Technical University of Denmark

Written examination, May 24, 2018

Page 1 of 7 pages

Course number: 02141

Aids allowed: All written aids are permitted

Exam duration: 4 hours

Weighting: 7-step scale

Old exam sets are not indicative of new exam sets.

Exercises on Formal Methods

Exercise 1 (25%)

Consider the program graph in Figure 1.

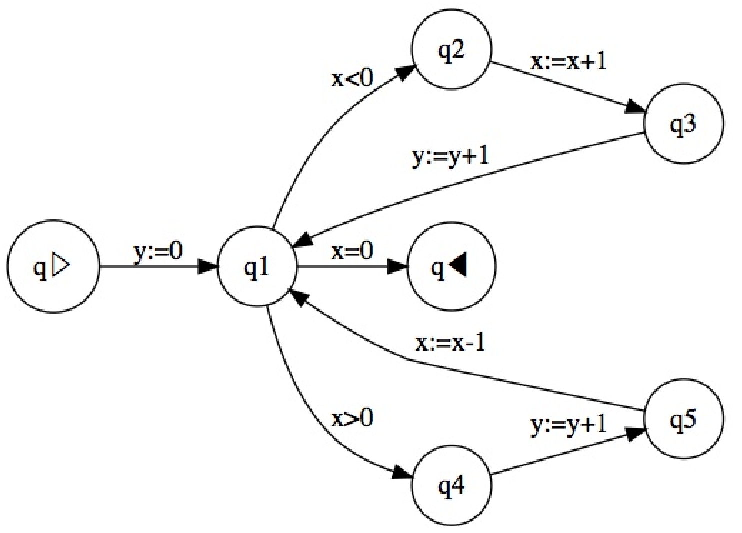


Figure 1: Program Graph for computing the absolute value.

A: Suppose that the initial memory $\sigma_\triangleright$ has $\sigma_\triangleright(\mathtt{x}) = -4$ and $\sigma_\triangleright(\mathtt{y}) = 27$ and imagine a complete execution sequence

$$\langle q_\triangleright; \sigma_\triangleright\rangle \overset{k}{\Longrightarrow} \langle q_\blacktriangleleft; \sigma_\blacktriangleleft\rangle$$

A1: What is the final memory $\sigma_\blacktriangleleft$?
A2: How many steps k were needed?
A3: Is the program graph deterministic?
A4: Is the program graph evolving?

B: Is there a program in Guarded Commands that gives rise to the program graph in Figure 1? If your answer is yes you should write the program in Guarded Commands. If your answer is no you should argue why this cannot be the case.

C: Recall that the absolute value $\mathsf{abs}(z)$ is defined by

$$\mathsf{abs}(z) = \begin{cases} z & \text{if } z \geq 0 \\ -z & \text{if } z < 0 \end{cases}$$

Complete the following incomplete partial predicate assignment

$$\begin{array}{rcl} \mathbf{P}(q_{\triangleright}) & = & \mathtt{x} = \underline{\mathtt{n}} \\ \mathbf{P}(q_1) & = & \cdots \\ \mathbf{P}(q_{\blacktriangleleft}) & = & \mathtt{y} = \mathsf{abs}(\underline{\mathtt{n}}) \end{array}$$

into one that is correct. Furthermore, argue that it is correct.

D: Suppose that the initial value of `x` is positive and that the initial value of `y` is positive and consider the detection of signs analysis.

D1: Construct the analysis assignment as computed by the chaotic iteration algorithm; for succinctness you may write $(+,-)$ for the abstract memory $\widehat{\sigma}$ that has $\widehat{\sigma}(\mathtt{x}) = +$ and $\widehat{\sigma}(\mathtt{y}) = -$.

D2: In case $\mathbf{A}(q_2)$ and $\mathbf{A}(q_3)$ are non-empty you should explain why this is so.

Exercise 2 (15%)

Consider the following program C in Guarded Commands:

```
i:=0;
j:=0;
do  i<n -> A[i]:=A[i]+3; i:=i+1
 [] j<m -> B[j]:=B[j]*B[j]; j:=j+1
od
```

Here `A` is intended to be an array of length `n` and `B` is intended to be an array of length `m`.

A: Draw the program graph as constructed by

$$\mathbf{edges}(q_{\triangleright} \rightsquigarrow q_{\blacktriangleleft})[\![C]\!]$$

of Chapter 2.

B: Draw the instrumented program graph as constructed by

$$\mathbf{edges_s}(q_{\triangleright} \rightsquigarrow q_{\blacktriangleleft})[\![C]\!](\{\,\})$$

of Chapter 5.

C: Suppose that the permissible information flow is given by $\{\mathtt{i},\mathtt{A},\mathtt{n}\} \rightrightarrows \{\mathtt{i},\mathtt{A},\mathtt{n}\}$ and $\{\mathtt{j},\mathtt{B},\mathtt{m}\} \rightrightarrows \{\mathtt{j},\mathtt{B},\mathtt{m}\}$.

Is the program secure, i.e. what is the value of $\mathbf{sec}[\![C]\!](\{\,\})$?

You should motivate your answer (but do not need to exhibit the detailed computation).

