Solution to Homework 9

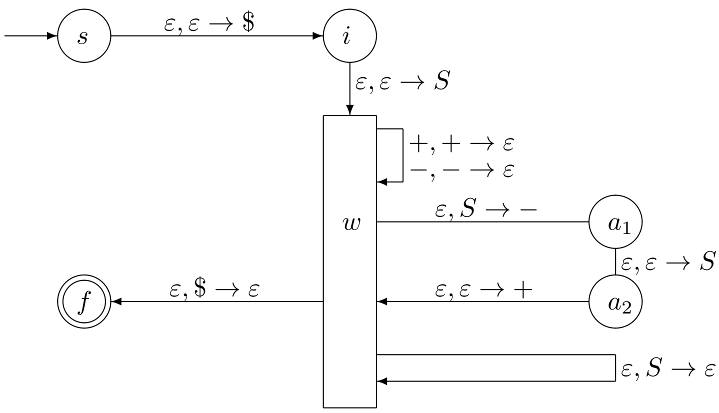
Background. In Problem 7, we considered a grammar with rules

$$S \to \varepsilon \text{ and } S \to +S- \,.$$

Tasks:

1. Use a general algorithm to construct a (non-deterministic) pushdown automaton that corresponds to context-free grammar described in Problem 7.
2. Show, step by step, how the word $++--$ will be accepted by this automaton.

Solution to Task 1. By using the general algorithm, we get the following pushdown automaton:



Solution to Task 2. Let us show how this is done on the example of the word $++--$ generated by the above automaton:

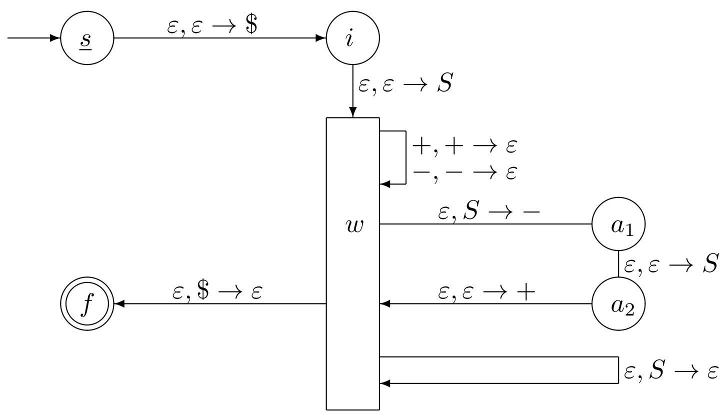
$$\underline{S} \to +\underline{S}- \to ++\underline{S}-- \to ++-- \,.$$

To make this derivation clearer, let us mark the variable S corresponding to different transitions by subscripts:

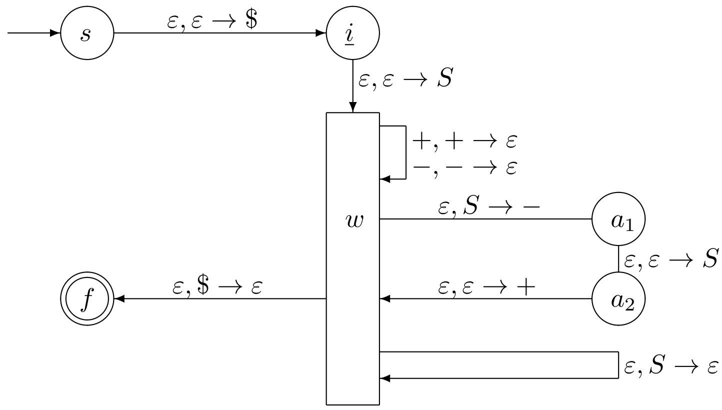
- we start with the first occurrence S_1 of the variable S;
- we then use the rule $S_1 \to +S_2-$ whose right-hand side contains the second occurrence S_2 of the variable S;
- this occurrence, in its turn, gets transformed into $S_2 \to +S_3-$ for yet another occurrence S_3 of the same variable S; so far, the derivation takes the form
$$S_1 \to +S_2- \to ++S_3--;$$
- finally, to the occurrence S_3, we apply the rule $S_3 \to \varepsilon$, resulting in the desired derivation:

$$S_1 \to +S_2- \to ++S_3-- \to ++--.$$

Let us now trace what our pushdown automaton will do. We start in the state s with an empty stack:



