PSI* 2025/26 TP info 2

Couverture optimale ; allocation de ressources

Réaménagement de l'épreuve X PSI 2012

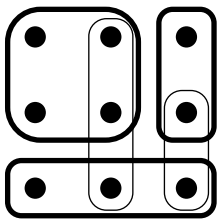
Motivation

Ce texte a pour motivation principale le problème d'allocation de ressources. L'approche choisie consiste ici à considérer ce problème comme une instance de celui de la couverture optimale d'ensemble.

Problème de la couverture optimale

Le problème s'énonce alors comme suit. Soit U un ensemble fini. Étant donnée une famille F constituée d'ensembles inclus dans U telle que F couvre U (c'est-à-dire que chaque élément de U appartient à au moins un ensemble de F), de combien d'ensembles de F a-t-on besoin, au minimum, pour couvrir U ?

Dans l'exemple ci-dessous, l'ensemble U est constitué de 9 points. La famille F contient 5 ensembles et recouvre U. Aucune sous-famille de F constituée d'au plus 2 ensembles n'est couvrante. En revanche, la sous-famille constituée des 3 ensembles en traits épais sur le dessin est couvrante. Une telle famille est dite *optimale*.



Quitte à numéroter les points de U, on peut considérer que U est l'ensemble $U_n = [\![0, n-1]\!] = \{0, 1, \ldots, n-1\}$.

Exemple préliminaire : allocation de ressources

Une école souhaite proposer des activités sportives à ses élèves. Il y a plusieurs activités possibles, et chaque élève est invité à dire lesquelles lui conviennent (en en choisissant au moins une). L'école souhaite minimiser le nombre d'activités à organiser, tout en garantissant que chaque élève puisse suivre une activité qui lui convient.

Question 1 Justifier de façon précise que ce problème peut être vu comme un problème de couverture optimale.

Opérations considérées comme "élémentaires" ou non

On pourra utiliser les opérateurs arithmétiques usuels sur les entiers (+,-,*, //, %, considérés chacun comme "élémentaire" dans les calculs de complexité) mais **pas les fonctions puissances et logarithmes prédéfinies** du langage Python.

La complexité, ou le temps d'exécution, d'une fonction ou d'un programme P est le nombre d'opérations élémentaires (opérations arithmétiques précédentes, opérateurs de test classiques, affectation, etc...) nécessaires à l'exécution de P. On ne donnera pas la valeur exacte de ce nombre d'opérations, mais on cherchera à l'estimer en fonction d'un paramètre n sous la forme $O(f(n))$: ainsi $O(n)$, $O(n^2)$ sont des formes de réponse convenables.

I Opérations sur les sous-ensembles finis de $\mathbb{N}$

Dans cette partie, on représente les ensembles finis d'entiers positifs ou nuls par des tableaux de booléens (en Python : utiliser le type list). L'ensemble $\mathcal{S}$ correspondant au tableau `S` contient exactement les entiers `i` tels que `S[i]` est égal à `True` ; S ne contient pas les entiers `i` tels que est égal à `False`, ou tels que `S[i]` n'est pas défini.

Ainsi, le tableau [`False`, `True`, `True`, `False`, `True`] représente l'ensemble $\{1, 2, 4\}$. On remarque qu'un même ensemble $\mathcal{S}$ peut être représenté de multiples façons, en jouant sur la longueur du tableau. Par exemple [`False`, `True`, `True`, `False`, `True`, `False`] et [`False`, `True`, `True`, `False`, `True`, `False`, `False`, `False`, `False`] sont d'autres représentations de l'ensemble $\{1, 2, 4\}$.

Question 2 Écrire une fonction `cardinal(S)`, qui prend en argument un tableau `S` de booléens représentant l'ensemble $\mathcal{S}$, et qui renvoie le cardinal de l'ensemble $\mathcal{S}$.