Exercises on Context-free Languages

Exercise 3 (30%)

The exercises in this part of the exam are based on PingPong, a programming language whose syntax is given by the following context-free grammar. The productions are

P	$\rightarrow$	A	(actions)
	$\mid$	P `>` P	(sequential composition of programs)
	$\mid$	P `+` P	(non-deterministic choice between programs)
	$\mid$	P `*` P	(parallel composition of two programs)
	$\mid$	`[`P`]`	(scoped program)
A	$\rightarrow$	`ping`	(do ping)
	$\mid$	`pong`	(do pong)

the set of non-terminal symbols is $N = \{P, A\}$, the set of terminal symbols is $T = \{ \texttt{[}, \texttt{]}, \texttt{>}, \texttt{+}, \texttt{*}, \texttt{ping}, \texttt{pong}\}$, and the initial symbol is P.

(a) Show that the following program is accepted by the grammar of PingPong by providing a parse tree for it:

```
[ ping > pong ]
```

(b) Show that the grammar is ambiguous by providing one PingPong program and two distinct parse trees for it.

(c) Consider the following associativity and precedence rules:

- `>` has the highest priority and associates to the right;
- `+` has priority over `*` and associates to the right;
- `*` has the lowest priority and associates to the left.

which of the two parse trees you provided in (b) adheres to these rules?

(d) Consider again the rules of (c). As explained in class, many grammar languages used by parser generators allow you to specify those rules in the grammar specification. Most likely, this is what you did in the mandatory assignment. Describe very briefly the grammar annotations that would implement the rules of (c) in one of the two grammar languages we saw in class (FsLexYacc or ANTLR4). Please specify which grammar language you have chosen. You don't need to give the entire grammar specification but just the part where the rules are specified.

(e) Consider again the rules of (c). Transform the grammar of PingPong to encode the rules directly in the grammar. You can apply the techniques seen in class, i.e. enforce associativity by forbidding recursion on one of the sides, and enforce precedence by stratifying the grammar into precedence layers.

(f) Construct a Pushdown Automaton (PDA) that accepts the same language as the the grammar of PingPong described at the beginning of the exercise. Use the construction that we saw in class to translate a grammar into an equivalent PDA.

(g) Use the PDA you have constructed to show that the example of (a) is in the language of the PDA. Show the sequence of configurations (instantaneous descriptions) of the PDA starting with the initial one (q, `[ ping > pong ]`, P), where q is the initial state of your PDA, `[ ping > pong ]` is the input and P is the initial stack.

(h) Design a set of datatypes that are suitable to store abstract representations (ASTs) of PingPong programs. You can take inspiration from your solution to the mandatory assignment, i.e. you can use F# types or Java classes. Show how the example program

```
[ ping > pong ] * [ ping + pong ]
```

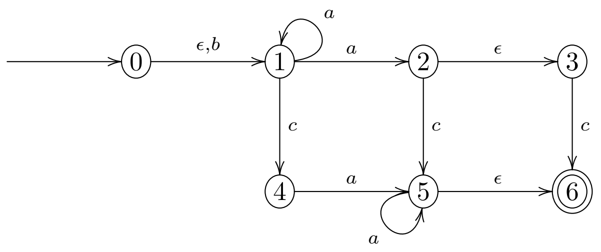
would be represented as a value of your datatype.

Regular languages

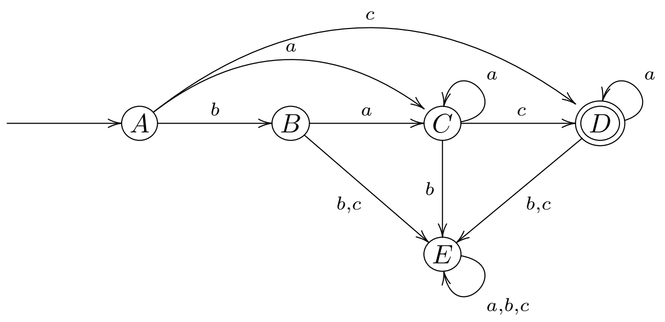
Exercise 4 (30%)

Let $\Sigma = \{a, b, c\}$ and consider the following languages:

- L_1 is given by the ϵ-NFA:



- L_2 is given by the regular expressions $(\epsilon + b)(a + c)^*$
- L_3 is given by the DFA:



- L_4 is given by $\{ba^n ca^m \mid n, m \geq 0, n \geq m\}$
- L_5 is given by the regular expression $(\epsilon + (b + \epsilon)aa^*)ca^*$

In this exercise we shall investigate the relationship between these languages.

(a) Fill out the following table with a *yes* if the string in the column is a member of the language in the row and a *no* if it is not a member of the language:

w	ϵ	c	bc	bac	bca	aac	$acaca$
L_1							
L_2							
L_3							
L_4							
L_5							

(b) Use the subset construction algorithm (Hopcroft, Motwani and Ullman chapter 2) to construct a DFA corresponding to the ϵ-NFA for L_1.

(c) Prove that L_4 is not a regular language.

(d) Fill out the following table with a *yes* if the language in the leftmost column is a subset of the language in the topmost row and a *no* if this is not the case.

$\subseteq$	L_1	L_2	L_3	L_4	L_5
L_1	*yes*				
L_2		*yes*			
L_3			*yes*		
L_4				*yes*	
L_5					*yes*

If you answer *yes*, please give an argument for why, and if you answer *no*, please give a counter example.