CSEP531 Homework 5 Solution

1. Suppose that 3-SAT is $PSPACE$-complete. Then all problems in $PSPACE$ reduces to 3-SAT, thereby being in NP. This means $PSPACE \subseteq NP$. But we know $NP \subseteq PSPACE$. Therefore, $NP = PSPACE$.

2. IPA is in $PSPACE$ because in linear space we can simulate M on x and keeping a counter of the number of steps. We rejects if M either (i) violates the "in space" constrain; or (ii) rejects; or (iii) operates for more than the number of possible configurations, which is $|Q||x||\Sigma|^{|x|}$ where Q is the set of state of M, and Σ is its tape alphabet.

   To show that IPA is $PSPACE$-hard, we show that *any* language L in $PSPACE$ reduces to IPA. Since $L \in PSPACE$, there is a machine M that decides if $x \in L$ in space $|x|^k$ for some constant k. Then obviously, M accepts x iff it accepts $xB^{|x|^k}$ (that is, x "padded" with $|x|^k$ blanks). Hence $x \in L$ iff $(M, xB^{|x|^k})$ is a "yes" instance of IPA.

3. For each of the questions, we will give two solutions. One is systematic; the other is shorter and quite "cute".

   (a) • **First solution.** Let $\mathcal{F}$ be the event that the last guest sits on her assigned seat and $\mathcal{E}_i$ be the event that the first guest sits on seat i. Furthermore, let p_k be the probability of $\mathcal{F}$ when there are totally k guests. It is easy to see that $p_1 = 1/2$.
   Next, suppose $k \geq 2$. Then

$$\begin{aligned}
p_k = Pr[\mathcal{F}] &= \sum_{i=1}^{k} Pr[\mathcal{F} \wedge \mathcal{E}_i] \\
&= \sum_{i=1}^{k} Pr[\mathcal{F}|\mathcal{E}_i]Pr[\mathcal{E}_i] \\
&= Pr[\mathcal{F}|\mathcal{E}_1]Pr[\mathcal{E}_1] + \sum_{i=2}^{k-1} Pr[\mathcal{F}|\mathcal{E}_i]Pr[\mathcal{E}_i] + Pr[\mathcal{F}|\mathcal{E}_k]Pr[\mathcal{E}_k] \\
&= \frac{1}{k} + \frac{1}{k}\sum_{i=2}^{k-1} Pr[\mathcal{F}|\mathcal{E}_i].
\end{aligned}$$

   The last equations follows from the following facts: (i) $Pr[\mathcal{F}|\mathcal{E}_1] = 1$; (ii) $Pr[\mathcal{F}|\mathcal{E}_k] = 0$; and (iii) $Pr[\mathcal{E}_i] = \frac{1}{k}$ for all i.
   Next, we compute $Pr[\mathcal{F}|\mathcal{E}_i]$ for $2 \leq i \leq k-1$, that is the probability that the last guest sits on her assigned seat given that the first guest sits on seat i. In this case, all guests from the second one to the $i-1$st one sits on their assigned seats. However, the ith guest has to sit on a seat chosen uniformly at random among seats $1, i+1, i+2, \ldots k-1$ and k. Renaming seats $i+1, i+2, \ldots k$ to $2, 3, \ldots k-i+1$, we get the original situation with $k-i+1$ guests. Therefore $Pr[\mathcal{F}|\mathcal{E}_i] = p_{k-i+1}$.

The rest is a matter of calculation. We have

$$\begin{aligned} p_k &= \frac{1}{k} + \frac{1}{k}\sum_{i=2}^{k-1} p_{k-i+1} \\ &= \frac{1}{k} + \frac{1}{k}\sum_{j=2}^{k-1} p_j \\ &= \frac{1}{k} + p_{k-1}/k + \frac{1}{k}\sum_{j=2}^{k-2} p_j \end{aligned}$$

Since $p_{k-1} = 1/(k-1) + 1/(k-1)\sum_{j=2}^{k-2} p_j$, we have

$$\sum_{j=2}^{k-2} p_j = ((p_{k-1} - 1/(k-1))\,(k-1).$$

Therefore

$$p_k = \frac{1}{k} + p_{k-1}/k + \frac{k-1}{k}\,(p_{k-1} - 1/(k-1)) = p_{k-1}.$$

Thus, we have $p_k = 1/2$ for all $k \geq 2$.

