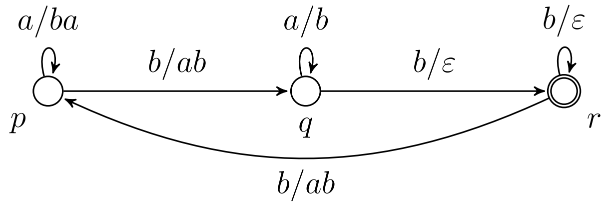
Prof Roland Meyer	**Advanced Automata Theory** **Exercise Sheet 9**	TU Braunschweig
Dr Prakash Saivasan		Summer term 2016

Out: June 13 Due: June 19, 10:00

Exercise 1: Büchi Pushdown Systems

Solve the accepting run problem for the Büchi-pushdown system over $\Gamma = \{a, b\}$ below:



(a) Find some/all $(s, \gamma) \in Q \times \Gamma$ such that $(s, \gamma) \rightarrow^+ (r, u) \rightarrow^* (s, \gamma v)$ for some $u, v \in \Gamma^*$.

(b) Compute $A_{\text{pre}^*(C)}$ for $C = \{(s, \gamma v) \mid v \in \Gamma^*, (s, \gamma) \text{ is a configuration found in (a)}\}$.

Exercise 2: Model Checking BPDS

We extend Büchi Pushdown Systems to accept words from a finite input alphabet $\Sigma = \mathbb{P}(\mathcal{P})$ for some finite set of propositions $\mathcal{P}$. The automaton definition now includes an initial configuration c_0 and transitions are now labeled, i.e. they take the form $q \xrightarrow{\gamma/w:\, a} q'$ with $q, q' \in Q$, $\gamma \in \Gamma$, $w \in \Gamma^*$ and $a \in \Sigma$, with the corresponding semantic rule $(q, \gamma v) \xrightarrow{a} (q', wv)$. Note that the constructions presented in the lecture are not affected by this change. The language of such a BPDS P is $L(P) := \{a_0 a_1 a_2 \ldots \mid c_0 \xrightarrow{a_0} c_1 \xrightarrow{a_1} c_2 \xrightarrow{a_2} \ldots \text{ is an accepting run}\}$.

a) Given an NBA A over Σ and a BPDS P over Σ, construct a BPDS $P \parallel A$ over Σ with $L(P \parallel A) = L(A) \cap L(P)$.

b) Given an LTL formula φ and a BPDS P, show that $L(P) \subseteq L(\varphi)$ is decidable and comment on the complexity.

Exercise 3: Modelling Recursive Programs with (B)PDSs

Consider the following pseudo-code:

```
def m() {
    x = 1 - x;
    if(x == input()) {
        s();
        m();
    }
}
```

```
def s() {
    x = 1 - x;
    if(x != input()) {
        m();
        s();
    }
}
```

Here, `x` is a global boolean variable (`1` is true, `0` is false), `input()` randomly returns `0` or `1` (it represents input from the user/environment modelled as non-determinism). Assume we start the program by calling `m()` with `x=0`.

a) Design a PDS that models the given program. Use $\Gamma = \{s, m\}$ to model the call stack.

b) Using a pre^* construction, describe how you would decide that `m` and `s` are always called in alternation.

Exercise 4: pre **computation for PDS**

Consider a PDS P and a P-NFA A.

a) Show how to construct a P-NFA A' with $\text{CF}(A') = \text{CF}(A)$ that has no transition leading to an initial state.

b) Show how to construct a P-NFA A_{pre} with $\text{CF}(A_{\text{pre}}) = \text{pre}(\text{CF}(A))$.
Prove that your construction is correct.