J Sched (2015) 18:631–644
DOI 10.1007/s10951-015-0438-9

Identical coupled task scheduling: polynomial complexity of the cyclic case

Vassilissa Lehoux-Lebacque[1] · Nadia Brauner[1] · Gerd Finke[1]

Published online: 24 June 2015

Abstract Coupled tasks are two-operation tasks, where the two operations are separated by a time interval of fixed duration. Coupled task scheduling problems refer then to the scheduling of a set of coupled tasks on a single machine. Applications of these problems, reported in the literature, arise in connection with radar systems, robotic cells, and in manufacturing. Most of the complexity issues for scheduling coupled tasks have been settled. However, the complexity status has been unknown for the identical coupled task problem, where multiple copies of a single coupled task are to be processed. The purpose of the article is to solve this open problem in the cyclic case, for which we prove the polynomial complexity.

Keywords High-multiplicity · Polynomial complexity · Coupled task · Cyclic scheduling

Mathematics Subject Classification 90B35

1 Introduction

In coupled task scheduling problems, one wants to process tasks on a single machine with each task j being 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 to maximize the throughput in the cyclic case (infinite number of tasks).

✉ Gerd Finke
Gerd.Finke@g-scop.grenoble-inp.fr

1 Univ. Grenoble Alpes, G-SCOP, CNRS, 38000 Grenoble, France

Coupled tasks were introduced by Shapiro (1980). Shapiro's model comes from a radar that is tracking an aircraft approaching a large airport.

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) is often lower bounded (see for instance Gupta 1996). We shall here consider only coupled tasks with fixed separation intervals as in Shapiro (1980). There is a vast literature on this subject, treating offline and online cases and proposing various algorithms (Duron 2002; Elshafei et al. 2003; Farina and Neri 1980; Milojevic and Popovic 1992; Orman et al. 1998; Shahani et al. 1996; Shapiro 1980).

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 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*.

The complexity of minimizing the makespan has been described by Orman and Potts (1997), see Table 1. Even the unit execution time (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 (Ageev and Baburin 2007; Békési et al. 2009). 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 Ahr et al. 2004).

Establishing the complexity status of the identical coupled task scheduling problem is therefore the remaining theoretical challenge. This very special case possesses nevertheless an interesting application in the field of robotic cells (Brauner et al. 2009).

Table 1 Complexities for the coupled task problem from Orman and Potts (1997)

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 polynomial	$a_j = a; L_j = L; b_j = b$
	$a_j = L_j = p; b_j$
	$a_j = b_j = p; L_j = L$

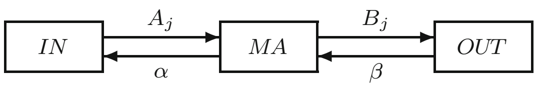

Fig. 1 1-Machine robotic cell

Brauner et al. (2009) show that this problem is equivalent to a no-wait one-machine robotic cell problem, with one robot, usually studied in cyclic mode. In Fig. 1, A_j is the transportation time of part j from the input station to the machine MA, α is the empty return time and similarly, from MA to the output station (B_j and β). For details on the equivalence, see Brauner et al. (2009). In particular, producing a large number of parts of a single type corresponds exactly to the cyclic identical coupled task scheduling problem.

In this paper, we concentrate on the identical coupled task problem. Note 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 (Brauner et al. 2005, 2007), 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, $\log_2 n + \log_2 a + \log_2 b + \log_2 L$). Ahr et al. (2004) propose an algorithm linear in n but exponential in L. In Baptiste (2010), it has been shown that for fixed a, b, and L, the optimal solution can be found in $O(\log n)$, which improves the $O(n)$ running time in Ahr et al. (2004). The constant is again highly exponential in L and does not yield any practical computation. This algorithm has been adjusted to the cyclic case in Ahr et al. (2004) showing that the cyclic problem corresponds to finding the minimum mean cycle in a certain weighted graph, where the mean cycle refers to the length of the cycle divided by the number of edges in the cycle (see also Brauner et al. 2009; Lebacque 2007). This problem is polynomial in the size of the underlying graph (Karp 1978) which, however in our case, has an exponential number of vertices. The computational experience in Brauner et al. (2009) 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 Pentium 4, 2.53 GHz (RAM: 1 Go) 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 due to memory space limitations.

