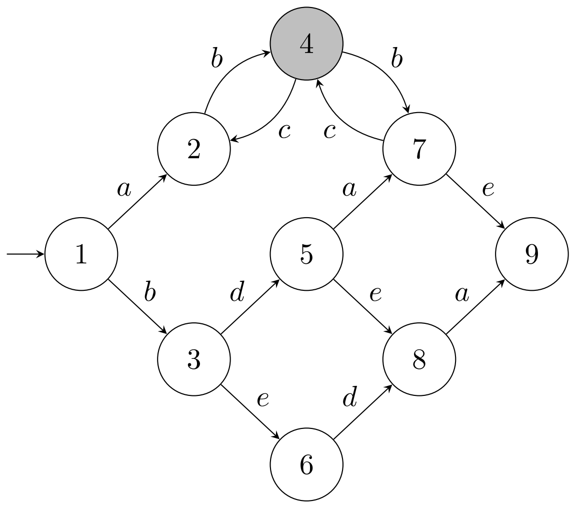
TD 7: Emptiness Test for Büchi Automata, Partial-Order Reduction

Exercise 1. Consider the labeled Kripke structure $\mathcal{K}$ shown below with actions $\{a, b, c, d, e\}$ and one atomic proposition q, where q holds only on state 4.



1. Determine a maximal independence relation I.

   (Recall that $I \subseteq A \times A$ is an independance relation for $\mathcal{K}$ if it is irreflexive, symmetric, and for all $(a, b) \in I$ and $s \in S$, if $a, b \in en(s)$, $s \xrightarrow{a} t$, and $s \xrightarrow{b} u$, then there exists $v \in S$ such that $a \in en(u)$, $b \in en(t)$, $t \xrightarrow{b} v$, and $u \xrightarrow{a} v$.)

2. Determine the maximal invisibility set U.

   (Recall that U is an invisibility set if for all $a \in U$ and $(s, a, s') \in \rightarrow$, $\nu(s) = \nu(s')$.)

Exercise 2 (Büchi Emptiness Test)**.** Consider an execution of Algorithm 1 on some Büchi automaton $\mathcal{B} = (\Sigma, S, s_0, \delta, F)$.

At each point during the DFS, we define the *search path* as the sequence of visited states for which the DFS call has not yet terminated (in the order in which they are visited), and the *explored graph* of $\mathcal{B}$ as the subgraph containing all visited states and explored transitions. We call an SCC of the *explored* graph *active* if the search path contains at least one of its states. A state is *active* if it is part of an active SCC in the explored graph (it is not necessary for the state itself to be on the search path). The *active graph* is the subgraph of the explored graph induced by the active states.

For all strongly connected component $C \subseteq S$ of $\mathcal{B}$, we call *root of* C the state of C that is visited first during the DFS, i.e. the node r_C such that $r_C.num = \min\{s.num \mid s \in C\}$ at the end of the DFS. We define similarly the root of an SCC in the explored graph.

Algorithm 1 Depth-first-search

```
nr = 0;
hash = { };
dfs(s0);
exit;
```

dfs(s) :

```
add s to hash;
nr = nr + 1;
s.num = nr;
for all t ∈ succ(s) do
   if t not in hash then
      dfs(t)
   end if
end for
```

1. Show that an inactive SCC in the explored graph is also an SCC of $\mathcal{B}$.
2. Show that the roots of the SCCs in the active graph are a subsequence $r_1 \dots r_m$ of the search path, and that an activated node s is in the active SCC of r_i if and only if $i < m$ and $r_i.num \leq s.num < r_{i+1}.num$, or $i = m$ and $r_i.num \leq s.num$.
3. Show that Algorithm 2 maintains the following invariants:
   - the stack W contains the sequence $(r_1, C_1) \dots (r_m, C_m)$ where $r_1 \dots r_m$ is the sequence of roots of the active graph, and C_i is the active SCC of r_i,
   - for all nodes s, $s.active$ is *true* if and only if s is active.
4. Show that Algorithm 2 returns *true* iff the language of the input Büchi automaton is empty, and that in that case, it terminates as soon as the explored graph contains a counterexample.
5. Adapt Algorithm 2 to test emptiness of a generalized Büchi automaton with acceptance sets $F_1, \dots, F_n$.
6. Compare with the nested DFS algorithm from the lectures.

Algorithm 2 Emptiness Test

```
nr = 0;
hash = { };
W = { };
dfs(s_0);
return true;
```

dfs(s):

```
add s to hash;
s.active = true;
nr = nr + 1;
s.num = nr;
push (s, {s}) onto W;
for all t ∈ succ(s) do
   if t not in hash then
      dfs(t)
   else if t.active then
      D = { };
      repeat
         pop (u, C) from W;
         if u is accepting then
            return false
         end if
         merge C into D;
      until u.num ≤ t.num;
      push (u, D) onto W;
   end if
end for
if s is the top root in W then
   pop (s, C) from W;
   for all t in C do
      t.active = false
   end for
end if
```