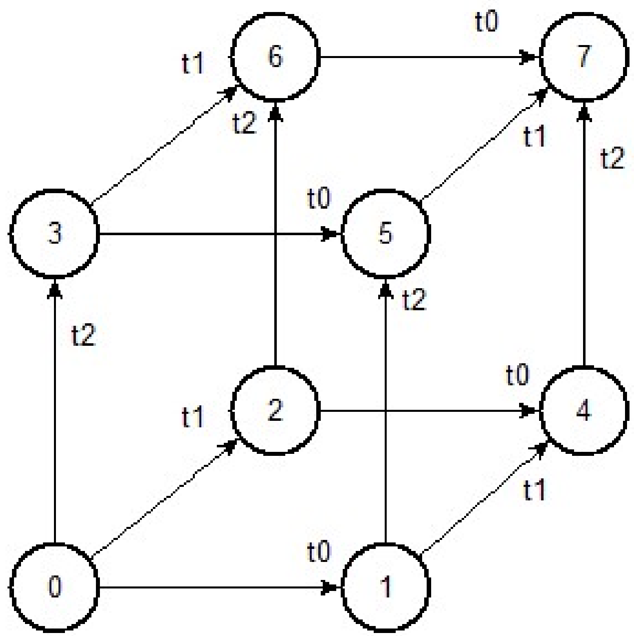
Idea: some of the complexity comes from the interleaving of “independent” actions

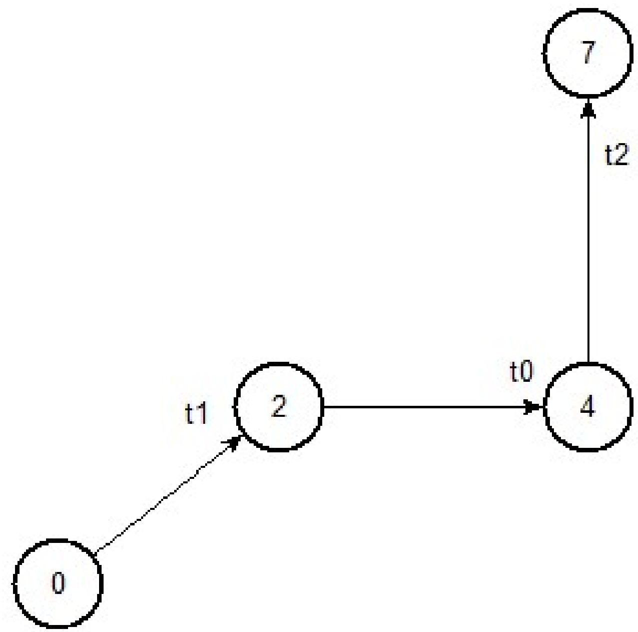




Partial-Order reduction

Solution: choose an arbitrary order between independent transitions when you can always postpone/reorder them (this relation may change during the evolution of the system)





Using abstraction

- Example has 23e6 states
- Exploration using *persistent states* is instantaneous for $N \leq 100$; … takes 125ms for 6000 banhistas
- Predicted size for 'N = 6000' is 1e525 states!

tina -q -P swimming-HUGE.ndr