Theoretical Aspects of Scheduling Coupled-Tasks in the Presence of Compatibility Graph

Gilles Simonin, Rodolphe Giroudeau, Jean-Claude König, Benoit Darties

Theoretical Aspects of Scheduling Coupled-Tasks in the Presence of Compatibility Graph

G. Simonin and **R. Giroudeau** and **J.-C. König**
{firstname.lastname}@lirmm.fr
LIRMM - UMR CNRS 5506
University of Montpellier II,
161 rue Ada
34392, Montpellier cedex 5, France

B. Darties
Benoit.Darties@u-bougogne.fr
LE2I - UMR CNRS 5158
University of Burgundy
BP 47 870, 21078, Dijon, France

Abstract

This paper presents a generalization of the coupled-task scheduling problem introduced by Shapiro (Shapiro 1980), where considered tasks are subject to incompatibility constraint depicted by an undirected graph. The motivation of this problem comes from data acquisition and processing in a mono-processor torpedo used for underwater exploration. As we add the compatibility graph, we focus on complexity of the problem, and more precisely on the border between $\mathcal{P}$ and $\mathcal{NP}$-completeness when some other input parameters are restricted (e.g. the ratio between the durations of the two sub-tasks composing a task): we adapt the global visualization of the complexity of scheduling problems with coupled-task given by Orman and Potts (Orman and Potts 1997) to our problem, determine new complexity results, and thus propose a new visualization including incompatibility constraint. In the end, we give a new polynomial-time approximation algorithm result which completes previous works.

Introduction

Motivation

This paper deals with the problem of data acquisition subject to incompatibility constraint in a submarine torpedo. Many scheduling issues arise in several situations, e.g. in a radar pulsing context (Sherali and Smith 2005; Ageev and Baburin 2007), radar system (Orman, Shahani, and Moore 1998; Orman et al. 1996), or particular application (Brauner et al. 2009). In our context, the torpedo is used to execute several submarine topographic surveys, including topological or temperature measurements. Its aim is to collect data and to process them on a mono-processor within a minimum timeframe. A collection of sensors acquire data for the torpedo. Each data acquisition consists in an acquisition task which is divided into two sub-tasks: a sensor first emits a wave which propagates in the water, then he gets a corresponding echo. Scheduling issues appear when several sensors using different frequencies can work in parallel, while acquisitions using the same frequency have to be delayed in order to avoid interferences. It is necessary for robotic engineers to have a good theoretical knowledge of this type of problem. Thus, the aim of the work is to study in many sub-configurations and determine complexity and approximation results on then.

Modelisation and related work

Coupled-tasks (Shapiro 1980) are a natural way to model such data acquisition by our torpedo. Each acquisition task can be viewed as a coupled-task A_i composed by two sub-tasks, respectively dedicated for wave transmission and echo reception.We note a_i and b_i the processing time of each sub-task. Between these two sub-tasks there is an incompressible and inextensible idle time L_i which represents the spread of the echo in the water. Due to hardware constraints, we do not work in a preemptive mode: once started, a sub-task cannot be stopped and then continued later. A valid schedule implies here that for any task started at t, the first sub-task is fully executed between t and $t + a_i$, and the second between $t+a_i+L_i$ and $t+a_i+L_i+b_i$. We note $\mathcal{A} = \{A_1, \ldots, A_n\}$ the collection of coupled-tasks to be scheduled. Incompatibility constraints also exist between tasks due to wave interferences. We say two tasks A_i and A_j are compatible if they use different wave frequencies; thus any sub-task of A_i may be executed during the idle time of A_j, as in Figure 1. We introduce a graph $G_c = (\mathcal{A}, E_c)$ to model such this compatibility, where edges from E_c link any pair of compatible coupled-tasks. In the torpedo problem, the conflict graph G_c owns several strong topology constraints. In order to achieve the best possible study and reuse presented results in more general configurations, we do not take into account these constraints and suppose G_c can be any graph.

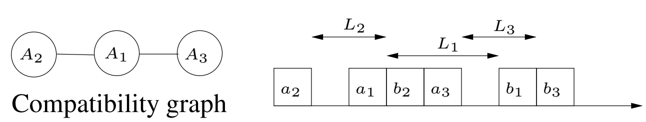

Figure 1: Example of compatibility constraint with $A_1 = (a_1 = b_1 = 1, L_1 = 3)$, $A_2 = (a_2 = b_2 = 1, L_2 = 2)$, $A_3 = (a_3 = b_3 = 1, L_3 = 2)$

Our contributions are the following, in such context the trellis of complexity results are completed by several results in complexity, and we design a polynomial-time approximation algorithm which completes previous works.

