TD 9: Pushdown Systems

Exercise 1 (Labelled Pushdown Systems)**.** Let $\mathcal{P} = (P, \Gamma, \Delta, \Sigma)$ be a labelled pushdown system, i.e. the rules in Δ are of the form $pA \overset{a}{\hookrightarrow} qw$, where $p, q \in P$ are control locations, $A \in \Gamma$ and $w \in \Gamma^*$ are stack symbols, and additionally $a \in \Sigma$ is an *action*. The set of configurations $Con(\mathcal{P})$ consists of the tuples qw with $q \in P$ and $w \in \Gamma^*$. For two configurations c, c' we write $c \overset{w}{\Rightarrow} c'$, where $w \in \Sigma^*$, if c can be transformed into c' by a sequence of rules whose labels yield w.

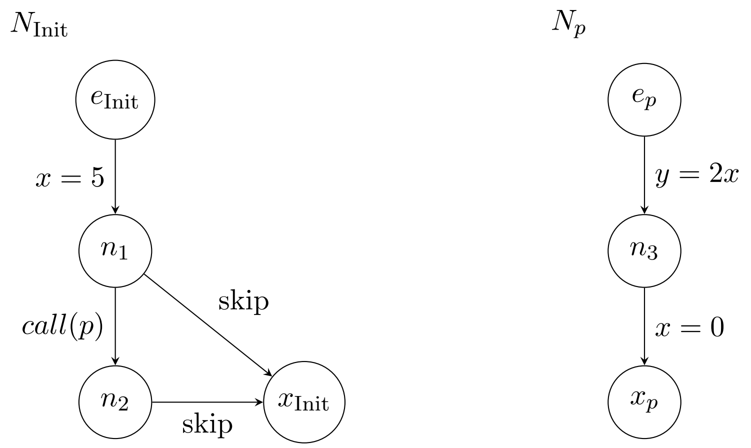
Given a regular set of configurations C, it is known how to compute $pre^*(C) = \{ c \in Con(\mathcal{P}) \mid \exists c' \in C, w \in \Sigma^* : c \overset{w}{\Rightarrow} c' \}$. If C is accepted by an automaton with n states, this takes $\mathcal{O}(n^2 \cdot |\Delta|)$ time.

1. Let $L \subseteq \Sigma^*$ be a regular language and C be a regular set of configurations. We define
$$pre^*[L](C) := \{ c \in Con(\mathcal{P}) \mid \exists c' \in C, w \in L : c \overset{w}{\Rightarrow} c' \}.$$
One can prove that $pre^*[L](C)$ is regular. Describe how to compute a finite automaton accepting $pre^*[L](C)$.
2. Give a bound on the amount of time it takes to compute $pre^*[L](C)$.

Exercise 2 (Dickson's Lemma)**.** A *quasi-order* $(A, \leq)$ is a set A endowed with a reflexive and transitive ordering relation $\leq$. A *well quasi order* (wqo) is a quasi order $(A, \leq)$ s.t., for any infinite sequence $a_0 a_1 \cdots$ in A^ω, there exist indices $i < j$ with $a_i \leq a_j$.

1. Let $(A, \leq)$ be a wqo and $B \subseteq A$. Show that $(B, \leq)$ is a wqo.
2. Show that $(\mathbb{N} \uplus \{\omega\}, \leq)$ is a wqo.
3. Let $(A, \leq)$ be a wqo. Show that any infinite sequence $a_0 a_1 \cdots$ in A^ω embeds an infinite non-decreasing subsequence $a_{i_0} \leq a_{i_1} \leq a_{i_2} \leq \cdots$ with $i_0 < i_1 < i_2 < \cdots$.
4. An element $x \in X \subseteq A$ is minimal in X if every $y < x$ is outside X. Show that $(A, \leq)$ is a wqo if and only if every subset $X \subseteq A$ has at least one but only finitely many minimal elements in X.
5. Let $(A, \leq_A)$ and $(B, \leq_B)$ be two wqo's. Show that the cartesian product $(A \times B, \leq_\times)$, where the product ordering is defined by $(a, b) \leq_\times (a', b')$ iff $a \leq_A a'$ and $b \leq_B b'$, is a wqo.