- **Second solution.** First, observe that when the last guest comes to the room, all the seats from 2 to $n-1$ have been occupied; for if a seat i, $2 \leq i \leq n$ was available, then the ith guest should have taken it. Second, observe that the two seats 1 and n look completely the same to the first $n-1$ guests. Therefore, they are available with the same probability, which is 1/2.

(b)

- **First solution.** Let Y be the (random variable representing the) number of guests sitting on their assigned seat, X be the number of such guests except the absent minded professor – the reason for introducing X will be clear later, and $e_k = E(X)$ if there are totally k guests. Note that $Y = X$ most of the time, except for when the absent minded professor sits on her assigned seat, in which case $Y = X + 1$. Since this case happens with probability $\frac{1}{k}$, we have $E(Y) = E(X) + \frac{1}{k}$. Thus, instead of computing $E(Y)$, we compute $E(X)$, i.e. e_k.
  We have $e_1 = 0$. Suppose $k \geq 2$, we have

$$\begin{aligned} e_k = E[X] &= \sum_{i=1}^{k} E[X|\mathcal{E}_i]Pr[\mathcal{E}_i] \\ &= E[X|\mathcal{E}_1]Pr[\mathcal{E}_1] + \sum_{i=2}^{k} E[X|\mathcal{E}_i]Pr[\mathcal{E}_i] \\ &= \frac{k-1}{k} + \frac{1}{k}\sum_{i=2}^{k} E[X|\mathcal{E}_i]. \end{aligned}$$

The last equality follows from the following facts: (i) $E[X|\mathcal{E}_1] = k-1$ since if the absent minded professor sits on her assigned seat, all others do as well; and (ii) $Pr[\mathcal{E}_i] = \frac{1}{k}$ for all i. Now, suppose that the absent minded professor sits on seat $i \neq 1$. Then all guests from the second one to the $i-1$st one – there are $i-2$ of them – sit on their assigned seats, while the ith guest sits on a seat chosen uniformly at random among seats $1, i+1, i+2 \ldots k$. If we rename seats $i+1, i+2, \ldots k$ to $2, 3, \ldots k-i+1$, we get the same situation with $k-i+1$ guests. (Here is where the definition of X is useful, since no matter where the ith guest sits, she does not sit on her assigned seat.) Thus, we have $E(X|\mathcal{E}_i) = (i-2) + e_{k-i+1}$.
The rest is a matter of calculation. We have

$$
\begin{aligned}
e_k &= \frac{k-1}{k} + \frac{1}{k}\sum_{i=2}^{k}((i-2) + e_{k-i+1}) \\
&= \frac{k-1}{k} + \frac{1}{k}\sum_{j=1}^{k-1}(k-1-j+e_j) \\
&= \frac{k-1}{k} + \frac{1}{k}\left(\sum_{j=1}^{k-2}(k-1-j+e_j) + e_{k-1}\right) \\
&= \frac{k-1}{k} + \frac{1}{k}\left(\sum_{j=1}^{k-2}(k-2-j+e_j) + (k-2) + e_{k-1}\right)
\end{aligned}
$$

Since

$$e_{k-1} = \frac{k-2}{k-1} + \frac{1}{k-1}\sum_{j=1}^{k-2}(k-2-j+e_j),$$

we have

$$\sum_{j=1}^{k-2}(k-2-j+e_j) = (k-1)e_{k-1} - (k-2).$$

Thus,

$$e_k = \frac{k-1}{k} + e_{k-1},$$

which yields

$$e_k = k - (1 + 1/2 + 1/3 + \dots 1/k).$$

Hence, $E(Y) = 1/k + e_k = k - (1 + 1/2 + \dots 1/(k-1))$, which is around $k - \ln k$.