We also want to mention that extensions of the identical coupled task problem turn out to be NP-hard. In Blazewicz et al. (2010) 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 (Simonin et al. 2011). 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 shall provide a new solution approach to the problem. First, we describe in Sect. 2 a class of solutions for the cyclic identical coupled task problem, for which we determine the best solution in polynomial time. Then, in Sect. 3, we show that this solution is in fact optimal for our general problem. The cyclic identical coupled task problem is therefore solvable in polynomial time.

2 A class of solutions

Cyclic scheduling is well known in the literature, see for instance the survey (Levner et al. 2010). Depending on the environment, one wants to find a feasible processing order of the tasks on the different machines, together with their starting times, that can be repeated identically (infinitely many times), *i.e.*, there is a constant time interval of length T, the period or cycle $\mathcal{C}$, and each operation of some task, assigned to a certain machine during one period at time t will be assigned exactly to the same machine at time $T + t$ in the next period.

Depending on the application, different criteria may be optimized. One may minimize the cycle time $T(\mathcal{C})$ or the number of tasks $N(\mathcal{C})$ of $\mathcal{C}$ (*Work-in-Process*). Another measure is to maximize the throughput $\frac{N(\mathcal{C})}{T(\mathcal{C})}$ or, equivalently, minimize the mean cycle time $\lambda(\mathcal{C}) = \frac{T(\mathcal{C})}{N(\mathcal{C})}$.

In our case, we use the last criterion $\lambda(\mathcal{C})$. We have a single machine and a single part type (the coupled task). Therefore, the cycle will be a feasible pattern of a's and b's. But since the tasks consist of two parts, a cycle does not necessarily contain both parts of the same coupled tasks. Let us give the detailed definition of the problem.

Definition 1 (*Cyclic identical coupled task scheduling problem*) Let three positive integers a, L, and b be given. One has multiple copies of a single coupled task $(a; L; b)$, consisting of two *operations* (or *elements*), the first having length a and the second length b, and both operations are exactly separated by L time units. A *cycle* $\mathcal{C}$ is a finite sequence of a's, b's, and idle times that can be appended repeatedly to each other, to form a feasible placement of the coupled tasks (*i.e.,* without overlapping of operations). Cycle $\mathcal{C}$ contains necessarily the same number of a's and b's. Let $N(\mathcal{C})$ be the number of coupled tasks in $\mathcal{C}$ in the cyclic sense (*i.e.,* the number of elements in $\mathcal{C}$ divided by 2) and let $T(\mathcal{C})$ be the length of $\mathcal{C}$.

Then the cyclic identical coupled task scheduling problem is to find a cycle $\mathcal{C}$ with minimum mean cycle time $\lambda(\mathcal{C}) = \frac{T(\mathcal{C})}{N(\mathcal{C})}$.

We call two cycles $\mathcal{C}_1$ and $\mathcal{C}_2$ *equivalent* if they have the same mean cycle times, $\lambda(\mathcal{C}_1) = \lambda(\mathcal{C}_2)$. Cycle $\mathcal{C}_1$ *dominates* cycle $\mathcal{C}_2$ if their mean cycle times verify the inequality $\lambda(\mathcal{C}_1) \leq \lambda(\mathcal{C}_2)$. The dominance is *strict* if $\lambda(\mathcal{C}_1) < \lambda(\mathcal{C}_2)$. A cycle $\mathcal{C}$ is optimal and minimizes $\lambda(\mathcal{C})$ if and only if it dominates all other cycles.

It is easy to construct feasible solutions for the cyclic identical coupled task problem with given integers a, L, *and* b. 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 $aaa \dots abbb \dots b$, 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 hence placed 9 coupled tasks on the length 91 (the mean cycle time is then $\lambda = \frac{91}{9}$).