The aim is to produce a shortest schedule, ie. minimize

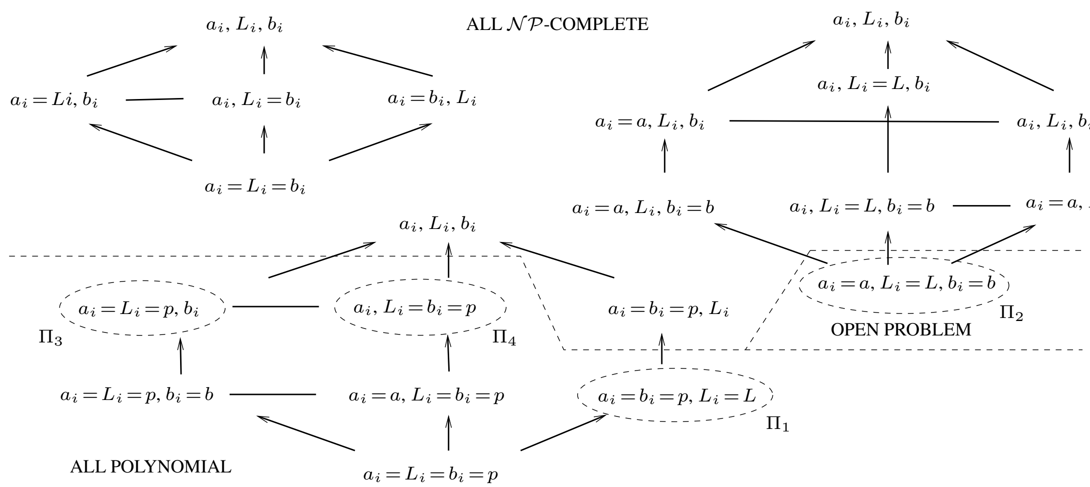

Figure 2: Global visualisation of the complexity of scheduling problems with coupled-tasks described by three distinct trellis in (Orman and Potts 1997). Triplet (a_i, L_i, b_i) describes the type of problem studied, where each variable a_i, b_i and L_i can take any value or be equal to a constant. Finally, there is an arc from a specific problem to a more general problem, and an edge between two symmetrical problems.

the date C_{max}[1] when all tasks are executed, respecting the incompatibility constraint between tasks. As this main problem is decomposable, we use the Graham's notation scheme $\alpha|\beta|\gamma$ (Graham et al. 1979) (respectively the machine environment, job characteristic and objective function) to characterize the sub-problems we study. We define the TORPEDO main problem as $1|a_i, L_i, b_i, G_c|C_{max}$. In the rest of this paper, given a valid schedule σ and a task A_i, we note $\sigma(A_i)$ the date when A_i is being executed, ie. sub-tasks are executed in respectively $\sigma(A_i)$ and $\sigma(A_i) + a_i + L_i$.

In existing works, complexity of scheduling problems with coupled-tasks and no incompatibility constraint has been investigated (Blażewicz et al. 2009; Orman and Potts 1997; Ahr et al. June 2004) (note that authors focus their studies on precedence constraint between the acquisition tasks, which is different from the work presented in this paper). We study here a generalization which consists in introducing a compatibility graph G_c between tasks, and measuring impact of G_c existence on the complexity and approximation actual results. In particular we focus on the limit between polynomial and $\mathcal{NP}$-complete problems and on the establishment of approximate solutions for difficult instances.

In (Orman and Potts 1997), authors give a global visualization of scheduling problems complexity with coupled-tasks through three trellis presented in Figure 2. Our approach is to achieve the same type of study in presence of a compatibility graph G_c. By comparing results of (Orman and Potts 1997) with those obtained by relaxing incompatibility constraint, we can measure impact of this constraint on this kind of problem.

This paper is organized as follow:

- In the first section, we present some $\mathcal{NP}$-complete and polynomial results for different sub-problem of TORPEDO. This lead us to present global visualization inspired by the one presented in Figure 2 which takes into account the presence of compatibility graph between tasks, and highlights the importance of G_c on problem complexity;
- In the second section we give a polynomial-time approximation algorithm for the first studied problem, taking into account the values of some instance parameters.

[1] C_{max} is the processing end of the latest executed task.

Study of the complexity in presence of a compatibility graph

In this section, we present several complexity results on different TORPEDO sub-problems. In order to perform a full study, we reuse problems identified on Figure 2. Taking into account incompatibility constraint make problems more difficult than they were. Thus problems which were $\mathcal{NP}$-complete without incompatibility constraint remain trivially $\mathcal{NP}$-complete when such constraint is introduced. Considering hierarchy of our problems, we will focus our study on adding incompatibility constraint to problems, which are at the limit of Polynomiality and $\mathcal{NP}$-completeness or still open, and identified as problems Π_1, Π_2, Π_3 and Π_4 according to the diagram of Figure 2. For a better visibility, we will use the problem notation Π'_i as a reference of problem Π_i on which compatibility graph is added. Results of this section are divided into four main parts, each part being devoted to the complexity study of a given sub-problem: first, we will prove the $\mathcal{NP}$-completeness of two scheduling problems:

- Π'_1 : $1|a_i = b_i = p, L_i = L, G_c|C_{max}$
- Π'_2: $1|a_i = a, L_i = L, b_i = b, G_c|C_{max}$

Then we show the polynomiality of following problems:

- Π'_3 : $1|a_i = L_i = p, b_i, G_c|C_{max}$

- Π'_4 : $1|a_i, L_i = b_i = p, G_c|C_{max}$

We will prove in particular that $\mathcal{NP}$-completeness of Π'_1 implies $\mathcal{NP}$-completeness of Π'_2. For these problems, we will set some parameters in order to measure the influence of G_c on evolution of the complexity.

