T-79.231 Parallel and Distributed Digital Systems

Place/Transition Nets

Marko Mäkelä

July 15, 2002

Place/Transition Nets

Let us generalise the rules of the token game introduced in the first lecture:

- A place may contain several tokens, which may be interpreted as resources.
- There may be several input and output arcs between a place and a transition. The number of these arcs is represented as the weight of a single arc.
- A transition is enabled if its each input place contains at least as many tokens as the corresponding input arc weight indicates.
- When an enabled transition is fired, its input arc weights are subtracted from the input place markings and its output arc weights are added to the output place markings.

Net

The triple $N = \langle S, T, F \rangle$ is a *net* if

- the sets of places S and transitions T are disjoint ($S \cap T = \emptyset$) and
- for the *flow relation* F it holds that $F \subseteq (S \times T) \cup (T \times S)$.

Note that for a net N and a transition $t \in T$, $\langle s, t \rangle \in F$ if s is an input place of t, and $\langle t, s' \rangle \in F$ if s' is an output place of t.

Let $a \in S \cup T$. The set ${}^\bullet a = \{a' \mid \langle a', a \rangle \in F\}$ is the *pre-set* of a, and the set $a^\bullet = \{a' \mid \langle a, a' \rangle \in F\}$ is its *post-set*.

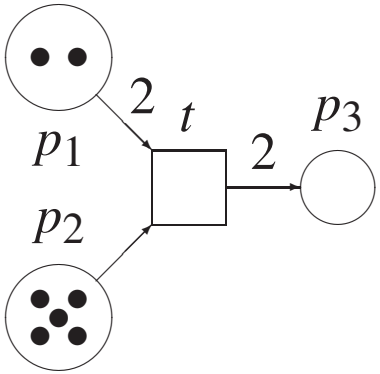
Place/Transition Net

The quadruple $\langle S, T, F, W \rangle$ is a *Place/Transition Net* (Stellen/Transition-Netz) if

- $\langle S, T, F \rangle$ is a finite net and
- the mapping $W : F \to (\mathbb{N} \setminus \{0\})$ determines the *arc weights* of the transitions.

The corresponding quintuple $\langle S, T, F, W, M_0 \rangle$ is a *Place/Transition System*, where $M_0 : S \to \mathbb{N}$ is the initial marking of the system.

Place/Transition Net: Example



The place/transition system $\langle S, T, F, W, M_0 \rangle$ above is defined as follows:

- $\langle S, T, F \rangle = \langle \{p_1, p_2, p_3\}, \{t\}, \{\langle p_1, t \rangle, \langle p_2, t \rangle, \langle t, p_3 \rangle\} \rangle$,
- $W = \{\langle p_1, t \rangle \mapsto 2, \langle p_2, t \rangle \mapsto 1, \langle t, p_3 \rangle \mapsto 2\}$ and
- $M_0 = \{p_1 \mapsto 2, p_2 \mapsto 5, p_3 \mapsto 0\} = \langle 2, 5, 0 \rangle$.

The firing rule of Place/Transition Nets

Let there be a place/transition net $\langle S, T, F, W \rangle$ and a marking $M : S \to \mathbb{N}$.

1. The transition $t \in T$ is *M-enabled*, $M[t\rangle$ if $\forall s \in {}^\bullet t : M(s) \geq W(s,t)$. (Firing condition)

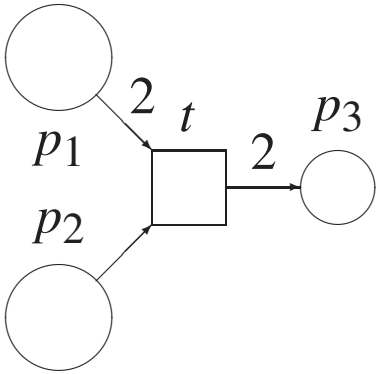
2. An M-enabled transition t may *fire*, producing the *successor marking* $M' := [M\rangle t$, $M'(s) = M(s) - W'(s,t) + W'(t,s)$ where $W' : (S \times T) \cup (T \times S) \to \mathbb{N}$:

$$\begin{cases} W'(f) = W(f) & \forall f \in F \\ W'(f) = 0 & \text{otherwise.} \end{cases} \qquad \text{(Firing rule)}$$

