UNIVERSITÉ LIBRE DE BRUXELLES

INFO-F-403 – Introduction to language theory and compiling
First session examination

August, 31st, 2021

Instructions

- This is a closed book test. You are not allowed to use any kind of reference.
- You can anwser in French or in English.
- Write your first and last names on each sheet that you hand in.
- Write clearly: you can use a pencil or a ballpen or even a quill as long as your answers are readable!
- Always provide full and rigourous justifications along with your answers.
- This test is worth 12 points out of 20. The weight of each questions is given as a reference.
- In your answers (diagrams representing automata, grammars,...), you can always use the conventions adopted in the course, without recalling them explicitely. If you deviate from these conventions, be sure to make it clear.

Question 1 — 4 points

We want to build a DFA for the following language L:

'The set of all words on the alphabet $\{\mathtt{a},\mathtt{b},\mathtt{c}\}$ in which, every time there is an a at position i, there is no b at position $i+2$ (if this position exists)'.

For example, the words aaaa, aacabc and ababa are in L, while abcabb and aaaab are not. To obtain such a DFA, apply the following steps:

1. give a *formal definition* of the language, using proper mathematical notation (with quantifiers like $\forall$ and $\exists$ if need be). For example, you can define the language under the form $L = \{w = w_1 w_2 \cdots w_n \mid \cdots\}$;
2. give a *formal definition* of the *complement* $\overline{L}$ of L, in the same spirit;
3. build an NFA that accepts $\overline{L}$ (it is much easier to build an NFA than a DFA to accept $\overline{L}$, so try and exploit non-determinism);
4. turn this NFA into an equivalent DFA using the procedure of the course (and justify it by giving the intermediary steps and computations); and, finally
5. compute the complement of this DFA.

Question 2 — 2 points

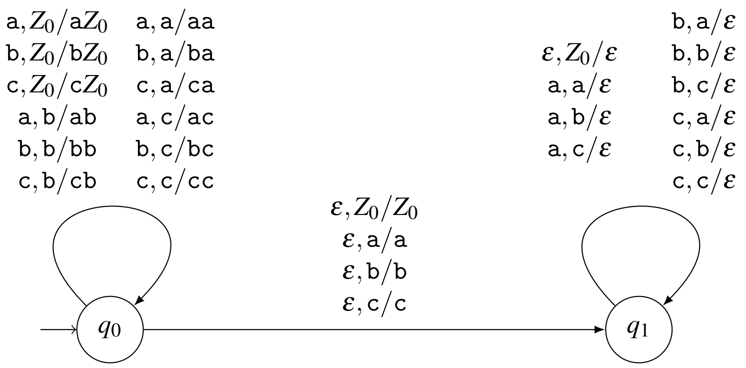
Here are three typical tasks that a compiler must perform:

1. identify variable names and figure out that several occurrences of the same variable name refer to the same variable;
2. check that, in arithmetic expressions, brackets are well-balanced (all opened brackets are eventually closed and all closing brackets correspond to a pending open bracket);
3. in an assignment such as `x=2*(i+3)`, check whether the type of the expression on the right-hand side (`2*(i+3)`) matches the type of the variable (`x`) and raise an error or introduce a conversion if needed.

For each of these tasks explain at which step of the compiling process they occur, and how they are performed.

Question 3 — 3 points

Consider the following PDA P on the alphabet $\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$



1. give a description (not necessarily formal) of the empty stack $N(P)$ language accepted by this PDA. Give an intuitive justification of your answer;
2. for each of the three following words, say whether the PDA accepts it (with empty stack). When the PDA accepts it, give an accepting run on this word, and justify why your run is accepting: (i) `abccba`, (ii) `abcabc`, (iii) `aaa`.
3. Is there a regular expression that accepts $N(P)$? Justify your answer (when the answer is 'yes', give the regular expression and explain why it accepts $N(P)$).

Question 4 — 3 points

1. Define the notion of $LL(k)$ grammar and provide, using your own words, the intuitions behind the definition.
2. For the three grammars below, say whether they are $LL(1)$ *and* whether they are $LL(2)$. Justify your answers (preferably, using the definition given in point 1).
3. When possible, transform the grammars that are not $LL(1)$ into $LL(1)$ equivalent ones.

4. For all the resulting $LL(1)$ grammars (i.e., those that were originally $LL(1)$ or those that were not but that you have managed to transform into $LL(1)$ equivalent ones), give the actions tables of their associated top-down parser.

(1)	S'	$\to$	$S\$$
(2)	S	$\to$	$A\texttt{a}$
(3)		$\to$	B
(4)	A	$\to$	$\texttt{ab}$
(5)		$\to$	ε
(6)	B	$\to$	$\texttt{b}$

(1)	S'	$\to$	$S\$$
(2)	S	$\to$	$\texttt{a}$
(3)		$\to$	$B\texttt{b}C$
(4)	B	$\to$	$\texttt{c}B$
(5)		$\to$	$\texttt{d}B$
(6)		$\to$	ε
(7)	C	$\to$	$\texttt{b}$
(8)		$\to$	$\texttt{c}$
(9)		$\to$	$\texttt{d}$

(1)	S'	$\to$	$S\$$
(2)	S	$\to$	$A\texttt{s}$
(3)		$\to$	$B\texttt{b}$
(4)	A	$\to$	$\texttt{a}$
(5)	B	$\to$	$\texttt{b}C$
(6)		$\to$	$\texttt{b}D$
(7)	D	$\to$	$D\texttt{e}$
(8)		$\to$	$\texttt{f}F\texttt{g}H$
(9)		$\to$	$\texttt{fe}$
(10)	F	$\to$	$\texttt{f}D\texttt{g}$
(11)	H	$\to$	$\texttt{h}$