Study of Problem Π'_1

In sub-problem $\Pi'_1 = 1|a_i = b_i = p, L_i = L, G_c|C_{max}$, all sub-tasks require the same execution time $p \in \mathbb{N}^*$ and idle time is fixed to a constant L. According to Orman and Potts, problem $\Pi_1 = 1|a_i = b_i = p, L_i = L|C_{max}$ is polynomial. We are going to study the complexity of Π'_1 by varying the value of parameter L according to the value of p. We study three disjoint cases, respectively $0 < L < p$, $p \le L < 2p$ and $2p \le L$, and prove that the first two are polynomial (Lemma 1 and 2), the last one $\mathcal{NP}$-complete (Lemma 3):

Lemma 1. *When $0 < L < p$, Π'_1 is solvable in polynomial-time.*

Proof. When $0 < L < p$, it is easy to see that no task can overlap with the execution of another task. An optimal schedule consists in executing tasks sequentially without delay side by side. This algorithm admits a linear time complexity and produce a schedule of length $C_{max} = |\mathcal{A}| \times (2p + L)$. □

Lemma 2. *When $p \le L < 2p$, Π'_1 is solvable in polynomial-time.*

Proof. When $p \le L < 2p$, at most one sub-task of duration p may be scheduled during the idle time L of another task. Thus, any scheduling of Π'_1 can be associated with a matching on G_c : tasks associated with the vertices covered by the matching edges are executed in pairs, creating "blocks" with an inactivity time of $(2L - 2p)$. For two tasks A_i and A_j we have $\sigma(A_j) = \sigma(A_i) + a_i$.

After ordering the tasks corresponding to the matching, we execute the remaining tasks which do not belong to the matching in the same manner as in the first case. The length of the schedule will therefore depend on the size of the matching, and thus finding a matching with maximum cardinality in G_c provides an optimal schedule. Finding a maximum matching in a general graph has complexity $O(n^3)$ using Gabow's algorithm (Gabow 1976), and therefore the case $p \le L < 2p$ is polynomial. □

When $L \ge 2p$, we can now overlap the execution of more than two acquisition tasks, which leads us to search for cliques in the compatibility graph to reduce the inactivity time on the processor. We show this result in $\mathcal{NP}$-completeness of Π'_1: we restrict our study of Π'_1 to the sub-case where $L_i = 2p$ for any task. We propose lemma 3 and prove the $\mathcal{NP}$-completeness of Π'_1 when $L_i = 2p$; then the generalization when $L_i \ge 2p$ is immediate.

Lemma 3. *Deciding if an instance of Π'_1 where $L_i = 2p$ for any task has a schedule of length $\beta = \sum_{i=1}^{n}(a_i + b_i) = 2np$ is a $\mathcal{NP}$-complete problem.*

Proof. Obviously, Π'_1 is in $\mathcal{NP}$. We prove the $\mathcal{NP}$-completeness of Π'_1 thanks to a polynomial time reduction from TRIANGLE PARTITION (Garey and Johnson 1979) which consists to determinate if the vertices of a graph can be covered by disjoint triangles:

Let I^* be an instance of TRIANGLE PARTITION, i.e. a graph $G = (V, E)$ with $|V| = 3q, q \in \mathbb{N}^*$. From I^* we construct in polynomial-time an instance I of Π'_1 with a compatibility graph $G_c = (\mathcal{A}, E_c)$ as follows:

- $\forall i \in V$, an acquisition task A_i is introduced in $\mathcal{A}$, composed by two sub-tasks a_i and b_i of executed length $a_i = b_i = p$ and by an incompressible and inextensible inactivity time between them of length $L_i = 2p$.
- For each edge $e = \{i, j\} \in E$, an edge $e_c = \{A_i, A_j\}$ is added in E_c. we have a non-exclusive relationship between the two tasks A_i and A_j.

Figure 3 illustrates such a transformation, which is clearly computable in polynomial-time.

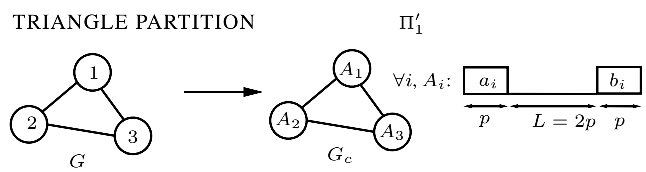

Figure 3: Example of the polynomial-time transformation

Let us prove that the existence of a perfect triangle-cover on G vertices implies the existence of an optimal schedule without idle time (then $C_{max} = n \times 2p$) , and reciprocally:

$\Rightarrow$ Suppose that there exists a triangles cover on G vertices. Then, let us show that there is a schedule without idle time of length $2np$, $2np$ being the sum of processing times. To do this, it is sufficient to combine the acquisitions tasks A_i three by three in G_c according to the triangle-cover found in G. The execution of these blocks forms a schedule without idle time. Figure 4 presents an example of block formed with three tasks. If all tasks can be included into such a block, then we obtain a schedule of length $2np$.

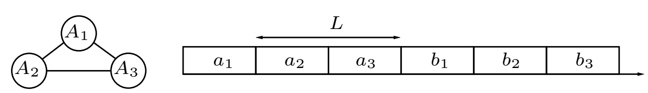

Figure 4: Illustration of a block of three acquisition tasks, $\sigma(a_3) = C(a_2) = C(a_1) + a_2$

$\Leftarrow$ Conversely, if there is a schedule of length $2np$ on instance I, then let us show that G vertices can be covered by exactly q triangles.
It is obvious that if $C_{max} = 2np$, then there is no idle time on the processor. This means that every idle slot of length $L_i = 2p$ is bound to be filled. However, we need three acquisition tasks carried into each other in order to

obtain a block of three tasks without idle time. So, with exactly q blocks, we obtain a schedule without inactivity time. Since three acquisition tasks carried into each other are necessarily compatible in G_c, there exists a triangles cover on G_c vertices and G vertices by construction.