3. The *successor markings* of the set $\mathcal{M}$ are $\mathcal{M}[\rangle := \bigcup_{M \in \mathcal{M}} \bigcup_{t \in T} \left\{ \{[M\rangle t\} \;\middle|\; M[t\rangle \right\}$.

4. The *reachable markings* of $\mathcal{M}$ are $[\mathcal{M}\rangle := \mathcal{M}[\rangle^* = \mathcal{M} \cup \mathcal{M}[\rangle \cup \mathcal{M}[\rangle[\rangle \cup \ldots$.

The firing rule of Place/Transition Nets: Example



Marking M	$M[t\rangle$	$\{M\}[\rangle$
$\{p_1 \mapsto 2, p_2 \mapsto 5, p_3 \mapsto 0\}$	enabled	$\{\{p_1 \mapsto 0, p_2 \mapsto 4, p_3 \mapsto 2\}\}$
$\{p_1 \mapsto 0, p_2 \mapsto 4, p_3 \mapsto 2\}$	disabled	$\emptyset$
$\{p_1 \mapsto 1, p_2 \mapsto 5, p_3 \mapsto 0\}$	disabled	$\emptyset$

Example: A multi-processor system (1/4)

Let us create a coarse model of a system consisting of 5 processors, 3 shared memory banks and 2 buses. The states of the processors are mapped to places, always containing a total of 5 tokens:

p_1: ready for executing an instruction

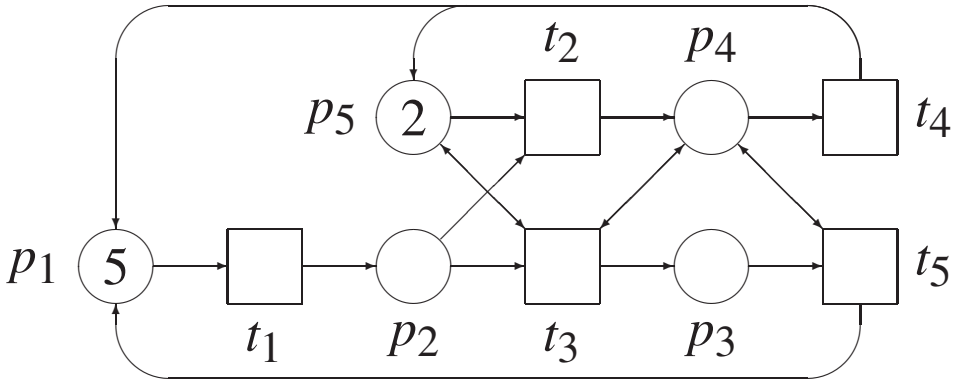
p_2: waiting for a bus to become available

p_3: waiting for a memory bank

p_4: occupying a bus

The place p_5 accounts for the status of the memories and the buses.

Example: A multi-processor system (2/4)



- t_1 access request (executing a new instruction)
- t_2 the desired memory is available
- t_3 the memory is occupied; queueing
- t_4 accessing memory and completing the request
- t_5 accessing previously occupied memory, completing

Example: A multi-processor system (3/4)

The model implies, among others, the following things:

- Even though one processor is occupying a memory, it is possible to enqueue access requests via the other bus.

- If two processors access separate memory banks simultaneously, the remaining processors will have to wait for an available bus.

