Identical coupled task scheduling: polynomial complexity of the cyclic case

Vassilissa Lehoux-Lebacque, Nadia Brauner, Gerd Finke

Identical coupled task scheduling: polynomial complexity of the cyclic case

Vassilissa Lehoux-Lebacque, Nadia Brauner and Gerd Finke

May 20, 2009

Abstract

Coupled task problems arise in connection with radar systems. For the transmission and reception of an electromagnetic pulse the radar (the processor) has to execute two tasks that are separated by some fixed time interval. Most of the complexity issues for scheduling a set of such pairs of two-operation tasks have been settled. However, the complexity status is still unknown for the identical coupled task problem, where multiple copies of a single coupled task are to be processed. The purpose of this article is prove the polynomial complexity of this problem in the cyclic case.

1 Introduction

Coupled tasks were introduced by Shapiro in 1980 for scheduling the operations of a radar [19]. The radar emits a pulse that is transmitted to a target and reflected back to the radar, which receives the pulse. Hence the radar must process two operations per task (emission and reception), and those operations are separated by a fixed duration.

The coupled task scheduling problem is then to process tasks on a single machine (the radar) with each task j composed of two operations of lengths a_j and b_j separated by exactly L_j time units. The objective is to minimize the makespan in the non-cyclic case (given a fixed number of coupled tasks) or the throughput rate in the cyclic case (infinite number of tasks).

Coupled task problems belong to the wider class of scheduling multi-operation tasks, where consecutive operations are separated by a certain time interval. In manufacturing processes, the time that has to elapse between operations (delays, time-lags) are often lower bounded (see for instance [12]). We shall here consider only coupled tasks with fixed separation intervals as they apply to radar systems. There is a vast literature on this subject, treating offline and online cases and proposing various algorithms [19, 18, 17, 15, 11, 10, 9].

Let us now consider offline coupled task problems, where a set of tasks $\{a_j; L_j; b_j\}$ has to be scheduled on a single processor (the radar) with interleaving the coupled tasks but without overlapping the operations. In particular, the case $a_j = a$, $L_j = L$, $b_j = b$ for all j is called the identical coupled task problem.

Recently Brauner *et al.* [8] have described a new application of this coupled task model. They show that this problem is equivalent to a no-wait one-machine robotic cell problem usually studied in cyclic mode. In Figure 1, A_j is the transportation time of part j from the input station to the machine, α is the empty return time and similarly, from M to the output station (B_j and β). For details on the equivalence, see [8]. In particular, producing a large number of single parts corresponds to the identical part production cycle. For this case, studying cyclic identical coupled tasks scheduling problems becomes a more relevant application.

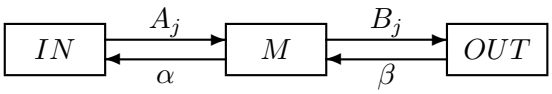

Figure 1: 1-machine robotic cell

The complexity of minimizing the makespan has been described by Orman and Potts [16]. Even the UET problem, $a_j = 1$, L_j, $b_j = 1$ for all j, is NP-hard and algorithms with worst-case performance ratio have been developed [1, 4]. However, the complexity status of the identical coupled task problem remains open for both the non-cyclic case (Table 1) and also for the cyclic case (see [2]).

Table 1: Complexities for the coupled task problem from [16]

Complexity	**Case**
Strongly NP-hard	a_j; L_j; b_j
	$a_j = L_j = b_j$
	$a_j = a$; L_j; $b_j = b$
	$a_j = a$; $L_j = L$; b_j
Open	$a_j = a$; $L_j = L$; $b_j = b$
Polynomial	$a_j = L_j = p$; b_j
	$a_j = b_j = p$; $L_j = L$

In this paper, we concentrate on the identical coupled task problem. Notice that the input of this problem is composed of four integers (three in the cyclic case): n, the number of tasks, a the duration of the first operation of a task, b the duration of the second operation and L the distance between both operations. Hence, it is a high-multiplicity scheduling problem [6, 7] for which even proving that it belongs to NP might be difficult. Indeed, a description of a schedule (giving for instance the starting time of each of the n tasks) is not polynomial in the input size (which is, in our case, $\log n + \log a + \log b + \log L$). Ahr *et al.* [2] propose an algorithm linear in n but exponential in L. This algorithm has been adjusted to the cyclic case in [2] showing that the cyclic problem corresponds to finding the minimum mean cycle in a certain graph of 0-1 patterns (see also [14, 8]). This problem is polynomial in the size of the underlying graph [13] which, however in our case, has an exponential number of vertices. The computational experience in [8] shows, even with a significant reduction of the number of vertices, that for $a = 5$, $b = 3$ and $L = 41$, we have almost 9 000 vertices in the graph and the solution takes almost an hour computation time on a standard PC. Increasing L to 43, results in more than 14 000 vertices and we were not able to solve this instance with such an approach. In [3] it has been shown that for fixed a, b and L the optimal solution can be found in constant time. The constant is

highly exponential in L and does not yield any practical computation. Note that one gets an $O(\log n)$ algorithm, including the input, which improves the $O(n)$ running time in [2].

We also want to mention that extensions of the identical coupled task problem turn out to be NP-hard. In [5] it is shown that the addition of strict precedences of identical coupled tasks $\{a = 1; L; b = 1\}$, *i.e.* ordered pairs of tasks are given that are not allowed to interleave, makes the problem NP-hard. Similarly, the problem $\{a; L; b\}$ becomes NP-hard if one adds a task-compatibility graph where two coupled tasks are compatible if they may interleave [20].

The identical coupled task problem is indeed very intriguing. It appears simple: a single type of oriented geometric object (the coupled task) is to be packed linearly on the real line in an optimal manner. The reason why the complexity status is still open after so many years seems to be that one does not know enough about the structural properties of the optimal solution patterns.

We first describe in Section 2 a class of solutions for the cyclic identical coupled task problem for which we determine the optimal solution in polynomial time. Then, in Section 3 we show that the optimal solution of this class is also optimal for our general problem. The cyclic identical coupled task problem is therefore solvable in polynomial time.

2 A class of solutions

It is easy to construct feasible solutions for the cyclic identical coupled task problem with given integers a, b, L. Take for instance $a = 5$, $b = 3$ and $L = 43$, a problem that could not be solved by the graph method. One can always pack, as long as possible, coupled tasks in succession without inserting unnecessary idle times. This would give the pattern $aaa \ldots abbb \ldots b$, which can then be repeated. For the example, there would be 9 a's, an idle time of 3 units, followed by 9 b's (which are separated by $a - b = 2$ units). We have placed 9 coupled tasks on the length 91 (the ratio is then $10\frac{1}{9}$).