- **Second solution**. Let x_i be the probability that the ith guest sits on her assigned seat, then x_i is also the probability that seat i is available when the ith guest comes. By linearity of expectation, the expected number of guests sitting on their assigned seats is $\sum_{i=1}^{n} x_i$. Clearly, $x_1 = 1/n$.
  Now, suppose $i > 1$. Consider the time when the ith guest comes. At that time, all seats from 2 to $i-1$ must be occupied. Therefore, exactly one among the remaining $n-i+2$ seats is unavailable. Since all these seats look exactly the same to the first $i-1$ guests, they are unavailable with the same probability. In particular, the probability that seat i is unavailable is $1/(k-i+2)$. Therefore, $x_i = 1 - 1/(n-i+2)$. Plug this in, we get the expected number of guests who sit on their assigned seats is:

$$
\begin{aligned}
1/n + \sum_{i=2}^{n}\left(1 - \frac{1}{n-i+2}\right) &= 1/n + \sum_{j=2}^{n}(1 - 1/j) \\
&= n - (1 + 1/2 + \dots 1/(n-1))
\end{aligned}
$$

4. We start with stating some observations. First, since the communication system has $|V_1| \cdot |V_2| \cdots |V_n|$ polynomial size states, we can enumerate all its states using polynomial space. Second, given two states $\mathbf{a} = (a_1, a_2, \dots a_n)$ and $\mathbf{b} = (b_1, b_2, \dots b_n)$, we can check if $(\mathbf{a}, \mathbf{b}) \in T$ using polynomial space by walking through all index i and checking if $a_i = b_i$ or $(a_i, b_i) \in P$. Third, we can also check if a state $\mathbf{a} = (a_1, a_2, \dots a_n)$ is a deadlock using polynomial space by walking through all indices and check if there are two indices i and j such that there exist b_i, b_j where $(a_i, b_i) \in P$ and $(a_j, b_j) \in P$.

With these observations, we can solve ReachableDeadlock as follows. The algorithm walk through all states of the communication system and check if the current state $\mathbf{d}$ is a deadlock. If it is, the algorithm check if $\mathbf{d}$ can be reached from the starting state $\mathbf{s}$ in exactly the same way with the proof of Savitch's theorem.

To see that this algorithm uses polynomial space, note that if $\mathbf{d}$ is reachable, then the minimum number of steps to go from $\mathbf{s}$ to $\mathbf{d}$ is at most the number of states, whose log is a polynomial on the size of the input. Thus, the recursion stack contains a polynomial number of items at any time. Furthermore, each of the items is of polynomial size. Therefore, the size of the recursion stack is a polynomial on the size of the input, which shows that the algorithm uses polynomial space.

5. We give a reduction from IPA to ReachableDeadlock. Given a machine M and a string x, we first modify M to get a machine M' that works exactly like M except that it rejects whenever the head is on the first cells outside of the input so that M' never uses more than $|x| + 2$ cells in its computation. Clearly, M accepts x inplace iff M' accepts x. In the following, we will refer to the two special cells next to either ends of the input the *bad* cells.

   We will build a system of communicating processes such that the states of the system correspond to the configurations of M', the transitions between states correspond to the transitions between configurations and the deadlock states correspond to the accepting configurations. To begin with, let $|x| = n$, we create $n + 2$ processes $G_0, G_1, \dots G_n, G_{n+1}$ where $G_1, G_2, \dots G_n$ correspond to the n cells holding the input and G_0 and G_{n+1} corresponding to the two bad cells. In the followings, we describe the vertex sets V_i and edge sets E_i of these processes.

