

# Chapter 17

## Reachability

We now consider a fundamental question for all high-level formalisms, but phrase the question in terms of Petri nets.

**The reachability problem:** Given a Petri net with an initial marking  $\mathbf{m}_0$ , is it possible to (eventually) reach a specified marking  $\mathbf{m}$ ?

This problem is known to be decidable, but requires exponential space and time in the worst case.

### 17.1 Coverability

**Definition 17.1** *Given a Petri net, a marking  $\mathbf{m}'$  covers another marking  $\mathbf{m}$  if*

$$\mathbf{m}'[p] \geq \mathbf{m}[p]$$

*for all places  $p \in \mathcal{P}$ .*

Note that a marking covers itself.

We will write  $\mathbf{m}' \geq \mathbf{m}$  if  $\mathbf{m}'$  covers  $\mathbf{m}$ .

We will write  $\mathbf{m}' > \mathbf{m}$  if  $\mathbf{m}'$  covers  $\mathbf{m}$  and  $\mathbf{m}' \neq \mathbf{m}$ .

**Definition 17.2** *Given a Petri net, a marking  $\mathbf{m}$  is said to be coverable if there exists a marking  $\mathbf{m}' \geq \mathbf{m}$  that is reachable from the initial marking  $\mathbf{m}_0$ .*

#### 17.1.1 The coverability tree

The *coverability tree* is a tree of markings that cover all markings reachable from the initial marking. Within these markings, we use a special symbol,  $\omega$ , to indicate “unbounded number of tokens”. For any integer  $n$ , we have  $\omega > n$ ,  $\omega + n = \omega - n = \omega$ , and  $\omega \geq \omega$ . The coverability tree can be constructed using the following algorithm:

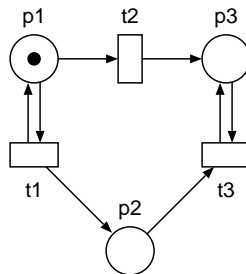
### Algorithm 17.1 Coverability Tree Construction

```

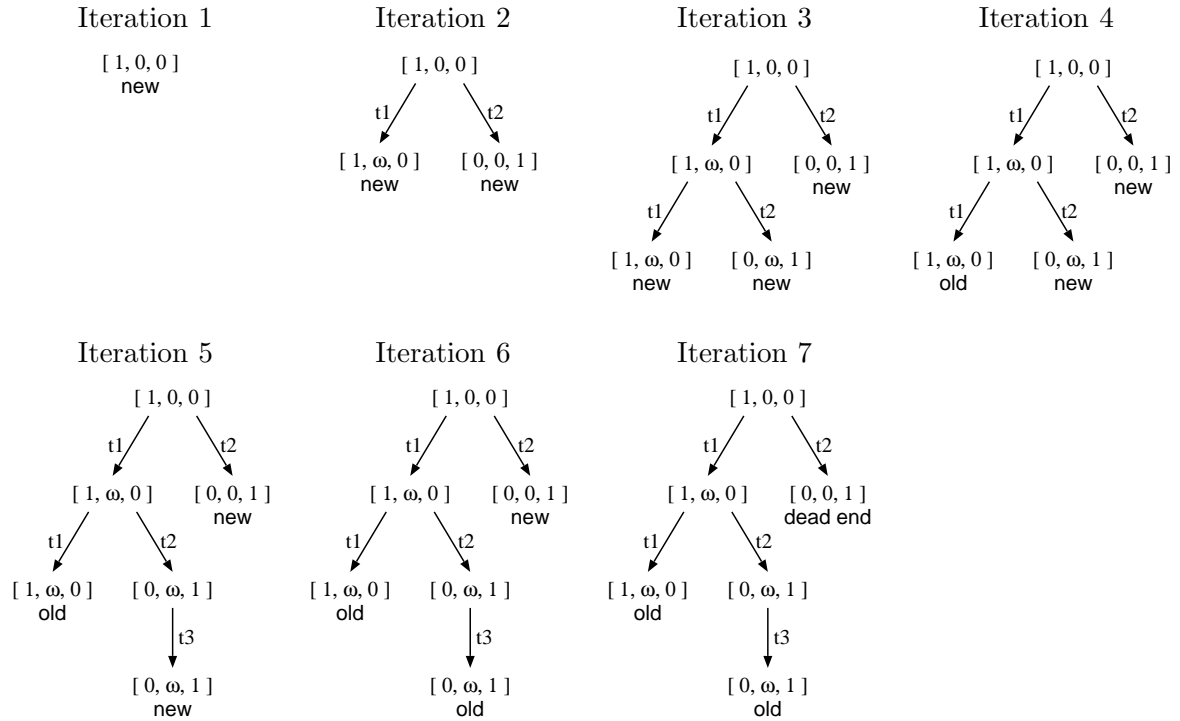
Set  $m_0$  as the root of the tree; tag it as "new".
while  $\exists$  "new" markings
  Select some "new" marking  $m$ ;
  Give  $m$  a blank tag;
  if  $\exists m'$ , on the path from  $m_0$  to  $m$ , with  $m' = m$  then
    Tag  $m$  as "old";
    continue;
  endif
  if no transitions are enabled in  $m$  then
    Tag  $m$  as "dead end";
    continue;
  endif
  for each transition  $t$  enabled in  $m$  do
     $m' \leftarrow$  the marking reached from  $m$  by firing  $t$ 
    if  $\exists m'' > m'$  on the path from  $m_0$  to  $m'$ 
      for each  $p$  such that  $m'[p] > m''[p]$  do
         $m'[p] \leftarrow \omega$ ;
      endfor
    endif
     $m'$  is a new node in the tree;
    Draw an arc from  $m$  to  $m'$  with label  $t$ ;
    Tag  $m'$  as "new";
  endfor
endwhile