Exercise 3 (Data-flow Analysis)**.** We consider a problem from interprocedural data-flow analysis. A program consists of a set *Proc* of procedures that can execute and recursively call one another. The behaviour of each procedure p is described by a flow graph, an example with two procedures is shown below.



Formally, a flow graph for procedure $p \in \mathit{Proc}$ is a tuple $G_p = (N_p, A, E_p, e_p, x_p)$, where

- N_p are the nodes, corresponding to program locations; we denote $N := \bigcup_{p \in \mathit{Proc}} N_p$.
- $A = A_I \cup \{\, \mathit{call}(p) \mid p \in \mathit{Proc} \,\}$ are the actions, where A_I are *internal actions* (such as assignments etc); additionally an action can call some procedure. A is identical for all procedures.
- $E_p \subseteq N_p \times A \times N_p$ are the edges, labelled with actions from A. We denote $E := \bigcup_{p \in \mathit{Proc}} E_p$.
- e_p is the *entry point* of procedure p, i.e. when p is called, execution will start at e_p.
- x_p is the *exit point* of p (without any outgoing edges); when x_p is reached, p terminates and execution resumes at last call site of p.

1. Construct a labelled pushdown system with one single control location that expresses the behaviour of the procedures in *Proc*.

Suppose that the internal actions in A_I describe assignments to global variables, i.e. they are of the form $v := \mathit{expr}$, where v is a variable and expr the right-hand-side expression. If v is a variable, then $D_v \subseteq A_I$ is the set of actions that assign a value to v and $R_v \subseteq A_I$ the set of actions where v occurs on the right-hand side.

Let $\mathit{Init} \in \mathit{Proc}$ be an initial procedure and $n \in N$ a node in the flow graph. We say that variable v is *live* at n if there exists a node n' and an execution that (i) starts at e_{Init}, (ii) passes n, (iii) finally reaches n' with an action from R_v, and (iv) there is no assignment to v between n and n' in this execution. (Intuitively, this means that the value that v has at n matters for some execution; this is used in compiler construction to determine whether an optimizing compiler may "forget" the value of v at n.) For instance, in the shown example, the variable x is live at n_1 and e_p, but not in the other nodes.

2. Describe a regular language $L \subseteq A^*$ that describes the sequences of actions that can happen along such executions between n and n'.
3. Describe how, given a variable v, one can compute the set of nodes n such that v is live at n.

Exercise 4 (Multi-Pushdown Systems)**.** An n-dimensional *multi-pushdown system (n-MPDS)* is a tuple $\mathcal{M} = (P, \Gamma, (\Delta_i)_{0<i\leq n})$ where $n \geq 1$ is the number of stacks, P a finite set of control states, Γ a finite stack alphabet, and each $\Delta_i \subseteq P \times \Gamma \times P \times \Gamma^*$ is a finite transition relation. A configuration of an n-MPDS is a tuple $c = (q, w_1, \ldots, w_n)$ in $P \times (\Gamma^*)^n$. The *transition relation* $\to$ on configurations is defined as $\to = \bigcup_{0<i\leq n} \to_i$, where

$$(q, w_1, \ldots, Aw_i, \ldots, w_n) \to_i (q', w_1, \ldots, w'_i w_i, \ldots, w_n) \quad \text{iff} \quad qA \hookrightarrow q' w'_i w_i \in \Delta_i$$

1. Show that the *control state reachability problem*, i.e. given an initial configuration c in $P \times \Gamma^n$ and a control state $p \in P$, whether there exist $w_1, \ldots, w_n$ such that $c \to^* (p, w_1, \ldots, w_n)$ is undecidable as soon as $n \geq 2$.
2. Let us consider a restriction on $\to^*$: *k-bounded* runs are defined as the k-iterates $c \Rightarrow^k c'$ of the relation

   $$c \to c' \qquad \text{iff} \qquad \exists i.\ c \to_i^* c'$$

   i.e. a k-bounded run can be decomposed into k subruns where a single PDS is running.

   Show that the k-bounded control-state reachability problem, i.e. given an initial configuration c in $P \times \Gamma^n$ and a control state $p \in P$, whether there exist $w_1, \ldots, w_n$ such that $c \Rightarrow^k (q, w_1, \ldots, w_n)$ is decidable.