Approximation algorithm for constrained coupled-tasks scheduling problem

Gilles Simonin *, Benoit Darties †, Jean-Claude König ‡ and Rodolphe Giroudeau ‡

†LIRMM UMR 5506 CNRS - Université de Montpellier II
161 rue Ada 34392 Montpellier Cedex 5, France
Email: {rodolphe.giroudeau, jean-claude.konig}@lirmm.fr

‡LE2I-CNRS UMR 6306 - University of Burgundy
8 Rue Alain Savary 21000 Dijon, France
benoit.darties@u-bourgogne.fr

* Insight Centre for Data Analytics - University College Cork - Ireland
Email: gilles.simonin@insight-centre.org

***Abstract*—We tackle the makespan minimization coupled-tasks problem in presence of compatibility constraints. In particular, we focus on stretched coupled-tasks, *i.e.*coupled-tasks having the same sub-tasks execution time and idle time duration. In such context, we propose some complexity results according to several parameters and we design an efficient polynomial-time approximation algorithm.**

I. Presentation of stretched coupled-task model

In this article, we consider a extended coupled-tasks scheduling problem (see [11], [12]) in which the tasks are subjected to compatibility constraints. Coupled-tasks, introduced by Shapiro [13], are an easy way to model some data acquisition processes: a coupled-task is composed by two sub-tasks of processing time a_i and b_i and whose execution must be separated by a fixed interval time L_i (called the idle time of the task). Applications from coupled-tasks includes, among others, radar detection process on embedded systems: a sensor emits a radio pulse as a first sub-task, and finally listen for an echo reply as a second sub-task. Between these emission and reception, there is an idle time due to the propagation, in both sides, of radio pulse. The aim is the makespan minimization of a set of non-preemptive coupled-tasks on a mono-processor. This requires to allot one or several different sub-tasks during the idle time of a coupled-task. (the problem is also defined as a scheduling problem with exact delays on single machine). The complexity of this problem has been previously studied for various durations of the sub-tasks and their idle time. In this paper, we complete preliminary results dedicated to a specific configuration and presented in [5]:

- All coupled-tasks are stretched coupled-tasks: for each task the processing time of the two sub-tasks and the idle time are equal to the same value called the task stretch factor.
- Compatibility constraint arise between coupled-tasks. We say two 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. We introduce a graph $G_c = (A, E_c)$ to model such this compatibility, with E_c link any pair of compatible coupled-tasks.

978-1-4799-6773-5/14/$31.00 ©2014 IEEE

Each coupled-task $A_i = (a_i, L_i, b_i)$ composed by two sub-tasks of processing time a_i and b_i, respectively dedicated for wave transmission and echo reception. Between these two sub-tasks there is a fixed idle time L_i which represents the spread of the echo in the medium. We work in a non-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. This paper focuses on *stretched* coupled-tasks, *i.e.* coupled-tasks for what the durations of the first sub-task, the second sub-task and the idle time are equal to a stretch-factor applied to an original task (by example $(a_i, L_i, b_i) = (1, 1, 1)$). Formally, a stretched coupled-task A_i is a task such that $a_i = L_i = b_i = \alpha(A_i)$, where $\alpha(A_i)$ is the stretch factor of the task. In the rest of the paper, coupled-tasks are always stretched coupled-tasks, and noted A_i when we need to refer to the values a_i, b_i and L_i, or with a single identifier, *i.e.* x, otherwise. In such configuration, for two compatible tasks A_j and A_j to be scheduled in parallel, one of the following conditions must hold:

1) either $\alpha(A_i) = \alpha(A_j)$: then the idle time of one task is fully exploited to schedule a sub-task from the other (i.e. b_i is scheduled during L_j, and a_j is scheduled during L_i), and the completion of the two tasks is done without idle time.
2) or $3\alpha(A_i) \leq \alpha(A_j)$: then task A_i is fully executed during the idle time L_j of A_j. For sake of simplify, we say we *pack* A_i into A_j.

