Homework 6 Solutions

DISCLAIMER: The solutions presented below are incomplete and might be insufficient to get full grade on the homework. They do not model acceptable solutions, but rather present an idea of how a certain problem can be approached. A diligent student should be able to work out complete solutions. Please report any mistakes that you find to the instructor and TA(s).

1. (a)

$$\begin{aligned}
\delta(q_0,\$) &= (q_0,\$,R)\\
\delta(q_0,1) &= (q_0,1,R)\\
\delta(q_0,0) &= (q_0,0,R)\\
\delta(q_0,B) &= (q_1,B,L)\\
\delta(q_1,0) &= (q_2,1,L)\\
\delta(q_1,1) &= (q_1,0,L)\\
\delta(q_1,\$) &= (q_2,1,L)\\
\delta(q_2,0) &= (q_2,0,L)\\
\delta(q_2,1) &= (q_2,1,L)\\
\delta(q_2,\$) &= (q_f,\$,R)\\
\delta(q_2,B) &= (q_f,B,R)
\end{aligned}$$

   Where q_0 is the state responsible for moving the reading head to the right, q_1 is responsible for addition of 1 (in particular it also handles the carry bit), q_2 is responsible for moving the reading head to the left, q_f is the accepting state.

   (b) $q_0\$111 \vdash \$q_0111 \vdash \$1q_011 \vdash \$11q_01 \vdash \$111q_0B \vdash \$11q_11 \vdash \$1q_110 \vdash \$q_1100 \vdash q_1\$000 \vdash q_2B1000 \vdash q_f1000$.

2. The language accepted by this NTM is all binary strings beginning with 0.

3. It suffices to show that a two-dimensional Turing Machine M can be simulated by a two-tape one-dimensional Turing Machine M', since we already know that any two-tape TM can be simulated by a single-tape TM.

   A particular state of M can be viewed as a finite rectangular box and a position of the head in a particular state within that box. This is because at any time of execution of M, it can only examine a finite number of cells. We can represent this state in M' in row-major form, separating rows with a new symbol, say, *. For example, suppose that M is in the following state

	⋮	⋮	⋮	⋮	
...	0	1	B	0	...
...	1	B	B	B	...
...	0	0	0	1	...
	⋮	⋮	⋮	⋮	

   These contents of 2D tape will be captured on 1D tape in M' as follows:

...	B	$*$	0	1	B	0	$*$	1	B	B	B	$*$	0	0	0	1	$*$	B	...

   Now, if M's head was in row 2 and column 2 in state q, then M' will be positioned on the second symbol of the second interval in between two stars, also in state

q. If the head moves left or right in M, then it also does so in M'. If the head moves up (down), then in M the head has to relocate to the nearest left (right) interval between the stars, positioning in the correct symbol from the right of the star.

There are two more technical points. Suppose that the head in M moves up and leaves the rectangle, then in M' we need to prepend a fresh new interval (adding a new star to the correct location) immediately to the left of its tape contents, i.e. add a new row to the simulation. Similarly, when M moves down off the rectangle. When M moves right off the rectangle, we have to increase the size of each row in simulation in M'. This amounts to lots of shifting of contents.

It is easy to see that all the above steps can be accomplished with an aid of the second tape, completing our argument.

Note: other solutions exist. One other nice approach is to map indices of cells in each quadrant using the enumeration from a proof that $\mathbb{N} \times \mathbb{N}$ has the same cardinality as $\mathbb{N}$, i.e., the proof that the set of rationals is countable.

4. Much of the solution to this question arises from understanding the hint. Let M be a DPDA for language L. Our goal is to build DPDA M' for language L/a. The idea is to make M' "look ahead". Let's see what this means. Suppose that the machine M' has seen input w so far. It then should pretend that it is the end of the input and check whether wa would be accepted by machine M. However, while performing this verification, the machine should not destroy the contents of the stack or alter its state, because it might be feeded with extra characters later (w might not be the end of the entire input). In other words, this verification procedure has to be performed "on-line". Formally we have the following.

   Let $M = (Q, \Sigma, \Gamma, \delta, g_0, Z_0, F)$ be the DPDA, as above. We shall define $M' = (Q', \Sigma, \Gamma', \delta', q_0', Z_0', F')$ to be the following:

   - $\Gamma' = \Gamma \times 2^Q$ - as per the hint, we shall represent stack $X_1, \ldots, X_n$ by $(X_1, S_1), \ldots, (X_n, S_n)$ such that S_i is the set of states q such that M accepts from ID $(q, a, X_i \ldots X_n)$.
   - $Q' = Q \times 2^Q$ - the states of new DPDA record the current state of M and the set of states S_1, which appears as the second component of an element at the top of the stack.
   - Letting $S_0 := \{q \in Q \mid M \text{ accepts from ID } (q, a, Z_0)\}$, we set $Z_0' = (Z_0, S_0)$.
   - With S_0 defined as above, we let $q_0' = (q_0, S_0)$.
   - $F' = \{(q, S) \mid q \in F \cap S\}$.
   - Lastly, we define δ'. Suppose that $\delta(q, a, X) = (p, \epsilon)$, then we define $\delta'((q, S), a, (X, S)) = ((p, S), \epsilon)$. Also, for all $q \in Q$, $S, S' \subseteq Q$ with $S \neq S'$ and $X \in \Gamma$ we introduce $\delta'((p, S), \epsilon, (X, S')) = ((p, S'), (X, S'))$ - this rule serves as updating the buffer in the state holding the current set of states on top of stack. If $\delta(q, a, X) = (p, YX)$, then we introduce $\delta'((q, S), a, (X, S)) = ((p, S'), (Y, S')(X, S))$, where S' consists of two kinds of states:
     - states g such that beginning from ID (g, a, YX) DPDA M pops Y without reading a and finding itself in a state from S, and

  – states g such that beginning from ID (g, a, YX) DPDA M accepts without popping Y.

It is easy to extend the above definition of δ' to the strings $\gamma \in \Gamma$, so we leave it as an exercise. (Note that here we don't allow M to accept immediately after ϵ-moves. Allowing such acceptance is a straitforward modification of the above construction).

With the above construction it is easy to see that the semantic description of the invariant described in Γ' actually holds for M'. The correctness of the simulation follows from this and the definition of F'.