The previous solution is working in both modes (cyclic and non cyclic). In Figure 2, we give another solution that is rather surprising and very simple. It turns out later that this is in fact an optimal solution. Note first that a cyclic solution is based on an infinite number of tasks. One is therefore looking for a feasible pattern of a's and b's, which can be attached identically and repeatedly to the left and the right to a copy of the given pattern. Thus in the cyclic mode there is no particular starting and finishing phase of the solution. The solution is somewhat unexpected since the placement is partially at fractional starting times for the coupled tasks. It is interesting to notice that such fractional placements cannot be detected by the graph methods in [2, 8, 14, 3]. Here we have placed one coupled task in the cyclic sense on 8.5 units which is composed of 'a', an idle time of 0.5 units and 'b' (ratio of $8\frac{1}{2}$).

In this section, we construct a class $\mathcal{L}$ of schedules for which we determine the optimal solution in polynomial time for given positive integers a, b and L.

Without loss of generality, we assume that $a > b$ (the problem is trivial for $a = b$ and if $b > a$ we can reverse the optimal cycle). Now consider a cyclic solution (in short, a cycle) $\mathcal{C}$. Each coupled task on $\mathcal{C}$ can be described by its *window* $W = [a, L, b]$ giving the sequence S of a-terms and b-terms that are placed in the L-section of W. We will

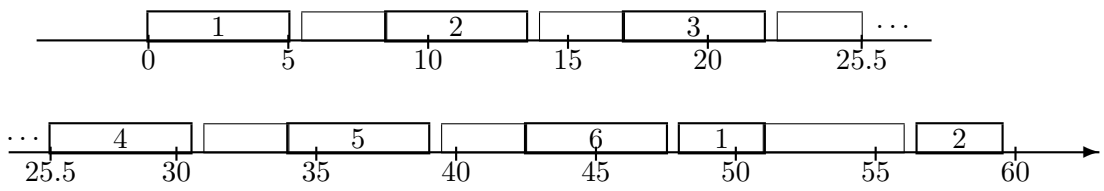

Figure 2: A feasible cyclic solution for $a = 5$, $b = 3$ and $L = 43$

also assume that $L \geq a + b$ since otherwise the problem is trivial: each task window can contain at most one operation on its L-section.

As an illustration, let's use $S = aabbbabaaabbbbab$. Notice that two consecutive b-terms are at least separated by $a - b$ time units. Equality with $(a - b)$ occurs in fact if the corresponding a-terms are consecutive without idle time. Therefore, the space utilization of the first b-term of two consecutive b-terms is at least a on the L-section of the window. Let us distinguish the two versions of the space length a: we keep the notation a for the a-term of a starting coupled task and we use the notation $\bar{a}$ for the b-term of a coupled task, followed by the idle time $a - b$. Then, our given sequence utilizes at least the space $aa\bar{a}\bar{a}babaaa\bar{a}\bar{a}\bar{a}ba\bar{a}$. Notice that the final b transforms to the space $\bar{a}$ since it is followed by the final b of the coupled task of W.

Writing the sequence for the space utilization in the form $aa\bar{a}\bar{a}(ba)(ba)aa\bar{a}\bar{a}\bar{a}(ba)\bar{a}$, we first count the ba-terms. There are $\beta = 3$ such terms. Then we count the remaining a-terms and $\bar{a}$-terms. There are $\alpha = 10$ such terms. In this way, we get, for each window of $\mathcal{C}$, a solution of the constraint set

$$\begin{cases} L = \alpha a + \beta(a + b) + \gamma \\ \alpha, \beta, \gamma \geq 0 \end{cases} \tag{1}$$

The sequence S consists of $\alpha + 2\beta$ elements. We call (α, β, γ), or in short (α, β), the *profile* of window W (where γ is the slack variable). Usually, different task windows of a cycle will have different profiles. Let us define the initial window W_0 of a given cycle $\mathcal{C}$ as a window that contains the most a's. Let W_1 be the window that starts with the last a-term in window W_0. The intersection $W_0 \cap W_1$ can contain some idle time ϵ, but there is no additional b-term. Otherwise there would be a window with more a-terms than in W_0. The union $W_0 \cup W_1$ has the total length of $(2L + a + b - \epsilon)$ units.

It is somewhat remarkable that, for any feasible solution (α, β) of (1), one can construct cycles so that every task window W of these solutions possesses the same profile (α, β). We shall now describe in detail this construction.

2.1 Construction of the feasible cycle $\mathcal{C}(\alpha, \beta)$

For a given profile (α, β), we normalize the order of elements in the initial task window as follows:

$$\mathbf{a}\ a^{\alpha}(ba)^{\beta}\ \mathbf{b}$$

We extend this sequence for $W_0 \cup W_1$ in the following way:

$$Z = a^{\alpha+1}(ba)^{\beta}b^{\alpha+1}(ab)^{\beta}$$

Here the term $a^{\alpha+1}$ is belonging to $b^{\alpha+1}$ and the a's of $(ba)^{\beta}$ are combined with the b's of $(ab)^{\beta}$ to form the coupled tasks. This means that we try to place exactly $(1+\alpha+2\beta)$ coupled tasks, in the cyclic sense, in $W_0 \cup W_1$. As mentioned before, two consecutive b's are separated by at least $(a-b)$ idle time units. The sequence hence transforms to

$$a^{\alpha+1}(ba)^{\beta}\bar{a}^{\alpha}(ba)^{\beta}b$$

as far as space utilization is concerned. So far, we just defined the letter-pattern Z (also called *letter-cycle*). To get a valid feasible cycle, if it exists, we have to insert idle times in the right places and eventually use a succession of several of these letter-cycles for a single cyclic solution. But we can already verify, from the last sequence, that every window does in fact have the same profile (α, β).

Without any restriction, we can assume that all coupled tasks of our cycles are left-shifted as much as possible. We will now give the construction of a cycle that has the same profile (α, β) in every window.