Thus, we have TRIANGLE PARTITION $\propto \Pi_1'$. We know that TRIANGLE PARTITION is $\mathcal{NP}$-complete (Garey and Johnson 1979), So we can conclude that the problem Π_1' is $\mathcal{NP}$-complete. □

From the proof of $\mathcal{NP}$-completeness of Π_1', we note that for $L = kp$ with $k \geq 2$, the existence of a schedule without idle slot is equivalent to finding a partition of the G_c vertices by disjoint cliques of size $(k+1)$ (which is equivalent to the $\mathcal{NP}$-complete problem PARTITION INTO SUBGRAPHS ISOMORPHIC TO H, where H is a clique of size $(k+1)$). The approximation study of problem Π_1' is presented on the second section of this paper

Study of problem Π_2'

From the results obtained by Orman and Potts (Orman and Potts 1997) (see Figure 2), we know that finding the complexity of Π_2 is still an open problem. We focus here on problem $\Pi_2' : 1|a_i = a, L_i = L, b_i = b, G_c|C_{max}$, with $a, b, L \in \mathbb{N}^*$. By observing the values of parameters a_i and b_i, we state the following observation: Π_2' is a generalization of Π_1'. Indeed, instances of Π_1 are particular cases of Π_2' when $a = b = p$. This lead us to propose Theorem 1:

Theorem 1. *Decision problem Π_2' is $\mathcal{NP}$-complete by generalization.*

We have shown that the problem $\Pi_2' : 1|a_i = a, b_i = b, L_i = L, G_c|C_{max}$ was $\mathcal{NP}$-complete in the general case, a deeper complexity study has been performed in (Simonin et al. 2010) when values of a and b are linked to each others.

Study of problem Π_3'

This problem consists of scheduling n acquisition tasks having the same model $a_i = L_i = p, b_i$. The first sub-task and idle time are set at the same constant p, $p \in \mathbb{N}^*$, while the second sub-task can take any value.

The set of these acquisition tasks contains two subsets: the first subset denoted K is composed of all the acquisition tasks A_i such that $b_i \leq p$, the second subset denoted S is composed by all other tasks. Two tasks (a_i, L_i, b_i) and (a_j, L_j, b_j) in S cannot be executed one inside the other, so the edge $\{i, j\} \notin G_c$ and automatically these edges are removed. For this section, we will use the following reformulation: G_c is still a complete graph when K is a clique in G_c, S is an independent set and $\forall x \in K, \forall y \in S$, we have $\{x, y\} \in G_c$.

Theorem 2. *The scheduling problem $\Pi_3' : 1|\ a_i = L_i = p, b_i, G_c\ |C_{max}$ is polynomial.*

Proof. The configuration proposed by problem Π_3' allows only at most one sub-task to be scheduled during the idle time of a task. By weighting each edge of the graph with the sequential time of the overlap of the two tasks linked by the edge, our problem has a solution if we find a matching that minimizes the weight of the matching edges and the isolated vertices.

For these purposes, we will search a similar problem which is known to be solved in polynomial time. This problem, which is equivalent to our problem through a polynomial-time transformation, consists in finding a minimum weight perfect matching : the minimum weight perfect matching problem consists in finding a perfect matching in a weighted graph where the sum of perfect matching edges is minimized, which can be done in polynomial time (Edmonds 1965). We propose the following polynomial-time construction:

Let $\mathcal{I}_1$ be an instance of our problem with a compatibility graph $G_c = (V_c, E_c)$, and $\mathcal{I}_2$ an instance of the minimum weight perfect matching problem in graph constructed from $\mathcal{I}_1$. Let $G_c' = (V_c', E_c')$ and $G_c'' = (V_c'', E_c'')$ be two copies of compatiblity graph G_c. The vertex corresponding to A_i is denoted A_i' in G_c' and A_i'' in G_c''. From G_c' and G_c'' we construct a graph $H_c = (V_c' \cup V_c'', E_c' \cup E_c'' \cup E_c''')$ with $E_c''' = \{\{A_i', A_i''\} | A_i \in V_c\}$. We define the following weights on the edges of H_c :

- Each edge $\{A_i', A_j'\}$ (resp. $\{A_i'', A_j''\}$), where $b_i > p$ or $b_j > p$, is weighted by $\frac{3p+max\{b_i,b_j\}}{2}$. This value represents half of the execution time used in the scheduling by the two coupled-tasks, where the second task belongs to S.
- Each edge $\{A_i', A_j'\}$ (resp. $\{A_i'', A_j''\}$), where $b_i \leq p$ and $b_j \leq p$, is weighted by $\frac{3p+min\{b_i,b_j\}}{2}$. This value represents half of the execution time used in the scheduling by the two coupled-tasks that belong to K. The second executed task will be the one with the smallest b_i.
- Each edge $\{A_i', A_i''\}$ is weighted by $2p + b_i$. This value represents the execution time used in the schedule by an isolated task.

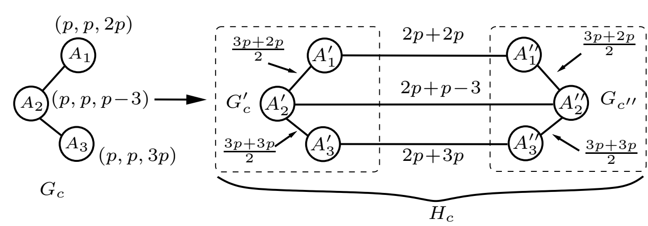

Figure 5: Example of the polynomial-time transformation.

Figure 5 illustrates such a construction. In order to design a polynomial-time algorithm solving the problem Π_3', we will prove firstly the following proposition: For a minimum weight perfect matching of C, a schedule of minimum processing times C exists and reciprocally (cf Figure 6).