In follows, we focus in specific compatibility graph G_C by considering a 1-stage bipartite graph *i.e.* a bipartite $G_c = (X, Y, E_c)$ graph of depth one. We introduce $\#X$ which count up the number of different elements in the set X. Moreover, $\Delta_{G_c}(X)$ (resp. $\Delta_{G_c}(Y)$) represents the maximum degree of the set X (resp. Y). In the same way, $d(Y)_{G_c}$ gives the exact degree of each task in the Y-set. The results are summarized

Topology	Complexity	Approximation
G_c=Star graph	$\mathcal{NPC}$ (see [5])	$\mathcal{FPTAS}$ (see [5])
G_c=Chain graph	$O(n^3)$ (see [5])	
G_c= 1-ST, $\#(X)=1$ $\#(Y)\leq n, \Delta_{G_c}(Y)=2$	$O(n^3)$ (Theo. 1)	
G_c=1-ST $\#(X)=1$ $\#(Y)\leq n$	$O(n^3)$ (Theo. 2)	
G_c= 1-ST, $\Delta(G_c)=3$	$\mathcal{NPC}$ (Theo. 3)	$\frac{7}{6}$-$\mathcal{APX}$(see [5])
G_c= complete 1-ST	$\mathcal{NPC}$ (see [10])	$\mathcal{PTAS}$ (see [5])
G_c= complete 1-ST $\alpha(x)=\alpha(y), \forall x,y\in X_1$	$\mathcal{NP}-\mathcal{C}$ (see [10])	$\mathcal{PTAS}$ (see [5]) $\frac{13}{12}$-$\mathcal{APX}$(see [5])
G_c= 2-ST	$\mathcal{NPC}$ (see [5])	$\frac{13}{9}$-$\mathcal{APX}$ (see [5])
G_c= k-ST	$\mathcal{NPC}$ (see [5])	$\frac{40}{21}$-$\mathcal{APX}$ (Theo. 4)

TABLE I. COMPLEXITY AND APPROXIMATION RESULTS.

in Table I (ST for stage bipartite). Notice that approximation results are based on results given by [1], [2], [3] and [6].

II. HARDNESS AND SOLVABLE CASE

We start by design a polynomial-time algorithm for the scheduling problem in which the maximum degree of incoming arcs on Y-tasks is at most two.

Theorem 1: The problem of deciding whether an instance of $1|\alpha(A_i), G_c = 1-stage\ bipartite, \#X \leq n, \#Y \leq n, \Delta_{G_c}(Y)=2|C_{max}$[1] is polynomial.

Proof: Let $G_c=(X,Y,E)$ be a 1-stage bipartite compatibility graph. Y-tasks will always be scheduled sequentially. The aim is to fill their idle time with a maximum of tasks of X, while the remained tasks will be executed after the Y-tasks. We just have to minimize the length of the remained tasks. Note that $\Delta_{G_c}(y)=2, \forall y\in Y$. The algorithm use three steps:

1) For each task $y\in Y$ such that $3\times\alpha(x_1)+3\times\alpha(x_2)\leq\alpha(y)$ where x_1 and x_2 are the only two neighbors of Y, we add y to the schedule and execute x_1 and x_2 sequentially during the idle time of y. Then we remove y, x_1 and x_2 from the instance.
2) Each remaining task $y\in Y$ admits at most two incoming arcs (x_1,y) and / or (x_2,y). We add a weight $\alpha(x)$ to the arc $(x,y), \forall x\in N(y)$, then perform a maximum weight matching on G_c in order to minimize the length of the remained tasks of X. Thus, the matched coupled-tasks are executed, and these tasks are removed from G_c.
3) Then, remaining X-tasks are allotted sequentially after the other tasks.

The complexity of an algorithm is $O(n^3)$ using the Hungarian method [9]. ■

Theorem 2: The problem of deciding whether an instance of $1|\alpha(A_i), G_c = 1-stage\ bipartite, \#(X)=1, \#(Y)\leq n|C_{max}$ is polynomial.

