Mapping, une optimisation Pseudo-Booléenne

Pierre Sauvage

20 mai 2014

1 Introduction

//TODO

2 Exposé du problème

On cherche à effectuer δ déploiements avec $\delta_{min} \leq \delta \leq \delta_{max}$
On a un ensemble de nœuds possédant des ressources typés. On a un ensemble de scopes possédants des ressources typés que l'on chercher à mapper. Toutes les transitions possèdent des arcs de sorties.
Ces arcs ont les caractéristiques suivantes :

— Une fréquence (entier) f d'utilisation sur une période donnée (ou une approximation de fréquence)

— Un poids (entier) p correspondant au nombre de paquets nécessaire pour envoyer la payload.

On considère donc le coût par arc de sortie a de s_1 vers un scope s_2 :

$$c_a = f \times p$$

On note l'ensemble A_{s_1,s_2} des arcs d'un scope s_1 vers un scope s_2
On peut donc déterminer, $\forall s1, s2 \in S$ le coût total :

$$c_{s_1,s_2} = \sum_{a \in A_{s_1,s_2}} c_a$$

Les arcs sont orientés : $A_{s_1,s_2} \neq A_{s_2,s_1}$ *donc* $c_{s_1,s_2} \neq c_{s_2,s_1}$

On peut classer les ressources d'un Scope dans deux catégories :

— Les ressources propres. Ce sont des ressources qui sont créés et utilisés par le scope que l'on souhaite mapper.

— Les ressources d'import. Ce sont par exemple des ressources métiers (propre au nœud), ou des ressources propres à un autre scope (d'un précédent mapping, généralement).

Un scope s est mappable sur un nœud n si et seulement si :

— Le nœud n possède une ressource libre de même type pour toute ressource propre.
— Le nœud n possède une ressource de même nom et de même type pour toute ressource d'import.

Les ressources de type "S" ne peuvent pas être créés par un scope, ce sont donc toujours des ressources d'import.
On cherche à minimiser :
— $\delta_{max} - \delta$
— la fonction de coût total.

3 Formalisation

Pour l'instant, on fixe δ : $\delta_{min} = \delta_{max}$

3.1 Approfondissement

Notre problème de mapping est une variante du problème du sac à dos (sac à dos multiple, sac à dos à choix multiple).
Soit N l'ensemble des n Nœuds $\{N_1, ..., N_n\}$
Soit S l'ensemble des m Scopes $\{S_1, ..., S_m\}$

Un type est une paire nom/ensemble On définit T l'ensemble des types, $|T| = t$.
Une ressource est un triplet composé de :
— Son nom
— Le nom de son type
— Sa valeur, élément de l'ensemble définit par son type

On appelle une ressource libre une ressource dont le nom est $\emptyset$.

$\forall n \in N$, on définit R_n l'ensemble des ressources. R_n admet un unique partitionnement par type admettant au plus t éléments. On notera $\dot{r}$, une ressource libre, $\dot{R_n}$ l'ensemble des ressources libres, et $\dot{R_n^j}$ l'ensemble des ressources libres de type j. On notera $\overline{r}$ une ressource existante, $\overline{R_n}$ l'ensemble des ressources existantes, et $\overline{R_n^j}$ l'ensemble des ressources existantes de type j.

Une ressource non-libre consomme un espace mémoire non nul. Les nœuds ont un espace mémoire limité. Cela rajoute une contrainte supplémentaire :

$$\forall n \in N : \sum_{r \in \overline{R_n}} \text{sizeof}(r) \leq \text{sizeof}(n)$$

$\forall s \in S$, on définit :
— $M(s) \in \mathbb{N}^+$, la multiplicité du scope

— $\dot{R_s}$ l'ensemble des ressources propres admettant au plus t éléments $\{\dot{R_s^1}, ..., \dot{R_s^k}\}$.
— $\overline{R_s}$ l'ensemble des ressources d'import admettant au plus t éléments

Un scope s est dit mappable sur un nœud n si et seulement si $\forall i \in T$:
— $\forall \dot{r_1} \in \dot{R_s^i}, \exists \dot{r_2} \in \dot{R_n^i}$
— $\forall \overline{r_1} \in \overline{R_s^i}, \exists \overline{r_2} \in \overline{R_n^i}$, tel que $\overline{r_1} = \overline{r_2}$

On appelle mapping l'opération qui consiste à associer d nœuds à un scope de multiplicité d. Le scope doit être mappable sur chacun des d nœuds

Les nœuds forment un réseau. Le graphe des nœuds est potentiellement orienté (communication unidirectionnelle). On associe donc à ce graphe une matrice de distance. On écrira chaque cellule $d_{l,c}$. Ainsi d_{n_1,n_2} représente la métrique entre n_1 et n_2. Si et seulement si le graphe est non-orienté, alors $\forall n_1, n_2 \in N : d_{n_1,n_2} = d_{n_2,n_1}$.
Dans le système EMMA, chaque nœud possède une table de routage aux dimensions finies. Cela implique qu'un nœud ne peut pas toujours communiqué avec un autre, et ce quelque soit la métrique. On définit la convention suivante : lorsqu'un nœud n_1 ne peut communiquer avec un nœud n_2, on associe une valeur clé $\lambda \in N^-$ à l'entrée d_{n_1,n_2} de la matrice de distance.