- The following *invariants* hold for all reachable markings $M \in \{M_0\}[\rangle$ of the net:

$$\begin{aligned} M(p_1)+M(p_2)+M(p_3)+M(p_4) &= M_0(p_1)+M_0(p_2)+M_0(p_3)+M_0(p_4) \\ M(p_4)+M(p_5) &= M_0(p_4)+M_0(p_5) \end{aligned}$$

Example: A multi-processor system (4/4): Remarks

- The processors, buses and memory banks are indistinguishable (no *identity*).
- The number of processors can be varied by modifying the initial marking.
- Can the number of buses be altered this easily?
- The model does not reflect the number of memory banks.
- By defining firing probabilities or durations for the transitions, it is possible to evaluate the performance of the system: what is the average number of processors waiting for a bus, or how long does it take to serve a request.

Reachability Graph

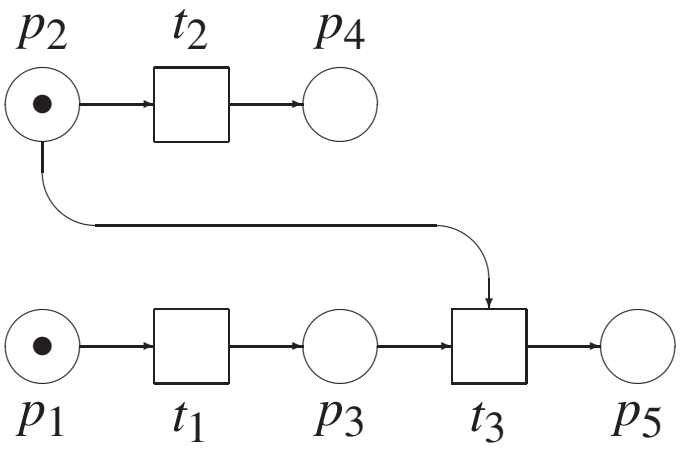
The *reachability graph* of a place/transition system $\langle S, T, F, W, M_0 \rangle$ is a rooted, directed and arc-labelled graph $G = \langle V, E \subseteq (V \times T \times V), v_0 \in V \rangle$ where

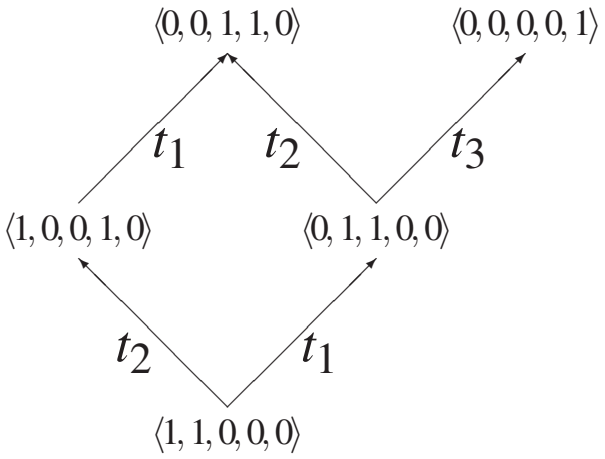
- $V = [\{M_0\}\rangle$ is the set of graph nodes,
- $v_0 = M_0$ is the root node and
- $E = \left\{ \langle M, t, M' \rangle \;\middle|\; M \in V \wedge M' = [M\rangle t \right\}$ is the set of edges.

The edges, leading from one marking to another, are labelled with the name of the firing transition.

In practice, all analysis tools and methods require the computation of the reachability graph. The graph depicts a parallel or distributed system as a regular automaton. In the graph, it is hard to recognise independent or concurrent actions.

Reachability Graph: Example





- The weight of each arc is 1.
- Often, the marking $\langle 1,1,0,0,0\rangle$ is denoted $\{p_1, p_2\}$.
- All *interleavings* of the mutually independent transitions t_1 and t_2 are shown in the graph. If concurrent steps are identified, the graph would be augmented with the edge $\{p_1, p_2\} \xrightarrow{t_1,t_2} \{p_3, p_4\}$.