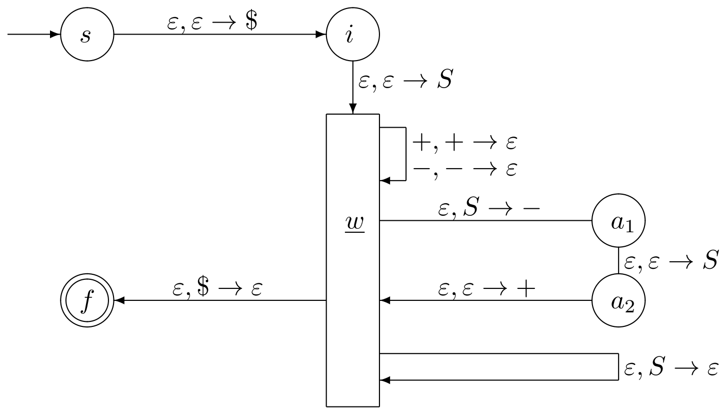
The only thing we can do when in the state s is push the dollar sign into the stack and get to the intermediate state i:



The contents of the stack is as follows:

$$\boxed{\$}$$

When we are in the state i, the only thing we can do is push the starting variable S (which corresponds to the first occurrence S_1 of this variable) into the stack and go into the working state w;



Now, the stack contains the starting variable on top of the dollar sign:

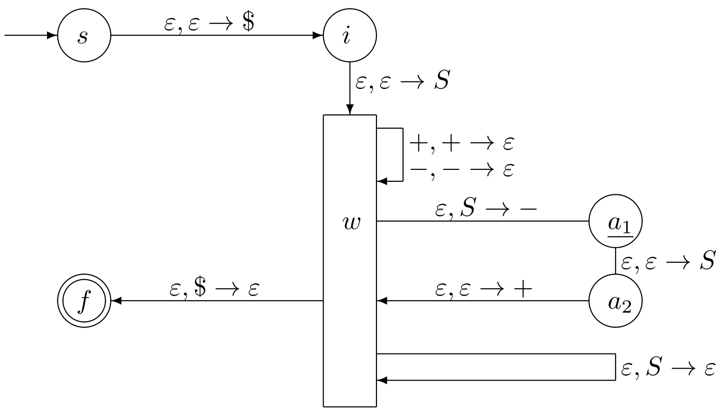
$$\begin{array}{|c|}\hline S \\ \hline \$ \\ \hline\end{array}$$

Now that we are in the working state, we can start following the rules that were used to derive the word $++--$. The first rule was $S \to +S-$, or, to be precise, $S_1 \to +S_2-$. As we have mentioned, this rule is implemented in three steps:

- first, we pop S (that corresponds to the first occurrence S_1) and push the last symbol of the right-hand side – in this cases, the symbol $-$ – into the stack, getting into the auxiliary state a_1;
- then, we push S (that corresponds to the second occurrence S_2) into the stack, getting into the auxiliary state a_2;
- finally, we push $+$ into the stack, and go back to the working state w.

Let us illustrate this step by step.

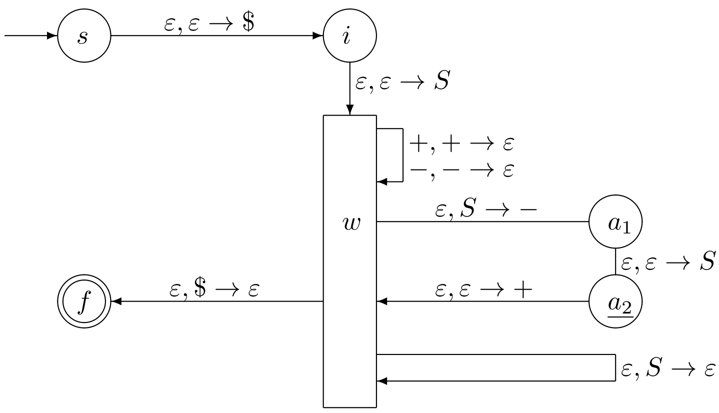
First, we pop S, push $-$, and go into the state a_1:



The stack will now have $-$ instead of S:

$-$
\$

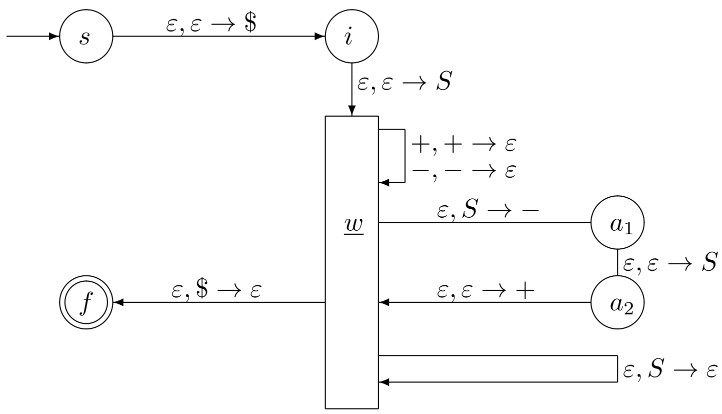
Then, we push S (corresponding to S_2) into the stack and go into the state a_2:



The stack will now have S on top of its previous contents:

S
$-$
$\$$

Finally, we push $+$ into the stack, and go back to the working state:



The stack will now have symbol $+$ at the top:

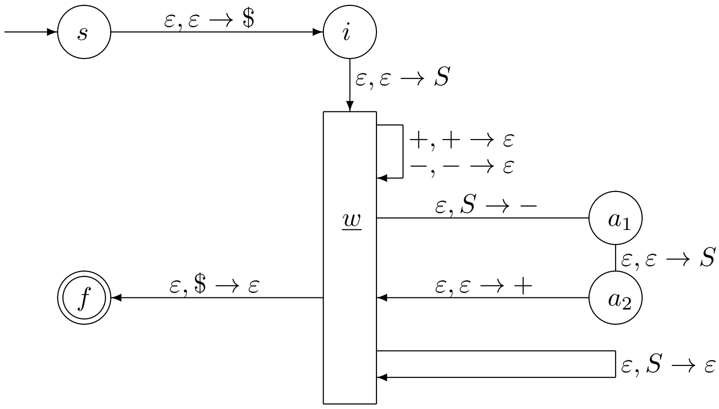
$+$
S
$-$
$\$$

Now, the symbol $+$ is top of the stack. The only thing we can do if a terminal symbol is on top of the stack is use one of the rules of the type $x, x \to \varepsilon$ where x stands for the corresponding terminal symbol.

In our case:

- since the terminal symbol on top of the stack is the symbol $+$,
- we need to use the rule $+, + \to \varepsilon$,

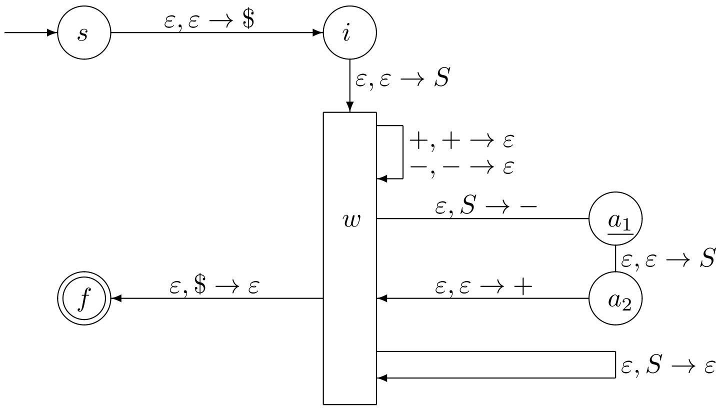
i.e., we read the symbol $+$ from the original word $++--$ and pop the top symbol $+$ from the stack:



After this popping, the variable S (corresponding to S_2) will be on top of the stack:

S
$-$
$\$$

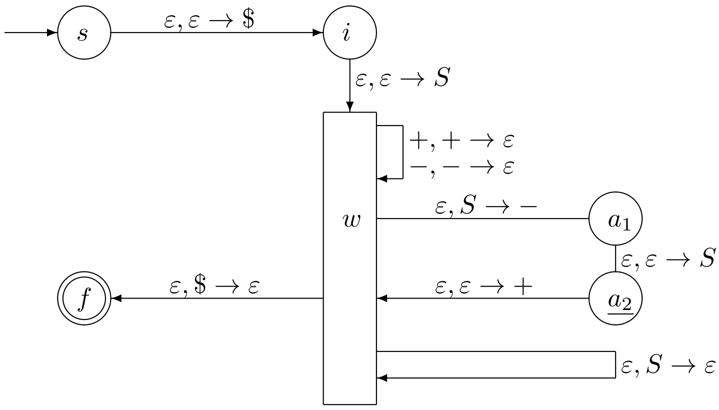
According to the original derivation of the word $++--$, to get rid of the second occurrence S_2 of the variable S, we also use the rule $S \to +S-$, or, to be precise $S_2 \to +S_3-$. So, similarly to what we have before when we used this rule, first, we pop S, push $-$, and go to the state a_1:



