Hash Code
by Google France

FR

Sujet de qualification, 12 mars 2015

Optimisation d'un centre de données

Introduction

Depuis plus de dix ans, Google construit des centres de données de sa propre conception, déployant des milliers d'ordinateurs tout autour du globe. Dans chacun de ces centres, des serveurs font fonctionner en permanence les services que nous utilisons tous les jours, du moteur de recherche Google et YouTube au système de juge de Hash Code.

La conception d'un centre de données est un problème d'optimisation à plusieurs facteurs. Nous souhaitons certainement obtenir autant de capacité de calcul que possible — nos services ont besoin de beaucoup de ressources aujourd'hui, et nous en demanderons encore plus à l'avenir. Cependant, maximiser la capacité brute n'est pas le seul objectif : il est aussi important de s'assurer que la capacité de traitement reste disponible même en cas de pannes matérielles.

Tâche

Les serveurs d'un centre de données sont **physiquement organisés en rangées**. Chaque rangée peut partager des ressources telles que l'alimentation électrique. Si une telle resource fait défaut, nous supposons que toute la rangée est perdue et tous les serveurs de la rangée deviennent **indisponibles**.

Les serveurs d'un centre de données sont également **organisés logiquement en groupes**. Chaque serveur appartient à exactement un groupe, et lui fournit une certaine quantité de ressources de calcul appellée **capacité**. La capacité d'un groupe est la somme des capacités des serveurs **disponibles** du groupe.

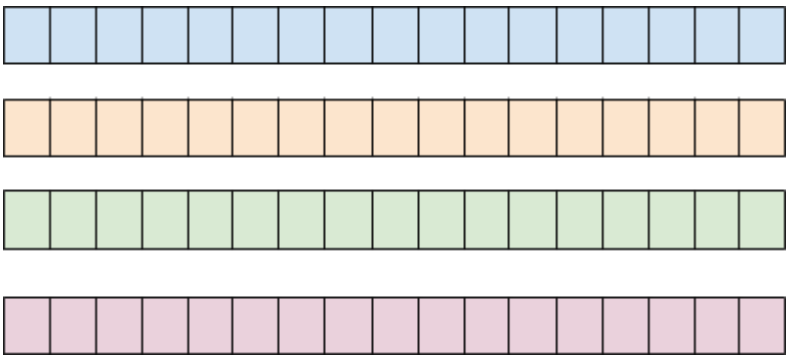
Pour assurer la fiabilité d'un groupe il est donc souhaitable de distribuer ses serveurs sur des rangées différentes, de telle sorte que, quand une rangée fait défaut, le groupe puisse continuer à opérer sur les serveurs des rangées restantes (quoique avec une capacité réduite à cause des serveurs indisponibles). La **capacité garantie** d'un groupe est la capacité minimum du groupe quand au plus une rangée du centre de données fait défaut.

Étant donnés le plan d'un centre de données et la liste des serveurs disponibles, votre mission est d'affecter les serveurs à des **emplacements au sein des rangées** et aux **groupes logiques** de manière à maximiser **la plus petite capacité garantie** de tous les groupes.

Description du problème

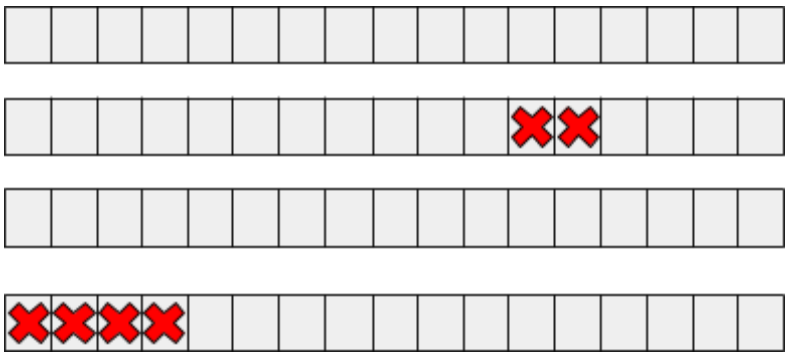
Emplacements

Un centre de données est modélisé par des **rangées** d'**emplacements** pouvant accueillir des serveurs.



Un centre de données avec quatre rangées.