The previous solution is working in both modes (cyclic and non-cyclic). In Fig. 2, we give another solution that is rather surprising and very simple. It will turn out later that this is in fact an optimal cycle. According to the definition, one is looking for a feasible pattern of a's, b's, and idle times, 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 cycle in Fig. 2 is somewhat unexpected since the placement is partially at fractional starting times for the coupled tasks. It is interesting to note that such fractional placements cannot be detected by the graph methods in Ahr et al. (2004), Baptiste (2010), Brauner et al. (2009), Lebacque (2007). Here, we have formed a cycle composed of 'a', an idle time of 0.5 units, and 'b' (we marked 'a' and 'b' of the same coupled task by an identical number). Therefore, we have placed two elements, *i.e.,* one coupled task in the cyclic sense, on 8.5 units (mean cycle time of 8.5). Therefore, this cycle is strictly dominating the previous one.

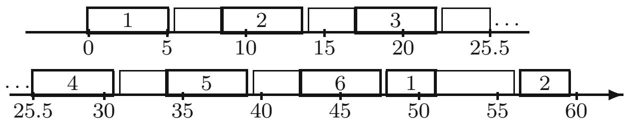

Fig. 2 A feasible cyclic solution for $a = 5$, $b = 3$, and $L = 43$

In this section, we construct a class of cycles, for which we determine the best cycle in polynomial time for given positive integers a, L, and b.

Now, consider a cycle $\mathcal{C}$. 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).

We will also assume that $L \geq a + b$ since otherwise the problem is trivial. Using subscripts to indicate the a-part and b-part of the same coupled task, the optimal solution for $L < a$ is simply the sequence

$$a_1(\text{idle} = L)b_1,\ a_2(\text{idle} = L)b_2,\ \dots$$

and for $a + b > L \geq a$, we get the following optimal sequence:

$$a_1a_2(\text{idle} = L - a)b_1(\text{idle} = a - b)b_2,$$
$$a_3a_4(\text{idle} = L - a)b_3(\text{idle} = a - b)b_4,\ \dots$$

We can make the following fundamental observation:

Remark 1 If two a's are placed consecutively without any idle time, then the corresponding b's are separated by an idle time of $a - b$ units.

Consider now a particular coupled task (a_S, b_S) in a cycle $\mathcal{C}$. We are interested in the sequence S of a's and b's that is placed on the L-section separating a_S and b_S. The *window* $W_S = [a_S, L, b_S]$ of the coupled task (a_S, b_S) in the cycle $\mathcal{C}$ is defined as the sequence $a_S S b_S$. The length of each window is equal to $a + L + b$. Remark 1 gives us an indication of the space utilization of S: whenever we have two consecutive b's in S, at least the space $b + (\text{idle} = a - b) + b = a + b$ is occupied on the L-section. To indicate this fact, we convert each sequence bb of S to $\bar{a}b$, where $\bar{a} = (b, \text{idle} = a - b)$ is of length a and where $(a - b)$ is called the *intrinsic idle time*.

Definition 2 Let $\mathcal{C}$ be a cycle and $W_S = [a_S, L, b_S] = a_S S b_S$ a window of $\mathcal{C}$. Obtain $\bar{S}$ by converting in S all subsequences bb to $\bar{a}b$. Should S terminate with b, then also this b becomes $\bar{a}$ since it is followed by b_S. Count the number β of terms (ba) occurring in $\bar{S}$, and then count the number α of remaining a's and $\bar{a}$'s. We call (α, β) the *profile* of window W_S.

In this way, we get with each profile (α, β) of a window in $\mathcal{C}$, a solution of the constraint set:

$$\alpha a + \beta(b + a) \leq L \text{ with integers } \alpha, \beta \geq 0. \tag{1}$$

As usual we may convert the inequality to an equation by introducing the integer slack variable:

$$\gamma = L - \alpha a - \beta(b + a) \geq 0. \tag{2}$$

We also refer to (α, β, γ) as the *profile in extended form*, whenever it is useful (for instance, if conditions on γ are to be specified).

Example 1 As illustration, consider the window $W_S = [a_S, L, b_S] = a_S S b_S$, where $S = aabbbabaaabbbbab$. Then

$$\bar{S} = aa\bar{a}\bar{a}babaaa\bar{a}\bar{a}\bar{a}ba\bar{a} = aa\bar{a}\bar{a}(ba)(ba)aa\bar{a}\bar{a}\bar{a}(ba)\bar{a}.$$

Hence, the profile is $(\alpha = 10, \beta = 3)$.

