# HW3 Randomized Algorithms

MPRI 1.24

Thu. Feb. 6, 2014 - **Due on Thu. Feb. 13, 2014**

You are asked to complete the exercises and send me your solutions by email at:
nicolas.schabanel@liafa.fr
(or drop it in my mail box on the 4th floor) on or before **Thu. Feb. 13, 2014 at noon**.

■ **Exercise 1 (FPT algorithm for spotting $k$ disjoint triangles).** Given $G = (V, E)$ an undirected graph ($n = |V|$ and $m = |E|$) and $k$ an integer, we are looking for $k$ vertex-disjoint triangles in $G$. We are looking for an algorithm of time complexity $O(f(k)n^c m^{c'})$ where the exponents $c$ and $c'$ are constant, independent of $k$. Consider the following randomized algorithm

**Algorithm 1** FPT randomized algorithm for $k$ disjoint triangles

- Choose independently for each vertex $u$ a color $c_u \in \{1, \ldots, 3k\}$ uniformly at random.
- **return** "Yes" if there is a *colorful* solution, i.e. a set of $k$ triangles whose $3k$ vertices use exactly once each color; **return** "I don't know" otherwise.

► ***Question* 1.1)** *Show that if $G$ contains $k$ disjoint triangles, then the probability that the algorithm answer "Yes" is at least $e^{-3k}$.*
▷ Hint. *use that : $k! \geqslant (k/e)^k$ for all $k$.*

► ***Question* 1.2)** *How many times should you run this algorithm to improve success probability to $1/2$?*

In order to check whether a colorful solution exists, we propose to try all permutations $\pi$ on $\{1, \ldots, 3k\}$ and check if there is are triangles of colors $(\pi_1, \pi_2, \pi_3), \ldots, (\pi_{3k-2}, \pi_{3k-1}, \pi_{3k})$.

► ***Question* 1.3)** *Describe an algorithm that decides if such a collection of triangles exists. What is the overall expected time complexity in $k$, $n$ and $m$, of the algorithm that uses this method to return $k$ disjoint triangles with probability at least $1/2$ if they exists in $G$? What is the time complexity if $k$ is fixed?*

■ **Exercise 2 (Traffic monitoring: uniformity detection).** Imagine that we are running a huge website and we want to prevent attacks by keeping track of the origins of the various clients currently connected to the server. Along time, clients connect and then disconnect from the website. And we want to detect if all the clients connected are from the same IP address. But we do not want to slow down the server and wish to dedicate to this task only a *constant* memory, i.e. only a constant number of integers. We model the problem as follows:

We are given an infinite stream of events $e_1, e_2, \ldots, e_n, \ldots$ where each $e_i$ is either **connect**$(x)$ or **disconnect**$(x)$ where $x$ is a positive integer standing for the IP address of the client (dis-)connecting. We assume that the stream is wellformed, i.e. that there are always at least as many events **connect**$(x)$ as **disconnect**$(x)$ from the beginning of the stream to any position for every integer $x$. We want to detect when all the clients connected have the same IP address $x$.

▶ ***Question* 2.1)** *Spot when to set the alarm on in the following sequence where $x$ denotes the event* **connect**$(x)$ *and $\bar{x}$ the event* **disconnect**$(x)$*:*

$$1, 2, 3, \bar{2}, \bar{3}, 1, 1, \bar{1}, 4, 6, 7, \bar{1}, \bar{6}, \bar{1}, 2, \bar{2}, \bar{4}, 8, 3, \bar{3}, \bar{7}, 9$$

We consider the following algorithm that uses only three integer variables:

- start with $\mathtt{n} := 0$, $\mathtt{a} := 0$ and $\mathtt{b} := 0$ at $t = 0$;
- on event **connect**$(x)$: do $\mathtt{n} := \mathtt{n} + 1$, $\mathtt{a} := \mathtt{a} + x$ and $\mathtt{b} := \mathtt{b} + x^2$;
- on event **disconnect**$(x)$: do $\mathtt{n} := \mathtt{n} - 1$, $\mathtt{a} := \mathtt{a} - x$ and $\mathtt{b} := \mathtt{b} - x^2$;
- set on the alarm every time that $\mathtt{n} > 0$ and $\mathtt{b} = \mathtt{a}^2/\mathtt{n}$.