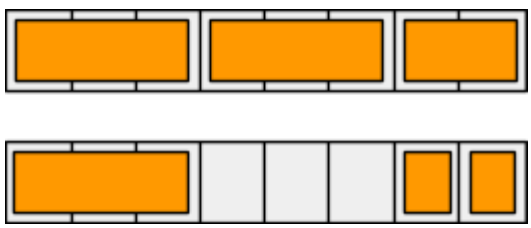
Certains de ces emplacements peuvent être indisponibles (par exemple à cause d'autres équipements les occupant).



Six emplacements indisponibles du centre de données marqués avec des croix rouges.

Serveurs

Chaque serveur est caractérisé par sa **taille** et sa **capacité**. La taille est le nombre d'emplacements consécutifs occupés par la machine. La capacité est la quantité totale de ressources CPU de la machine (une valeur entière).



Des serveurs de différentes tailles placés sur deux rangées.

Données d'entrée

Les données d'entrée sont fournies dans un fichier texte contenant exclusivement des caractères ASCII avec des lignes terminées par le caractère '\n' (fins de ligne UNIX).

Le fichier est constitué des lignes suivantes :

- une ligne contenant les cinq entiers naturels suivants, séparés par des espaces :
  - ***R*** $(1 \leq R \leq 1000)$: le nombre de rangées du centre de données,
  - ***S*** $(1 \leq S \leq 1000)$: le nombre d'emplacements de chaque rangée du centre de données,
  - ***U*** $(0 \leq U \leq R \times S)$: le nombre d'emplacements indisponibles,
  - ***P*** $(1 \leq P \leq 1000)$: le nombre de groupes à créer,
  - ***M*** $(1 \leq M \leq R \times S)$: le nombre de serveurs à allouer;
- ***U*** lignes consécutives décrivant les emplacements indisponibles. Chacune de ces lignes contient deux entiers naturels séparés par une espace, r_i et s_i $(0 \leq r_i < R, 0 \leq s_i < S)$, représentant le numéro de la rangée (r_i) et le numéro dans la rangée (s_i) de l'emplacement indisponible ;
- ***M*** lignes consécutives décrivant les serveurs à allouer. Chacune de ces lignes contient deux entiers naturels séparés par une espace, z_i et c_i $(1 \leq z_i \leq S, 1 \leq c_i \leq 1000)$, représentant la taille du serveur (z_i), i.e. le nombre d'emplacements qu'il occupe, et la capacité (c_i) du serveur.

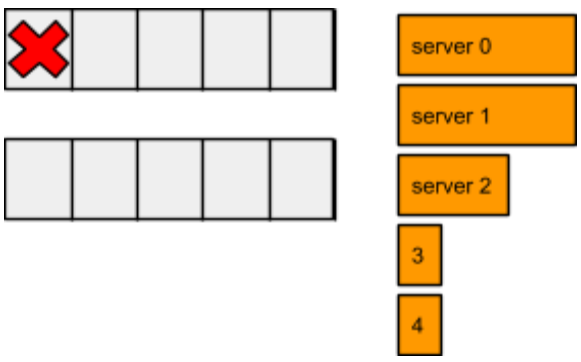
Exemple

Voici un exemple de fichier d'entrée.

`2 5 1 2 5`	`2 rangées de 5 emplacements, 1 emplacement indisponible, 2 groupes et 5 serveurs.`
`0 0`	`Coordonnées de l'unique emplacement indisponible.`
`3 10`	`Le premier serveur prend trois emplacements et a une capacité de 10.`
`3 10`	`De même pour le second serveur.`
`2 5`	`Le troisième prend trois emplacements et a une capacité de 5.`
`1 5`	`Le quatrième prend seulement un emplacement et a une capacité de 5.`
`1 1`	`Le cinquième prend un emplacement et a une capacité de 1.`

Exemple de fichier d'entrée.

Le fichier ci-dessus décrit un centre de données avec deux rangées de 5 emplacements chacune et des serveurs de différents paramètres.



Centre de données et serveurs décrits dans l'exemple ci-dessus.

