Theoretical Computer Science Exam, January 19th, 2022

The exam consists of **4 exercises**. Available time: 1 hr 30 min.; you may write your answers in Italian, if you wish.

All exercises must be addressed for the written exam to be considered sufficient.

F.NAME …………………………….. L.NAME …………………………………… PERSON CODE …………………

Exercise 1 (8 pts)

Consider the following language

$$L = \{ b\, a^i\, b^j\, a^k \mid 1 \le i \le k \,\wedge\, j > 0 \}$$

For instance, strings *baabbbaaa*, *baabbaa* $\in L$, while ε, *bbabaa*, *baaaba*, *aabaaa* $\notin L$.

- Design an automaton, of the least powerful type, that accepts language L.
- Write a grammar, of the least general type, that generates language L.

Exercise 2 (8 pts)

Consider a subprogram, called *decrIncr*, having three parameters: an array of integers *in*, its size *nIn*, an array of integers *out*. The input parameters are: *in* and *nIn*, whereas the array *out* is an output parameter. The two arrays can be assumed to have the same size (i.e., number of components), greater than 0; all the elements of array *in* are assumed to be strictly positive; in the input array *in* there is a *minimum* element, that is, an element whose value is strictly less than all the other ones.

After execution of the subprogram the minimum value of array *in* (let us call it *m*) is written, in the same position, into the array *out*; furthermore, all the components of array out in the preceding positions are equal to the elements having the same position in array *in* decremented by the value *m*, and all elements in the following positions are equal to the elements having the same position in array *in* incremented by the value *m*. For instance, the specification is satisfied by the following values: *nIn*=6, *in*=[7, 9, 5, 2, 11, 6], *out*=[5, 7, 3, 2, 13, 8]; the minimum element is 2 at the fourth position, all elements of array *in* in the previous positions (7, 9, 5) are inserted into array *out* decremented by 2 (5, 7, 3), and all elements of array *in* in the following positions (11, 6) are inserted into array *out* incremented by 2 (13, 8).

Please answer the following questions:

1. write pre- and post-conditions specifying the requirements of the above described procedure; assume that indices of the arrays start at 1;
2. for each of the following input/output data say if the procedure *decrIncr* satisfies the specification if it computes the given value of the *out* array when it receives the given input parameter values; please provide suitable justifications for your answers.

   a) *nIn*=6, *in*=[2, 7, 9, 5, 11, 6], *out*=[2, 9, 11, 7, 13, 8]

   b) *nIn*=6, *in*=[7, 9, 2, 5, 11, 6], *out*=[9, 11, 2, 3, 9, 4]

   c) *nIn*=6, *in*=[2, 2, 2, 2, 2, 2], *out*=[4, 9, 2, 5, 11, 6]

Exercise 3 (8 pts)

Assume that a generic regular grammar G_R is given.

Answer the following questions, providing precise, clear justifications.

- Given also a generic Turing machine *M*, is the problem of determining wether *M* accepts the language generated by G_R solvable?

- Given also a generic finite state automaton *A*, is the problem of determining wether *A* accepts the language generated by G_R solvable?

Exercise 4 (8 pts)

Consider the language composed of strings $x \in \{a, b\}^*$ such that $\#a(x)=2\cdot\#b(x)$, i.e., strings for which the number of *a* characters is twice the number of *b* characters. Please answer the following questions.

- Sketch a simple Turing machine that accepts this language. Design this machine so that its time complexity is minimal, and provide an estimation of its time and space complexity as a function of $n=|x|$, the length of the input string; suitably justify the complexity figures.

- Sketch another Turing machine that accepts this language with a minimal space complexity, and provide an estimation of its time and space complexity as a function of $n=|x|$, the length of the input string; suitably justify the complexity figures.

Solutions

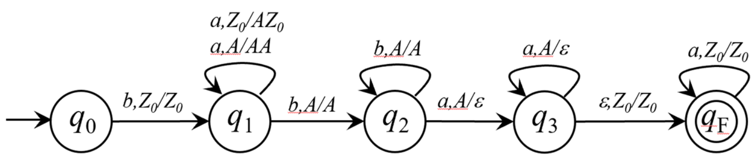
Exercise 1

Consider the following language

$$L = \{\, b\, a^i\, b^j\, a^k \mid 1 \le i \le k \;\wedge\; j > 0 \,\}$$

For instance, strings *baabbbaaa*, *baabbaa* $\in L$, while ε, *bbabaa*, *baaaba*, *aabaaa* $\notin L$.

- The following *deterministic* pushdown automaton accepts language L.



- The following context-free grammar generates language L.