In a general cycle, different windows will usually have different profiles. It is somewhat remarkable that, for any feasible solution (α, β) of (1), one can construct cycles so that every 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 cycles $\mathcal{C}(\alpha, \beta)$

For a given profile (α, β), consider the first coupled task (a_1, b_1) and normalize the order of elements in the initial window $W_1 = [a_1, L, b_1]$ as follows:

$$W_1 = a_1\, a^\alpha (ba)^\beta\, b_1 \tag{3}$$

or in short

$$W_1 = a^{\alpha+1}(ba)^\beta\, b.$$

Then, we extend this sequence to the following letter-pattern Z:

$$Z = a^{\alpha+1}(ba)^\beta b^{\alpha+1}(ab)^\beta. \tag{4}$$

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 the letter-pattern Z contains exactly $(1 + \alpha + 2\beta)$ coupled tasks, in the cyclic sense.

Having marked the position of a single coupled task, here (a_1, b_1), all other coupled tasks can be identified, since corresponding a's and b's must follow in the same order. Note also the duality in the structure of Z. Replacing a's with b's and conversely, the first part of Z, $a^{\alpha+1}(ba)^\beta$, transforms to the second part $b^{\alpha+1}(ab)^\beta$. Let us now analyze the letter sequence $Z, Z, \ldots$ and present an example, where we indicate the coupled tasks with subscripts.

Example 2 Let us use the profile $(\alpha = 2, \beta = 2)$. Consider the corresponding sequence Z, Z:

$$a_1a_2a_3b_{-2}a_4b_{-1}a_5b_1b_2b_3a_6b_4a_7b_5,$$
$$a_8a_9a_{10}b_6a_{11}b_7a_{12}b_8b_9b_{10}a_{13}b_{11}a_{14}b_{12}.$$

We shall select some typical windows in this sequence:

$W_1 = a_1a_2a_3b_{-2}a_4b_{-1}a_5b_1$ or $a_1aa(ba)(ba)b_1$; profile (2, 2)
$W_3 = a_3b_{-2}a_4b_{-1}a_5b_1b_2b_3$ or $a_3(ba)(ba)\bar{a}\bar{a}b_3$; profile (2, 2)
$W_5 = a_5b_1b_2b_3a_6b_4a_7b_5$ or $a_5\bar{a}\bar{a}(ba)(ba)b_5$; profile (2, 2)
$W_7 = a_7b_5a_8a_9a_{10}b_6a_{11}b_7$ or $a_7(ba)aa(ba)b_7$; profile (2, 2)

As suggested by Example 2, we get the following general result:

Proposition 1 *Let the sequence $Z, Z, Z, \ldots$ be associated with the profile (α, β). Then this profile is in fact an invariant for each window in the sequence.*

Proof Consider the sequence Z, Z:

$$a^{\alpha+1}(ba)^\beta b^{\alpha+1}(ab)^\beta,$$
$$a^{\alpha+1}(ba)^\beta b^{\alpha+1}(ab)^\beta.$$

There are three different types of location for a coupled task (a_t, b_t) in this sequence: two are within Z, and one is in the transition. The statement of Proposition 1 is obviously true if $\alpha = 0$ or $\beta = 0$. Now let $\alpha > 0$ and $\beta > 0$ and set $\alpha = u + v$ and $\beta = 1 + x + y$.