Proof: Assume that the arcs are oriented $X\rightarrow Y$. Let I be a instance with $d_G(y_i)\geq k_i+1, \forall y_i\in Y$. Indeed, if $d_G(y_i)\leq k_i$, the schedule of Y-tasks and the $\Gamma^-(y_i)$ are obvious. We assume that G is connected. The scheduling problem is transformed into a maximum flow problem in the following ways:

- We add a source s and a sink t. We add arcs $(s,x), \forall x\in X$, (resp. $(y,t), \forall y_i\in Y$) with capacity $c(sx)=1$ (resp. $c(y_it)=k_i$ with $k_i=\lfloor\frac{3\alpha(x)}{\alpha(y_i)}\rfloor$).
- Finally, we add $c(xy_i)=1, \forall(x,y_i)\in E$.

The resulted graph is denoted by $G'=(s,t,X,Y,E')$.

Now, we show that the flow f admits a maximum flow with $F=\sum_{i=1}^{l}k_i$ with $l=|Y|$. Suppose that the flow f is maximum with value $F=\sum_{i=1}^{l}k_i-1$. Therefore it exists one arc $(\tilde{y_i},t)$ unsaturated by the flow f. Consequently, we have one arc $(x^*,\tilde{y_i})\in E, f(x^*,\tilde{y_i})=0$. Then it exists one vertex $y_i^*\neq\tilde{y_i}\backslash f(x^*,y_i^*)=1$ otherwise the task x^* must be packed into $\tilde{y_i}$. Moreover, we assume that it exists one arc $(\tilde{x},\hat{y_i})$ with $f(\tilde{x},\hat{y_i})=0$ and $f(\hat{x},\hat{y_i})=1$.

Since G_c is connected, the chain $[x^*,y_1,x_1,\ldots,y_k,\tilde{x}], k\geq 1$ of length even exists. We change the flow value $f(x,y)=z\rightarrow f(x,y)=\bar{z}$, for all pair of vertices in this chain, with $z\in\{0,1\}$ and finally we put $f(x^*,\tilde{y})=1$. Therefore, it exists a new feasible flow f with a value strictly greater than F. Since it exists a $s-t$ cut with capacity $\sum_{i=1}^{l}k_i$, thus the maximum value of f is necessarily $\sum_{i=1}^{l}k_i$.

The X-tasks such that $f(x,y_i)=1$ in G' are packed into the y_i-task., otherwise the X-tasks are executed consequently after the Y-tasks.

■

Theorem 3: The problem of deciding whether an instance of $1|\alpha(A_i), G_c = 1-stage\ bipartite, \#(X)=2, \#(Y)=1, d(X)_{G_c}\in\{1,2\}, d(Y)_{G_c}\in\{3,4\}|C_{max}$ is $\mathcal{NP}$-hard.

Sketch of Proof 1: The proof is based on a reduction from the 3 DIMENSIONAL MATCHING (see [7]): let A, B, and C be finite, disjoint sets of the same cardinality n, and T be a subset of $A\times B\times C$ of cardinality t.

The aim is to find $M\subseteq T$ with maximum cardinality, such that for two distinct triples $(u_a,u_b,u_c)\in M$ and $(v_a,v_b,v_c)\in M$ we have $u_a\neq v_a$, $u_b\neq v_b$, and $u_c\neq v_c$. This problem is well known to be $\mathcal{NP}-$-hard even if the occurrence of each element of $A\cup B\cup C$ is at most 2 in the T-subsets [4] (in this case, we have $t=2n$).

