Solution to Homework 11

**Task.** Use the general algorithm to transform the pushdown automaton from Problem 6 into a context-free grammar. Show, step-by-step, how the resulting grammar will generate the word ABAA.

**Solution.** Let us recall how the word ABAA is accepted by this automaton:

<table>
<thead>
<tr>
<th>read</th>
<th>state</th>
<th>stack</th>
<th>A</th>
<th>B</th>
<th>A</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>s</td>
<td>$</td>
<td>$</td>
<td>A</td>
<td>$</td>
<td>$</td>
</tr>
<tr>
<td></td>
<td>w</td>
<td></td>
<td>$</td>
<td>A</td>
<td>$</td>
<td>$</td>
</tr>
<tr>
<td></td>
<td>a₁</td>
<td></td>
<td></td>
<td>A</td>
<td>$</td>
<td>$</td>
</tr>
<tr>
<td>stack</td>
<td>w</td>
<td></td>
<td>$</td>
<td>A</td>
<td>$</td>
<td>$</td>
</tr>
<tr>
<td></td>
<td>a₂</td>
<td></td>
<td></td>
<td>A</td>
<td>$</td>
<td>$</td>
</tr>
<tr>
<td></td>
<td>w</td>
<td></td>
<td>$</td>
<td>A</td>
<td>$</td>
<td>$</td>
</tr>
<tr>
<td></td>
<td>f</td>
<td></td>
<td></td>
<td>A</td>
<td>$</td>
<td>$</td>
</tr>
<tr>
<td></td>
<td>f</td>
<td></td>
<td></td>
<td>A</td>
<td>$</td>
<td>$</td>
</tr>
</tbody>
</table>

We start with the state $s$, we end up in the final state $f$. Thus, the first rule we apply if the rule $S \rightarrow Asf$;

$$ S \rightarrow Asf $$

The first symbol we push is the dollar sign, this dollar sign is popped at the end. Thus, we have the following combination of pop-push rules:

$$ s \rightarrow \varepsilon, \varepsilon \rightarrow $ $w $ $f \rightarrow \varepsilon, \varepsilon \rightarrow $ $f $$

In general, we have the two transitions

$$ p \rightarrow x, \varepsilon \rightarrow t $ $q $$

$$ r \rightarrow y, t \rightarrow \varepsilon $ $s $$

What do we need to plug in instead of $p, q, etc.$ in the general 2-rule picture to come up with this particular picture:

- instead of $p$, we place $s$;
• instead of $q$, we place $w$;
• instead of $s$ and $t$, we place $f$;
• instead of $x$ and $y$, we place $\varepsilon$.

If we make these substitutions in the general rule:

$$A_{ps} \rightarrow xA_{qt}y,$$

we get the rule

$$A_{sf} \rightarrow \varepsilon A_{wf}\varepsilon.$$

Since concatenation with the empty string does not change anything, this means

$$A_{sf} \rightarrow A_{wf}.$$

Thus, the derivation so far takes the following form:

$$S \quad A_{sf} \quad A_{wf}$$

We covered how we push the dollar sign and how we pop it. Let us underline what we have covered:

<table>
<thead>
<tr>
<th>read</th>
<th>A</th>
<th>B</th>
<th>A</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>s</td>
<td>w</td>
<td>$a_1$</td>
<td>w</td>
</tr>
<tr>
<td>stack</td>
<td>$$$</td>
<td>$$$</td>
<td>A</td>
<td>$$$</td>
</tr>
</tbody>
</table>

If we ignore the dollar signs – since we already took care of them – then we see that we have three intermediate states with the empty stack. So, we need to use the transitivity rule, which in this case takes the form

$$A_{wf} \rightarrow A_{wa_1}A_{a_1}wA_{wa_1}A_{a_1}f.$$

Thus, the derivation tree takes the following form:
Transition $A_{wa1}$. In the first transition from $w$ to $a_1$ we have only one rule: $A, \$ \rightarrow \$. In this rule, we pos $\$ and then pop it, so, in effect, nothing changes in the stack. Thus, in this case, the use of this rule is equivalent to $A, \epsilon \rightarrow \epsilon$. Since there is only one rule, according to the general algorithm, we need to add a fictitious rule to form a pair. We ignore the dollar signs that do not change:

This combination leads to the rule $A_{wa1} \rightarrow AA_{a1}a_1B$, i.e., $A_{wa1} \rightarrow AA_{a1}a_1B$.

Here, the remaining transition between $a_1$ and $a_1$ does not include any additional steps, so we can use the rule $A_{a1a1} \rightarrow \epsilon$. So, we get the rule $A_{wa1} \rightarrow A$.

Transition $A_{a1w}$. For the transition between $a_1$ and $w$, we push A and then pop A:

So, we have a transition $A_{a1w} \rightarrow \epsilon A_{ww}B$, i.e., $A_{a1w} \rightarrow A_{ww}B$. Here, the remaining transition between $w$ and $w$ does not include any additional steps, so we can use the rule $A_{ww} \rightarrow \epsilon$. So, we get the rule $A_{a1w} \rightarrow B$.

Transition $A_{wa1}$. The next transition $A_{wa1}$ is the same as the first transition from $w$ to $a_1$, so it corresponds to the rule $A_{wa1} \rightarrow A$.

Transition $A_{a1f}$. In the last transition $A_{a1f}$, we first push A, then pop A:

So, we get the transition $A_{a1f} \rightarrow \epsilon A_{wf}A$, i.e., $A_{a1f} \rightarrow A_{wf}A$. This takes care of the second A, so we get:
Now, if we ignore the second A, we get an intermediate state $a_2$ with an empty stack, so we need to use transitivity $A_{wf} \rightarrow A_{wa_2}A_{a_2f}$.

For $A_{wa_2}$, there is only one rule — in which the stack does not change, so its application is equivalent to $A, \varepsilon \rightarrow \varepsilon$. Since there is only one rule, we pair it with a fictitious trivial rule:

$w \rightarrow A, \varepsilon \rightarrow \varepsilon \quad a_2 \rightarrow A, \varepsilon \rightarrow \varepsilon \quad a_2$\hfill

So, we get $A_{wa_2} \rightarrow AA_aA_a \varepsilon$, i.e., $A_{wa_2} \rightarrow AA_aA_a$. There are no rules for the transition $A_{a_2a_2}$, so we get $A_{a_2a_2} \rightarrow \varepsilon$ and thus, $A_{wa_2} \rightarrow A$.

Finally, for the transition $A_{a_2f}$, we push and pop the last A by using the following rules:

$A, \varepsilon \rightarrow A \quad w \rightarrow A, \varepsilon \rightarrow \varepsilon \quad f$\hfill

Here, we get the rule $A_{a_2f} \rightarrow \varepsilon A_{wa_2} \varepsilon$, i.e., $A_{a_2f} \rightarrow A_{wa_2} \varepsilon$. There are no rules for the transition $A_{wa_2}$, so we get $A_{wa_2} \rightarrow \varepsilon$ and thus, $A_{wa_2} \rightarrow \varepsilon$. So, the transition takes the following form: