Université Joseph Fourier GRENOBLE UFR IM²AG — Master of Science in Informatics at Grenoble — Grenoble INP ensimag

UE Mathematics for Computer Science

First session exam December 18, 2013 (3 hours)
Only personal hand-written notes are allowed.
Use separated sheets for problems 1-2 (part I) and problems 3-4 (part II).
All problems are independent from each other.
Number of points given for each problem is given for information purposes only and is subject to modifications without notice.

Part I

Problem : The Dining Philosophers

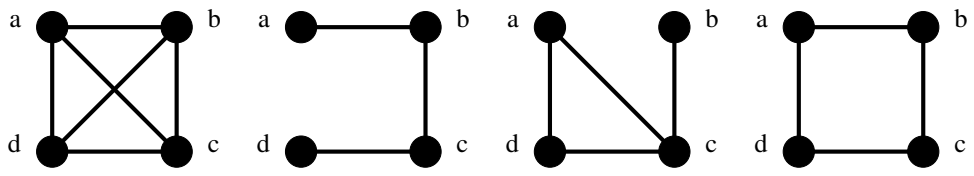
Consider the general problem of n dining philosophers. We suppose that a philosopher has neighbors and the set of philosophers $X = \{1, \cdots, n\}$ is modeled by a non directed graph $\mathcal{G} = (X, E)$ where nodes represent philosophers and edges the neighbor relation.

Philosophers could either eat or think, but two philosophers that are neighbors cannot eat simultaneously. Consequently, a configuration of the system is a subset A of X satisfying the constraint that for two nodes taken in A they are not neighbors in $\mathcal{G}$. In graph theory, these type of subsets are called *independent subsets*.

1 Configuration counting *(5 points)*

Question 1.1 : Configuration examples

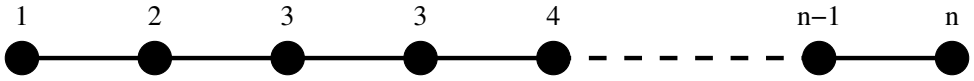
For the four following graphs give all the independent subsets.



Graph 1 Graph 2 Graph 3 Graph 4

Line graph

Let L_n be the number of independent subsets of the line graph with n nodes.



Line graph

Question 1.2 : Initial values

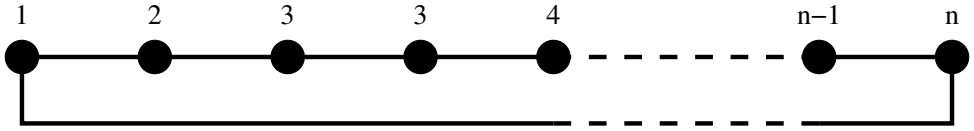
Compute L_1, L_2, L_3, L_4, L_5.

Question 1.3 : Recurrence formula

Use a combinatorial argument to establish a recurrence relation on the L_n. Then compute L_n, or refer to well known numbers.

Ring graph

Let C_n be the number of independent subsets of the ring graph with n nodes.



Ring graph

Question 1.4 : Initial values

Compute C_1, C_2, C_3, C_4, C_5.

Question 1.5 : Recurrence formula

Use a combinatorial argument to establish a relation between the C_n and the L_n. Then compute C_n.

2 Dynamic of the system *(5 points)*

Consider now the general problem with a given graph $\mathcal{G}$. The state of the system is the set of philosophers eating, an independent set. For the notation, we denote by 0 (respectively 1) the state of the philosopher thinking (resp eating). The state of the system is modeled by a n-dimensional vector $x = (x_1, \cdots, x_n)$ where $x_i = 0$ or 1 depending on the state of philosopher i. We suppose that the dynamic of the system is given by Algorithm 1 (next page).

```
Evolution ()
    Set initial value of philosophers ← {0, 0, ··· , 0}
    // initial independent set
    repeat
        i ← Pick uniformly a philosopher()
        if X[i]==1 Philosopher i is eating
            X[i]=0
        else Philosopher i is thinking
            if X[j]==0 for all j ∈ V(i) Philosopher i could eat
                // the neighbors are not eating
                if Random() < p the philosopher decides to eat with probability p
                    X[i] ← 1
    until the end
```

Algorithm 1: The dynamic of philosophers

Question 2.1 : A philosopher alone

Model the system with only one philosopher by a Markov chain (draw the graph and label the transitions) and compute the steady-state.