This cycle is denoted by $\mathcal{C}(\alpha, \beta, \gamma)$ or in short $\mathcal{C}(\alpha, \beta)$, and may eventually be based on k repetitions of the letter-cycle Z. The value of k has to be found. In order to get a correct cycle, also the pattern of idle times has to repeat exactly. Let us now define the class $\mathcal{L}$ of cycles that consists of the set $\{\mathcal{C}(\alpha, \beta)$: (α, β) is a profile$\}$. We have assumed that the placements are left-shifted as much as possible. When starting the cycle with the first letter-cycle, we don't know in advance the best placements of the b-terms since they belong to earlier placed coupled tasks. We consider the initial window W_0, which by definition contains the most a's.

For the construction, we place the idle times in front of the ba-terms (ϵ_1 to ϵ_β) in W_0, and additionally an idle time (ϵ) in front of the b-term of the initial coupled task in order to equilibrate correctly the total idle time to γ as required. We assume that all a-terms are scheduled without idle time.

With this assumption, the vector of idle times in the first window W_0 is of the form

$$[\epsilon_1, \epsilon_2, \ldots, \epsilon_\beta, \epsilon = \gamma - \sum \epsilon_i]$$

Remember that after W_0, all coupled tasks are systematically placed as early as possible. Therefore, all following idle times in the sequence Z, Z, ... are uniquely determined as a function of ϵ_i and ϵ.

Now consider an arbitrary window in the sequence. We have the intrinsic idle time of length $(a-b)$ between two b-terms, which we write as $\bar{a}b$. The other idle times may be obtained as follows.

Rule 1 (idle time pattern) In the given window, one or several consecutive $\bar{a}$ appear necessarily in the form $\ldots a\bar{a}\bar{a}\ldots\bar{a}(ba)\ldots$. With the earliest placement principle for all a's, the only (non-intrinsic) idle time in this sequence is located in front of the first $\bar{a}$-term. This idle time may be associated with the subsequent ba-term. Again, one obtains in this way idle times $\epsilon_1, \ldots, \epsilon_\beta$ and ϵ.

Using Rule 1, the idle times for the sequence $Z, Z, \ldots$ take the following form

$$
\begin{array}{lcl}
R_1 & = & [\ \epsilon_1\ \epsilon_2\ \epsilon_3\ \ldots\ \epsilon_\beta\ \ \epsilon\ \ \epsilon_1\ \ldots\ \epsilon_{\beta-2}\ \epsilon_{\beta-1}\ \epsilon_\beta\ ] \\
R_2 & = & [\ \epsilon\ \ \epsilon_1\ \epsilon_2\ \ldots\ \epsilon_{\beta-1}\ \epsilon_\beta\ \ \epsilon\ \ \ldots\ \epsilon_{\beta-3}\ \epsilon_{\beta-2}\ \epsilon_{\beta-1}\ ] \\
R_3 & = & [\ \epsilon_\beta\ \epsilon\ \ \epsilon_1\ \ldots\ \epsilon_{\beta-2}\ \epsilon_{\beta-1}\ \epsilon_\beta\ \ \ldots\ \epsilon_{\beta-4}\ \epsilon_{\beta-3}\ \epsilon_{\beta-2}\ ] \\
\vdots & & \\
R_{\beta+1} & = & [\ \epsilon_2\ \epsilon_3\ \epsilon_4\ \ldots\ \epsilon\ \ \epsilon_1\ \ \epsilon_2\ \ldots\ \epsilon_{\beta-1}\ \epsilon_\beta\ \ \epsilon\ \ ] \\
R_{\beta+2} & = & R_1
\end{array}
$$

We define the *cycle time* $\lambda(\mathcal{C})$ of a cycle as the ratio of the *cycle length* divided by the number of coupled tasks in $\mathcal{C}$. We obtain for $\mathcal{C}(\alpha,\beta)$ after $(\beta+1)$ repetitions of Z the cycle length

$$(\beta+1)(2L+a+b)-(\epsilon+\epsilon_\beta+\ldots+\epsilon_1)=(\beta+1)(2L+a+b)-\gamma$$

on which we have placed $(\beta+1)(1+\alpha+2\beta)$ coupled tasks.

Proposition 1 *The cycle time $\lambda(\mathcal{C}(\alpha,\beta))$ is given by the formula*

$$\lambda(\mathcal{C}(\alpha,\beta))=\frac{(\beta+1)(2L+a+b)-\gamma}{(\beta+1)(1+\alpha+2\beta)} \tag{2}$$

independent of the choice of ϵ_i.

We call two cycles *equivalent* if they have the same cycle times. According to Proposition 1, the cycles $\mathcal{C}(\alpha,\beta)$ are equivalent for all feasible ϵ_i (*i.e.* $\epsilon_i \geq 0$ for all i and $\sum \epsilon_i \leq \gamma$).

A special choice of ϵ_i is to set $\epsilon_i = \epsilon = \frac{\gamma}{\beta+1}$ for all i. Then we get $R_2 = R_1$ and we obtain the cycle $\mathcal{C}(\alpha,\beta)$ in compact form, based on a single letter-cycle Z, which requires however a rational placement of the coupled tasks. Otherwise, we can use the equivalent cycle $\mathcal{C}(\alpha,\beta)$ above with $(\beta+1)$ letter-cycles. This case is particularly interesting. It allows the construction of different cycles, for instance $\epsilon=\gamma, \epsilon_1=\cdots=\epsilon_\beta=0$. This gives an integer placement of all coupled tasks, but at the expense of very long cycles.

As shown in [3], using a linear programming formulation with a totally unimodular coefficient matrix, one can always find the best solution for a given sequence of coupled tasks with integer placements. For our example ($a=5, b=3, L=43$), the profile $(0,5)$ and $\epsilon=\gamma=3$, $\epsilon_i=0$ $(i=1,2\ldots5)$, we get integer placements for all tasks, but the cycle length increases already to 561. These rapidly increasing lengths are certainly one of the reasons why their detection is so difficult if the graph approach is used [2, 14, 8, 3].

The particular cycle $\mathcal{C}(0,\beta)$ possesses additional symmetries. Its letter-pattern is simply a succession of 'ab'. Starting again with window W_0, we follow here the idle times in the next (non overlapping) adjacent window W_1'. It is easy to see that the idle times $[\epsilon_1,\epsilon_2,\ldots,\epsilon_\beta,\epsilon]$ of W_0 repeat identically in W_1'. We get, therefore, additional equivalent cycles of the length of a simple window, $a+b+L$, where we have placed $(\beta+1)$ coupled tasks in the cyclic sense. Hence we can express the cycle time, equivalently to (2), in the form

$$\lambda(\mathcal{C}(0,\beta))=\frac{a+b+L}{\beta+1}$$

Taking finally $\epsilon_i = \frac{\gamma}{\beta+1}$ for all i, one gets the most compact form of the cycle $\mathcal{C}(0,\beta)$, which is simply the pattern $(a, \frac{\gamma}{\beta+1}, b)$ as in Figure 2.