Question 3 Écrire une fonction `diff(S1,S2)`, qui prend en argument deux tableaux de booléens `S1` et `S2` représentant les ensembles $\mathcal{S}_1$ et $\mathcal{S}_2$, et qui renvoie un tableau de booléens représentant la différence ensembliste $\mathcal{S}_1 \setminus \mathcal{S}_2$. Ici $\mathcal{S}_1 \setminus \mathcal{S}_2$ désigne l'ensemble constitué des éléments de $\mathcal{S}_1$ qui ne sont pas dans $\mathcal{S}_2$.

Question 4 Écrire une fonction `union(S1,S2)`, qui prend en argument deux tableaux de booléens `S1` et `S2` représentant les ensembles $\mathcal{S}_1$ et $\mathcal{S}_2$, et qui renvoie un tableau de booléens représentant l'union de $\mathcal{S}_1$ et $\mathcal{S}_2$.

II Représentation des parties finies de $\mathbb{N}$ par des entiers

Par "ensemble" on désigne ici exclusivement des parties finies de $\mathbb{N}$. Afin de pouvoir manipuler simplement des familles de tels ensembles, on souhaite maintenant représenter les ensembles par des entiers. Pour ce faire, on associe à un ensemble $\mathcal{S}$ (d'entiers positifs ou nuls) l'entier $s = \sum_{i \in \mathcal{S}} 2^i$. Par exemple, l'ensemble $\{1, 2, 4\}$ correspond à l'entier $2^1 + 2^2 + 2^4$, soit 22.

Contrairement à la représentation par tableau de booléens, cette représentation est unique. On prendra garde à ne pas confondre les entiers représentant des ensembles avec les entiers contenus dans ces ensembles.

Question 5 Écrire une fonction `set2int(S)` qui prend en argument un tableau `S` de booléens représentant un ensemble $\mathcal{S}$, qui renvoie l'entier `s` représentant $\mathcal{S}$, **et dont le temps d'exécution est linéaire en la taille de `S`.** *On rappelle que l'utilisation de fonctions puissance et logarithme prédéfinies dans le langage est interdit.*

Question 6 Si $U = \{0, 1, 2, ..., n-1\}$, quel est le plus grand entier représentant un sous-ensemble de U ? Quel est le plus petit ?

Question 7 Écrire une fonction `int2set(s)` qui prend en argument un entier `s` représentant un ensemble $\mathcal{S}$, et qui renvoie un tableau de booléens représentant $\mathcal{S}$.

En estimer le temps d'exécution en fonction de `s`.

On spécifiera en en-tête de la fonction quel tableau particulier la fonction renvoie parmi les multiples représentations possibles.

Question 8 On souhaite manipuler dorénavant la représentation des parties finies de $\mathbb{N}$ par des entiers. En utilisant les fonctions précédemment définies, rédiger des fonctions répondant aux spécifications suivantes :

- `cardE(s)` : prend en argument un entier `s` et renvoie le cardinal de l'ensemble $\mathcal{S}$ codé par s.
- `unionE(s1,s2)` : prend en argument deux entiers `s1` et `s2`, codant deux ensembles $\mathcal{S}_1$ et $\mathcal{S}_2$, et renvoie l'entier représentant l'ensemble $\mathcal{S}_1 \cup \mathcal{S}_2$.
- `diffE(s1,s2)` : prend en argument deux entiers `s1` et `s2`, codant deux ensembles $\mathcal{S}_1$ et $\mathcal{S}_2$, et renvoie l'entier représentant l'ensemble $\mathcal{S}_1 \setminus \mathcal{S}_2$.

III Couvertures optimales

On aborde maintenant le problème de couverture optimale, défini en préambule du sujet.