1. Z= $a^u a_t a^v (ba)^\beta b^u b_t b^v (ab)^\beta$: then the window

   $$[a_t, L, b_t] = a_t a^v (ba)^\beta \bar{a}^u b_t$$

   has, according to Definition 2, the profile $(\alpha = u+v, \beta)$.
2. $a^{\alpha+1}(ba)^x(ba_t)(ba)^y b^{\alpha+1}(ab)^x(ab_t)(ab)^y$. We get

   $$[a_t, L, b_t] = a_t(ba)^y\bar{a}^\alpha(ba)^{x+1}b_t$$

   with profile $(\alpha, \beta = 1 + x + y)$.
3. Consider the sequence

   $$a^{\alpha+1}(ba)^\beta b^{\alpha+1}(ab)^x(a_tb)(ab)^y,$$
   $$a^{\alpha+1}(ba)^x(b_ta)(ba)^y b^{\alpha+1}(ab)^\beta$$

   Here, we have $[a_t, L, b_t] = (a_tb)(ab)^y a^{\alpha+1}(ba)^x b_t = a_t(ba)^{y+1}a^\alpha(ba)^x b_t$. Again the profile is $(\alpha, \beta = 1 + x + y)$. □

For a given profile (α, β), consider the letter-pattern Z. Let $W_1 = [a_1, L, b_1]$ be its first window and $W_2 = [a_2, L, b_2]$ its last window. Note that a_2 is the last 'a' preceding 'b_1.' We know that each window has length $a + L + b$. The letter-pattern Z has the form

$$a_1 \dots a_2(\text{idle} = \epsilon)b_1 \dots b_2. \tag{5}$$

Here, the intersection $W_1 \cap W_2 = a_2(\epsilon)b_1$ so that the length of Z is given by

$$2L + a + b - \epsilon. \tag{6}$$

This is independent of the position of the other elements in the two windows. On Z, we have exactly $(1+\alpha+2\beta)$ a's and $(1+\alpha+2\beta)$ b's, which means $(1+\alpha+2\beta)$ coupled tasks in the cyclic sense. But in order to obtain from the letter sequence $Z, Z, \dots$ a feasible cyclic schedule, we have to insert all idle times. A complete cycle must then repeat identically, including also all idle times. The cycle to be constructed will be denoted by $\mathcal{C}(\alpha, \beta, \gamma)$ or in short $\mathcal{C}(\alpha, \beta)$, remembering that γ can be calculated from α and β, using (2).

We know that all idle times in a window, excluding the intrinsic idle times, add up to $\gamma = L - \alpha a - \beta(b + a)$. Starting from an initial window $W_1 = a_1 a^\alpha (ba)^\beta b_1$, we adopt the left-shifted placement strategy: we place a new 'a', following a 'b' or another 'a', without inserting any idle time. It remains to define for W_1 the non-intrinsic idle times $\epsilon_1, \dots \epsilon_\beta$, satisfying $\epsilon_i \geq 0$ for all i and $\sum \epsilon_i \leq \gamma$, in front of each of the β terms (ba) and additionally the idle time $\epsilon = \gamma - \sum \epsilon_i$ in front of b_1. Then as required we have

$$\sum \epsilon_i + \epsilon = \gamma \tag{7}$$

Example 3 (*Example 2 continued*) Let us assume the extended profile $(\alpha = 2, \beta = 2, \gamma)$ with some not specified value $\gamma = L - \alpha a - \beta(b + a)$. In the first window, we start with idle times ϵ_1, ϵ_2 and ϵ, satisfying (7). We want to follow the idle time pattern in the sequence Z, Z, where we omit all intrinsic idle times. When advancing in this sequence, one obtains the next idle time to be inserted by looking at the neighboring windows. We get for the first letter-pattern Z:

$$a_1 a_2 a_3 (\boldsymbol{\epsilon_1}) b_{-2} a_4 (\boldsymbol{\epsilon_2}) b_{-1} a_5 (\epsilon) b_1 b_2 b_3 a_6 (\boldsymbol{\epsilon_1}) b_4 a_7 (\boldsymbol{\epsilon_2}) b_5$$

For instance, having the idle times of $W_1 = a_1 a_2 a_3 (\boldsymbol{\epsilon_1}) b_{-2} a_4 (\boldsymbol{\epsilon_2}) b_{-1} a_5 (\epsilon) b_1$, the window $W_4 = a_4 (\boldsymbol{\epsilon_2}) b_{-1} a_5 (\epsilon) b_1 b_2 b_3 a_6 (\boldsymbol{\epsilon_1}) b_4$ requires only one additional (non-intrinsic) idle time in front of b_4, which is necessary equal to ϵ_1 since (7) has to be fulfilled. The idle times are as follows for the next copy of Z:

$$a_8 a_9 a_{10} (\epsilon) b_6 a_{11} (\boldsymbol{\epsilon_1}) b_7 a_{12} (\boldsymbol{\epsilon_2}) b_8 b_9 b_{10} a_{13} (\epsilon) b_{11} a_{14} (\boldsymbol{\epsilon_1}) b_{12}$$