Computing the reachability graph

REACHABILITY-GRAPH($\langle S, T, F, W, M_0 \rangle$)

1 $G : \langle V, E, v_0 \rangle \leftarrow \langle \{M_0\}, \emptyset, M_0 \rangle$;
2 $Markings$: stack $\leftarrow \langle M_0 \rangle$;
3 **while** $Markings \neq \langle\rangle$
4 **do** $M \leftarrow Markings$.pop();
5 **for** $t \in$ enabled(M)
6 **do** $M' \leftarrow$ fire(M, t);
7 **if** $M' \notin V$
8 **then** $V \leftarrow V \cup \{M'\}$
9 $Markings$.push(M');
10 $E \leftarrow E \cup \{\langle M, t, M' \rangle\}$;
11 **return** G;

The algorithm makes use of two functions:

- enabled$(M) := \{t \mid M[t\rangle\}$
- fire$(M, t) := [M\rangle t$

By replacing the search stack with a search queue, the graph will be traversed breadth first instead of depth first. Breadth first search will find the shortest transition path from the initial marking to an erroneous marking. Some applications require depth first search.

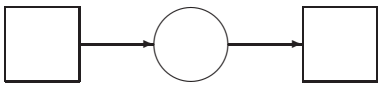
The size of the reachability graph

How many nodes can the reachability graph of a place/transition system contain? The maximum is the number of potential markings of the system.

If the system is known to limit the number of tokens in place p_i to at most m_i, there can be at most $\prod_{i=1}^{n} m_i = m_1 \cdot m_2 \cdot \ldots \cdot m_n$ distinct markings (and graph nodes).

A reachability graph can contain at most $|E| \leq |V| \cdot |T| \cdot |V|$ edges, one for each combination of two nodes and a transition.

In the general case, there is no upper limit for the number of tokens in a place, and there may be an infinite number of reachable markings, as below:



The finiteness of the reachability graph

If each place of a place/transition system can contain at most k tokens in each reachable marking, the system is said to be *k-safe*.

A k-safe system has at most $(k+1)^n$ markings; for 1-safe systems, the limit is 2^n.

The reachability graph can be detected to be infinite by introducing the concept of *coverability* ($M \leq M'$ if $\forall s \in S : M(s) \leq M'(s)$) and by modifying the computation algorithm so that if a successor marking covers a previous marking on the same transition path, the graph is infinite.

Computational complexity

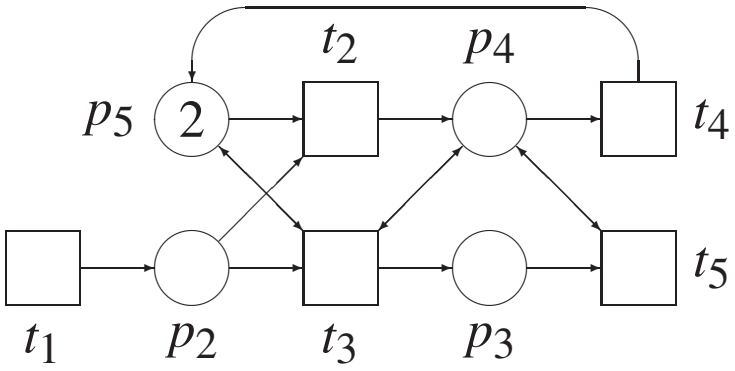
Even though a place/transition system can have an infinite number of reachable markings, the problem of reachability (whether any reachable marking satisfies the given condition) is decidable ($PSPACE$-complete for 1-safe systems and $EXPSPACE$-hard for others).

An extension to place/transition nets are *inhibitor arcs*. An inhibitor arc leading from a place to a transition inhibits the transition to become enabled if the place contains at least as many tokens as the weight of the arc indicates. As these systems can simulate Turing machines, their reachability problem is undecidable.

The general decidability of the reachability problem has more theoretical than practical implications. In practice, a problem is undecidable if the memory or disk capacity is exceeded or a search takes days. Some problems can be solved in an easier way, if the system is described using an “easier” theoretical model.