On peut représenter une famille finie $\mathcal{F} = (\mathcal{F}_0, \mathcal{F}_1, ..., \mathcal{F}_p)$ constituée d'ensembles finis d'entiers (positifs ou nuls) par un tableau f, dont chaque case f[ℓ] est l'entier qui représente l'ensemble $\mathcal{F}_\ell$ au sens de la partie II. Par exemple, le tableau $[17, 3, 8, 22]$ représente la famille $(\{0,4\},\{0,1\},\{3\},\{1,2,4\})$, puisque $2^0 + 2^4 = 17$, $2^0 + 2^1 = 3$, $2^3 = 8$ et $2^1 + 2^2 + 2^4 = 22$.

On fixe désormais $U = \{0, 1, 2, ..., n-1\}$ et une famille $\mathcal{F}$ d'ensembles inclus dans U telle que $\mathcal{F}$ couvre l'ensemble U. Dans tout le reste du sujet, on suppose que $\mathcal{F}$ est donnée par un tableau f d'entiers. On fera donc systématiquement passer le tableau f et l'entier n dans toutes les fonctions, sans répéter ce qu'ils représentent.

Codage d'une sous-famille de $\mathcal{F}$ par un entier

Une sous-famille $\mathcal{G}$ de $\mathcal{F}$ est constituée de certains des ensembles de la famille $\mathcal{F}$, et est donc de la forme $(\mathcal{F}_{\ell_0}, \mathcal{F}_{\ell_1}, ...)$, où $\ell_0, \ell_1, ...$ est une sous-suite finie strictement croissante des indices de f. Par conséquent $\mathcal{G}$ est définie par l'ensemble des indices $\{\ell_0, \ell_1, ...\}$. Le cardinal de $\mathcal{G}$ est donc le cardinal de l'ensemble $\{\ell_0, \ell_1, ...\}$. Au sens de la partie II, cet ensemble d'indices peut une nouvelle fois être représenté par un entier.

Par exemple, si $\mathcal{F}$ est la famille $(\{0,4\},\{0,1\},\{3\},\{1,2,4\})$, la sous-famille $\mathcal{G} = (\{0,4\},\{3\})$, qui est constituée des ensembles $\mathcal{F}_0$ et $\mathcal{F}_2$, est de cardinal 2 et est représentée par l'entier 5, puisque $2^0 + 2^2 = 5$.

Noter que désormais trois objets distincts sont représentés par des entiers.

- Les éléments de $U = \{0, 1, 2, ..., n-1\}$: dans ce cas la lettre i sera de préférence utilisée, et il n'y a pas de codage particulier, i représente l'entier i ;
- Les sous-ensembles de U : dans ce cas la lettre s sera de préférence utilisée ;
- Les sous-familles de $\mathcal{F}$: dans ce cas la lettre g sera de préférence utilisée.

Question 9 Écrire une fonction `estCouverture(g,f,n)` qui prend en outre en argument un entier `g` représentant une sous-famille $\mathcal{G}$ de $\mathcal{F}$ et qui renvoie `True` si $\mathcal{G}$ est couvrante et `False` sinon.

Estimer son temps d'exécution en fonction de n et du nombre p d'élements de la famille $\mathcal{F}$.

Question 10 Écrire une fonction `optimale(f,n)` qui parcourt toutes les sous-familles possibles de $\mathcal{F}$ afin de renvoyer un entier représentant une sous-famille couvrante de $\mathcal{F}$ optimale. Estimer son temps d'exécution.

IV Algorithme glouton

On pourrait penser à un démarche plus rapide pour trouver une sous-famille couvrante de petit cardinal : dans cet algorithme dit "glouton", l'idée est de construire la sous-famille étape par étape, en ajoutant à chaque fois l'ensemble qui permet de couvrir le plus de nouveaux éléments.

Question 11 Écrire une fonction `glouton(f,n)`, qui renvoie un entier `g` représentant une sous-famille couvrante $\mathcal{G}$ de $\mathcal{F}$ grâce à l'algorithme glouton décrit ci-dessus. Estimer son temps d'exécution.

Question 12 Proposer un exemple de famille où l'algorithme glouton ne renvoie pas une sous-famille couvrante optimale.