The idle time vector of Z has changed from $R_1 = [\epsilon_1\, \epsilon_2\, \epsilon\, \epsilon_1\, \epsilon_2]$ to $R_2 = [\epsilon\, \epsilon_1\, \epsilon_2\, \epsilon\, \epsilon_1]$. From this transition, we can deduce the idle times for the subsequent Z: $R_3 = [\epsilon_2\, \epsilon\, \epsilon_1\, \epsilon_2\, \epsilon]$, then $R_4 = R_1 = [\epsilon_1\, \epsilon_2\, \epsilon\, \epsilon_1\, \epsilon_2]$. We have reached our goal. The idle time pattern repeats identically after three copies of Z and we have obtained a feasible cycle with the sequence of a's and b's, given by ZZZ, and corresponding (non-intrinsic) idle time pattern $R_1 R_2 R_3$. In order to fix completely the cycle, we have to choose feasible values for ϵ_1, ϵ_2 and ϵ according to (7). One possible choice is for instance $\epsilon_1 = 0$, $\epsilon_2 = 0$ and $\epsilon = \gamma$.

In general, the vector of idle times in the first window W_1 for profile (α, β, γ) is of the form

$$[\epsilon_1, \epsilon_2, \dots, \epsilon_\beta, \epsilon = \gamma - \sum \epsilon_i] \tag{8}$$

for non-negative values ϵ_i and ϵ.

Remember that after W_1, all coupled tasks are systematically placed as early as possible. Therefore, all placements and idle times in the sequence $Z, Z, \dots$ are uniquely determined.

Now consider an arbitrary window W in the sequence Z, $Z, \dots$ associated with the profile (α, β). The positions of all non-intrinsic idle times of W can be described as follows (compare with Example 3):

Rule 1 (*idle time pattern*) In a given window $W = [a_S, L, b_S]$, the terms (ba) and b_S may be preceded by a certain number of $\bar{a}$'s. There are $\beta + 1$ non-intrinsic idle times $\varphi_1, \varphi_2 \dots \varphi_{\beta+1}$, which are in fact equal to the idle times in the starting window $\epsilon_1, \dots \epsilon_\beta$ and ϵ in some order. These idle times φ_i are in positions $a\,(\varphi_i)\,\bar{a}\,\bar{a} \dots \bar{a}(ba)$; $i = 1, \dots, \beta$ for the β terms (ba); and $a(\varphi_{\beta+1})\,\bar{a}\,\bar{a} \dots \bar{a} b_S$ for the term b_S.

We want to construct a feasible cycle, denoted by $\mathcal{C}(\alpha, \beta, \gamma)$ or in short $\mathcal{C}(\alpha, \beta)$.

Using Rule 1, one can, as in Example 3, obtain all idle time vectors R_i for the sequence Z, Z,... as follows:

$$\begin{array}{lcl}
R_1 & = & [\ \epsilon_1\ \ \epsilon_2\ \epsilon_3 \dots \epsilon_\beta \quad \epsilon \quad\ \ \epsilon_1\ \dots \epsilon_{\beta-2}\ \epsilon_{\beta-1}\ \epsilon_\beta \quad] \\
R_2 & = & [\ \epsilon\ \ \epsilon_1\ \epsilon_2 \dots \epsilon_{\beta-1}\ \epsilon_\beta \quad \epsilon \quad \dots \epsilon_{\beta-3}\ \epsilon_{\beta-2}\ \epsilon_{\beta-1}\ ] \\
R_3 & = & [\ \epsilon_\beta\ \epsilon\ \ \epsilon_1 \dots \epsilon_{\beta-2}\ \epsilon_{\beta-1}\ \epsilon_\beta \dots \epsilon_{\beta-4}\ \epsilon_{\beta-3}\ \epsilon_{\beta-2}\ ] \\
\vdots & & \\
R_{\beta+1} & = & [\ \epsilon_2\ \epsilon_3\ \epsilon_4 \dots \epsilon \quad\ \ \epsilon_1 \quad\ \epsilon_2\ \dots \epsilon_{\beta-1}\ \epsilon_\beta \quad \epsilon \quad] \\
R_{\beta+2} & = & R_1
\end{array} \tag{9}$$

