Randomised Algorithms
Winter term 2022/2023, Exercise Sheet No. 7

Authors:
Ben Ayad, Mohamed Ayoub
Kamzon, Noureddine

December 4, 2022

Exercise 1.

(a) *We have for $t \geq 1$:* $X_{t+1} = \begin{cases} X_t + [X_t/2] & \text{with probability 18/37} \\ X_t - [X_t/2] & \text{Otherwise} \end{cases}$

Technically speaking, we can never reach the state 0, as the player gets stuck when $X_t = 1$. We'll define $Y_t = X_t - 1$. We have: $Y_{t+1} = \begin{cases} Y_t + [\frac{Y_t+1}{2}] & \text{with probability 18/37} \\ Y_t - [\frac{Y_t+1}{2}] & \text{Otherwise} \end{cases}$

Which yields, for $s \geq 1$:

$$\begin{aligned} \mathbb{E}[Y_t - Y_{t+1} | Y_t = s] &= s - \mathbb{E}[Y_{t+1} | Y_t = s] \\ &= s - (\frac{18}{37}(s + [\frac{s+1}{2}]) + \frac{19}{37}(s - [\frac{s+1}{2}])) \\ &= \frac{1}{37}[\frac{s+1}{2}] \end{aligned}$$

Hence, $\forall s \geq 1 : \mathbb{E}[Y_t - Y_{t+1} | Y_t = 1] = \frac{1}{37} \leq \mathbb{E}[X_t - X_{t+1} | X_t = s]$. Using the Additive Drift Theorem, we get:

$$\mathbb{E}[T | Y_0] \leq 37 Y_0 \implies \mathbb{E}[T | X_0] \leq 37(X_0 - 1)$$

(b) *We have, using the Multiplicative Drift Theorem with tail bounds, $\forall r \geq 0$:*

$$\mathbb{P}[T > [\frac{r + \ln(x_0)}{\delta}] | X_0 = x_0] \leq e^{-r}$$

For $r = 2000\delta - \ln(x_0)$, we get $[\frac{r+\ln(x_0)}{\delta}] = 2000$, and hence:

$$\mathbb{P}[T > 2000 | X_0 = 10^6] \leq e^{-(2000\frac{1}{37} + \ln(10^6)} \leq e^{-40}$$

Exercise 2.

(a) *The worst case is to have all pairs $\{a_i, a_j\}$ s.t: $i < j$ not ordered, in this case $I(S_0) = \binom{n}{2} = \frac{n(n-1)}{2}$. We can achieve so, by taking a strictly ordered list, and inverse the order.*

(b) *We let $S_t = [a_1, \ldots, a_n]$ and $S_{t+1} = [b_1, \ldots, b_n]$. We have, $b_i = a_j$ and $b_j = a_i$. And let $I(S_t)$ be the set of pairs in the wrong ordering (similary for $I(S_{t+1})$). We need to prove that:*

$$I(S_{t+1}) \subsetneq I(S_t)$$

For the strictness, obsiously, $\{a_i, a_j\} \in I(S_t)$ *but* $\{a_i, a_j\} = \{b_j, b_i\} \notin I(S_{t+1})$ *(by definition of* S_{t+1}*), now we need to prove the inclusion:*

Let $\{b_k, b_l\} \in I(S_{t+1})$ *s.t:* $k < l$*: we have of course* $b_l < b_k$*. We distinguish 5 cases (and we ommit the trivial/extreme cases where k=1 or l=n):*

If $k \neq i$ *and* $l \neq j$*:*
Then $(b_k, b_l) = (a_k, a_l)$ *and hence,* $\{a_k, a_l\} \in I(S_t)$*.*

If $k = i$ *and* $j < l$ ***(Case-A)*** *:*
We have: $a_l = b_l < b_k = b_i = a_j \implies \{a_l, a_j\} \in I(S_t) \implies \{b_l, b_i\} \in I(S_t)$