Now, we have $-$ instead of S on top of the stack:

$-$
$-$
$\$$

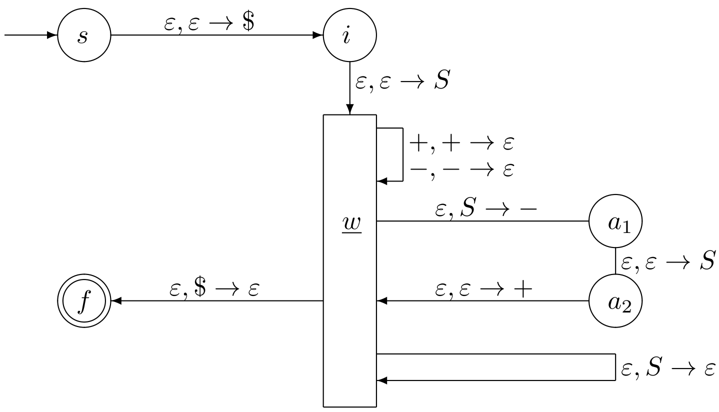
After that, we push S (that corresponds to the third occurrence S_3) and go to state a_2:



The stack now has the form:

S
$-$
$-$
$\$$

Finally, we push $+$ into the stack and go back to the working state w:

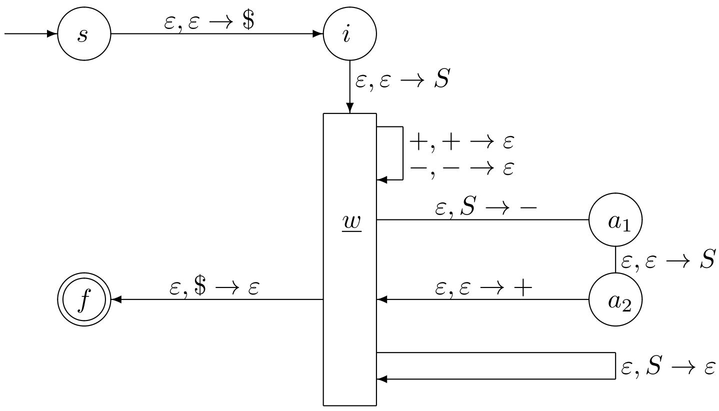


‘

Now, the stack has the following form:

$+$
S
$-$
$-$
$\$$

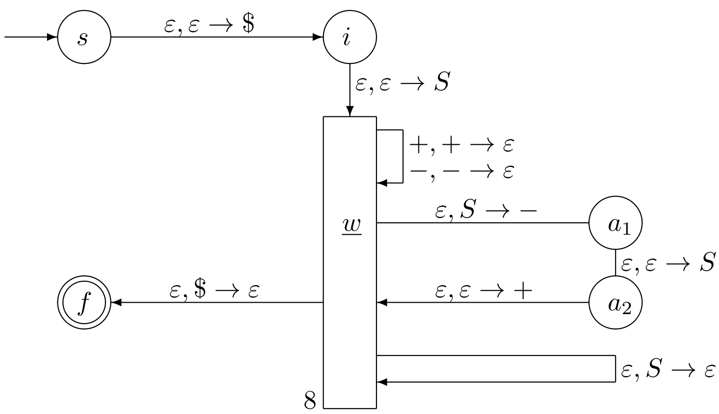
Now again, we have a terminal symbol $+$ on top of the stack, so the only thing we can do is use the rule $+, + \to \varepsilon$: we read the second symbol $+$ of the word $++--$ (the first one we have already read, so the cursor points to the second one) and pop $+$ from the stack. As a result, we get the following state:



The stack has the following form:

S
$-$
$-$
$\$$

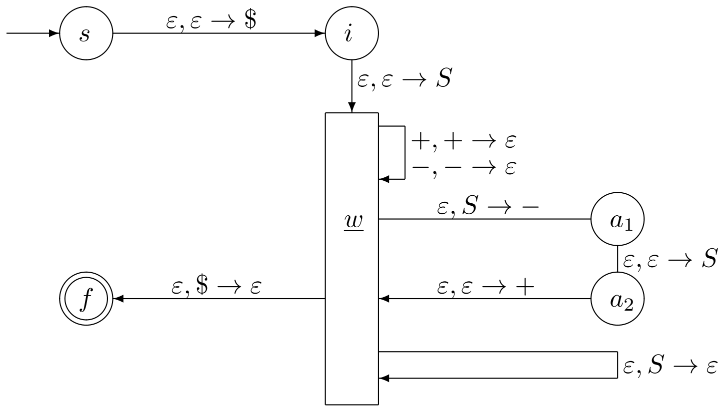
On top of the stack is the variable S corresponding to the third occurrence S_3. To get rid of this variable, in the original derivation of the sequence, we used the rule $S \to \varepsilon$ – or, to be more precise, $S_3 \to \varepsilon$. This rule of the grammar corresponds to the transition $\varepsilon, S \to \varepsilon$ of the pushdown automaton, i.e., we pop S from the stack:



The stack now takes the following form:

$-$
$-$
$\$$

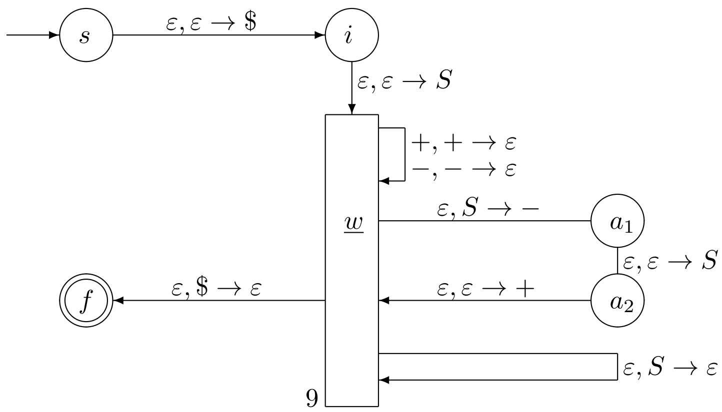
There is a terminal symbol on top of the stack – in this case, the symbol $-$. We want an empty stack at the end. The only way to get rid of $-$ is to use the rule $-,- \to \varepsilon$, i.e., to read the next symbol $-$ from the word $++--$, and to pop $-$ from the stack:



Now, the stack has the following form:

$-$
$\$$

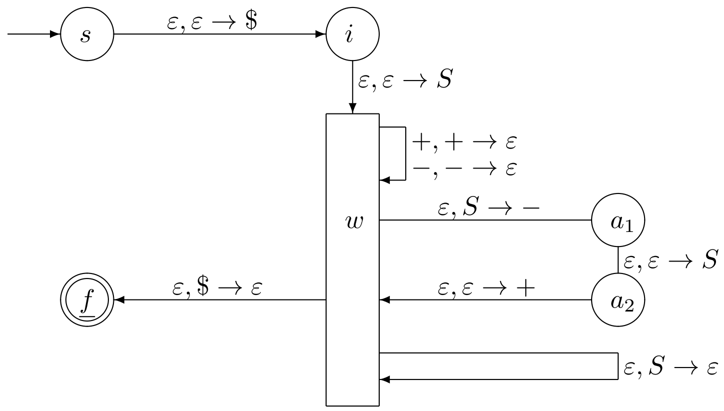
Again, we have a terminal symbol $-$ on top of the stack, so we again use the rule $-,- \to \varepsilon$, i.e., we read the next symbol $-$ of the word $++--$, and we pop $-$ from the stack:



Now, the stack only contains the dollar sign:

$$\boxed{\$}$$

We have read all the letters of the original word, and all we have in the stack is the dollar sign. So now, we can use the rule $\varepsilon, \$ \to \varepsilon$ to pop the dollar sign and to go to the final state:



Now, we are in the final state f with the empty stack. This means that the word $+ + - -$ is accepted by this pushdown automaton.

A graphical description of the transitions.

read							$+$			
state	s	i	w	a_1	a_2	w	$\to$	w	a_1	a_2
stack		$\$$	S $\$$	$-$ $\$$	S $-$ $\$$	$+$ S $-$ $\$$	$\to$	S $-$ $\$$	$-$ $-$ $\$$	S $-$ $-$ $\$$

read		$+$			$-$		$-$		
state	w	$\to$	w	w	$\to$	w	$\to$	w	f
stack	$+$ S $-$ $-$ $\$$	$\to$	S $-$ $-$ $\$$	$-$ $-$ $\$$	$\to$	$-$ $\$$	$\to$	$\$$	