Overmind : ordonnancement pour la planification de projets industriels

Tristan Ézéquel, Pierre-François Dutot

INRIA Grenoble Rhône-Alpes, Montbonnot, France

{tristan.ezequel}@inria.fr

{pierre-françois.dutot}@imag.fr

Mots-clés : *ordonnancement, planification, heuristique*

1 Introduction

Le FUI Overmind est un projet dirigé par le studio d'animation TeamTO (Paris, Bourg-lès-Valence), en collaboration avec le studio Folimage (Bourg-lès-Valence), l'école des Gobelins (Paris) et INRIA Grenoble Rhône-Alpes (Montbonnot). Il vise au développement d'un nouvel outil de gestion de projet, avec l'idée d'allier planification et suivi de production dans un contexte industriel.

Notre rôle dans ce projet est de proposer un planificateur adaptatif avec des techniques d'ordonnancement pour réduire le coût de production des projets. L'objectif est informatif : on vise à fournir une aide à la décision via un panel de plannings possibles, en supposant que les décisions finales seront prises par des superviseurs humains. Le projet final devrait ainsi se distinguer de logiciels des gestion existants par sa capacité à évaluer des choix locaux en exprimant un coût et un risque empiriques par simulation (voir section 3).

Les projets concernés ici sont des productions de séries et de films animés, comprenant généralement plusieurs centaines d'employés et plus de 10000 tâches.

2 Problématique

2.1 Contexte

Nous nous situons dans un problème d'ordonnancement avec un ensemble de n tâches et m ressources [1]. Les tâches et les ressources sont hétérogènes : on dispose d'estimations sur les durées des tâches (moyennes et variances) qui peuvent être de l'ordre de l'heure ou du jour. De plus, chaque tâche est associée à un type de ressource qui pourra l'exécuter. L'objectif est de minimiser une fonction de coût définie par l'utilisateur via une pondération sur des caractéristiques d'un planning : makespan (durée du projet), coût en main-d'oeuvre, degré de respect des deadlines. On considère plusieurs types de contraintes, à savoir :

— des dépendances entre les tâches décrites via un DAG (graphe orienté sans cycle) ;
— des deadlines sur les tâches ;
— des restrictions pour chaque tâche à un type de ressource compatible ;
— un système d'assignation par groupes de tâches vers des groupes de ressources.

2.2 Incertitudes

Le fait de s'inscrire dans un scénario réel implique la présence d'incertitudes : les temps d'exécution des tâches seront inconnus, bien qu'estimés par l'utilisateur. Selon le type de projet, on peut également voir apparaître des tâches pendant son déroulement.

L'incertitude complique le problème par le fait qu'un planning donné porte sur ce qu'on appellera un *scénario*. On exprime un scénario par un vecteur de dimension n décrivant, pour chaque tâche, sa durée d'exécution.

3 Solutions apportées

Pour un projet donné, plusieurs heuristiques basées sur des listes de priorités [2] sont mises en place et testées sur un panel de scénarios (choisis selon des distributions gaussiennes définies par les estimations fournies sur les tâches). On assigne aux tâches une valeur de priorité qui déterminera laquelle doit être lancée lorsque plusieurs sont candidates. Parmi ces heuristiques, on retrouve notamment l'algorithme HEFT [3] qui utilise le bottom level [4] des tâches comme valeur de priorité, calculé via les estimations moyennes de temps d'exécution des tâches et le DAG.

On évalue chaque heuristique avec une fonction de coût, après simulation sur un ensemble de scénarios donné. Les résultats se révèlent très dépendants de la structure du projet étudié : selon le projet considéré, certaines heuristiques seront plus intéressantes que d'autres en termes de coût. Il s'agit donc d'adapter une heuristique particulière à chaque instance de projet.

L'utilisation de listes de priorités permet également de créer de nouvelles heuristiques en associant un nombre arbitraire d'heuristiques par somme pondérée. On observe que dans la plupart des cas, il existe une composition d'heuristiques meilleure que celles déjà existantes. En effet, une heuristique comme HEFT assigne parfois la même valeur de priorité à plusieurs tâches qui pourront être en concurrence pendant la planification ; le fait d'utiliser plusieurs heuristiques permet de réduire drastiquement le nombre de conflits de ce genre, donc d'éviter beaucoup de prises de décision arbitraires.

Une autre démarche suivie a permis d'améliorer les résultats obtenus par les heuristiques (ou combinaisons) classiques. Elle consiste à créer de nouvelles heuristiques en assignant aux tâches des valeurs de priorité de manière aléatoire. Pour certaines structures de projet [1] on trouve facilement de nouvelles heuristiques plus efficaces.

En sélectionnant une partie des meilleures heuristiques trouvées en termes de coût, on observe que la valeur de priorité de certaines tâches varie peu au sein de ces heuristiques. Cette information nous permet de cibler des tâches dont la position a tendance à beaucoup influer sur le coût d'un projet : elle s'avère capitale dans un contexte d'aide à la décision.

Références

[1] Blazewicz J., Ecker K.H., Pesch E., Schmidt G., Weglarz J. *Handbook on Scheduling.* Springer, 2007.

[2] Michel Cosnard, Denis Trystram. *Parallel Algorithms and Architectures.* International Thomson Computer Press, 1995.

[3] Topcuoglu Haluk, Hariri Salim, Wu M. Performance-effective and low-complexity task scheduling for heterogeneous computing. IEEE Transactions on Parallel and Distributed Systems, 2002

[4] Henri Casanova, Arnaud Legrand, Yves Robert. *Parallel Algorithms.* Numerical Analysis and Scientific Computing. Chapman & Hall/CRC, 2008.

1. Projets contenant de l'information non utilisée par les heuristiques implémentées. Par exemple, présence de deadlines à coûts très variables