```

### Example 17.1



Consider the above Petri net, drawn with initial marking  $[1, 0, 0]$ . Note:  $t_1$  can fire arbitrarily many times, until  $t_2$  fires; then,  $t_3$  can fire as many times as  $t_1$  fired. Using the above algorithm, we can construct the coverability tree, drawn below at the beginning of each iteration of the while loop.



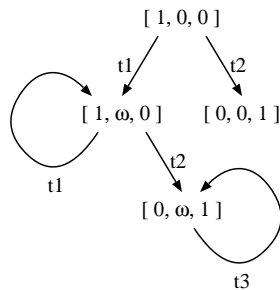
### 17.1.2 The coverability graph

The *coverability graph* is similar to the coverability tree. Nodes in the coverability graph correspond to the unique markings in the coverability tree. In the coverability graph, there is an edge from  $\mathbf{m}$  to  $\mathbf{m}'$ , labeled with transition  $t$ , if

- $t$  is enabled in marking  $\mathbf{m}$ , and
- the firing of  $t$  in  $\mathbf{m}$  leads to marking  $\mathbf{m}''$  with  $\mathbf{m}' \geq \mathbf{m}''$ .

#### Example 17.2

The coverability graph for the previous example is shown below.



The coverability graph (or tree) does *not* contain enough information to definitively solve the reachability problem:

- A marking  $\mathbf{m}$  is *not* reachable if there is no marking  $\mathbf{m}' \geq \mathbf{m}$  in the coverability graph.

- A marking  $\mathbf{m}$  *may be* reachable if there is a marking  $\mathbf{m}' \geq \mathbf{m}$  in the coverability graph.

The inability to say for certain that a marking is reachable is due to the loss of information that occurs due to the symbol  $\omega$ .

## 17.2 The reachability graph

We can simplify the coverability graph algorithm by eliminating the symbol  $\omega$ , and storing the actual markings reached. This gives us instead the *reachability graph*, whose set of nodes is equal to the markings that can be reached from the initial marking  $\mathbf{m}_0$  in zero or more steps. This set of markings is called the *reachability set*. The reachability graph can be constructed using the following algorithm.