Indeed, the weight of each edge $e = \{A_i', A_j'\} \in \{V_c', V_c'\}$ (resp. $e = \{A_i'', A_j''\} \in \{V_c'', V_c''\}$), with $i \neq j$, corresponds to half the length of the scheduling on the processor for the acquisition tasks A_i' and A_j' (A_i'' and A_j'') if

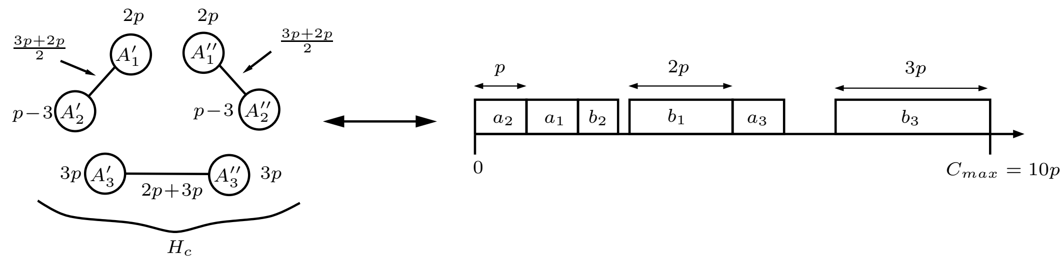

Figure 6: Example of correspondence between a perfect matching and an optimal schedule.

they overlap. This overlap can be represented by a block. The weight of each edge $e = \{A'_i, A''_i\} \in \{V'_c, V''_c\}$ is the length of the scheduling on the processor for a simple acquisition task.

By construction H_c contains an even number of vertices, and the fact that each vertex of G'_c is connected to an equivalent vertex in G''_c, finding a perfect matching on the graph H_c is possible. This means that there exists a schedule such that each task is executed only once. Note that the matching in G'_c is not necessarily identical to the one in G''_c, but they still have the same weight. So, we can take the same matching in G'_c and G''_c without lost of generality. The makespan obtained is equal to sum of the processing times of the obtained blocks and those of isolated tasks. And since each isolated task (resp. block) has an execution time equal to the weight of the equivalent edge (resp. the two equivalent edges on G'_c and G''_c) in the perfect matching, we have the sum of edges weights of the matching which is equal to the blocks sum of the scheduling obtained. Thus, for a minimum weight perfect matching C, there exists a schedule of minimum length C and reciprocally.

This shows the relationship between a solution to the problem Π_3 and a solution of a minimum weight perfect matching in H_c. This relationship is illustrated on Figure 6. Edmonds algorithm gives a minimum weight perfect matching in $O(n^2m)$ (Edmonds 1965). Thus the optimization problem Π_3 is polynomial. □

The polynomial-time algorithm, which gives an optimal solution to the problem $\Pi_3 : 1|\ a_i = L_i = p, b_i, G_c\ |C_{max}$, is decomposed into two steps: the first step consists in creating the graph H_c then in finding a perfect matching in it, while the second step consists in executing the acquisition tasks on the processor according to the matching edges. Algorithm 1 gives a such solution with a complexity time of $O(n^2m)$.

Study of problem Π'_4

Problem $\Pi'_4 : 1|\ a_i, L_i = b_i = p, G_c|C_{max}$ is composed by n acquisition tasks which have the same model $(a_i, L_i = b_i)$. Each acquisition task is different from the others, and for each of them the two sub-tasks and the idle time have the same execution time. In this section we announce in corollaries 1 that Π'_4 can be solved in a polynomial time using an Orman and Potts's result.

Corollary 1. *Problem Π'_4 admits a polynomial-time algorithm.*

Algorithm 1: An optimal scheduling in polynomial time

input : $\mathcal{A} = \{A_1, A_2, \ldots, A_n\}, H_c, G_c$
output: C^{opt}_{max}
begin
- Search in H_c a perfect matching M minimizing the weight of the matching edges;
- For each edge $e = (A'_i, A'_j) \in H_c$ (resp. $e = (A''_i, A''_j) \in H_c$) of the matching M, such that A'_i and A'_j (resp. A''_i et A''_j) belong to the same graph G'_c (resp. G''_c), the acquisition tasks A_i and A_j associated to the graph G_c are scheduled into each other according to the edge weight.; Two cases are possible, if $p \geq b_i \geq b_j$ then $\sigma(A_j) = \sigma(A_i) + a_i$, and if $b_i \geq p$ then $\sigma(A_i) = \sigma(A_j) + a_i$. ;
- For each edge $e = (A'_i, A''_i) \in H_c$ of the matching M, such that $A'_i \in G'_c$ and $A''_i \in G''_c$, the acquisition task A_i associated to the graph G_c is executed after the scheduling.;

Proof. Orman and Potts (Orman and Potts 1997) gave us a theorem that says that a scheduling problem with acquisition tasks, where the objective is makespan, have the same complexity than its symmetrical problem[2] (this is not true for approximation). In Figure 2, there is a symmetry between $1|a_i = L_i = p, b_i|C_{max}$ and $1|a_i, L_i = b_i = p|C_{max}$. By relaxing the incompatibility constraint, the two problems stay symmetric. Thus, problem Π'_4 is symmetric to Π'_3, and we deduce that problem Π'_4 is *polynomial* as Π'_3. The scheduling is optimal with Algorithm 1 by changing b_i by a_i. □

Summary of complexity results

We have shown the $\mathcal{NP}$-completeness of Π'_1 and Π'_2, and the polynomiality of Π'_3 and Π'_4. As we indicated in the introduction of this paper, all problems which were already $\mathcal{NP}$-complete without compatibility graph (see Figure 2) stay $\mathcal{NP}$-complete when G_c is introduced.

