Malleable Task Scheduling

July 2020

1 Related Work

The problem of malleable task scheduling has been studied for more than thirty years. Du and Leung [1] have proved that the problem is strongly NP-hard for more than 5 processors. Turek [9] has first addressed the approximation algorithms with a ratio 2, which was improved to $\sqrt{3}$ in [6] and 3/2 in [5].

Several discussed the problem with precedence constraint [2] [3] [4] [7]. Du and Leung [1] proved that malleable task scheduling with precedence constraints with more than 3 precessors is NP-hard. Jansen and Klaus [3] developed an approximation algorithm with ratio 4.730598 in 2005, which deciding the numbers of processors allotted to execute the tasks such that the maximum between both opposite criteria of critical path length and average work in the first phase. In the second phase, it generates a new allotment and to schedule all tasks according to the precedence constraints. The approximation ratio is improved to 3.291919 in [4] which implements linear program and a variant of the list scheduling algorithm and still have potentialfor further application in practice. Wu discussed the problem while the task is deadline-sensitive which aims to maximize social welfare and minimize machine number and the maximum weighted completion time. Dynamic programming is implemented for social welfare maximization and greedy programming for social welfare maximization [10].

Nishikawa and Shimada discussed Malleable Fork-Join (MFJ) tasks which aims to minimize the overall schedule length. [7] [8] Nishikawa apply Integer linear programming (ILP) and constraint programming (CP) and the result shows that CP-based scheduling for malleable fork-join tasks can quickly obtain the better schedules than ILP-based [7]. Shimada's algorithm improved the performance while when the system has a large number of cores but failed to find good results for large task graphs in a practical time [8].

References

[1] Jianzhong Du and Joseph Y-T Leung. Complexity of scheduling parallel task systems. *SIAM Journal on Discrete Mathematics*, 2(4):473–487, 1989.

[2] Elisabeth Günther, Felix G. König, and Nicole Megow. Scheduling and Packing Malleable Tasks with Precedence Constraints of Bounded Width. *SpringerLink*, pages 170–181, 9 2009.

[3] Klaus Jansen and Hu Zhang. An approximation algorithm for scheduling malleable tasks under general precedence constraints. *ACM Trans. Algorithms*, 2(3):416–434, 7 2006.

[4] Klaus Jansen and Hu Zhang. Scheduling malleable tasks with precedence constraints. *Journal of Computer and System Sciences*, 78(1):245–259, 1 2012.

[5] G Mounié, C Rapine, and D Trystram. A 3/2-dual approximation algorithm for scheduling independent monotonic malleable tasks, manuscript, 2002.

[6] Gregory Mounie, Christophe Rapine, and Dennis Trystram. Efficient approximation algorithms for scheduling malleable tasks. In *Proceedings of the eleventh annual ACM symposium on Parallel algorithms and architectures*, pages 23–32, 1999.

[7] Hiroki Nishikawa, Kana Shimada, Ittetsu Taniguchi, and Hiroyuki Tomiyama. Scheduling of Malleable Fork-Join Tasks with Constraint Programming. *2018 Sixth International Symposium on Computing and Networking (CANDAR)*, pages 133–138, 11 2018.

[8] Kana Shimada, Ittetsu Taniguchi, and Hiroyuki Tomiyama. ILP-based scheduling for malleable fork-join tasks. *SIGBED Rev.*, 16(3):21–26, 11 2019.

[9] John Turek, Joel L Wolf, and Philip S Yu. Approximate algorithms scheduling parallelizable tasks. In *Proceedings of the fourth annual ACM symposium on Parallel algorithms and architectures*, pages 323–332, 1992.

[10] Xiaohu Wu and Patrick Loiseau. Algorithms for scheduling malleable cloud tasks. 2015.