**Algorithm 17.2** Reachability graph construction

```

 $\mathcal{U} \leftarrow \{\mathbf{m}_0\};$            // set of unexplored markings
 $\mathcal{S} \leftarrow \{\mathbf{m}_0\};$        // reachability set so far
 $\mathcal{E} \leftarrow \emptyset;$          // edges in the reachability graph
while  $\mathcal{U} \neq \emptyset$ 
    Remove some marking  $\mathbf{m}$  from  $\mathcal{U}$ ;
    for each transition  $t$  enabled in  $\mathbf{m}$  do
         $\mathbf{m}' \leftarrow$  the marking reached from  $\mathbf{m}$  by firing  $t$ ;
        if  $\mathbf{m}' \notin \mathcal{S}$  then
            Add  $\mathbf{m}'$  to  $\mathcal{S}$ ;
            Add  $\mathbf{m}'$  to  $\mathcal{U}$ ;
        endif
        Add edge from  $\mathbf{m}$  to  $\mathbf{m}'$  with label  $t$  to  $\mathcal{E}$ ;
    endfor
endwhile

```

Note:

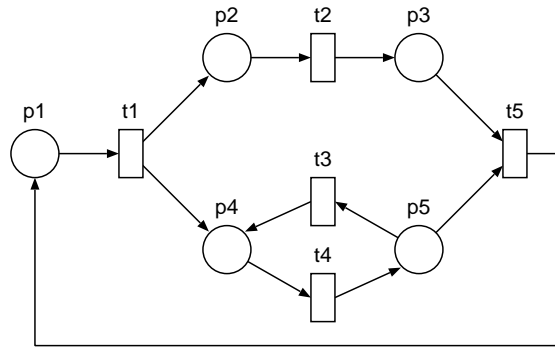
- The algorithm terminates iff the reachability set is finite.
- If markings can be added to and removed from sets  $\mathcal{U}$  and  $\mathcal{S}$  in constant time (not a realistic assumption), then the algorithm has complexity  $\mathcal{O}(|\mathcal{S}| \cdot |\mathcal{T}|)$ , if we check each transition in each marking.
- If markings are removed from  $\mathcal{U}$  in FIFO order, we get “breadth-first” search.
- If markings are removed from  $\mathcal{S}$  in LIFO order, we get “depth-first” search.
- The reachability graph contains enough information to answer any reachability question, and can be used to determine a firing sequence from  $\mathbf{m}_0$  to any reachable marking  $\mathbf{m}$ .

Differences between the coverability graph and the reachability graph:

- The coverability graph can be used for unbounded Petri nets; the reachability graph will be infinite in this case.

- If the coverability graph contains no markings with  $\omega$ , then it is equal to the reachability graph, and the reachability graph is finite.
- If the reachability graph is infinite, then the coverability graph will contain a marking with  $\omega$ .

