Les outils de parallélisation du Big Data pour la résolution efficace des problèmes d'ordonnancement : premier pas avec le Job-Shop

Philippe Lacomme et Nikolay Tchernev

Université Blaise Pascal, LIMOS (UMR 6158), F-63178 Aubière, France
placomme@isima.fr, tchernev@isima.fr

Mots-clés *: Ordonnancement, STORM*

1 Introduction

Les techniques utilisées dans le domaine du Big Data pour traiter de grand volume de données, se basent sur des outils spécifiques à cette communauté. Ces outils sont dans la continuité des travaux initiés par la définition des approches de type MapReduce et permettent (Dean and Ghemawat, 2008) d'exécuter facilement un code sur un ensemble de machines reliées par Internet. Hadoop et Storm sont les deux projets les plus connus et les plus utilisés. Storm définit la notion de « topologie » comme étant l'architecture informatique de déploiement et utilise plusieurs types de nœuds pour à la fois assurer le déploiement du code, la coordination (nœud Nimbus par exemple) et la communication (Zookeeper). Les approches de parallélisation issues du « big data », commencent à être utilisées pour résoudre des problèmes d'optimisation. Google utilise MapReduce pour des problèmes d'indexation, de « machine learning » ou encore d'indexation de pages web (Dean and Ghemawat, 2004). Récemment, Cohen en 2009 (Cohen, 2009) a montré que ce type d'approche était utilisé pour le calcul de composantes connexes, de clustering et d'énumération. (Bunch *et al.*, 2009) a montré comment utiliser des approches de type MapReduce pour des transformées de Fourier ou encore la résolution d'équations différentielles. En ce qui concerne les problèmes d'optimisation discrets (Hu *et al.*, 2016) ont défini récemment un Parallel Niche Genetic Algorithm (MR-PNGA) qui tire profit de ce type d'approche et qui permet de tirer profit de ressources de calcul dans le « Cloud ». (Tauer and Nagi, 2013) ont définit une heuristique de type approche lagrangienne basée sur MapReduce pour un problème d'affectation qui a été testée sur un cluster de 8 nœuds. Le problème de la clique maximale a aussi fait l'objet d'étude par (Svendsen *et al.*, 2015). Notre travail se situe à l'intersection d'une part des travaux réalisés dans le big data sur les outils de parallélisation de code et d'autre part les travaux autour des problèmes d'ordonnancement/transport. Notre objectif est de montrer que les nouveaux paradigmes issus du big data présentent un intérêt pour la communauté ordonnancement/transport et peuvent nous amener à définir de nouvelles approches de résolution. Le problème traité est le job-shop.

2 Storm pour le Job-shop

Le principe de communication proposé par Storm tire profit des clés/valeurs couramment utilisé dans les bases de données NoSQL. La topologie se compose d'un ensemble de Bolts (un Bolt peut être considérée comme une unité de traitement) munie d'une méthode spécifique « Execute » qui est lancée chaque fois d'un tuple clé/valeur est reçu. Les informations à envoyer aux bolts sont contenues dans la partie valeur et correspondent, pour un algorithme d'optimisation, aux différents paramètres de la méthode. Les bolts s'exécutent en parallèle sur les différentes machines du cluster et des opérations de synchronisation peuvent avoir lieu pour relancer une recherche autour d'une solution prometteuse. Pour le job-shop nous avons adopté un schéma d'optimisation de type GRASP×ELS et nous avons choisi d'exécuter en parallèle plusieurs ELS comme le montre la figure 1.