If $k = i$ *and* $i < l < j$ ***(Case-B)****:*
We have: $a_l = b_l$ *(as* $l \neq i$ *and* $l \neq j$*)*
& $b_l < b_k$ *(as* $\{b_l, b_k\} \in I(S_{t+1})$*)*
& $b_k = b_i$ *(We are in the case where* $k = i$*)*
& $b_i < b_j$ *(By contsruction of* S_{t+1}*)*
& $b_j = a_i$ *(By construction of* S_{t+1}*)*

And hence: $a_l < a_i$*, but* $i < l$*, which yields,* $\{a_l, a_i\} \in I(S_t)$ *(we have* $a_i = a_k$*, so this case is complete).*

If $k < i$ *and* $j = l$*: similar logic to* ***Case-A****.*
If $i < k < l$ *and* $j = l$*: similar logic to* ***Case-B****.*

And hence, $I(S_{t+1}) \subset I(S_t)$*, we already proved that the inclusion is strict, given rise:*

$$INV(S_{t+1}) < INV(S_t) \implies INV(S_{t+1}) \leq INV(S_t) - 1 \implies 1 \leq INV(S_t) - INV(S_{t+1})$$

(c)

Exercise 3.

Let X_i *be the RV associated with step* i *of the algorithm and defined as follows:*

$$X_i = \begin{cases} 1 & \text{if at step } i\text{, we picked } j \text{ such as } B[j] = null \text{ (in one try)} \\ 0 & \text{otherwise.} \end{cases}$$

We have $\mathbb{P}[X_i = 1] = \frac{n-i+1}{n}$

Now let Y_i *be the number of tries we need at each step* i*, to find a null element in* B*. Abviously, the running time* T *could be expressed as follows:* $T = \sum_{i=1}^{n} Y_i$*, which yields,* $\mathbb{E}[T] = \sum_{i=1}^{n} \mathbb{E}[Y_i]$

Y_i *is a geometric RV, with parameter* $p = \mathbb{P}[X_i = 1]$*, hence,* $\mathbb{E}[Y_i] = \frac{n}{n-i+1}$*, which yields:*

$$\begin{aligned}\mathbb{E}[T] &= \sum_{i=1}^{n} \frac{n}{n-i+1} \\ &= n(1+\frac{1}{2}+\cdots+\frac{1}{n}) \\ &= nH_n \\ &= \Theta(n\log(n))\end{aligned}$$

Exercise 4.

The case where $n = 1$, is trivial, we will start our induction from $n = 2$.

Let's say $A = [a_1, a_2]$, we have two permutations in total, $A_1 = A$ and $A_2 = [a_2, a_1]$. In this case, the main loop of the algorithm has two runs, but the second run (and generally when $i = n$) doesn't not change the outcome of the list. Hence, the list is only changed in the first step ($i = 1$). Which yields two possibilities with equal probability, we either pick $j = 2$ (the algorithm swipes the two elements, and outputs, A_2), or we pick $j = 1$, and nothing changes to the list, and the algorithm outputs A_1. Hence, both A_1 and A_2 have equal probability 1/2 of being chosen by the algorithm. This proves the correctness in the case of $n = 2$.

Now, we suppose that the algorithm is correct for some $n \geq 2$, and prove that the algorithm is correct for an input of size $n + 1$:

Let T be a list of size $n + 1$, and let $L = [L_1, L']$ be a random permutation of T, where L_1 is the first element of L, and L', is random permutaion of $T/\{L_1\}$. We need to prove that $FastRandomPermutation(T)$, has a probability of $\frac{1}{(n+1)!}$ of outputing L.

For the first step, the algorithm picks at uniformely random an element of T, and swipes it with the first elemenet of T (T_1), Hence, we have get a probability of $\frac{1}{n+1}$ of actually picking L_1. For the rest, we are left with a list of size n, $T/\{L_1\}$, and since L' is a valid permutation of $T/\{L_1\}$, we get by induction, that the probability of picking L' in the steps $i \in \{2, \ldots, n+1\}$ is $\frac{1}{n!}$.

By the the multiplication theorem of probability, we conclude that the probability of outputing L by the algorithm is $\frac{1}{n+1}\frac{1}{n!} = \frac{1}{(n+1)!}$. This completes our proof by induction.