  $S \to b\, A$

  $A \to a\, A\, a \mid A\, a \mid a\, B\, a$

  $B \to b\, B \mid b$

Exercise 2

1. write pre- and post-conditions specifying the requirements of the above described procedure; assume that indices of the arrays start at 1.

precondition:

$nIn > 0 \wedge \exists i\, (\, 1 \le i \le nIn \wedge \forall j\, (1 \le j \le nIn \wedge j{\neq}i \to in[i] < in[j]\ )\ )$

postcondition:

$\exists i\, (\, (\, 1 \le i \le nIn \wedge \forall j\, (1 \le j \le nIn \wedge j{\neq}i \to in[i] < in[j]\ )\,) \wedge$

$\quad (\, out[i] = in[i]\,) \wedge$

$\quad \forall h\, (1 \le h < i \to out[h] = in[h] - in[i]\ ) \wedge$

$\quad \forall k\, (i < k \le nIn \to out[k] = in[k] + in[i]\ )$

$)$

2. for each of the following input/output data say if the procedure *decrIncr* satisfies the specification if it computes the given value of the *out* array when it receives the given input parameter values; please provide suitable justifications for your answer.

nIn=6, in=[2, 7, 9, 5, 11, 6], out=[2, 9, 11, 7, 13, 8]

Yes, because both the precondition and the postcondition are satisfied; notice that the segment of the array *in* preceding the position of the minimum element is empty, so that the clause $\forall \boldsymbol{h}\ \mathbf{(1 \le h < i \to out[h] = in[h] - in[i]\ )}$ is satisfied vacuously.

nIn=6, in=[7, 9, 2, 5, 11, 6], out=[9, 11, 2, 3, 9, 4]

No, because the the precondition is satisfied, but the postcondition is not satisfied, because the element in the segment of the array *in* preceding the position of the minimum element have been increased, and those in the segment following it have been decreased.

nIn=6, in=[2, 2, 2, 2, 2, 2], out=[4, 9, 2, 5, 11, 6]

Yes, because the precondition is not satisfied. Notice that in this case the values in the output parameter are immaterial.

Exercise 3

- Given also a generic Turing machine M, is the problem of determining wether M accepts the language generated by G_R solvable?

No, because determining wether a generic Turing machine M accepts a language is equivalent to determine if M computes a given function, which notoriously is an unsolvable problem, from the Rice theorem.

- Given also a generic finite state automaton A, is the problem of determining wether A accepts the language generated by G_R solvable?

Yes, because from grammar G_R one can algorithmically build an equivalent finite state automaton A_R , that is an automaton that accepts the language generated by G_R. Then it is possible to check algorithmically that A and A_{LR} are equivalent (that is, they accept the same language): for instance by (algorithmically) building the product automaton $A_\times$ that accepts the intersection of the languages $L(A)$ and $\neg L(A_{LR})$ (where $\neg L(A_{LR})$ is the complement of $L(A_{LR})$) and then verify (still in an algorithmic way) that the language accepted by automaton $A_\times$ is empty.

Exercise 4

Consider the language composed of strings $x\in\{a, b\}^*$ such that $\#a(x)=2\cdot\#b(x)$, i.e., strings for which the number of a characters is twice the number of b's.

- Sketch a simple Turing machine that accepts this language. Design this machine so that its time complexity is minimal, and provide an estimation of its time and space complexity.

The Turing machine that recognizes the language with minimal time complexity uses two memory tapes (T1 and T2) to store a count in unary of the numebr of a's read from the input tape into T1 and of the b's into T2. Then it can check if $\#a(x)=2\cdot\#b(x)$ by scanning the two memory tapes from right to left. For each symbol in T2 the head on T1 moves two cells to the left. The input string is accepted if and only if the two heads on T1 and T2 reach the initial tape position simultaneously.

The space complexity is $S(n) = \Theta(n)$, with $n = |x|$, because a total number of n symbols is written overall on the two memory tapes when the input string is read. The time complexity is $T(n) = \Theta(2n) = \Theta(n)$, because after reading thr input string an additional number of moves is executed that is less than n in all cases.

- Sketch another Turing machine that accepts this language with a minimal space complexity, and provide an estimation of its time and space complexity.

The Turing achine that recognizes the language with minimal space complexity also uses two memory tapes to record the number of a's and b's, but in binary number form. Once the input string has been completely read, it scans the two numbers to check if one number is twice the other: the string representing the number of a's must be equal to the one for the number of b's, with an additional 0 in the least significant position. The space complexity is therefore reduced to $S(n) = \Theta(\log n)$.

The time complexity is $T(n) = \Theta(n \log n)$, because, for each character read, one of the two counters must be incremented, which requires $\Theta(\log k)$ for a number encoded by a string of length k. Notice that the complexity of the check that one number is twice the other is dominated by the increment operations on the counters, because it is $\Theta(\log n)$.