2.2 Finding an optimal solution in $\mathcal{L}$

We say that cycle $\mathcal{C}_1$ *dominates* $\mathcal{C}_2$ if the cycle times verify $\lambda(\mathcal{C}_1) \leq \lambda(\mathcal{C}_2)$. The dominance is *strict* if $\lambda(\mathcal{C}_1) < \lambda(\mathcal{C}_2)$.

Lemma 1 *A cycle $\mathcal{C}(\alpha,\beta,\gamma)$ is strictly dominated by the following cycles:*

1. *$\mathcal{C}(\alpha+1,\beta,\gamma-a)$ if $\gamma \geq a$*
2. *$\mathcal{C}(\alpha-1,\beta+1,\gamma-b)$ if $\alpha \geq 1$ and $a > \gamma \geq b$*

Proof. In order to show *[1]*, compare the cycle times of cycles $\mathcal{C}(\alpha,\beta)$ and $\mathcal{C}(\alpha+1,\beta)$. Likewise, *[2]* is obtained by comparing cycles $\mathcal{C}(\alpha,\beta)$ and $\mathcal{C}(\alpha-1,\beta+1)$. □

Lemma 2 *A cycle $\mathcal{C}(\alpha,\beta,\gamma)$ is dominated by*

3. *$\mathcal{C}(\alpha+2,\beta-1,\gamma-(a-b))$ if $\beta \geq 1$ and $\gamma \geq (\beta+1)(a-b)$*
4. *$\mathcal{C}(\alpha-2,\beta+1,\gamma+(a-b))$ if $\alpha \geq 2$ and $\gamma \leq (\beta+1)(a-b)$*

This dominance is strict if $\gamma > (\beta+1)(a-b)$ in [3] and $\gamma < (\beta+1)(a-b)$ in [4].

Proof. Setting $\alpha' = \alpha+2$ and $\beta' = \beta-1$ and $\gamma' = \gamma-(a-b)$ in *[3]* and $\alpha' = \alpha-2$, $\beta' = \beta+1$ and $\gamma' = \gamma+(a-b)$ in *[4]*, we have for both cases $\alpha' + 2\beta' = \alpha + 2\beta$. A glance at the cycle time (equation (2)) shows that $\mathcal{C}(\alpha,\beta,\gamma)$ is dominated by $\mathcal{C}(\alpha',\beta',\gamma')$ whenever

$$\frac{\gamma'}{\beta'+1} \geq \frac{\gamma}{\beta+1}$$

This inequality is easily verified. For instance, for *[3]* we have

$$\begin{aligned} \gamma'(\beta+1) - \gamma(\beta'+1) &= (\gamma-(a-b))(\beta+1) - \gamma\beta \\ &= \gamma - (\beta+1)(a-b) \\ &\geq 0 \end{aligned}$$

□

The domination rules imply that all cycles $\mathcal{C}(\alpha,\beta)$ with $\alpha \geq 2$ and $\beta \geq 1$ are dominated. Therefore, one can find an optimal cycle among the ones of the form $\mathcal{C}(\psi,\beta)$, $\psi \in \{0,1\}$ and $\mathcal{C}(\alpha,0)$.

Let us denote by $\lceil u \rceil$ and $\lfloor u \rfloor$ the integer part of u rounded up and down, respectively. We get immediately the following characterization.

Proposition 2 *An optimal cycle* $\mathcal{C}^N = \mathcal{C}(\alpha^N, \beta^N, \gamma^N)$ *in* $\mathcal{L}$ *is one of the following*

(a) $\alpha^N = \psi \in \{0,1\}$, $\beta^N = \left\lfloor \frac{L}{a+b} \right\rfloor$, $\gamma^N = L - \psi a - (a+b)\left\lfloor \frac{L}{a+b} \right\rfloor$ *with* $\psi = 1$ *if* $L-(a+b)\left\lfloor \frac{L}{a+b} \right\rfloor \geq a$ *and* $\psi = 0$ *otherwise, provided that* $\gamma^N \leq (\beta^N+1)(a-b)$*; else*

(b) $\alpha^N = \left\lfloor \frac{L}{a} \right\rfloor$, $\beta^N = 0$, $\gamma^N = L - a\left\lfloor \frac{L}{a} \right\rfloor$ *with* $b > \gamma^N \geq a - b$.

We obtain directly the following result

Theorem 1 *Cycle* $\mathcal{C}^N$ *is optimal in the class* $\mathcal{L}$ *and can be determined in polynomial time.*

Notice that the profile of $\mathcal{C}^N$ is obtained by simple divisions, the cycle is presented in compact form, and the cycle time is obtained by a simple formula. Hence they are calculated in polynomial time in the input size.

So far, we have considered the particular letter-cycle Z, associated with the profile (α, β). Let us call in general a cycle that has the same profile (α, β) in every window an (α, β)-*profile extension*. Our cycle $\mathcal{C}(\alpha, \beta)$ is such a profile extension. Notice that, starting with Z, the following windows of $\mathcal{C}(\alpha, \beta)$ follow different orders of the a-terms and ba-terms. Of course, one may start the cycle with any of these windows. The resulting cycles are all equivalent to $\mathcal{C}(\alpha, \beta)$.

A general realization of a window with profile (α, β) with $\alpha > 0$ contains in some order the terms $\{a^{\alpha_1}, \overline{a}^{\alpha_2}, (ba)^{\beta}\}$ where $\alpha_1 + \alpha_2 = \alpha$ and $\overline{a}$ is a 'b' followed by the intrinsic idle time $(a-b)$. One may define the corresponding letter cycle Z' as before and the previous theory will be the same, since applying Rule 1 for the idle times does not depend on the order of the different terms. Consequently, all these cycles are equivalent to $\mathcal{C}(\alpha, \beta)$.

Lemma 3 *The cycle* $\mathcal{C}(\alpha, \beta)$ *is an optimal* (α, β)*-profile extension.*

Proof. As explained, we can, without restriction, refer to the letter-cycle Z. It remains to be shown that further idle times can not improve the cycle. There are two possible additional idle time insertions.