The right way to the correctness of this deterministic algorithm passes through the analysis of a random variable. Consider a random variable $X$ taking positive integer values. We denote by $\text{supp}(X) = \{ x : \Pr\{X = x\} > 0 \}$ and assume that $|\text{supp}(X)| < \infty$. We denote by $\mathbb{E}[X]$ and $\mathbb{V}\text{ar}[X] = \mathbb{E}[(X - \mathbb{E}(X))^2]$ respectively the expectation and the variance of $X$.

▶ ***Question* 2.2)** *Show that $|supp(X)| = 1$ if and only if $\mathbb{V}\text{ar}[X] = 0$.*

▶ ***Question* 2.3)** *Conclude that the algorithm is correct.*

■ **Exercise 3 (Matrix multiplication testing).**

Consider $\mathcal{H} = \{0, 1\}^2 \setminus \{0\}$ the set of non-zero vectors with $0, 1$-coordinates. We denote by $\langle a|b\rangle = a^t\, b = \sum_{i=1}^{n} a_i b_i$ the standard scalar product of two vectors $a, b \in \mathbb{R}^n$. For $a \in \mathbb{R}^n \setminus \{0\}$, let us denote by: $a^\perp = \{x \in \mathbb{R}^n : \langle x|a\rangle = 0\}$, the set of vectors orthogonal to $a$.

▶ ***Question* 3.1)** *Show that if $a \in \mathbb{R}^n$ is a non-zero vector, then there is an one-to-one function (an injection) from $\mathcal{H} \cap a^\perp$ to $\mathcal{H} \setminus a^\perp$, that is to say we can associate to every $x \in \mathcal{H}$ such that $\langle x|a\rangle = 0$, an $y \in \mathcal{H}$ such that $\langle y|a\rangle \neq 0$ in a one-to-one manner.*
▷ Hint. *Flip a well-chosen coordinate of $x$.*

▶ ***Question* 3.2)** *Conclude that if $a \in \mathbb{R}^n$ is a non-zero vector, the probability that $\langle a|u\rangle \neq 0$ when $u$ is chosen uniformly at random in $\mathcal{H}$, is at least $\frac{1}{2}$.*

We are given a program $M(A, B)$ which is supposed to compute efficiently the product of two $n \times n$ matrices $A$ and $B$ using a $O(n^{2.37})$-time sophisticated algorithm.

We consider the following testing procedure which verifies whether a $n \times n$-matrix $C$ is the product of the two $n \times n$-matrices $A$ and $B$:

**Procedure** MatrixMultiplicationTesting($A$,$B$,$C$)
1: Draw a random vector $u \in \{0, 1\}^n \setminus \{0\}$ uniformly at random
2: Compute the vectors $ABu$ and $Cu$
3: **if** $ABu \neq Cu$ **then**
4: **return** « $C$ is **not** the product of $A$ and $B$ »
5: **else**
6: **return** « No error detected »

▶ ***Question* 3.3)** *Show how to compute the product $ABu$ in $O(n^2)$ time (i.e. without computing the matrix product $AB$).*

▶ ***Question* 3.4)** *Show that in $C \neq AB$, then the* MatrixMultiplicationTesting *procedure detects the error with probability at least $1/2$.*
▷ Hint. *$AB = C \Leftrightarrow AB - C = 0 \Leftrightarrow$ all lines of $AB - C$ are $0$.*

► ***Question* 3.5*)*** *Design an algorithm that takes three $n \times n$-matrices $A$, $B$, $C$ and $\epsilon > 0$ as inputs and detects with probability at least $1 - \epsilon$ if $C = AB$. What is its time complexity in $n$ and $\epsilon$? What is its failure probability when $C = AB$?*