PECSN, 7° appello 2022

**Exercise 1**

A lake contains an unknown number of fishes, call it $n$. The environment protection officers take a (random) sample of $m$ fish from the lake, *mark* each of them with red ink (so that it will be recognizable later on), and return it to the lake.
Later, they take a second (random) sample of $k$ fish.
Call $X$ the RV that counts the number of red fishes in the second sample

1) Assume that $k = 1$. Compute the probability that $X = 1$.
2) Assume that $k = 2$. Compute the probability that $X = 0,1,2$.
3) For a generic $k$, and compute the probability that $X = x$, $0 \le x \le k$ (hint: reason about the probability model *first*).

Assume that $n \gg m \gg k$ from now on.

4) Find a suitable approximation for the previous expression;
5) Assume you observe $x$ red fish in a sample of $k$. Using the result of point 4), compute an estimate of $n$ and justify your result.

**Exercise 2**

Consider a system that can solve the same problem by running one among $n$ different algorithms for that problem on a single processor. The system solves one problem at a time, and only accepts a new problem when it is idle. Problems arrive at rate $\lambda$. When a problem is admitted, the system selects the algorithm to be run in a probabilistic way: the probability that algorithm $j$ is selected is equal to $\pi_j$. Each algorithm $j$ has an exponential running time, whose mean is $\frac{1}{\mu_j}$.

1) Provide a suitable model for the system

2) find the steady-state probabilities and the stability condition

3) Compute the condition on the probabilities $\pi_j$ such that it is *equally likely* to observe the system running any of the $n$ algorithms at the steady state

4) Compute the mean number of jobs in the system and the mean response time. Justify the result for the latter.

**Exercise 1 – solution**

1) The probability that a fish is red is clearly $\frac{m}{n}$, hence this is the answer.

2) The probability that both fishes are red is $p_2 = \frac{m}{n} \cdot \frac{m-1}{n-1}$. The probability that none are is $p_0 = \frac{n-m}{n} \cdot \frac{n-m-1}{n-1}$. The probability that one fish is marked is

$$1-p_2-p_0 = 1-\frac{m}{n}\cdot\frac{m-1}{n-1}-\frac{n-m}{n}\cdot\frac{n-m-1}{n-1}$$
$$= \frac{n^2-n-\left(m^2-m\right)-\left[\left(n-m\right)^2-\left(n-m\right)\right]}{n\cdot\left(n-1\right)}$$
$$= \frac{n^2-n-m^2+m-\left[n^2+m^2-2nm-n+m\right]}{n\cdot\left(n-1\right)}$$
$$= \frac{-m^2-m^2+2nm}{n\cdot\left(n-1\right)}$$
$$= 2\cdot\frac{\left(n-m\right)\cdot m}{n\cdot\left(n-1\right)}$$

3) The sample space is the set of all possible subset of $k$ fish taken from a set of $n$. Each subset is equally likely (sampling is done at random), hence we are in a UPM. The number of possible outcomes is therefore $\binom{n}{k}$. In order to count the favorable outcomes, we use the basic principle of counting: the favorable outcomes will be $A \cdot B$, where $A$ is the number of subsets of $x$ red fish taken from a set of $m$, and $B$ is the number of $k-x$ non-red fish taken from a set of $n-m$. Therefore, the answer is:

$P\{X = x\} = \frac{\binom{m}{x}\cdot\binom{n-m}{k-x}}{\binom{n}{k}}$.

4) There are at least two ways to answer this question. The first one is to observe that, if $n \gg m \gg k$, then $\frac{m-\alpha}{n-\alpha} \approx \frac{m}{n}$, $0 \le \alpha \le k$, hence the probability that the next fish will be red does not change significantly after you removed some fish from the lake. Therefore, you can regard picking red fish as a repeated trial experiment in (almost) independent conditions. Accordingly, the probability that you catch $x$ red fish in a set of $k$ will be (approximately) binomial, i.e.:

$P\{X = x\} \approx \binom{k}{x} p^x (1-p)^{k-x}$, with $p = \frac{m}{n}$.

You can get to the same result by simplifying the previous formula according to the approximations:

$$P\{X=x\}=\frac{\binom{m}{x}\cdot\binom{n-m}{k-x}}{\binom{n}{k}}$$

$$=\frac{\left(\frac{m!}{x!\cdot(m-x)!}\right)\cdot\left(\frac{(n-m)!}{(k-x)!\cdot[(n-m)-(k-x)]!}\right)}{\frac{n!}{k!\cdot(n-k)!}}$$

$$\approx\frac{\left(\frac{m^x}{x!}\right)\cdot\left(\frac{(n-m)^{k-x}}{(k-x)!}\right)}{\frac{n^k}{k!}}$$

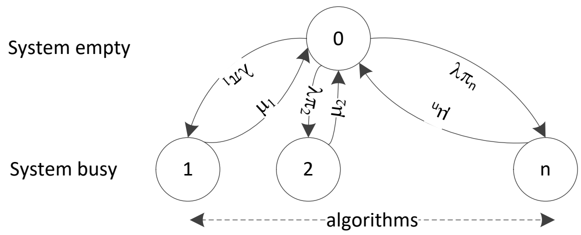
$$=\binom{k}{x}\cdot\left(\frac{m}{n}\right)^x\cdot\left(\frac{n-m}{m}\right)^{k-x}$$

5) Again, there are two ways to answer this question. The first one is acknowledging that taking the second sample is a bernoullian experiment. An estimate for the success probability of a bernoullian, given a sample of $k$ observations, is $p=\frac{x}{k}$. Since we know that $p=\frac{m}{n}$, then it follows that $n=\frac{m\cdot k}{x}$.

Alternatively, you can reason that, since you observed $x$ red fish in a sample of $k$, this can expected be the most likely outcome. The mode of a binomial distribution is around its mean value, which is $k\cdot p=k\cdot\frac{m}{n}$. Hence $x=k\cdot\frac{m}{n}$, which yields the same result.

**Exercise 2 – solution**

1) The system can be modeled by splitting probabilistically the arrival (Poisson) process using probabilities $\pi_j$. Algorithms are modeled as servers. The CTMC is the one below.




2) The system is always stable, since it has a finite queue (of one job). The SS probabilities can be computed using global equilibrium equations as follows:

$$\begin{cases} P_0\cdot\lambda=\sum_{i=1}^{n}P_i\cdot\mu_i \\ P_i\cdot\mu_i=P_0\cdot\lambda\cdot\pi_i \qquad 1\le i\le n\end{cases}$$

From which – by imposing normalization - one easily finds:

$$\begin{cases} P_0 = \dfrac{1}{1 + \sum_{j=1}^{n} \dfrac{\lambda \cdot \pi_j}{\mu_j}} \\ P_i = \dfrac{\lambda \cdot \pi_i}{\mu_i} \cdot \dfrac{1}{1 + \sum_{j=1}^{n} \dfrac{\lambda \cdot \pi_j}{\mu_j}} \qquad 1 \le i \le n \end{cases}$$

3) The condition by which all algorithms have the same probability to be observed running at the steady state is the condition by which $P_i = K$, $1 \le i \le n$. This is achieved if $\pi_i \propto \mu_i$: algorithms are as likely to be run as they are fast.

4) The system is empty in state 0 and holds one job in every other state. Hence $E[N] = 0 \cdot P_0 + (1 - P_0) \cdot 1 = 1 - P_0$. Moreover, it is $\overline{\lambda} = \lambda \cdot P_0$, since the system does not accept jobs while an algorithm is running, hence $E[R] = \frac{E[N]}{\overline{\lambda}} = \frac{1}{\lambda}\left(\frac{1}{P_0} - 1\right) = \sum_{i=1}^{n} \frac{\pi_i}{\mu_i}$. This last result has a straightforward interpretation: the only component of the response time is the service time, which is $\frac{1}{\mu_j}$ for algorithm $j$. However, algorithm $j$ is run with probability $\pi_j$, hence the sum.