Complement places

Sometimes one would like to limit the number of tokens that can enter a place. For instance, a model could be restricted to serve at most k clients at a time. The most natural way is to define a "closed flow" of tokens with a *complement place*.

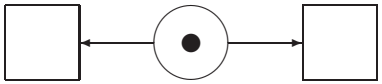


The arrival of a client is modelled by the firing of the transition t_1. Transitions t_4 and t_5 model the departure of clients. The firings of t_1 can be limited by introducing a place as its input place. This place would be made the output place of t_4 and t_5. The initial marking of this place denotes the number of clients circulating in the system.

Conflict and confusion

Transitions are *independent* if and only if they do not share input or output places. In this case, they may fire in arbitrary order or simultaneously. The transitions are *concurrent*.

A *conflict* may arise among transitions that share input places:

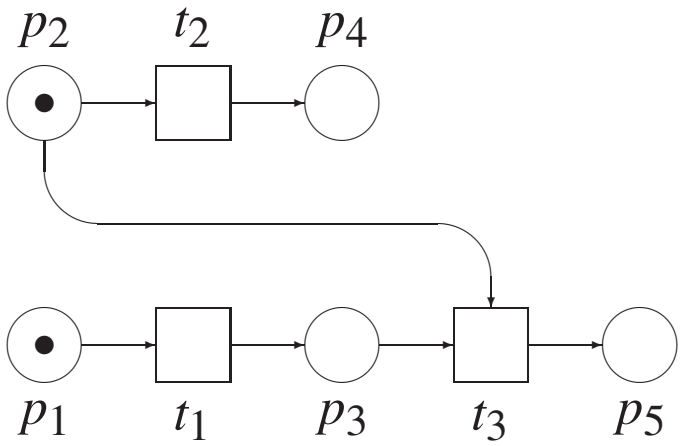


Only one of the transitions may fire. Conflicts denote nondeterminism.

A situation where the firing order of concurrent transitions may affect the amount of arising conflicts is called *confusion*.

Confusion: Example

Let us assume that the system is monitored by two observers (O_1 and O_2) who observe the firing of one transition at a time.

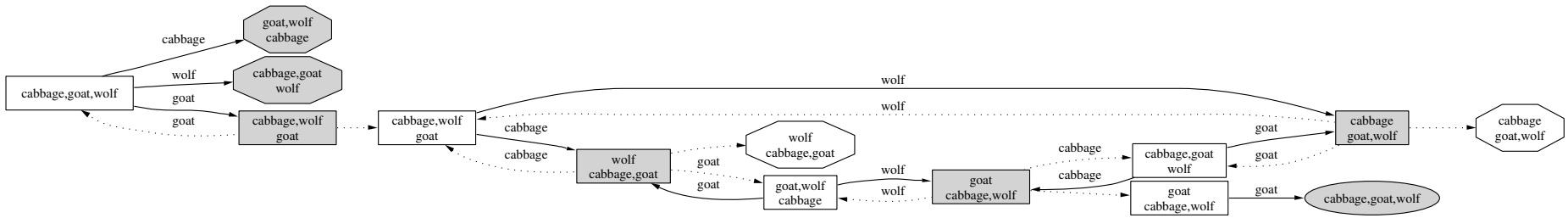


- O_1 observes that t_1 and t_2 fire in said order.
- O_2 observes that t_2 and t_1 fire in said order.
- O_1 observes a conflict between t_2 and t_3, while O_2 sees no conflict.

Ice-breaker: Solving logical puzzles

Not only parallel or concurrent systems can be modelled with place/transition systems. There is an old problem involving a ferry-man, a cabbage, a goat and a wolf who have to cross a river. The boat carries the ferry-man and one other thing. Without surveillance, the goat would eat the cabbage, and the wolf would eat the goat.

A model is constructed that describes possible actions (the ferry-man crossing the river alone or carrying something) and forbidden states (where something can be eaten). The reachability graph of the model contains all possible solutions.



Marko Mäkelä