We define a set of tasks $X\cup Y$ and model the compatibility constraint with a graph $G_c=(X,Y,E)$. For each element $x_i\in A\cup B\cup C$, we add an *item* coupled-task x_i into X with $\alpha(x_i)=1$. For each 3-element subset $s=(x_a,x_b,x_c)\in T$, we add a *box* coupled-task y_s to Y with $\alpha(y_s)=9$, and an *item* coupled-task x^s with $\alpha(x_{y_s})=2+\epsilon$. We also add the compatibility arcs $\{(x^s,y_i),(x_a,y_i),(x_b,y_i),(x_c,y_i)\}$ to E. Clearly we have $t=2n$ *box* coupled-tasks (each with an idle time of 9 units) of degree 4 in G_c, $t=2n$ *item* coupled-tasks x^s with stretch factor $2+\epsilon$ of degree 1 in G_c, and $3n$ *item* coupled-tasks with stretch factor 1 of degree 2 in G_c. Moreover G_c is a bipartite graph.

The construction implies that for any valid schedule, if 1 *item* coupled-tasks with stretch factor $2+\epsilon$ is packed into a *box* coupled-task, then no other task from X can be packed with it. Suppose we obtain on this instance a schedule of length $27.t+m.3(2+\epsilon)+3.3(n-m)=63n-3m(1-\epsilon)$, with

[1]We use the Graham's notation scheme $\alpha|\beta|\gamma$ [8]

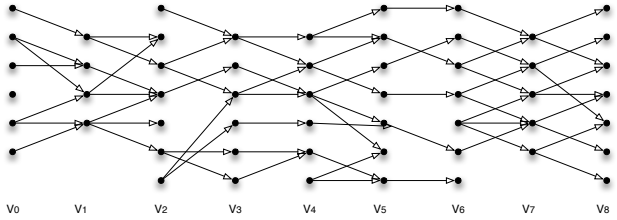

Fig. 1. A k-stage bipartite graph, here $k = 8$

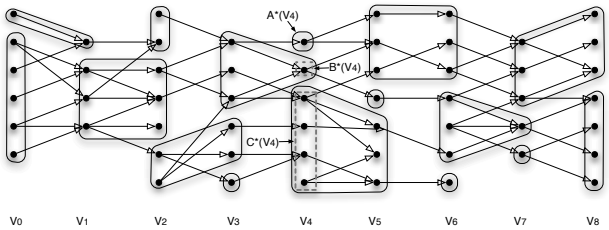

Fig. 2. An optimal solution on a k-stage bipartite graph

$m \leq n$, then the idle time of m *box* coupled-tasks is used to schedule $3.n$ *item* coupled-tasks with stretch factor 1, the idle time of $t-m$ other *box* coupled-tasks is used to schedule *item* coupled-tasks with stretch factor $2+\epsilon$, and $3(n-m)$ remaining tasks from X are scheduled sequentially.

Then one can find a solution to $3 - DM$ with cardinality m.

III. New approximation results

A k-stage bipartite graph is a digraph $G = (V_0 \cup \cdots \cup V_k, E_1 \cup \cdots \cup E_k)$ where $V_0 \dots V_k$ are disjoint vertex sets, and each arc in E_i is from a vertex in V_i to a vertex in V_{i+1}. The vertex of V_i are said to be at rank i, and the subgraph $G_i = (V_{i-1} \cup V_i, E_i)$ is called the i-th stage of G, and we write $G = G_1 + \cdots + G_k$.

Figure 1 presents such a k-stage bipartite graph with $k = 8$.

In the following, we note S the cost of the solution produced by our algorithm, and S^* the cost of the optimal one, using in the rest of the paper a $*$-character exponent to denotes the sets / values related to this optimal solution. We note $S[W]$ (resp. $S^*[W]$) the cost of a valid (resp. optimal) schedule on the instance induced by tasks $W \subseteq V$ with compatibility graph $G[W]$. We aim to show one can produce a schedule with cost $S = S[V] \leq \frac{41}{30} S^*[V] = \frac{41}{30} S^*$ in a polynomial time.

Compatibilities existing only between tasks of two consecutive ranks and arcs-transitive are not considered: none schedule can Suppose that x is executed during idle of y, then if y is already processed during z-idle time, therefore these allocations lead to a non-feasible schedule, for any x, y, z. In any solution, V-tasks can be 3-partitioned as (1) tasks scheduled alone, noted $A(V)$, (2) tasks with at least one other task scheduled during it idle time noted $B(V)$, and (3) tasks scheduled during the idle time of another one, noted $C(V)$. This notation is extendable to any subset of V, i.e. X_i.

