Synthèse du cours 11 : Introduction à l'analyse amortie

2 décembre 2024

François Laroussinie

NB : Ces synthèses ont pour but de compléter les notes prises en cours. Elles ne les remplacent pas ! En particulier, la plupart des preuves n'y figurent pas. Rappel : il faut programmer les algorithmes vus en cours.

1 Analyse amortie

Voir le livre *Algorithmes* de Cormen et al. pour plus de détails.

L'objectif est ici d'analyser la complexité d'une séquence de n instructions (toujours dans le pire cas) sur une structure de données, afin de pouvoir obtenir des résultats de complexité plus précis que l'approche classique (lorsque par exemple des opérations sont potentiellement coûteuses mais que celles-ci sont rares). Pour cela, on peut utiliser un des trois points de vue suivants :

- Méthode de l'agrégation : On évalue la complexité totale $T(n)$ de la séquence complète, et on dit que chaque instruction a une complexité amortie de $T(n)/n$.
- Méthode comptable : On attribue des *charges* à chaque instruction (ce sera le coût amorti de cette instruction). Si la charge est supérieure au coût réel de l'instruction, la différence est un crédit stocké dans la structure qui pourra être utilisé ultérieurement. Si la charge est inférieure au coût réel, on doit utiliser les crédits stockés (à condition qu'il y en ait !). Chaque opération/instruction a sa complexité amortie.
- Méthode du potentiel : on définit une fonction de potentiel (notée Φ) pour la structure de données analysée. Une opération peut conduire à augmenter le potentiel (que l'on pourra utiliser plus tard) ou diminuer le potentiel (on consomme alors des réserves... si il y en a !). Le coût amortie d'une instruction est alors défini par le coût réel PLUS la différence de potentiel induit par l'opération. NB : on prend, en général, une fonction de potentiel qui est nulle pour la structure dans l'état initial. On doit s'assurer qu'à tout moment le potentiel est positif (c'est-à-dire que l'on n'utilise pas des réserves dont on ne dispose pas).

2 Le compteur binaire

On considère un compteur binaire sur k bits et l'instruction (unique) d'incrémentation (+1). L'algorithme d'incrémentation est alors défini par l'algorithme suivant :

Procédure Incrément (C)
begin
  $i = 0$
  tant que $i < k \wedge C[i] == 1$ **faire**
    $C[i] = 0$
    $i++$
  si $i < k$ **alors** $C[i] = 1$

Notons que le bit de poids faible est le bit 0.
Le coût correspond ici au nombre de bascules d'un bit de 0 à 1 ou de 1 à 0. Notons que lors d'un incrément, il peut y avoir plusieurs bascules de 1 à 0, mais une seule bascule de 0 à 1.

Une séquence de n incréments a une complexité en $O(nk)$ selon les techniques habituelles.

En appliquant les outils de l'analyse amortie, on peut obtenir une résultat plus précis. On le présente selon les 3 approches évoquées précédemment :

— Méthode de l'agrégation : On peut observer que le bit de poids faible (0) change à chaque incrément, celui de poids 1 change un incrément sur deux, le bit de poids 2 change un incrément sur 4, etc. On peut alors observer que le coût total des n incréments est alors :

$$C(n) \;=\; \sum_{i=0}^{k-1} \lfloor \frac{n}{2^i} \rfloor \;<\; n \sum_{i=0}^{k-1} \frac{1}{2^i} \;<\; 2 \cdot n$$

On en déduit que que coût amorti d'un incrément est majoré par 2.

— Méthode comptable : on va attribuer deux crédits à un incrément. Un de ces crédit sert à la (unique) bascule de 0 à 1 et le second crédit est "stocké" avec ce nouveau 1 (et servira plus tard lors de sa bascule vers 0). Les bascules vers 0 sont ainsi déjà payées d'avance. On peut facilement vérifier (fait en cours) que cela suffit. Le coût amorti d'un incrément est alors de 2.

— Méthode du potentiel : on va définir $\Phi(C)$ (le potentiel associé à un codage du compteur C) comme le nombre de 1 présents dans la valeur du compteur écrite en binaire (*i.e.* le tableau de taille k). Le coût réel d'un incrément est alors égal à $1+t$ où t est le nombre de 1 qui passent à 0 lors de cet incrément (le 1 étant le coût de la bascule d'un 0 vers 1). Quel est le coût amorti (noté $\widehat{C_i}$) du i-ème incrément ? Après le i-ème incrément, le compteur vaut $\mathcal{C}_i = i \,\%\, 2^k$. Si on note b_i le nombre de 1 dans $\mathcal{C}_i$ et t_i le nombre de 1 qui passent de 0 à 1 lors de ce i-ème incrément, on a :

$$\widehat{C_i} \;=\; (t_i + 1) + \Phi(\mathcal{C}_i) - \Phi(\mathcal{C}_{i-1}) \;=\; t_i + 1 + b_i - b_{i-1}$$

Pour évaluer $b_i - b_{i-1}$, on distingue deux cas : le cas du reset (lorsque tous les bits sont passés à 0, donc lorsque i est un multiple de 2^k) et les autres :

— si $b_i = 0$ (reset), $b_{i-1} = k$ et $t_i = k$, d'où $b_i = b_{i-1} - t_i$ et donc $b_i - b_{i-1} \leq 1 - t_i$.

— si $b_i > 0$, $b_i = b_i - 1 - t_i + 1$, et donc $b_i - b_{i-1} \leq 1 - t_i$.

Dans les deux cas, on obtient : $\widehat{C_i} \leq (t_i + 1) + 1 - t_i = 2$. Et donc là encore, le coût amorti d'un incrément est en temps constant.

Dans chacune de ces approches, on trouve un coût amorti d'un incrément qui est constant, et ne dépend donc pas de k contrairement à l'approche standard !