Une heuristique à base de recherche locale pour la planification de tâches avec affectation de ressources

Bertrand Estellon[1], Frédéric Gardi[1,2], et Karim Nouioua[1]

[1] Laboratoire d'Informatique Fondamentale, Parc Scientifique et Technologique de Luminy, Marseille
{bertrand.estellon,karim.nouioua}@lif.univ-mrs.fr

[2] Experian-Prologia, Parc Scientifique et Technologique de Luminy, Marseille
frederic.gardi@experian-prologia.fr

Nous proposons une heuristique à base de recherche locale pour résoudre le problème proposé par la société France Telecom dans le cadre du Challenge ROADEF 2007. De manière générale, celui-ci peut être vu comme un problème de planification de tâches avec affectation de ressources. Ici, les tâches à planifier sont des interventions et leur réalisation nécessite l'affectation de ressources humaines, des techniciens, possédant certains niveaux de compétence dans différents domaines. Les interventions sont plus ou moins prioritaires ; au total, quatre priorités 1, 2, 3 et 4 sont définies. L'objectif est alors de minimiser un coût qui est fonction des temps de complétion des interventions de chacune des priorités.

1 L'heuristique générale

L'heuristique générale se décompose en quatre phases successives, chaque phase i s'intéressant à la planification des interventions de priorité i. L'objectif d'une phase i est de minimiser la date de fin t_i des interventions de priorité i, sans dégrader les dates de fin des interventions de priorité $i' < i$. Pour cela, un algorithme glouton complète la solution admissible héritée de la phase précédente à l'aide des interventions de priorité i. Ensuite, cette solution est modifiée par recherche locale de manière à diminuer t_i tout en maintenant les dates de fin $t_{i'}$ pour chaque priorité $i' < i$.

Plus précisément, l'étape de minimisation d'une date de fin t_i se fait comme suit. Étant donnée une solution admissible avec comme dates de fin $t_1, t_2, \ldots, t_i$, une nouvelle solution admissible avec comme dates de fin $t_1, t_2, \ldots, t_i - 1$ est recherchée. Au cours de la recherche, une intervention est dite non satisfaite si elle dépasse la date de fin $t_i - 1$ ou si l'équipe de techniciens à laquelle celle-ci est affectée ne possède pas toutes les compétences pour la réaliser. Dans ce cadre, l'objectif de la recherche locale est de minimiser le nombre d'interventions non satisfaites. Lorsqu'elle parvient à toutes les satisfaire, une nouvelle solution admissible est obtenue et le processus est réitéré.

Notons qu'une phase de prétraitement a été adjointe à l'heuristique générale de manière à pouvoir gérer les phénomènes de compensation entre les différents termes de la fonction de coût. En effet, dans certains cas, placer les interventions d'une priorité i avant les interventions d'une priorité $i' < i$ peut être plus favorable en terme de coût, les coefficients de la fonction objectif étant peu discriminants.

2 La recherche locale

La recherche locale consiste à exécuter de façon stochastique un ensemble de transformations modifiant la solution courante. Ces transformations sont de deux types, déplacement ou échange, et de deux catégories, les unes agissent sur les techniciens, les autres sur les interventions. Une transformation est acceptée à condition que la solution obtenue respecte les contraintes de précédence entre interventions, la durée maximale $H_{\max}$ d'une journée de travail et que le nombre d'interventions non satisfaites ne soit pas augmenté. Comme pour tout problème d'optimisation combinatoire, l'efficacité de la recherche locale repose essentiellement sur les deux points suivants, assurant un brassage important de solutions et donc une convergence rapide vers un optimum local de qualité : limiter le taux de rejet de chacune des transformations et accélérer le processus d'évaluation de chaque transformation.