**Example 17.3**

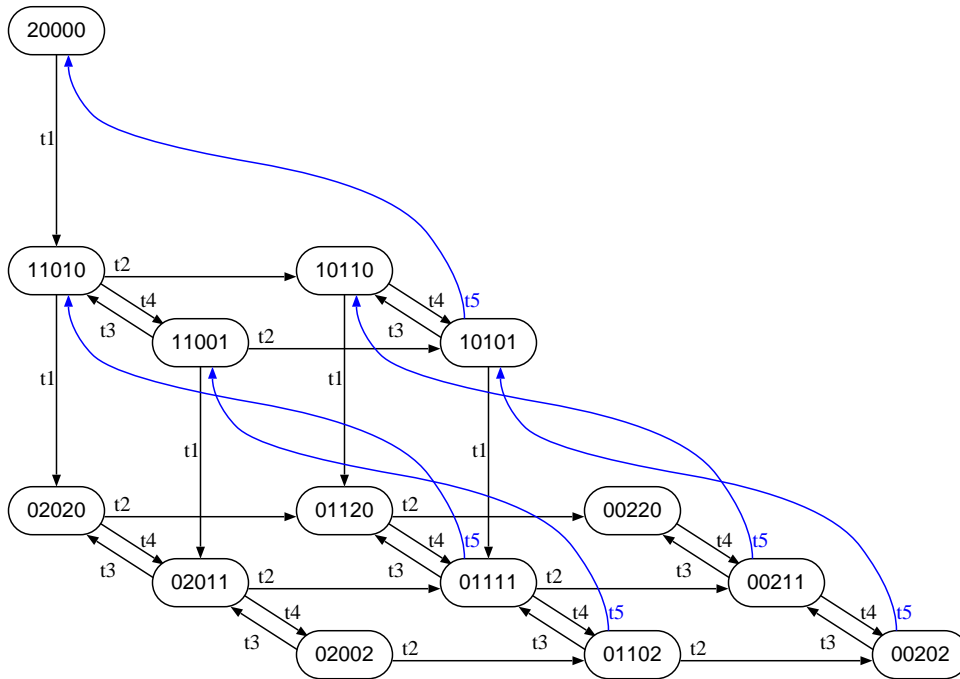


For the above Petri net, with initial marking

$$\begin{array}{ccccc}
 p_1 & p_2 & p_3 & p_4 & p_5 \\
 [ & 2 & 0 & 0 & 0 & ]
 \end{array}$$

construct the reachability graph.

Using the reachability graph construction algorithm and some patience, we obtain the following graph.



## 17.3 CTL model checking

If the choice of which Petri net transition to fire is non-deterministic, and assuming the reachability graph is finite, then a (high-level) Petri net model can be used to describe a (low-level) finite state machine, where the finite state machine is simply the reachability graph of the Petri net. We can then write CTL formulas, where atomic propositions are described in terms of the Petri net itself, and check them automatically:

1. Generate the reachability graph of the Petri net.
2. Perform CTL model checking as usual, on the reachability graph.

### Example 17.4

Does the Petri net in the previous example satisfy  $AF(\text{Place } p_1 \text{ is empty})$ ?

First, rewrite the expression in terms of EG:

$$AF(\text{Place } p_1 \text{ is empty}) \equiv \neg EG(\text{Place } p_1 \text{ is not empty})$$

The expression “Place  $p_1$  is not empty” is an atomic proposition, because we can easily determine if a given state (marking) satisfies it or not. This is satisfied by states  $\{[20000], [11010], [11001], [10110], [10101]\}$ . Upon inspection of the reachability graph, it should be clear that the expression

$$EG(\{[20000], [11010], [11001], [10110], [10101]\})$$

is also satisfied by the same set of states. Finally, taking the complement of this set, we obtain the set of states satisfying  $AF(\text{Place } p_1 \text{ is empty})$ , which is the set

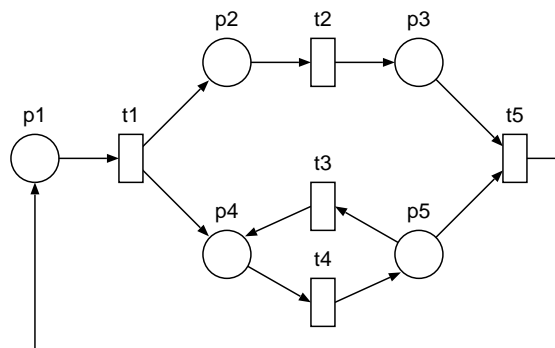
$$\{[02020], [02011], [02002], [01120], [01111], [01102], [00220], [00211], [00202]\}.$$

Since this set does not contain the initial marking, we conclude that the Petri net does *not* satisfy  $AF(\text{Place } p_1 \text{ is empty})$ .