1. Idle time between two consecutive a's: Since the corresponding b-terms are already separated by $(a-b)$, the left-shift assumption implies that there must be at least one additional 'a' between the two b-terms. This, however, is impossible, since the profile would have changed.

2. Idle time inside a ba-term: Let $\mathcal{C}$ be any (α, β)-profile extension, based on the letter-cycle Z. We compare $\mathcal{C}$ with $\mathcal{C}(\alpha, \beta)$, taking this time the cycle in the form with integer placements (*i.e.* $(\beta+1)$ repetitions of the letter cycle), setting all $\epsilon_i = 0$ and $\epsilon = \gamma$. The initial letter-cycle of $\mathcal{C}(\alpha, \beta)$ is left-shifted so that the first idle time of size γ occurs just before the end of the initial coupled task. Then all a's are placed as early as possible. Consequently, corresponding a's of cycle $\mathcal{C}$ cannot occur earlier than the a's in $\mathcal{C}(\alpha, \beta)$. This implies the lemma.

□

3 Optimal solution of the identical coupled task scheduling problem

From the dominance rules of Lemmas 1 and 2 and their proofs, we can see that the profile (α^N, β^N) is in fact an optimal solution of the integer program

$$\begin{array}{lll} \max & \alpha + 2\beta & \\ \text{s.t.} & L = \alpha a + \beta(a+b) + \gamma & \\ & \alpha, \beta, \gamma \geq 0 \qquad \text{integer} & \end{array} \tag{3}$$

This is true since every optimal solution $\mathcal{C}(\alpha, \beta)$ of (3) with $\beta \geq 1$ and $\alpha \geq 2$ can be reduced to $\mathcal{C}^N$ without ever decreasing the number of elements $\alpha + 2\beta$ (*i.e.* by using uniquely the dominance rules *[3]* and *[4]*).

Consequently, each window W of $\mathcal{C}^N$ with profile (α^N, β^N) contains the maximum possible number of elements $M = \alpha^N + 2\beta^N$. We call such windows *tightly-packed* and say that a cycle is *tightly-packed* if all its windows are tightly-packed. Notice that there is always such a cycle for given a, b and L since $\mathcal{C}^N$ is tightly-packed.

We also call a profile *tight* if it is an optimal solution of (3). Our cycles $\mathcal{C}(\alpha, \beta)$, whose profile (α, β) is tight, are tightly-packed cycles, and the cycle $\mathcal{C}^N$ is the best tightly-packed cycle in this set.

One may verify that a profile (α, β) is tight if, and only if, it satisfies

$$\alpha = M - 2\beta; \qquad \max\left\{0, \left\lceil \frac{Ma - L}{a - b} \right\rceil\right\} \leq \beta \leq \left\lfloor \frac{M}{2} \right\rfloor \tag{4}$$

In particular, the profile $(\psi, \lfloor M/2 \rfloor)$ is tight, where $\psi = 0$ if M is even and $\psi = 1$ if M is odd. The profile of the form $(\alpha, 0)$ may be tight or not.

Theorem 2 *The cycle $\mathcal{C}^N$ solves the cyclic identical coupled task problem, and this problem is therefore polynomially solvable.*

The proof is based on several lemmas.

Lemma 4 *All neighboring tightly-packed windows of a feasible cycle have necessarily the same profile.*

Proof. Consider two neighboring tightly-packed coupled tasks. Then we have two consecutive a-terms that are only separated by a certain number x of b's. Observe that the cases $x > M + 1$ and $x = M$ cannot occur.

Let $x = M + 1$. The two coupled tasks are non-overlapping and the second must contain exactly $M = x - 1$ a's on its L-section. Both windows have the same profile.

If $x = 0$, then we know (from the proof of Lemma 3) that the two a-terms are following each other without any idle time. Otherwise for left-shifted tasks, the second window would contain more than M elements, which is impossible. Again, both windows have the same profile.

Let $1 \leq x < M$. Then we have two overlapping coupled tasks. The two windows are tightly-packed. Hence we must also have exactly x elements between the ending b-terms of these windows. By construction, these x elements can only be a's. Then both windows have necessarily the same profile: exactly one ba-term from the first window disappears in the second window, but exactly one new ba-term is created. □

We get immediately the following result.

Corollary 1 *All tightly packed cycles are necessarily profile extensions.*

With Corollary 1, Theorem 2 seems to be quite natural since $\mathcal{C}^N$ is the best tightly-packed cycle. But to complete the proof one has to exclude all "irregular" cycles where only pieces of tightly-packed cycles are patched together. The remainder of the article is devoted to this issue.

Consider an arbitrary feasible cycle $\mathcal{C}$. We define a *block* B of $\mathcal{C}$ as a sequence of exactly $2(M+1)$ consecutive elements. The idle time following the last element of a block is to be included. We call a block *tight* if its sequence of elements follows the pattern of some (α_i, β_i)-profile-extension for a tight profile (α_i, β_i). Such a block contains exactly $(1+\alpha_i+2\beta_i) = M+1$ terms 'a' and $(1+\alpha_i+2\beta_i) = M+1$ terms 'b'. The *length* $|B|$ of a tight block B satisfies $|B| \leq (2L+a+b)$. The actual length, based on the tight profile (α_i, β_i) may vary from $2L+a+b-\gamma_i$ to $2L+a+b$. For the tightly-packed cycle $\mathcal{C}(\alpha_i, \beta_i)$ in its compact form each block is a complete cycle of length $2L+a+b-\gamma_i/(\beta_i+1)$.

Conversely, suppose B is a block with length $|B| \leq 2L+a+b$. $2(M+1)$ is the maximal number of elements that can be placed on $(2L+a+b)$ units and this is only possible if the windows of B are tightly-packed. But then these windows have the same tight profile (Lemma 4) and B follows the pattern of some tight profile-extension. Note that two consecutive tight blocks must have the same tight profile, using again Lemma 4. All blocks B having length $|B| > (2L+a+b)$ are called *non-tight*. For the following, we reserve the index $i = 0$ for the tight profile of the cycle $\mathcal{C}^N$. We take $\mathcal{C}^N$ in its compact form with the cycle and block length $2L+a+b-\gamma_0/(\beta_0+1)$. This ensures that $\mathcal{C}^N$ is a feasible cycle on any integer number of blocks. We also set $\mathcal{C}_i = \mathcal{C}(\alpha_i, \beta_i)$.