For problems which were polynomials without a compatibility graph, the introduction of G_c vary the complexity for some problems (e.g. for Π'_1 and Π'_2) while other problems stay polynomial (e.g. Π'_3 and Π'_4). This leads us to conclude

[2]Two scheduling problems defined by $1|a_i, L_i, b_i, |C_{max}$ and $1|b_i, L_i, a_i, |C_{max}$, with $i = 1, \ldots, n$, are said to be symmetric.

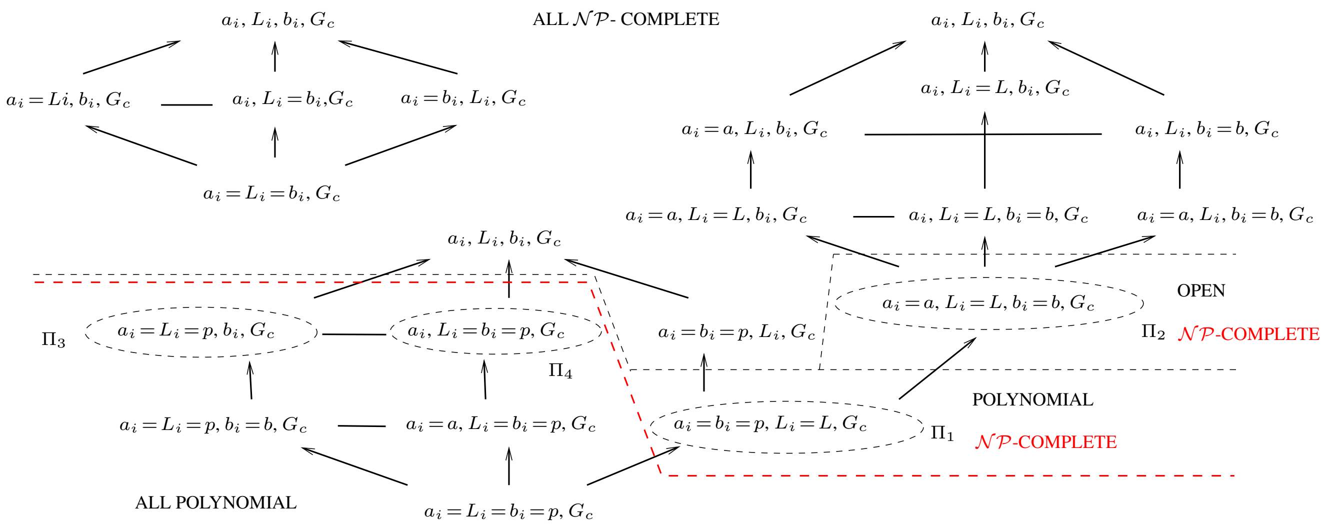

Figure 7: Global visualization of the impact of the incompatibility constraint introduction on scheduling problems complexity with acquisition tasks on single processor. The black dotted line represents results without incompatibility constraint, and the red dotted line when we introduce it.

that the introduction of compatibility graph is an important but not deterministic factor in the complexity of coupled-task scheduling problems.

Figure 7 summarizes the complexity results presented in this paper by reusing the global visualization introduced by Orman and Potts. In the following section, we continue our analysis by proposing a polynomial-time approximation algorithm.

Approximation algorithm for problem Π'_1

This section will be about the approximation study focalized on the $\mathcal{NP}$-complete problem $\Pi'_1 : 1|a_i = b_i = p, L_i = L, G_c|C_{max}$. The problem Π'_2 has been studied in two respective papers (Simonin, Giroudeau, and König July 2010; Simonin et al. 2010).

We are interested in the approximation of $\mathcal{NP}$-complete problem Π'_1. Recall that we work with n acquisition tasks, and when $L \geq 2p$ the adding of the incompatibility constraint means to the $\mathcal{NP}$-completeness of the problem. In order to achieve a schedule closest to the optimal, our research of an heuristic with non-trivial performance guarantee will focus on a study on the compatibility graph G_c. We will give two lower bounds, and an upper bound obtained by a maximal cliques cover of G_c vertices. In the following, let us call C^{opt}_{max} (resp. C^{h}_{max}) the length of an optimal schedule (resp. a schedule from our heuristic) for Π'_1.

Lemma 4. *By considering a maximum matching M of size m in G_c, our lower bound will be $C^{opt}_{max} \geq max\{2np, (n - 2m)(L + 2p) + 2m\}$.*

Proof. The optimal scheduling is obtained when there is no inactivity time, i.e. when the acquisition tasks form blocks, each of them composed by $\beta = (\frac{L}{p} + 1)$ A_i tasks, where $L = kp$ with $k \in \mathbb{N}^*$. These blocks are associated to a cover of vertices from G_c by cliques of size β. Thus, the lower bound satisfies the following inequation :

$$C^{opt}_{max} \geq \text{Sequential Time } = 2np \quad (1)$$

For the second lower bound, by considering a maximum matching M of size m in the compatibility graph, the number of isolated vertices equal $(n-2m)$. In the worst case, the optimal scheduling length need to be superior to the scheduling length obtained by isolated vertices, which form an independent set. Furthermore, we know that a task cannot be executed entirely into another, thus we can add at least $2m$ times the execution time p of a sub-task to the scheduling length (see Figure 8). Thus, we obtain a second lower bound according to a maximum matching M of size m:

$$C^{opt}_{max} \geq (n - 2m)(L + 2p) + 2mp \quad (2)$$

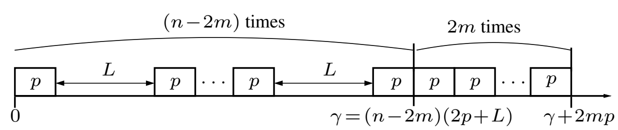

Figure 8: Illustration of the second lower bound

Therefore, according to the parameters values in our study, our lower bound will be the maximum between the two lower bounds (1) and (2):

$$C^{opt}_{max} \geq max\{2np, (n - 2m)(L + 2p) + 2mp\} \quad (3)$$

□

Lemma 5. *The heuristic, based on the research of a vertices cover in G_c by $\mathcal{K}$ maximal cliques of size less than L/p, gives an upper bound equal to $\mathcal{K}(L + p) + np$.*

Proof. The general idea consists in researching maximal cliques of size less than L/p in G_c in order to fill a maximum of slots created by the acquisition tasks. Each maximal clique is associated to the execution of a block of acquisition tasks as previously, but this time the block will not be without inactivity time. In order to compute the achieved scheduling length C^h_{max}, we sum the number of obtained blocks, which create each of them a slot of length L. We add the number of tasks to execute which represents the sequential time of all sub-tasks b_i to execute (See Figure 9).

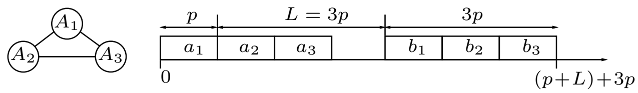

Figure 9: Possible scheduling for a block

The obtained makespan with a vertices cover in G_c by $\mathcal{K}$ maximal cliques gives the following upper bound:

$$C^h_{max} \leq \mathcal{K}(L+p) + \sum_{i=1}^{n} b_i = \mathcal{K}(L+p) + np \qquad (4)$$

□

The relative performance ρ using this heuristic is given by Theorem 3:

Theorem 3. *This heuristic, based on the maximal cliques covering, gives a relative performance equal to* $\rho \leq \frac{4p+L}{4p}$.

Proof. By using the obtained bounds (equations (3) et (4)), we obtain the following relative performance:

$$\rho \leq \frac{C^h_{max}}{C^{opt}_{max}} \leq \frac{\mathcal{K}(L+p)+np}{2np} \qquad (5)$$

This ratio is a general result for our problem, but we can search an other approach using the second lower bound with the matching. We can analyze the value of the relative performance ratio when the heuristic, used to approximate the problem, consists in finding a maximum matching M of size m. In this case, $\mathcal{K} = \boldsymbol{(n-m)}$ because the matching creates $\boldsymbol{m}$ blocks of size $(L+3p)$ and the isolated tasks form $\boldsymbol{(n-2m)}$ blocks of size $(L+2p)$. By substituting $\mathcal{K}$ in the obtained bound in equation (4), we find a new upper bound:

$$C^h_{max} \leq (n-m)(L+p) + np \qquad (6)$$

From the study of the max function in the lower bound (given by equation (2)), we can analyze the behavior of the relative performance. Since $C^{opt}_{max} \geq max\{2np, (n-2m)(L+2p)+2mp\}$, following cases should be considered[3]:

- For $m \in [0, \frac{Ln}{2(p+L)}[$, $C^{opt}_{max} \geq (n-2m)(L+2p)+2mp$
- For $m \in [\frac{Ln}{2(p+L)}, \frac{n}{2}]$, $C^{opt}_{max} \geq 2np$

According to m values, we obtain a new upper bound for our heuristic and a new lower bound for an optimal schedule (see Figure 10). When $m = \frac{Ln}{2(p+L)}$, we see in Figure 10 that the optimal ratio is obtained. The following equations give us the researched value:

$$\begin{aligned}
\rho &\leq \frac{C^h_{max}}{C^{opt}_{max}} \leq \frac{(n-\frac{Ln}{2(p+L)})(p+L)+np}{2np} \\
\rho &= \frac{\frac{(2(p+L)-L)}{2(p+L)}(p+L)+p}{2p} \\
\rho &= \frac{2p+\frac{L}{2}}{2p} = \frac{4p+L}{4p} \qquad (7)
\end{aligned}$$

Note that for $m = 0$, $\rho = 1$ (obviously, since the compatibility graph is a set of independent tasks). Moreover, for $m = \frac{n}{2}$, $\rho = \frac{3p+L}{4p}$. □

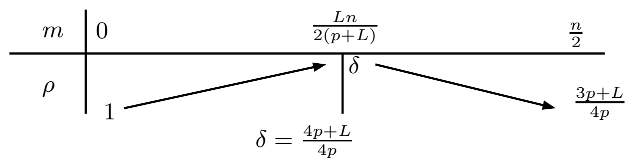

Figure 10: Behavior of the relative performance ρ according to the value of m

This ends the problem Π'_1 analysis. On negative side, we have shown that the problem is $\mathcal{NP}$-complete. On positive side, we gave an approximation algorithm with relative performance bounded by $\rho < \frac{4p+L}{4p}$, where L and p are problem parameters. The fact that the value of relative performance ρ, associated to the algorithm, depends on parameters L and p, leads to continue our work in research of approximation algorithms with a constant performance guarantee.

The approximation study of Π'_2 had been done in (Simonin et al. 2010). For this problem, we study the limit between polynomiality and $\mathcal{NP}$-completeness according to the values of parameter L when it depends on a and b.