We obtain the cycle $\mathcal{C}(\alpha, \beta)$ with a sequence of exactly $(\beta + 1)$ letters Z and the corresponding idle time pattern $R_1, R_2 \dots R_{\beta+1}$. Using (5) and (6), the cycle time of $\mathcal{C}(\alpha, \beta)$ is

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

on which we have placed $(\beta+1)(1+\alpha+2\beta)$ coupled tasks. We therefore obtain the following.

Proposition 2 *The mean cycle time $\lambda(\mathcal{C}(\alpha,\beta))$ of $\mathcal{C}(\alpha,\beta)$ is given by the formula*

$$\lambda(\mathcal{C}(\alpha,\beta))=\frac{(\beta+1)(2L+a+b)-\gamma}{(\beta+1)(1+\alpha+2\beta)} \tag{10}$$

independent of the choice of ϵ_i.

Hence, according to Proposition 2, for each letter-pattern Z, with non-intrinsic idle times such as described above, the mean cycle time of $Z, Z, \ldots$ depends only on the profile (α,β) associated with Z. Therefore, the cycles $\mathcal{C}(\alpha,\beta)$ are equivalent for all feasible ϵ_i (*i.e.,* $\epsilon_i \geq 0$ for all i and $\sum\epsilon_i \leq \gamma$). In the following, we shall give some particular choices of feasible ϵ_i's.

Setting $\epsilon_i=\epsilon=\frac{\gamma}{\beta+1}$ for all i, we get $R_i=R_1$ for all i. With this *equilibrated* setting of idle times, we obtain the cycle $\mathcal{C}(\alpha,\beta)$ in compact form, based on a single letter-pattern Z, which requires however a rational placement of the coupled tasks. For other choices of ϵ_i, we can use the equivalent cycle $\mathcal{C}(\alpha,\beta)$ above with $(\beta+1)$ repetitions of Z and $R_1, \ldots, R_{\beta+1}$ as idle time pattern. Rather than presetting ϵ_i and ϵ, satisfying (7) in the initial window W_1 and then applying the left-shifted placement strategy, one may also apply the left-shifted placement strategy in W_1. This yields $\epsilon=\gamma, \epsilon_1=\cdots=\epsilon_\beta=0$ and gives an integer placement of all coupled tasks, but at the expense of very long cycles.

As shown in Baptiste (2010), 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 ($a=5$; $L=43$; $b=3$), the profile $(0,5)$ and $\epsilon=\gamma=3$, $\epsilon_i=0$ ($i=1,2,\ldots,5$), we get integer placements for all tasks, but the cycle time increases already to $(\beta+1)(2L+a+b)-\gamma=561$. These rapidly increasing cycle times are certainly one of the reasons why their detection is so difficult if the graph approach is used (Ahr et al. 2004; Baptiste 2010; Brauner et al. 2009; Lebacque 2007).

The particular cycle $\mathcal{C}(0,\beta)$ possesses additional symmetries. Its letter pattern is simply a succession of '*ab*'. Starting again with window W_1 and their idle times $[\epsilon_1,\epsilon_2,\ldots,\epsilon_\beta,\epsilon]$, it is easy to see that the idle times in W_1 are identically repeating themselves in the following (non-overlapping) adjacent window. We get, therefore, additional equivalent cycles of the length of a single window, $a+b+L$, where we have placed $(\beta+1)$ coupled tasks in the cyclic sense. Hence, we can express the mean cycle time, equivalently to (10), in the form

$$\lambda(\mathcal{C}(0,\beta))=\frac{a+b+L}{\beta+1}. \tag{11}$$

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

We want to consider the set of all cycles $\mathcal{C}(\alpha,\beta)$ such as described above, *i.e.,*

$$\mathcal{L}=\{\mathcal{C}(\alpha,\beta):(\alpha,\beta)\text{ is a profile}\}. \tag{12}$$

Definition 3 Let $\mathcal{S}$ be a set of cycles. Then, the *best* cycle in $\mathcal{S}$ is one that has the smallest mean cycle time of all cycles in $\mathcal{S}$.