We call a cycle $\mathcal{C}$ *compact* with respect to some chosen starting window, if one cannot take out a window and without any change and readjustment, insert it earlier in $\mathcal{C}$. A compact version of a cycle $\mathcal{C}$ is dominating $\mathcal{C}$. Therefore, it is sufficient to consider only compact cycles.

We want to consider the (compact) cycle $\mathcal{C}$ as a sequence of blocks. We take in general the blocks, defined above, but with one exception. This is the case where $\mathcal{C}^N$ possesses the tight profile $(0, \beta_0)$ and this profile also occurs in $\mathcal{C}$. Note that in this case, we only have $\gamma_0 < a$, whereas one has $\gamma_i < b$ for all other cycles $\mathcal{C}_i$. The cycle $\mathcal{C}_0 = \mathcal{C}(0, \beta_0)$ may appear in $\mathcal{C}$ with any idle time distribution ϵ_i and $\mathcal{C}^N$ is the compact form of $\mathcal{C}_0$, based on the cycle pattern $(a, \frac{\gamma_0}{(\beta_0+1)}, b)$. These cycles have additional properties. We know that they possess a further cycle of the length of a single window, $a+L+b$, where they place exactly $(M+2)$ elements. In this particular case, we use reduced blocks with only $(M+2)$ elements, and hence both cycles, $\mathcal{C}^N$ and $\mathcal{C}_0$, have the identical length $(a+L+b)$ for such reduced tight blocks. We may take, if necessary, a multiple of $\mathcal{C}$ to get an integer number of complete standard and reduced blocks.

Proof of Theorem 2. Let a cycle $\mathcal{C}$ be given and let us consider the block decompositions of $\mathcal{C}$ as defined above. A block decomposition is not unique since one can change the cut-off points, which might modify its structure.

1. If there is a decomposition where all blocks are non-tight, then $\mathcal{C}^N$ as well as all other tightly-packed cycles $\mathcal{C}_i$ are strictly dominating $\mathcal{C}$. This class contains for instance all cycles where none of the windows is tightly-packed.

2. If there is a block decomposition where all blocks are tight, then $\mathcal{C}$ is a tightly-packed cycle and $\mathcal{C}^N$ is dominating $\mathcal{C}$ (Lemma 3, Corollary 1, Theorem 1).

3. Now suppose $\mathcal{C}$ is partially tightly-packed, which means that for all possible block decompositions there are tight and non-tight blocks. Then decompose $\mathcal{C}$ into several *B-D-block sequences* of the form $B_1, \ldots, B_u$; $D_1, \ldots, D_v$, $u \geq 1$, and $v \geq 1$ where all B-blocks are tight and all D-blocks are non-tight. The D-blocks are there to carry out the transition from one tight profile to another tight profile. According to Lemma 4, a direct transition is impossible. We also know that all windows of the B-blocks have the same tight profile. For this case, there might be the chance, although rather remote, to have very few B-blocks, much less than in a full cycle, and gain γ units, do a very efficient transition and beat this way the tightly-packed cycles. The technical difficulty comes from the fact that the lengths of tight blocks only possess an upper bound $(2L + a + b)$.

Theorem 2 is established if the cycle $\mathcal{C}^N$ performs better than $\mathcal{C}$ on each of the B-D-block sequences. Although $\mathcal{C}^N$ is the best tightly-packed cycle, these cycles have different cycle lengths, which will influence their performance on the B-blocks, whose number may vary. The standard blocks have the reference length $(2L + a + b)$. Let us define the *deviation* $\Delta(\mathcal{C}, v)$ from the total reference length $v(2L+a+b)$ of a cycle $\mathcal{C}$ on v blocks. By definition, $\Delta(\mathcal{C}, v) \leq 0$ on v consecutive tight (standard) blocks and $\Delta(\mathcal{C}, v) > 0$ on v non-tight blocks. In particular $\Delta(\mathcal{C}^N, v) = -v\frac{\gamma_0}{(\beta_0+1)}$ and similarly $\Delta(\mathcal{C}^N, v) = \Delta(\mathcal{C}_k, v) = 0$ on the reduced blocks for $\alpha_0 = \alpha_k = 0$, where the reference length is $(a + L + b)$.

Now consider any B-D-block sequence, where the B-blocks have the tight profile (α_k, β_k) for some k. Let us use the notation $\Delta(\mathcal{C}, Tot)$ for the deviation of the total number of blocks in the entire B-D-block sequence.

Theorem 2 is established if we can show that

$$\Delta(\mathcal{C}^N, Tot) \leq \Delta(\mathcal{C}, Tot) \tag{5}$$

for-each B-D-block sequence. To get this result, we shall compare with the deviation $\Delta(\mathcal{C}_k, Tot)$ which can only perform better than the actual (α_k, β_k)-profile extension in $\mathcal{C}$ on the B-blocks and then on the entire B-D-block *i.e.* $\Delta(\mathcal{C}_k, Tot) \leq \Delta(\mathcal{C}, Tot)$.

Case 1 (Reduced blocks) This refers to the case $\alpha_0 = \alpha_k = 0$. We know that $\mathcal{C}^N$ and $\mathcal{C}_k$ perform identically on the reduced B-blocks, *i.e.* $\Delta(\mathcal{C}^N, u) = \Delta(\mathcal{C}_k, u) = 0$ and $\mathcal{C}^N$ performs better than $\mathcal{C}$ on the D-blocks, which implies directly inequality (5).

Case 2. Standard blocks ($\alpha_0 \neq 0$ or $\alpha_k \neq 0$). It remains to prove (5) for this case. First it is shown that $\mathcal{C}^N$ is superior to $\mathcal{C}_k$ on a complete cycle of $\mathcal{C}_k$.

Lemma 5 *Suppose $\alpha_0 \neq 0$ or $\alpha_k \neq 0$ (Case 2). Then the inequality $\Delta(\mathcal{C}^N, \beta_k + 1) \leq \Delta(\mathcal{C}_k, \beta_k + 1)$ is satisfied.*

Proof. We show this property by treating separately the two different forms that $\mathcal{C}^N$ can take.