$\forall s \in S$, on définit $N_s \subseteq N$ tel que $\forall n \in N_s, \forall i \in T$:
— $|\dot{R_s^i}| \leq |\dot{R_n^i}|$
— $\forall \overline{r_1} \in \overline{R_s^i}, \exists \overline{r_2} \in \overline{R_n^i}$, tel que $\overline{r_1} = \overline{r_2}$
Si $\exists s$ tel que $N_s = \emptyset$ alors il n'y a pas de solution.

On définit $\forall s \in S, \forall n \in N_s$:
— $M(s)$ booléens $x^n_{s,1}, ..., x^n_{s,M(s)} \in X$
— $\forall i \in T, \forall r \in \dot{R_s^i}$, $M(s)$ booléens $y^n_{r,1}, ..., y^n_{r,M(s)} \in Y$

On remarque rapidement que :

$$|X| = \sum_{s \in S} M(s) \times |N_s|$$

$$|Y| = \sum_{s \in S} M(s) \times |\dot{R_s}| \times |N_s|$$

3.2 Formulation

On définit l'optimisation pseudo-booléenne à $|X| + |Y|$ littéraux :

$$\min\ z(X) = \sum_{s \in S} \sum_{n \in N_s} \sum_{s' \in S} \sum_{n' \in N_{s'}} c_{s,s'} d_{n,n'} \sum_{k=1}^{M(s)} \sum_{k'=1}^{M(s')} x_{s,k}^{n} x_{s',k'}^{n'}$$

On impose alors les contraintes suivantes :

$$\forall s \in S : \sum_{n \in N_s} \sum_{k=1}^{M(s)} x_{s,k}^{n} = \delta \times M(s) \tag{1}$$

$$\sum_{s \in S} \sum_{n \in N_s} \sum_{s' \in S} \sum_{\substack{n' \in N_{s'} \\ d_{n,n'} = \lambda}} c_{s,s'} \sum_{k=1}^{M(s)} \sum_{k'=1}^{M(s')} x_{s,k}^{n} x_{s',k'}^{n'} = 0 \tag{2}$$

$$\forall s \in S : \sum_{k=1}^{M(s)} \sum_{n \in N_s} [x_{s,k}^{n} \prod_{r \in \dot{R}_s} y_{r,k}^{n} \vee \neg x_{s,k}^{n} \prod_{r \in \dot{R}_s} \neg y_{r,k}^{n}] = M(s) \times |N_s| \tag{3}$$

$$\forall n \in N, \forall i \in T : \sum_{\substack{s \in S, \\ n \in N_s}} \sum_{k=1}^{M(s)} \sum_{r \in \dot{R}_s^i} y_{r,k}^{n} \leq |\dot{R}_n^i| \tag{4}$$

$$\forall n \in N : \sum_{\substack{s \in S, \\ n \in N_s}} \sum_{k=1}^{M(s)} \sum_{i \in T} \sum_{r \in \dot{R}_s^i} y_{r,k}^{n} \mathrm{sizeof}(r) \leq \mathrm{sizeof}(n) - \sum_{r' \in \overline{R_n}} \mathrm{sizeof}(r') \tag{5}$$

(1) Le nombre de map du scope équivaut au nombre de déploiement par la multiplicité du scope.

(2) Si un scope s mappé sur un nœud n possédant des arcs de sorties vers un scope s' mappé sur n'. ($c_{s,s'} x_s^n x_{s'}^{n'} > 0$) alors n doit pouvoir communiquer avec n' ($d_{n,n'} \neq \lambda$). Donc, la somme des coûts des paires de scopes ne pouvant communiquer doit être nulle.

(3) L'état des k^{e} ressources d'un scope sur un nœud est **équivalent** à l'état du k^{e} scope sur le nœud.

(4) Pour tout type, la quantité de ressources de ce type mappées sur un nœud ne doit pas dépasser la quantité disponible (le nombre de ressources libres) de ce nœud.

(5) La somme de l'espace mémoire emprunté par les ressources mappés doit être inférieur ou égal à l'espace disponible sur ce nœud.