Complexity and approximation for scheduling problem for coupled-tasks in presence of compatibility tasks

G. Simonin [1], R. Giroudeau[1] and J.C König[1]

[1] LIRMM UMR 5506, 161 rue Ada 34392, Montpellier France
{simonin,rgirou,konig}@lirmm.fr

Keywords: coupled-tasks, compatibility graph, complexity, approximation.

1 Introduction

In this paper, we present the problem of data acquisition according to compatibility constraints in a submarine torpedo. The torpedo is used in order to make cartography, topology studies, temperature measures and many other tasks in the water. The aim of this torpedo is to collect and process a set of data as soon as possible on a mono processor. In this way, it possesses few sensors, a mono processor and one type of tasks which must be scheduled: acquisition tasks. The acquisition tasks $\mathcal{A} = \{A_1, \ldots, A_n\}$ can be considered as coupled-tasks introduced by Shapiro (1980), indeed the torpedo sensors emit a wave which propagates in the water in order to collect the data. Each acquisition task A_i has two sub-tasks, the first one a_i sends an echo, the second one b_i receives it. For a better reading, we will denote the processing time of each sub-task a_i and b_i. Between the sub-tasks, there is an incompressible idle time L_i which represents the spread of the echo in the water.

At last, there exists compatibility constraints between acquisition tasks, due to the fact that some acquisition tasks cannot be processed at the same time as another task. In order to represent this constraint, a compatibility graph $G_c = (\mathcal{A}, E_c)$ is introduced, where $\mathcal{A}$ is the set of coupled-tasks and E_c represents the edges which link two coupled-tasks which can be executed simultaneously. In other words, at least one sub-task of a task A_i may be executed during the idle time of another task A_j (see example in Figure 1).

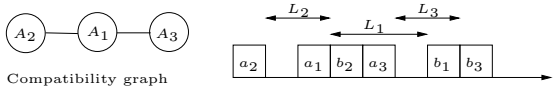

Fig. 1. Example of compatibility constraints with $a_i = b_i = 1$, $L_1 = 3$, $L_2 = L_3 = 2$

The aim of this problem is to produce a shortest schedule (i.e. to minimize the completion time of the last processed task) denoted by C_{max}, in presence of compatibility contraints between acquisition tasks. In the scheduling theory, a problem is categorized by its machine environment, job characteristic and objective function. So using the notation scheme $\alpha|\beta|\gamma$ proposed by Graham et al. (1979), this problem will be defined as $1|(a_i, L_i, b_i), G_c|C_{max}$.

Our work consists in measuring the impact of the compatibility graph on the complexity and approximation of scheduling problems with coupled-tasks on a mono processor. This paper is focusing on the limit between the polynomiality and the $\mathcal{NP}$-completeness of our problem, when the compatibility constraint is introduced with different topologies.

The complexity of the scheduling problem, with coupled-tasks and a complete compatibility graph[1], has been investigated first by Orman and Potts (1997), Błażewicz et al. (2009), and Ahr et al. (2004). Nevertheless, in this article we study a different problem in which coupled-tasks (or acquisition tasks) must respect a compatibility graph. By comparing the results of Orman and Potts (1997) and those obtained by relaxing the constraint of compatibility, we can measure the impact of compatibility constraint on this kind of problem. In such context, several results (Simonin et al. 2009) have been recently obtained according to coupled-tasks parameters. In the following, we consider the specific problem $\Pi = 1|(a_i = L_i = b_i), G_c|C_{max}$ where all the coupled-tasks have a different length[2]. Then, we study the complexity of Π according to the topology of the compatibility graph.

2 Polynomial-time approximation algorithm for Π

2.1 For general compatibility graph

We will develop a polynomial-time approximation algorithm for the coupled-tasks scheduling problem in presence of compatibility constraints where the duration of all the tasks is different. Therefore, for all A_i and A_j, $i \neq j$, we have $a_i = L_i = b_i \neq a_j = L_j = b_j$. With these hypotheses, the tasks are sorted according to their length L_i (we have $L_1 < L_2 < \ldots < L_n$).

Theorem 1. *A simple algorithm, which consists in scheduling the tasks consecutively, admits a ratio ρ equal to $\frac{3}{2}$.*

Proof. The length of the schedule obtained by our algorithm is $\sum_i 3L_i$, whereas the optimal length is greater or equal to $\sum_i 2L_i$. Thus we obtain a ratio $\rho \leq \frac{3}{2}$.

The bound is trivial, and in the following we will propose a better ratio for a specific compatibility graph.

2.2 For a complete oriented bipartite graph

This section is devoted to the complexity and approximation in the case where the compatibility graph is a complete oriented[3] bipartite graph $G_c = (X, Y, E)$. Let $X = \{X_1, \ldots, X_k\}$ where x_i is the length of task X_i, and $Y = \{Y_1, \ldots, Y_m\}$ where y_j is the length of task Y_i. Without lost of generality, we have $x_1 > \ldots > x_k \geq 3y_1 > \ldots > 3y_m$.