Conclusion

We have studied throughout this paper the scheduling problems on single processor with coupled-tasks in presence of arbitrary compatibility graph G_c. The different problems encountered arise because we vary basic parameters (a_i, L_i, b_i) of coupled-tasks in the same manner as do Orman and Potts in their paper on the study of coupled-tasks without incompatibility constraint. The goal sought throughout our paper was to determine the impact of incompatibility constraint on these problems, and to analyze critical cases located at the limit between polynomiality and $\mathcal{NP}$-completeness according to parameters value.

We have presented two $\mathcal{NP}$-completeness proofs for problems Π'_1 and Π'_2, and two polynomial proofs for problems Π'_3 et Π'_4. Figure 7 summarizes the complexity results presented in this paper. The first observation is that the introduction of incompatibility constraint has a significant impact on the complexity of some problems: e.g. problem Π_1 which was solvable in polynomial time becomes $\mathcal{NP}$

[3] We search the value of m in order to obtain $C^{opt}_{max} \geq (n-2m)(L+2p)+2mp$ or $C^{opt}_{max} \geq 2np$.

Problem	Complexity	Ratio.	Ref.
$\Pi_1' : (a_i = b_i = p, L_i = L), G_c$	$\mathcal{NP}$-complete	$\frac{4p+L}{4p}$	this paper
$(a_i = L_i = b_i), G_c$	$\mathcal{NP}$-complete	$\frac{3}{2}$	(Simonin, Giroudeau, and König July 2010)
$\Pi_2' : (a_i = a, L_i = L = a+b, b_i = b), G_c$	$\mathcal{NP}$-complete	$\left[\frac{3}{2}, \frac{5}{4}\right]$	(Simonin et al. 2010)
$\Pi_3' : (a_i = L_i = p, b_i), G_c$	Polynomial	1	this paper
$\Pi_4' : (a_i, L_i = b_i = p), G_c$	Polynomial	1	this paper

Table 1: Summarize of results

in the presence of compatibility graph (problem Π_1'), leading to the $\mathcal{NP}$-completeness of Π_2' while Π_2 was still open. From these results, we deduce the $\mathcal{NP}$-completeness of all more general problems. In a second point, we have proposed a polynomial-time algorithm for problems Π_3' and Π_4', and show the polynomiality of more specific problems.

In a second part we have presented a polynomial approximation algorithm for Π_1' with a performance ratio $\frac{4p+L}{4p}$, where p and L are fundamental parameters of the problem. This heuristic completes previous approximation results investigated in previous works, summarized in Table 1.

It is interesting to observe that problems complexity depends largely on link between parameter L_i and one of the other two: a_i or b_i. If L_i is equal to a_i or b_i, the only way to schedule tasks is either to overlap them two by two, or to execute them consecutively. This configuration leads us to search maximum matching or perfect in compatibility graph. When L_i is independent of the other two parameters, the possible schedules of tasks lead to seek chains, or cliques in G_c, and most of these problems are known to be $\mathcal{NP}$-complete.

General observation that we can do on approximation of studied problems is the following: introduction of incompatibility constraint is fundamentally changing traditional approach to this kind of problem, and led to study graph problems known to be hard to approximate. As obtained approximation bounds depend on L_i most of the time, perspectives of this work consist in determining existence or not of constant factor approximation algorithms for NP-complete problems.

References

Ageev, A. A., and Baburin, A. E. 2007. Approximation algorithms for uet scheduling problems with exact delays. *Operations Research Letters* 35(4):533 –540.

Ahr, D.; Békési, J.; Galambos, G.; Oswald, M.; and Reinelt, G. June 2004. An exact algorithm for scheduling identical coupled-tasks. *Mathematical Methods of Operations Research* 59:193–203(11).

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

Brauner, N.; Finke, G.; Lehoux-Lebacque, V.; Potts, C.; and Whitehead, J. 2009. Scheduling of coupled tasks and one-machine no-wait robotic cells. *Computers & OR* 36(2):301–307.

Edmonds, J. 1965. Maximum matching and a polyhedron with $0, 1$ vertices. *Journal of Research the National Bureau of Standards* 69 B:125–130.

Gabow, H. N. 1976. An efficient implementation of edmonds' algorithm for maximum matching on graphs. *Journal of the ACM* 23(2):221 – 234.

Garey, M. R., and Johnson, D. S. 1979. *Computers and Intractability: A guide to the theory of NP-completeness*. New York, NY, USA: W. H. Freeman & Co.

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

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

Orman, A. J.; Potts, C.; Shahani, A. K.; and Moore, A. R. 1996. Scheduling for a multifunction phased array radar system. *European Journal of Operations Research* 90:13–25.

Orman, A. J.; Shahani, A. K.; and Moore, A. R. 1998. Modelling for the control of a complex radar system. *Computers & OR* 25(3):239–249.

Shapiro, R. 1980. Scheduling coupled tasks. *Naval Research Logistics Quarterly* 27:477–481.

Sherali, H. D., and Smith, J. C. 2005. Interleaving two-phased jobs on a single machine. *Discrete Optimization* 2(4):348–361.

Simonin, G.; Darties, B.; Giroudeau, R.; and König, J.-C. 2010. Isomorphic coupled-task scheduling problem with compatibility constraints on a single processor. *To appear in Journal of Scheduling*.

Simonin, G.; Giroudeau, R.; and König, J.-C. July 2010. Polynomial-time algorithms for scheduling problem for coupled-tasks in presence of treatment tasks. *Electronic Notes in Discrete Mathematics* 32:647–654.