## 17.4 State explosion

High level models (including Petri nets) often produce an extremely large number of reachable states, even for “small” models. This is known as the “state explosion problem”. We will discuss ways to cope with this problem later in the semester.

### Example 17.5



For the above Petri net, with initial marking

$$\begin{matrix} p_1 & p_2 & p_3 & p_4 & p_5 \\ [ & N & 0 & 0 & 0 & 0 & ] \end{matrix}$$

how many reachable states are there?

In general, this question requires to construct the reachability graph. However, this Petri net is simple enough that we can answer this question without generating the reachability graph. Note that, if transition  $t_1$  fires  $n$  more times than transition  $t_5$ , then

- The sum of tokens in places  $p_2$  and  $p_3$  is exactly  $n$ . There are  $n + 1$  ways that this can occur.
- Similarly, the sum of tokens in places  $p_4$  and  $p_5$  is exactly  $n$ .
- The above two sums are completely independent.

Therefore, the number of reachable markings, given that transition  $t_1$  has fired  $n$  more times than transition  $t_5$ , is  $(n+1)^2$ . For the given initial marking, the number of times  $t_1$  may fire, before  $t_5$  fires, is  $0, 1, 2, \dots, N$ . The number of reachable markings is therefore

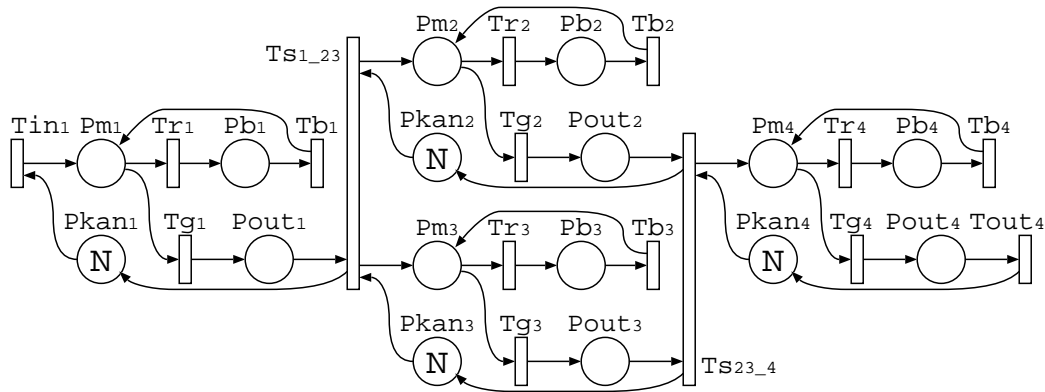
$$\sum_{n=0}^N (n+1)^2 = \sum_{n=0}^N n^2 + 2 \sum_{n=0}^N n + \sum_{n=0}^N 1 = \dots = \frac{(2N+3)(N+2)(N+1)}{6}$$

In particular, note that (as a sanity check):

- If  $N = 0$ , the number of markings is  $(3 \cdot 2 \cdot 1)/6 = 1$
- If  $N = 2$ , the number of markings is  $(7 \cdot 4 \cdot 3)/6 = 14$

The size of the reachability set grows as  $\mathcal{O}(N^3)$ .

### Example 17.6



The above Petri net is a model of a kanban manufacturing system, comprised of four components. In each component, a part is processed, which is either successful and the part can move on to the next stage, or unsuccessful, which requires additional work to “un-do” the previous processing. Component 1 is used by components 2 and 3, and components 2 and 3 together are used by component 4. The initial marking specifies

the initial number of raw parts  $N$  for each component. Using an analysis similar to the one in the previous (smaller) example, it can be shown that the number of reachable markings is exactly

$$|\mathcal{S}| = \frac{(N+1)^3(N+2)^3(N+3)^3(3N^2+12N+10)}{2160}$$

which grows as  $\mathcal{O}(N^{11})$ .