Computational complexity

Theorem 2. *The problem $\Pi' : 1|a_i = L_i = b_i, G_c = \text{complete oriented bipartite}|C_{max}$ is $\mathcal{NP}-$complete.*

Proof. In the following we consider a complete oriented bipartite graph $G_c^b = (X, Y, E)$, with $|X| = 2$. The proof is established by a reduction from EVEN PARTITION problem (Garey and Johnson 1979):

Instance: A finite set $\mathcal{K}$ of n elements $\{e_1, \ldots, e_n\}$, a bound $B \in \mathbb{N}^+$ and a size $s(e_i) \in \mathbb{N}$ for each $e_i \in \mathcal{K}$ such that each $s(e_i)$ is even and such that $\sum_{e_i \in \mathcal{K}} s(e_i) = 2B$. Remark that the problem remains $\mathcal{NP}$-complete even if $\forall i \neq j, s(e_i) \neq s(e_j)$.

[1] Note that the lack of compatibility graph is equivalent to a fully connected graph. In this way, all the tasks may be compatible with each other.

[2] The problem with complete compatibily graph is $\mathcal{NP}$-complete (see Orman and Potts (1997), so the relaxed version is obviously $\mathcal{NP}$-complete).

[3] the graph is oriented because of the Y-tasks can be scheduled in the X-tasks.

Question: Can $\mathcal{K}$ be partitioned into two disjoint sets $\mathcal{K}_1, \mathcal{K}_2$ of $\mathcal{K}$ such that $B = \sum_{e_i \in \mathcal{K}_1} s(e_i) = \sum_{e_i \in \mathcal{K}_2} s(e_i) \in \mathbb{N}$?

Let $\mathcal{I}^*$ be an instance from EVEN PARTITION, we construct an instance $\mathcal{I}$ for our scheduling problem where $C_{max} = 18B+3$ as follows:

We consider $(n+2)$ tasks denoted by, $X_1, X_2 \in X$ of length x_1, x_2, and $Y_1, \ldots, Y_n \in Y$ of length $y_1, \ldots, y_n$ such that:

- $X_1 = (3B+1, 3B+1, 3B+1)$ and $X_2 = (3B, 3B, 3B)$
- $Y_i = (a_i = s(e_i), L_i = s(e_i), b_i = s(e_i)), \forall i \in \{1, \ldots, n\}$

We add edges between $(X_i, Y_j), \forall i \in \{1, 2\}, \forall j \in \{1, \ldots, n\}$. Obviously, this transformation can be computed in polynomial time.

- Assume that $\mathcal{K}$ can be partitioned into two disjoint sets $\mathcal{K}_1, \mathcal{K}_2$ of $\mathcal{K}$ such that $B = \sum_{e_i \in \mathcal{K}_1} s(e_i) = \sum_{e_i \in \mathcal{K}_2} s(e_i) \in \mathbb{N}$. The task X_1 (resp. X_2) is executed at $t = 0$ (resp. at $t = 9B + 3$). The task Y_i, corresponding to the element $e_i \in \mathcal{K}_1$ (resp. $e_i \in \mathcal{K}_2$) of length $s(e_i)$, is scheduled as soon as possible in the idle slot of the task X_1 (resp. X_2).
- Reciprocally, we suppose that the length of the schedule is $C_{max} = 18B + 3$. The two tasks X_1 and X_2 cannot be executed into each other. It is clear that the sum of durations of these two tasks is $18B + 3$. All the other tasks must be executed in idle slots of length $3B$ or $3B + 1$, which implies the existence of a partition.

Polynomial-time Approximation Algorithm

We propose the following algorithm: initially the X-tasks, $|X| = k \geq 1$, and the Y-tasks, $|Y| = m \geq 1$, are sorted in non-increasing order with respect to their duration, separately. The tasks are executed in the input order according to the following rule: X-tasks are executed consecutively. A Y-task is assigned to the first used X-task where there is enough available space to include it; if no X-tasks can contain the Y-tasks, they are assigned after the last executed task in the schedule.

This algorithm is similar to the classical Best-Fit algorithm. The sets X and Y are sorted in $O(nlog(n))$, where $|X| + |Y| = n$.

Definition 1. *The set of Y-tasks, which are not executed in the X-tasks, is called remainder R.*

Theorem 3. *The previous algorithm admits a relative performance bounded by $7/6$ for the problem $\Pi' : 1|(a_i = L_i = b_i = x_i), G_c = \text{complete oriented bipartite}|C_{max}$.*

Proof. We suppose that there exists a remainder R (otherwise there would be an optimal solution). We consider $C_1 = \sum_{i=1}^{m} 3y_i$ and $C_2 = \sum_{i=1}^{k} x_i$.

Lemma 1. *The size of the slot (i.e. the inactivity time of a couple-task) x_i, $\forall X_i \in X$, is at least greater than $\frac{x_i}{2}$ for an execution of the Y-tasks if there exists a remainder.*

Proof. By contradiction, let X_i be the first task such that the used time in its slot is less than $\frac{x_i}{2}$, and let Y_j be the first task of the remainder. We have $y_j > \frac{x_i}{6}$, otherwise the task Y_j may be executed into the task X_i. In such context, there is no task from the Y-tasks which are executed into X_i when Y_j is considered, because the Y-tasks are sorted in non-incresang order, so $y_j > \frac{x_i}{3}$ (in contradiction with the fact that the graph G_c is an oriented bipartite complete graph, and that any Y-task can be scheduled in the X-tasks).