1. Consider the case where $(\alpha_0, 0) = (M, 0)$ is an optimal tight profile so that $\mathcal{C}^N = \mathcal{C}(M, 0)$. Using Lemma 2, its proof and Proposition 2, we get all tight profiles: $(\alpha_i, \beta_i) = (\alpha_0 - 2i, i)$ satisfying $\gamma_i = \gamma_0 + i(a-b)$; $\gamma_i \geq (\beta_i + 1)(a-b) = (i+1)(a-b)$ for all $i \geq 0$ satisfying $M - 2i \geq 0$. After a complete cycle of $\mathcal{C}(\alpha_k, \beta_k)$, we get:

$$\begin{aligned}
\Delta(\mathcal{C}^N, \beta_k + 1) - \Delta(\mathcal{C}_k, \beta_k + 1) &= -(\beta_k + 1)\frac{\gamma_0}{(\beta_0 + 1)} + \gamma_k \\
&= -(k+1)\gamma_0 + (\gamma_0 + k(a-b)) \\
&= -k(\gamma_0 - (a-b)) \\
&\leq 0.
\end{aligned}$$

2. Suppose now that (ψ, β_0) is an optimal tight profile, *i.e.* $\mathcal{C}^N = \mathcal{C}(\psi, \beta_0)$, where ψ is 0 or 1. Then all tight profiles are of the form $(\alpha_i, \beta_i) = (\psi + 2i, \beta_0 - i)$, $i \geq 0$. From Lemma 2 and Proposition 2, one gets the following information:

$$\begin{aligned}
&\gamma_i \leq (\beta_i + 1)(a-b); \\
&\gamma_i = \gamma_0 - i(a-b) \geq 0; \\
&\gamma_i < b, \text{ except } \gamma_0 < a \text{ if } \psi = 0.
\end{aligned}$$

We obtain

$$\begin{aligned}
\Delta(\mathcal{C}^N, \beta_k + 1) - \Delta(\mathcal{C}_k, \beta_k + 1) &= -\frac{(\beta_k + 1)\gamma_0}{\beta_0 + 1} + \gamma_k \\
&= -\frac{(\beta_0 + 1 - k)\gamma_0}{\beta_0 + 1} + \gamma_0 - k(a-b) \\
&= -\frac{k}{\beta_0 + 1}[(\beta_0 + 1)(a-b) - \gamma_0] \\
&\leq 0.
\end{aligned}$$

□

Using Lemma 5, it is sufficient for the given B-D-block sequence $B_1, \ldots B_u; D_1, \ldots D_v$ to assume that $u \leq \beta_k - 1$.

Lemma 6 *The following inequalities are valid for $u \leq \beta_k - 1$:*

$$\Delta(\mathcal{C}, u+1) \geq 0 \text{ if } \alpha_k \neq 0 \text{ and } \Delta(\mathcal{C}, u+1) \geq b - a \text{ for } \alpha_k = 0 \text{ and } \alpha_0 \neq 0. \qquad (6)$$

Let us take Lemma 6 for granted and finish the proof of Theorem 2. The case $\alpha_k = \alpha_0 = 0$ has been settled earlier (Case 1). We want to show that $\Delta(\mathcal{C}^N, u+1) \leq \Delta(\mathcal{C}, u+1)$ for

Case 2. This is obviously true if $\alpha_k \neq 0$. There $\Delta(\mathcal{C}^N, u+1)$ is negative. Now suppose that $\alpha_k = 0$ and $\mathcal{C}^N$ has the profile $(\alpha_0, 0)$. Then $b > \gamma_0 > a - b$ and $\Delta(\mathcal{C}^N, u+1) = -(u+1)\gamma_0 \leq -\gamma_0 < -(a-b)$, which implies again $\Delta(\mathcal{C}^N, u+1) \leq \Delta(\mathcal{C}, u+1)$. Continuing with the D-blocks, finally shows (5) $\Delta(\mathcal{C}^N, Tot) \leq \Delta(\mathcal{C}, Tot)$, thus terminating the proof of Theorem 2.

It remains to show the validity of Lemma 6.

Proof of Lemma 6. We shall compare the length of the first D-block D_1 with the tight block B_{u+1}, where we just continue with the profile (α_k, β_k). One can observe that the change one can make to get from B_{u+1} to D_1 is to drop at least one 'a', *i.e.* to take out at least one coupled task and start it later, thereby adjusting the rest of the sequence.

The proof will be done by listing several properties and subcases. Let us also remember that the underlying partially tightly-packed cycle $\mathcal{C}$ is compact.

a) For the transition from B_{u+1} to D_1 in a compact cycle, one never drops an 'a' that belongs to two consecutive a's in B_{u+1}.

Consider such a sequence $aa - - - b(\text{idle}{=}a-b)b$. Suppose for instance that one drops the first coupled task. If nothing is inserted, one could instead place one of the following coupled task into this position, contradicting the compactness of the cycle $\mathcal{C}$. The other possibility is to drop the first 'a' and start a new 'a' in the place of $b(\text{idle}{=}a-b)$. Also this coupled task could be advanced to the former position, which is again impossible for a compact cycle. We conclude that an 'a' may only be dropped if it is between two b's. This also implies that dropping an 'a' leaves an incompressible additional idle time of a units.

One may have $\mathcal{C}^N = \mathcal{C}(M, 0)$. But whenever $\mathcal{C}_k = \mathcal{C}(M, 0)$, no transition to a non-tight block is possible in a compact cycle $\mathcal{C}$, according to a), *i.e.* $\mathcal{C}_k = \mathcal{C}$ which is not possible; $\mathcal{C}$ was supposed to be only partially tightly-packed.

We want to consider the structure of cycles $\mathcal{C}_k$ in B_{u+1} and $\mathcal{C}$ in D_1. Let a^f be the first 'a' in B_{u+1} and b^f its b-term. a^f may be preceded in B_{u+1} by x b-terms. We can exclude the cases $x = M+1$ and $x = M$, which lead to $\mathcal{C}_k = \mathcal{C}(M, 0)$. Then we get the structure displayed in Figure 3 with $0 \leq x \leq M-1$, where Y is a sequence of $(M-1-x)$ a's and b's and $\overline{Y}$ is the dual sequence ('a' is replaced with 'b' and vice versa).

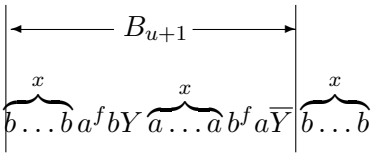

Figure 3: Structure of B_{u+1}

b) If the inequality

$$|D_1| \geq |B_{u+1}| + b \tag{7}$$

is verified, then (6) is valid.