Soumissions

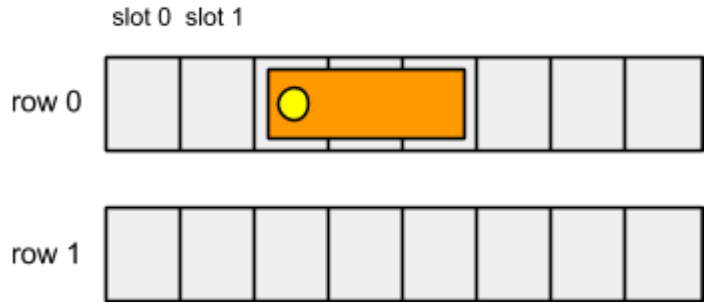
Format de fichier

Un fichier de soumission doit être un fichier texte brut contenant seulement des caractères ASCII, avec des lignes terminées soit par le seul caractère '\n' (style UNIX) ou les caractères '\r\n' (style Windows).

Le fichier doit contenir ***M*** lignes décrivant l'allocation des serveurs individuels, dans le même ordre que celui du fichier d'entrée. Chacune de ces lignes doit contenir soit:

- trois entiers naturels séparés par des espaces : ar_i, as_i et ap_i $(0 \le ar_i < R,\ 0 \le as_i < S,\ 0 \le ap_i < P)$, dénotant la rangée ($ar_i$) et l'emplacement dans la rangée (as_i) où le serveur est placé, et le groupe logique alloué pour le serveur (ap_i);
- la lettre minuscule "x" seule si le serveur n'est pas utilisé.

Pour les serveurs qui occupent plus d'un emplacement, l'emplacement de plus petit index doit être utilisé comme position du serveur.



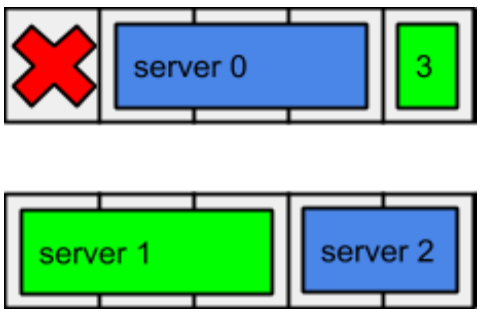
La position d'un serveur de taille 3 ci-dessus doit être décrite par ar_i = 0 et as_i = 2.

Exemple

Le fichier de soumission suivant correspond à l'exemple de fichier d'entrée donné ci-dessus.

`0 1 0`	`Serveur 0 placé rangée 0, emplacement 1 et affecté au groupe 0.`
`1 0 1`	`Serveur 1 placé rangée 1, emplacement 0 et affecté au groupe 1.`
`1 3 0`	`Serveur 2 placé rangée 1, emplacement 3 et affecté au groupe 0.`
`0 4 1`	`Serveur 3 placé rangée 0, emplacement 4 et affecté au groupe 1.`
`x`	`Serveur 4 non utilisé.`

Exemple de fichier de soumission.



Allocation des serveurs correspondant à l'exemple précédent.

Validation

Pour être acceptée, une solution doit satisfaire les critères suivants :

- le format du fichier doit correspondre à la description précédente,
- chaque emplacement du centre de données doit être occupé par au plus un serveur,
- aucun serveur ne doit occuper un emplacement non disponible du centre de données,
- aucun serveur ne doit s'étendre au-delà des emplacements de sa rangée.

Score

La **capacité garantie** de chaque groupe est définie comme la plus petite capacité totale de ces serveurs quand exactement une rangée du centre de données n'est pas disponible (parmi toutes les rangées existantes). Le score attribué à la soumission est la **plus petite** capacité garantie de tous les groupes. Le but est de maximiser ce score.

Exemple

Dans l'exemple de soumission ci-dessus, le score total est **5**, car la capacité garantie de chacun des deux groupes est 5 et $min(5, 5) = 5$. Voir la figure pour les détails.

	rangée 0	rangée 1	capacité garantie
groupe 0	10	5	**5**
groupe 1	5	10	**5**

Capacités garanties de chaque groupe dans la soumission d'exemple.

Formellement

Pour ***i*** ($0 \leq i < P$), la **capacité garantie *gc*$_i$** peut être définie par :

$$gc_i = min_{0 \leq r < R}(\sum_{k = 0, \text{ serveur } k \text{ dans le groupe } i}^{M-1} c_i - \sum_{k=0, \text{ serveur } k \text{ dans le groupe } i, \text{ serveur } k \text{ dans la rangée } r}^{M-1} c_i)$$

et le score total peut être défini par :

$$score = min_{0 \leq i < P} gc_i$$