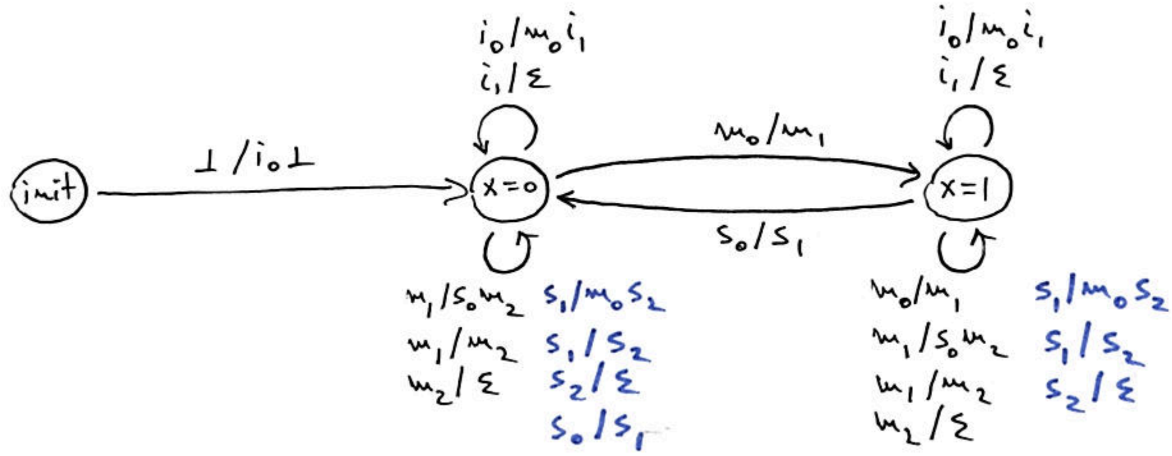
11.1.

```
void m() {
m0 x=1;
m1 if (r()==1) s();
m2 }
```

```
void s() {
s0 x=0;
s1 if (r()==1) m();
s2 }
```

```
int x=0;
void main() {
i0 m();
i1 }
```

The program counter can be encoded by the stack and the value of variable by the state of the PDS:



11.2. The method can be summarized by three steps:

- compute A such that $C = CF(A)$
- duplicate initial states
- compute <u>one</u> more pre step

Computing <u>one single</u> pre step follows the „intuition" mentioned in the slides:

„configuration $(q_1, \gamma w')$ is an immediate predecessor of (q_2, ww') wrt. $q_1 \xrightarrow{\gamma/w} q_2$ so if ww' is accepted from s_{q_2} by

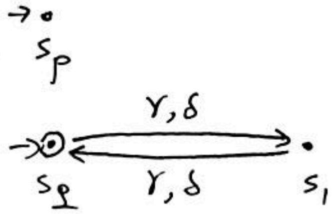
$$s_{q_2} \xrightarrow{w}_i s \xrightarrow{w'}_i s_F \in S_F$$

then the new transition accepts $\gamma \cdot w'$ from s_{q_1}:

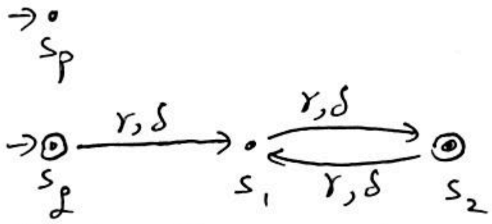
$$s_{q_1} \xrightarrow{\gamma}_{new} s \xrightarrow{w'}_i s_F \in S_F \text{ ."}$$

So in the 3rd step above we add $s^{pre}_{q_1} \xrightarrow{\gamma} s$ when $s_{q_2} \xrightarrow{w} s$ in A and $q_1 \xrightarrow{\gamma/w} q_2$ in the PDS.

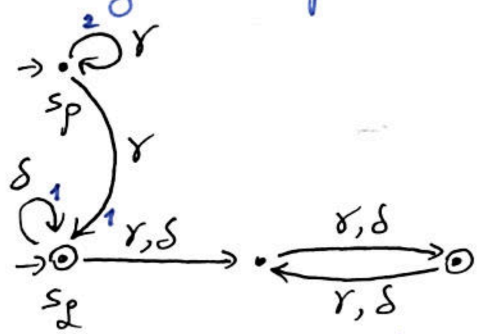
11.3. The P-NFA A_0 s.t. $CF(A_0) = C$ is



After unrolling (i.e. no incoming transitions to initial states) we get the P-NFA

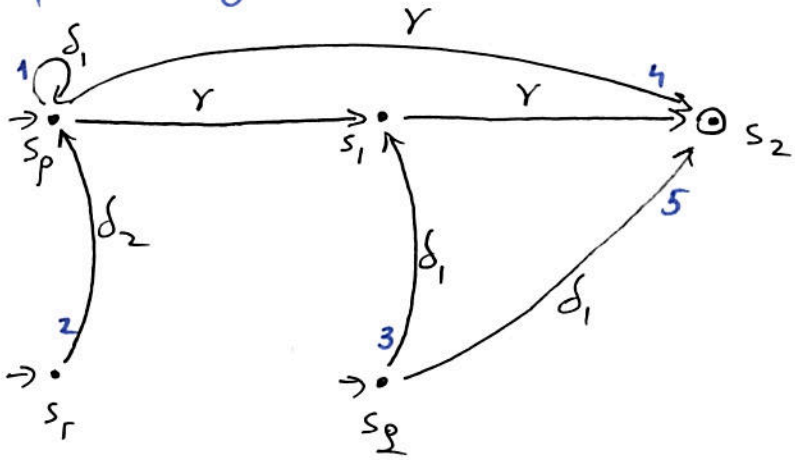


Applying the pre^* algorithm produces the P-NFA below:



Transitions labelled by 1 are added in a 1^{st} step by the <u>lazy</u> algorithm.

11.4. A_{pre^*} for the given P-NFA and PDS is



Note: $s_{\varrho_1} \xrightarrow{\gamma} s$ is added when $\varrho_1 \xrightarrow{\gamma / w} \varrho_2$
and $s_{\varrho_2} \xrightarrow{w} s \xrightarrow{w'} s_F \in S_F$.