We know that $\mathcal{C}_k$ is at least as good as $\mathcal{C}$ on the B-blocks (Lemma 3) and will gain exactly γ_k on one complete cycle. Remember that the cycle length is equal to $(\beta_k + 1)(2L+a+b)-\gamma_k$. Using (7), we get $\Delta(\mathcal{C}, u+1) \geq -\gamma_k+b$ and (6) follows immediately.

c) If two or more a's are dropped in B_{u+1}, then (6) is verified.

In this case, D_1 contains at least $2a$ idle time units, implying (7) since then $|D_1| \geq (2L + a + b) + a$. Consequently, we are left with the case where exactly one single 'a' is dropped in B_{u+1}.

d) Only the coupled task (a^f, b^f) is dropped.

Let us take out (a^f, b^f). Then $x \geq 1$ since $x = 0$ would extend B_u by a^f units and B_u would no longer be tight. Not more than one extra 'a' can be inserted in the place of b^f. The first 'b' following B_{u+1} remains in fixed position and has to be included in D_1. Thus (7) is satisfied.

e) A single 'a' different from a^f is dropped from B_{u+1}.

Now take out a coupled task $T = (a^g, b^g)$, different from (a^f, b^f), and start it later, thereby adjusting the rest of the sequence. There are several possible positions for T: the b-part of T may fall outside the block B_{u+1}, or it is located inside. In any case, one may at the best start an extra 'a' at the position of b^g. But as long as the first 'b' following B_{u+1} is different from b^g, one obtains (7) as in d).

However, if this 'b' belongs to T, then B_{u+1} terminates with 'a, id' (inside B_{u+1})' and 'b^g' (outside B_{u+1}), where id is the idle time in front of b^g. If b^g is followed by a 'b', the inequality (7) is implied as before.

Now let b^g be followed by an 'a'. In this case, (7) may not be fulfilled. This depends on the size of id. A glance at the idle time pattern of $\mathcal{C}_k$ shows that one can gain at most an additional idle time $(\gamma_k - id)$ before the end of a complete cycle $(u + 1 \leq \beta_k)$. In D_1 two a-terms may be shifted into (id, b^g). Then we have

$$\Delta(\mathcal{C}, u + 1) \geq -(\gamma_k - id) + (2a - (id + b)) = -\gamma_k - b + 2a \geq 0$$

for all id and (6) is valid.

This completes the proof for all possible cases. In fact, it has been shown that $\mathcal{C}^N$ is strictly dominating all partially tightly-packed cycles.

4 Conclusion

In Ahr et al. [2], Table 2, the minimum mean cycle times have been computed for small values ($a \leq 10$, $b \leq 5$, $L \leq 30$), using the graph method. The correctness of their results can now be verified.

They also present in their conclusion a conjecture for the cycle times in the very special case $b = 1$, which is as follows:

$\frac{a+1+2L}{2\frac{L+1}{a+1}}$ if $L \equiv -1 \mod (a + 1)$

$\frac{a+1+L}{\lfloor\frac{L}{a+1}\rfloor+1}$ otherwise

Also this formula is correct using (2). Notice that the case in Proposition 2a) applies. Setting $\beta = \lfloor \frac{L}{a+1} \rfloor$, the optimal profile is $(\alpha, \beta, \gamma) = (1, \beta, 0)$ whenever $L \equiv -1 \mod (a+1)$, otherwise the profile $(0, \beta, \gamma < a)$ is optimal.

Having established the polynomial complexity for the cyclic case, it remains the finite problem, where n identical coupled tasks have to be placed optimally. If n is very large, the optimal schedule will have to follow in the "middle section" the optimal cyclic configuration. However the starting and the finishing part of the schedule will in general depend on n.

References

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

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

[3] Ph. Baptiste. A note on scheduling identical coupled tasks in constant time. Technical report, private communication, 2009.

[4] J. Békési, G. Galambos, M. Oswalda, and G. Reinelt. Improved analysis of an algorithm for the coupled task problem with UET jobs. *Operations Research Letters*, 37(2):93 – 96, 2009.

[5] J. Blazewicz, K. Ecker, T. Kis, and M. Tanaș. A note on complexity of scheduling coupled tasks on a single processor. *Journal of the Brazilian Computer Society*, 7(3):23–26, 2001.

[6] N. Brauner, Y. Crama, A. Grigoriev, and J. van de Klundert. A framework for the complexity of high-multiplicity scheduling problems. *Journal of Combinatorial Optimization*, 9:313–323, 2005.

[7] N. Brauner, Y. Crama, A. Grigoriev, and J. van de Klundert. Multiplicity and complexity issues in contemporary production scheduling. *Statistica Neerlandica*, 61(1):75–91, 2007.

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

[9] C. Duron. *Ordonnancement en temps-réel des activités des radars*. PhD thesis, Université de Metz, 2002.

[10] M. Elshafei, H. D. Sherali, and J. C. Smith. Radar pulse interleaving for multi-target tracking. *Naval Research Logistics*, 51:72–94, 2003.

[11] A. Farina and P. Neri. Multitarget interleaved tracking for the phased radar array. *IEE Proceedings F*, 27:312–318, 1980.

[12] J. N. D. Gupta. Comparative evaluation of heuristic algorithms for the single machine scheduling problem with two operations per job and time-lags. *Journal of Global Optimization*, 9:239–250, 1996.

[13] M. Karp. A characterization of the minimum cycle mean in a digraph. *Discrete Mathematics*, 1978.

[14] V. Lebacque. *Théories et applications en ordonnancement : contraintes de ressources et tâches agrégées en catégories*. PhD thesis, Université Joseph Fourier, 2006.

[15] D. J. Milojevic and B. M. Popovic. Improved algorithm for the interleaving of radar pulses. *IEE Proceedings F*, 139:98–104, 1992.

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

[17] A. J. Orman, A. K. Shahani, and A. R. Moore. Modelling for the control of radar system. *Computers and OR*, 25:239–249, 1998.

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

[19] R. D. Shapiro. Scheduling coupled tasks. *Naval Research Logistics Quarterly*, 27(2):489–497, 1980.

[20] G. Simonin, B. Darties, R. Giroudeau, and J.-C. König. Isomorphic coupled-task scheduling problem with compatibility constraints on a single processor. In *MISTA*, 2009.