Voici quelques idées concernant le premier point. Les transformations agissant sur les techniciens se déclinent sous trois formes. La première, générique, consistent à déplacer ou échanger des techniciens de façon aléatoire au sein d'une journée. La deuxième vise également à déplacer ou échanger des techniciens de façon aléatoire, mais au sein de journées où apparaissent des interventions non satisfaites. Enfin, la troisième consiste à déplacer ou échanger des techniciens appartenant à des équipes auxquelles sont planifiées des interventions non satisfaites. Les transformations agissant sur les interventions se déclinent également sous ces trois formes, mais avec une dimension supplémentaire : le déplacement ou l'échange peut se faire entre journées, dans une journée, ou bien au sein d'une équipe. En sus ont été ajoutées des transformations spécifiques pour gérer le budget et satisfaire les contraintes de précédence. Au total, 31 transformations ont été définies. En moyenne, le taux de réussite d'une transformation est supérieur à 10 % et peut monter à plus de 50 % à certains moments de la recherche. Il est à noter que la vitesse de convergence de la recherche locale dépend fortement du taux d'utilisation de chacune des transformations.

Quant au second point, accélérer le processus d'évaluation, il a pu être atteint en exploitant les nombreux invariants de structures lors de la modification de la solution courante. Nous ne pouvons détailler ici toute l'algorithmique qui a été mise en œuvre pour accélérer le processus d'évaluation. Cependant, il est à noter qu'un soin particulier a été apporté à l'implémentation du test de non dépassement de la durée $H_{\max}$ de chaque journée de travail ainsi que des dates de fin $t_1, t_2, \ldots, t_i - 1$; ce test, compliqué par les contraintes de précédence entre interventions, est réalisé par mise à jour incrémentale des plus longs chemins dans un graphe acyclique orienté (en partie inspirée du travail de Katriel, Michel et Van Hentenryck [1,2] sur le sujet). Vérifier qu'une équipe possèdent toutes les compétences requises par les interventions qui lui sont planifiées est également un test coûteux, car nécessitant le parcours d'une matrice domaines/niveaux. Nous avons veillé à ce que le parcours de cette matrice se réduise aux seules cases pour lesquelles un test est nécessaire.

En définitive, notre implémentation permet la tentative de plus d'un million de transformations par seconde, et ce même dans le cas d'instances volumineuses (800 interventions, 150 techniciens, 10 domaines de compétence) ; sur 20 minutes d'exécution, notre heuristique dépasse le milliard de solutions visitées.

3 Résultats expérimentaux

Nous avons implémenté notre heuristique en langage C ANSI (12000 lignes de code environ) et testé celle-ci sur un ordinateur doté d'un processeur Pentium 4 cadencé à 3 GHz et de 1024 Mo de RAM (notons que l'exécution de notre algorithme ne nécessite que quelques Mo de RAM). La Figure 1 ci-dessous récapitule les résultats que nous avons obtenus sur les deux jeux d'instances A et B fournis par France Telecom. Pour chaque instance, nous avons effectué trois essais et conservé le moins bon résultat (les écarts de résultat entre les trois essais sont faibles).

Instances A	Solutions FT	Solutions EGN	Instances B	Solutions FT	Solutions EGN
data1	2490	2340	data1	69960	33900
data2	4755	4755	data2	34065	16260
data3	15840	11880	data3	34095	16005
data4	14880	14040	data4	50340	23685
data5	41220	29400	data5	150360	88680
data6	30090	18795	data6	47595	27675
data7	38580	30540	data7	56940	36900
data8	26820	20100	data8	51720	36840
data9	35600	27440	data9	44640	32700
data10	51720	38460	data10	61560	41280

Fig. 1. Résultats expérimentaux : FT = France Telecom, EGN = Estellon, Gardi, Nouioua.

Références

1. I. Katriel, L. Michel et P. Van Hentenryck (2005). Maintaining longest paths incrementally. *Constraints* 10(2), pp. 159–183.

2. L. Michel et P. Van Hentenryck (2003). Maintaining longest paths incrementally. In *Proceedings of CP 2003, the 9th International Conference on Principles and Practice of Constraint Programming* (Kinsale, Ireland), *Lecture Notes in Computer Science* 2833, pp. 540–554. Springer, Berlin.