   Observe that we can represent configurations of M' as strings of $n + 2$ elements, where each element is either a tape symbol or a pair of a tape symbol and a state of M'. For example, the string

$$\mathbf{y} = y_0 y_1 \dots y_{i-1}(y_i, q) y_{i+1} \dots y_n y_{n+1}$$

   represent the configuration in which the content of the tape is $y_0 y_1 \dots y_{i-1} y_i y_{i+1} \dots y_n y_{n+1}$, the head of M' is at position i and M' is in state q. Clearly, in order for a string to be a valid representation of a configuraiton, exactly one of its element must be a (tape symbol, state) pair.

   Let S be the set of all possible elements of such string, that is $S = \Sigma \cup (\Sigma \times Q)$ where Σ is the set of tape symbols and Q is the set of states of M'. We set V_i to be a "marked copy" of S, i.e. $V_i = \{s^i | s \in S\}$ so that V_i and V_j are disjoint for all $i \neq j$. Finally, we set $E_i = V_i \times V_i$. It can be verify that each configuration can be represented by a state of the described system. On the other hand, there are states of the system that are not the representation of any configuration; for example, the state $(a_0, a_1, \dots a_{n+2})$ where both a_1 and a_2 are (tape symbol, state) pairs. However, by setting the starting state to be the representation of the starting configuration of M' and specifying the set of transition pairs appropriately, we will make sure that all reachable states represent configurations.

   Now, we specify the set P of transition pairs so that each pairs represent a step in the computation of M. P contains all pairs $((s_1^i, s_2^i), (s_3^{i+1}, s_4^{i+1}))$ such that: (i) exactly one of s_1 and s_3 and exactly one of s_2 and s_4 are (tape symbol, state) pairs; and (ii) (s_1, s_3) and (s_2, s_4) are y_i, y_{i+1} and z_i, z_{i+1} respectively where $\mathbf{y}$ and $\mathbf{z}$ represents 2 consecutive configurations in the computation of M'.

   To see the correspondence between elements of P and the computation of M', consider any particular rule of M':

   > *If the current symbol is α and the current state is q, write β, move to the right and change to state q'.*

   This rule corresponds to all pairs of the form

$$(((\alpha, q)^i, \beta^i), (\gamma^{i+1}, (\gamma, q')^{i+1}))$$

   for all i and γ.

   Up to this point, it is easy to verify that for any two states $\mathbf{a} = (a_1, a_2, \dots a_n)$ and $\mathbf{b} = (b_1, b_2, \dots b_n)$ which represents configurations of M', $(\mathbf{a}, \mathbf{b}) \in T$ iff they represent two consecutive configurations.

Therefore, the reachable states of the system represents the reachable configurations of M'. Now, as stated above, we will make sure that the only reachable deadlocks are those represent the accepting configurations. Clearly, the only configurations where M' (therefore, the system) gets stuck is the rejecting and accepting ones. So, we add the following transition pairs:

$$(((\alpha^i, r), (\alpha^i, r)), (\gamma^{i+1}, \gamma^{i+1})) \text{ and } ((\alpha^i, \alpha^i), ((\gamma^{i+1}, r), (\gamma^{i+1}, r)))$$

for all α, γ and $0 \leq i \leq n$ where r is the rejecting state of M'. This make sure that the system of communicating processes doesn't get stuck on rejecting configurations of M; thus completes the reduction.

The corresponding between the reachable states and the configuration of M is clear. Furthermore, we can verify that starting from a state representing a reachable configuration, there is exactly one state that the system can transform into, and that state represents the next configuration. Thus, the system reach a deadlock iff M' reach an accepting configuration. Finally, given two states $\mathbf{a}$ and $\mathbf{b}$, we can check if $(\mathbf{a}, \mathbf{b}) \in P$ in polynomial time by looking at the transition table of M'.

This completes the proof that ReachableDeadlock is $PSPACE$-hard.