Thus, we have $0 \leq R \leq C_1 - \frac{C_2}{2}$, and one upper bound is: $C^h_{max} \leq$ Remainder + sequential duration of the X-tasks $= \left(C_1 - \frac{C_2}{2}\right) + 3C_2 = \frac{5C_2}{2} + C_1$. Two cases must be considered:

- If $C_1 \leq C_2$ then $C_{max}^{opt} \geq 3C_2$. Indeed, in the best case the Y-tasks can be executed in the idle slots of the X-tasks. Thus, the ratio is $\rho = \frac{C_{max}^h}{C_{max}^{opt}} \leq \frac{\frac{5C_2}{2}+C_1}{3C_2} \leq \frac{7}{6}$.
- If $C_1 > C_2$ then $C_{max}^{opt} \geq 2C_2 + C_1$. Indeed, in the best case all the inactivity times of the X-tasks are used. The ratio is $\rho = \frac{C_{max}^h}{C_{max}^{opt}} \leq \frac{\frac{5C_2}{2}+C_1}{2C_2+C_1} \leq \frac{\frac{5C_2}{2}+y}{2C_2+y} \leq \frac{7}{6}, \forall y \leq C_1$.

The worst case analysis of the relative performance is not rigorous enough. We conjecture that the ratio is $\frac{8}{7}$. In such context, we may exhibit an instance for which the bound is tight. We consider the four tasks $X_1 \in X$, $Y_1 \in Y$, $Y_2 \in Y$ et $Y_3 \in Y$ with following characteristics: $X_1 = (M, M, M)$, $Y_1 = (\frac{M}{6} + 2\epsilon, \frac{M}{6} + 2\epsilon, \frac{M}{6} + 2\epsilon)$, $Y_2 = (\frac{M}{6} + \epsilon, \frac{M}{6} + \epsilon, \frac{M}{6} + \epsilon)$, $Y_3 = (\frac{M}{6} - \epsilon, \frac{M}{6} - \epsilon, \frac{M}{6} - \epsilon)$ with $M \in \mathbb{N}^*$. Using the previous algorithm, we obtain a schedule of length $\mathbf{C_{max}^h = 4M}$. An optimal solution consists in executing the tasks Y_2 and Y_3 within the idle time of the task X_1. Therefore, we obtain $\mathbf{C_{max}^{opt} = \frac{7M}{2}}$. The ratio is equal to $\rho = \mathbf{\frac{8}{7}}$.

3 Conclusion

We investigate a particular coupled-tasks scheduling problem $1|a_i = L_i = b_i, G_c|C_{max}$ in presence of a compatibility graph with regard to the complexity and appproximation. We design a $\frac{3}{2}$-approximation algorithm for the general case. We also establish the $\mathcal{NP}$-completeness for the specific case where there is a bipartite compatibility graph. In such context, we propose a $\frac{7}{6}$-approximation algorithm and the bound is tight. For further research, it would be interesting to propose a polynomial-time approximation algorithm with a non-trivial ratio for non complete oriented bipartite graph.

References

D. Ahr, J. Békési, G. Galambos, M. Oswald, and G. Reinelt. An exact algorithm for scheduling identical coupled tasks. *Mathematical Methods of Operations Research*, 59:193–203, 2004.

J. Błażewicz and K. Ecker and T. Kis and C.N. Potts and M. Tanas and J. Whitehead. Scheduling of coupled tasks with unit processing times. *Technical report, Poznan University of Technology*, 2009.

M.R. Garey and D.S. Johnson. *Computers and Intractability, a Guide to the Theory of $\mathcal{NP}$-Completeness*. Freeman, 1979.

R.L. Graham, E.L. Lawler, J.K. Lenstra, and A.H.G. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling theory: a survey. *Annals of Discrete Mathematics*, 5:287–326, 1979.

A.J. Orman and C.N. Potts. On the complexity of coupled-task scheduling. *Discrete Applied Mathematics*, 72:141–154, 1997.

R.D. Shapiro. Scheduling coupled-tasks. *Naval Research Logistics Quaterly*, 20:489–498, 1980.

G. Simonin and R. Giroudeau and J.-C. König. Complexity and approximation for scheduling problem for a torpedo *CIE'39 : The 39th International Conference on Computers and Industrial Engineering, IEEE*, Troyes - France, 300–304, 2009.

G. Simonin and B. Darties and R. Giroudeau and J.-C. König Isomorphic Coupled-Task Scheduling Problem with Compatibility Constraints on a Single Processor. *MISTA'04 : 4th Multidisciplinary International Scheduling Conference : Theory and Applications*, Dublin - Irland, 378–388, 2009.

G. Simonin and R. Giroudeau and J.-C. König Extended Matching Problem for a Coupled-Tasks Scheduling Problem. *TMFCS'09 : International Conference on Theoretical and Mathematical Foundations of Computer Science*, Orlando - USA, 082–089, 2009.