

COMP 481: Automata, Formal Languages, and Computability
Spring 2009
Solutions to Homework Assignment #6

1. (a) Solving this problem depends on the fact (in the notes I handed out in class) that $L(G)$ is infinite if and only if G generates at least one string w , where $b^n < |w| \leq b^{n+1} + b^n$, where n is the number of non-terminals in G and b is the maximum number of symbols in the right-hand side of a production rule in G . So, first check all strings of length less than or equal to b^n . If G generates at least three of these strings, then print YES. Otherwise, check if G generates at least one string w , where $b^n < |w| \leq b^{n+1} + b^n$. If this is the case, print YES (since $L(G)$ is infinite in this case). Otherwise, print NO.
(b) Let M be a DFA that accepts all even-length strings. Using the construction shown in the textbook, convert G into a PDA M' , build a PDA M'' for $L(M) \cap L(M')$, and test for emptiness of $L(M'')$. If $L(M'') = \emptyset$, then print NO; otherwise, print YES.
(c) print YES. (every regular grammar is also a CFG!)
2. (a) A 2-tape TM M that decides the language scans the input from left to right to check if it's of the form a^*b^* . If not, it rejects. Then, it copies the b 's to the second tape, and resets the reading heads on both tapes to the leftmost cells. Then, it moves both reading heads right as long as it's seeing an ' a ' on the first tape, and a ' b ' on the second tape. Once it sees a blank on the first and an ' a ' on the second, it halts and accepts. If it sees two blanks on both tapes, or a blank on the second and an ' a ' on the first, it halts and rejects.
(b) A 3-tape TM M that decides the language scans the input from left to right and checks if it's of the form $(a \cup b)^*$. If not, it rejects. Then, it computes the length of the string, copies its second third to the second tape, and its third part to the third tape (if the string's length is not divisible by 3, M halts and rejects). Then, M moves all three reading heads to the leftmost cells, and move them right in parallel, while checking that all three heads read the same symbol at all times. If the three reading heads see blanks, M halts and accepts. Otherwise (not the same symbol on all three tapes), it halts and rejects.
(c) A 4-tape TM M that decides the language computes the numbers of a 's, b 's, and c 's and stores them on tapes 2, 3, and 4, respectively. Then, it compares all three numbers (when they are represented in unary, for example). If they are equal, it halts and accepts; otherwise, it rejects. While counting, if M encounters any symbol not in the alphabet, it halts and rejects.
3. (a) A 3-tape TM M that computes the sum of two numbers a and b , represented in binary, works as follows (a and b are both on the first tape when M starts):
 - i. If either a or b equals 0, M halts.
 - ii. M copies b to the second tape, positions the reading heads on the first and second tape to point to the rightmost letters in both a and b , and enters a state of 'no carry'.

- iii. If in a ‘no carry’ state:
 If the two letters are 0, M writes 0 on its third tape and moves the reading head right; if the two letters are 0 and 1, M writes 1 on its third tape and moves the reading head right; if the two letters are 1, M writes 01 on its third tape, keeps the reading head under the 1, and moves to a ‘carry’ state.
 Else: (in a ‘carry’ state)
 If the two letters are 0, M moves the reading head right, and transitions to a ‘no carry’ state; if the two letters are 0 and 1, M writes 01 on its third tape, and keeps the reading head under the 1; if the two letters are 1, M writes 11 on its third tape, and positions the reading head under the rightmost 1.
 - iv. M Moves the reading heads on tapes 1 and 2 one position to the left, and repeats step 3(a)iii. If at some point one of the tapes is empty and the other has letters left, M copies them to the third tape (while reversing the order and handling it correctly if in a ‘carry’ state).
 - v. When the input on the first and second tapes has been completely handled (the numbers are added), M erases the contents of tape 1, and puts the contents of tape 3 on it in a reversed order, and halt.
- (b) A 3-tape TM M that converts a binary number a on its first tape to a unary number, works as follows:
- i. M writes 1 on its second tape, positions the reading head of the first tape under its rightmost letter.
 - ii. If the reading head of the first tape reads a 1, M copies the contents of the second tape to the third tape.
 - iii. M duplicates the string on the second tape, moves the reading head on the first tape one position to the left, and goes back to Step 3(b)ii.
 - iv. When the first tape is empty, M erases its first tape, copies the contents of the third tape to the first one, and halts.
4. Let M_i be a TM that semidecides L_i . M'_i that decides whether $x \in L_i$ runs all machines M_i , $1 \leq i \leq k$, “in parallel”, on string x . If M_i accepts, then M'_i accepts. If M_j , for some $j \neq i$ accepts, M'_i rejects.
- Since the L_i ’s form a partition, we have that $\cup_{i=1}^k L_i = \Sigma^*$. In other words, for every $x \in \Sigma^*$, $x \in L_i$ for some $1 \leq i \leq k$, in which case M_i will halt and accept x . Therefore, one of the machines M_i is guaranteed to halt and accept x , and hence M'_i always halts (and decides L_i).
5. D and SD are closed under Union, Intersection, Concatenation, and Kleene Star. To prove D languages are closed under union and intersection, run the two TM’s for the two languages, and accept if either or both accept, respectively. The same technique applies to SD languages, but in the case of Union, the run of the two machines must be interleaved (for the Intersection, interleaving is not needed).
- To prove D languages are closed under concatenation, the TM M (for $L = L_1L_2$) considers all possible partitions xy of string w , runs TM M_1 (of L_1) on x and TM M_2 (of L_2) on y and

accepts iff both accept for some partition $w = xy$ (notice that the number of such partitions is finite). The same technique applies for SD, but M must interleave the runs of all possible partitions at the same time, to avoid getting stuck in one of the partitions, while another might accept.

To prove D languages are closed under Kleene Star, the TM M (for L^*) considers all possible partitions $w = x_1x_2 \cdots x_k$ ($0 \leq k \leq |w|$), runs M_1 (of L) on each x_i . M accepts iff M_1 accepts all x_i in one of the possible partitions. As before, a similar technique applies to SD languages, but M must interleave the runs of all possible partitions at the same time, to avoid getting stuck in one of the partitions, while another might accept.

6. The solutions to Problems (5) and (6) are combined in (5).