Let's consider an optimal solution S^*, also depicted by Figure 2. Each stage X_i is partitioned into $\{A^*(X_i), B^*(X_i), C^*(X_i)\}$ represented with "tirets". "Potatoes" on the figure helps to identify the different set of tasks scheduled in parallel: $C^*(X_i)$-tasks are allotted during the idle time of $B^*(X_{i-1})$-tasks. During the idle time of $C^*(X_{i+1})$ the $B^*(X_i)$-tasks are processed. Lastly, $A^*(X_i)$-tasks are scheduled alone.

Theorem 4: The problem $1|\alpha_i = a_i = L_i = b_i, k - stage\ bipartite|C_{max}$ admits a $41/30$-approximation algorithm.

Proof: Due the lack of place, we prove is omitted. Only the main idea is given. The idea consists to compute a $\frac{7}{6}$-approximation on each stage G_i for each i odd thanks to previous results [5], and to merge these parts to get a $41/30$-approximated solution. This requires $\sum_{v \in V_i | i even} \alpha(v) > \sum_{v \in V_i | i odd} \alpha(v)$. If not, use the same algorithm with i even instead of odd. ■

IV. Conclusion

We propose some news results on complexity and approximation for constrained coupled-tasks scheduling problem in presence of specific compatibility graph (bipartite). This article supplement the previous works given by [5].

References

[1] A. Caprara, H. Kellerer, and U. Pferschy. A PTAS for the Multiple Subset Sum Problem with different knapsack capacities. *Information Processing Letters*, 2000.

[2] A. Caprara, H. Kellerer, and U. Pferschy. The Multiple Subset Sum Problem. *Siam Journal on Optimization*, 11(2):308–319, 2000.

[3] A. Caprara, H. Kellerer, and U. Pferschy. A 3/4-Approximation Algorithm for Multiple Subset Sum. *J. Heuristics*, 9(2):99–111, 2003.

[4] M. Chlebík, and J. Chlebíková, Approximation Hardness for Small Occurrence Instances of NP-hard Problems, *Proceedings of the 5th Italian Conference on Algorithms and Complexity*, 2003.

[5] B. Darties, G. Simonin, R. Giroudeau, J.-C. König. Coupled-tasks in presence of bipartite compatibilities graphs *Proceedings of ISCO 2014, 3th International Symposium on Combinatorial Optimization*, LNCS, No. 8596, pages 161–172. Springer-Verlag, 2014.

[6] M. Dawande, J. Kalagnanam, P. Keskinocak, F.S. Salman, and R. Ravi. Approximation Algorithms for the Multiple Knapsack Problem with Assignment Restrictions. *Journal of Combinatorial Optimization*, 4(2):171–186, 2000.

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

[8] R L Graham, E L Lawler, J K Lenstra, and A H G Rinnooy Kan. Optimization and Approximation in Deterministic Sequencing and Scheduling: a Survey. *Annals of Discrete Mathematics*, 5, 1979.

[9] H. Kuhn, The Hungarian Method for the assignment problem *Naval Research Logistics Quarterly*, 1955

[10] G. Simonin, R. Giroudeau and J.-C. König. Complexity and approximation for scheduling problem for coupled-tasks in presence of compatibility tasks. In *Project Management and Scheduling*. 2010.

[11] G. Simonin, B. Darties, R. Giroudeau, and J.C. König. Isomorphic coupled-task scheduling problem with compatibility constraints on a single processor. *Journal of Scheduling*, 14(5):501—-509, 2011.

[12] G. Simonin, R. Giroudeau, J.C. König, and B. Darties. Theoretical Aspects of Scheduling Coupled-Tasks in the Presence of Compatibility Graph. *Algorithmic in Operations Research*, 7(1):1—-12, 2012.

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