Question 2.2 : A couple of philosophers

Model the system with two connected philosophers by a Markov chain and compute the steady-state.

Question 2.3 : A ring of philosophers

Model the system with 4 connected philosophers on a ring (Graph 4) by a Markov chain and compute the steady-state.

Question 2.4 : Theoretical result ($\star\star$)

Consider a homogeneous irreducible and aperiodic Markov chain on $\{1, \cdots, K\}$ with a transition matrix $P = ((p_{i,j}))$. Suppose that we find a probability vector $\pi = (\pi_1, \cdots, \pi_K)$ satisfying

$$\pi_i p_{i,j} = \pi_j p_{j,i}, \text{ for all couple of states } (i, j).$$

Prove that π is the steady-state of the chain and give an interpretation of the previous balance equation.

Question 2.5 : General philosophers case ($\star\star\star$)

Prove that for philosopher's system for a graph $\mathcal{G}$ the steady state has the form

$$\pi(x_1, \cdots, x_n) = \pi(0, \cdots, 0) \left(\frac{p}{n}\right)^{\sum_i x_i}.$$

How $\pi(0, \cdots, 0)$ could be computed ?

Part II

3 Exercises *(5 points)*

This part contains some (easy) independent problems.

Question 3.1 :

Show that $\sqrt{2}$ is irrational.

I recall that an irrational number is any real number that can not be written as a fraction of two integers. Intuitively, this means that such numbers can not be represented by a simple fraction. They are those numbers that have an infinite number of digits in the decimal notation.

The proof is by contradiction and it uses two simple parity arguments. Thus, assume by contradiction that $\sqrt{2} = \frac{a}{b}$ where $GCD(a, b) = 1$.

Question 3.2 :

Show by the method of your choice that the product of any four consecutive integers is equal to a square minus 1:

$n(n+1)(n+2)(n+3) = K^2 - 1$

Question 3.3 :

Show the two following expressions using the pattern of Figure 1.

$$F(n+1)^2 = 4.F(n-1)^2 + 4.F(n-1).F(n-2) + F(n-2)^2 \quad (1)$$

$$F(n+1)^2 = 4.F(n)^2 - 4.F(n-1).F(n-2) - 3.F(n-2)^2 \quad (2)$$

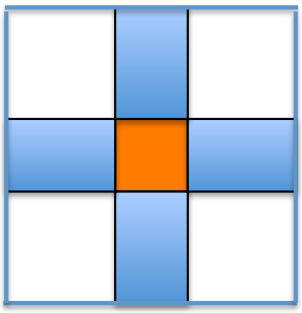


Figure 1: Basic pattern

4 Problem : Filling a knapsack *(5 points)*

This exercise is an analysis of algorithms for solving the knapsack problem.
This problem is well-known: we are given n *items* (non negative integers). Each item i has a volume w_i and a profit c_i, K is the global capacity of the sack (its maximum volume). The problem is to determine a subset of these items that maximize the total profit without exceeding the capacity.
Let us consider the following algorithm (called algorithm A) which sorts the items by non-increasing density ratio (where the density of item i is defined by $\frac{c_i}{w_i}$) and select them *greedily* according to this order as long as they fit into the sack. Greedily means that the choices are done one after the other, and once a choice is done, it is never reconsidered.

Question 4.1 :

Show that there exist some instances where the total value is arbitrarily far from the optimal[1].

Question 4.2 :

We propose now to slightly modify the previous algorithm as follows: Consider the first item that does not fit into the sack while using algorithm A (let call k this item). Justify the following relation: $c(OPT) \leqslant c(A) + c_k$ where $c(A)$ is the cost of all selected items by using algorithm A ($c(OPT)$ is the cost of an optimal solution for the considered instance).

Question 4.3 :

Using the previous relation, derive an algorithm with an approximation ration of $\frac{1}{2}$.

[1] it is enough here to consider a simple instance with only two items...