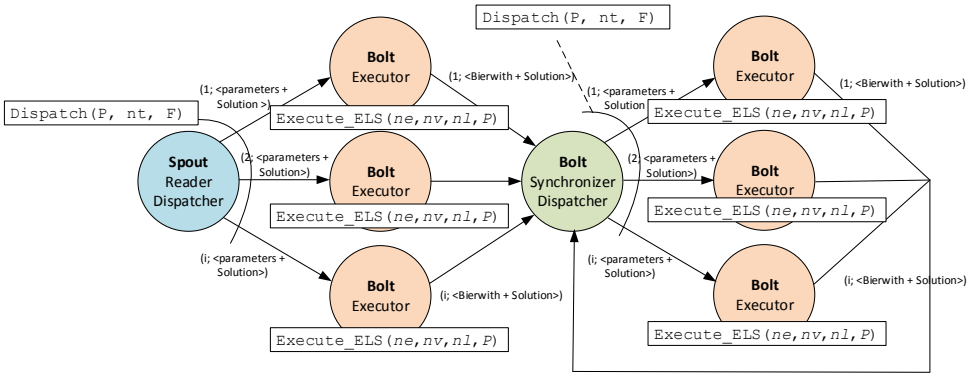

Figure 1. La topologie du GRASP×ELS

L'efficacité de la topologie dépend à la fois du degré de parallélisassions, de la fréquence des synchronisations mais aussi du type à savoir intensification/diversification.

3 Evaluations numériques

			GRASP×ELS			STORM based GRASP×ELS		
Instances	***n***	***BKS***	***BFS***	***TT***	***T****	***BFS***	***TT***	***T****
LA21	150	1046	**1055**	675	354	1060	57	5
LA22	150	927	935	579	316	**927**	4	4
LA23	150	1032	**1032**	0	0	**1032**	<1	<1
LA24	150	935	941	524	126	**938**	46	41
LA25	150	977	984	476	13	**977**	18	18
LA26	200	1218	**1218**	111	111	**1218**	7	7
LA27	200	1235	1264	1276	889	**1263**	97	37
LA28	200	1211	**1223**	1083	909	1224	90	85
LA29	200	1152	1211	1178	808	**1199**	98	88
LA30	200	1355	**1355**	1	1	**1355**	4	4
LA31	300	1784	**1784**	1	1	**1784**	1	1
LA32	300	1850	**1850**	1	1	**1850**	2	2
LA33	300	1719	**1719**	1	1	**1719**	2	2
LA34	300	1721	**1721**	15	15	**1721**	3	3
LA35	300	1888	**1888**	2	2	**1888**	1	1
LA36	225	1268	**1291**	718	322	**1278**	68	6
LA37	225	1397	**1415**	907	861	1419	82	72
LA38	225	1196	1232	1050	865	**1218**	87	16
LA39	225	1233	1251	1006	759	**1240**	89	21
LA40	225	1222	1242	1116	795	**1240**	89	24

(BKS : Meilleure solution connue, BFS : Meilleure solution trouvée, TT : Temps total, T* : temps pour trouver BFS)

TAB 1 – Résultats comparatifs

Deux versions du GRASP × ELS ont été comparées sur les mêmes machines. Une version séquentielle et une version basée sur la topologie de la figure 1 avec un cluster de 8 machines. Les résultats (TAB 1) montre une amélioration très importante des résultats obtenus avec des temps de calcul qui sont fortement amélioré et le facteur d'amélioration (supérieur au nombre de machines utilisées) montre un « speedup » superlinéaire.

4 Conclusion

Les premiers résultats montre qu'il y a un intérêt certain à s'intéresser aux outils de parallélisation issus du big data, car ils offrent de nouvelles opportunités pour résoudre efficacement les problèmes de recherche opérationnelle.

Références

[1] Cohen J., Graph twiddling in a MapReduce world, Computing in Science and Engineering. Vol. 11, pp. 29–41, 2009.

[2] Bunch C., B. Drawert and M. Norman, Mapscale: a cloud environment for scientific computing, Technical Report, University of California, Computer Science Department, 2009.

[3] Dean J. and S. Ghemawat, Mapreduce: simplified data processing on large clusters, Communications of the ACM, Vol. 51, pp. 107–113, 2008.

[4] Svendsen M., A.P. Mukherjee and S. Tirthapura. Mining maximal cliques from a large graph using MapReduce: Tackling highly uneven subproblem sizes. Journal of Parallel and Distribributed Computing, pp. 79–80, 2015.

[5] Tauer G. and Nagi R.. A map-reduce lagrangian heuristic for multidimensional assignment problems with decomposable costs. Parallel Computing, Vol. 39(11), pp. 653-668. 2013.