2.2 Finding the best solution in $\mathcal{L}$

Comparing the mean cycle times for different profiles, one obtains the dominance rules given by the following lemmas.

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

[i] $\mathcal{C}(\alpha+1,\beta,\gamma-a)$ if $\gamma \geq a$
[ii] $\mathcal{C}(\alpha-1,\beta+1,\gamma-b)$ if $\alpha \geq 1$ and $a > \gamma \geq b$.

It is easy to show *[i]* by comparing the mean cycle times of cycles $\mathcal{C}(\alpha,\beta)$ and $\mathcal{C}(\alpha+1,\beta)$. Likewise, *[ii]* is obtained by comparing the mean cycle times of $\mathcal{C}(\alpha,\beta)$ and $\mathcal{C}(\alpha-1,\beta+1)$.

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

[iii] $\mathcal{C}(\alpha+2,\beta-1,\gamma-(a-b))$ if $\beta \geq 1$ and $\gamma \geq (\beta+1)(a-b)$
[iv] $\mathcal{C}(\alpha-2,\beta+1,\gamma+(a-b))$ if $\alpha \geq 2$ and $\gamma \leq (\beta+1)(a-b)$.

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

Proof Setting $\alpha'=\alpha+2$ and $\beta'=\beta-1$, we get $\gamma'=L-\alpha' a-\beta'(a+b)=\gamma-(a-b)$ in *[iii]* and $\alpha'=\alpha-2$, $\beta'=\beta+1$ and $\gamma'=\gamma+(a-b)$ in *[iv]*. We have for both cases $\alpha'+2\beta'=\alpha+2\beta$. A glance at the mean cycle times (Eq. (10)) 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 *[iii]* 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 dominance rules imply that all cycles $\mathcal{C}(\alpha,\beta)$ with $\alpha \geq 2$ and $\beta \geq 1$ are dominated. In fact, each such cycle can be improved by reducing β according to *[iii]* if $\gamma \geq (\beta+1)(a-b)$ or by increasing β according to *[iv]* whenever $\gamma \leq (\beta+1)(a-b)$. We necessarily end with a non-reducible cycle, called $\mathcal{C}^N$, which is the best for class $\mathcal{L}$ in Eq. (12) and can be found among the ones of the form $\mathcal{C}(0,\beta)$, $\mathcal{C}(1,\beta)$, and $\mathcal{C}(\alpha,0)$. To decide which case occurs one simply may check whether the cycle $\mathcal{C}(0,\beta)$ or $\mathcal{C}(1,\beta)$ is dominated, using *[iii]*. Since we know how to construct the cycles $\mathcal{C}(\alpha,\beta)$, it is sufficient to determine the profile $(\alpha^N,\beta^N,\gamma^N)$ of $\mathcal{C}^N$, which we call the *best* profile of class $\mathcal{L}$.

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.

Algorithm 1 Best profile of class $\mathcal{L}$

```
1: **Input**: Integers $a, b, L$ ($a > b, L \geq a+b$)
2: **Output**: Best profile ($\alpha^N, \beta^N, \gamma^N$) and mean cycle time $\lambda$
3:
4: Let $\beta^N = \lfloor L/(a+b)\rfloor$ and $R = L - \beta^N(a+b)$
5: **if** $R < a$ **then**
6:    $\alpha^N = 0$ and $\gamma^N = R$
7: **else**
8:    $\alpha^N = 1$ and $\gamma^N = R - a$
9: **end if**
10:
11: **if** $\gamma^N < (\beta^N+1)(a-b)$ **then**
12:    Return ($\alpha^N, \beta^N, \gamma^N$)
13: **else**
14:    $\alpha^N = \lfloor L/a\rfloor$, $\beta^N = 0$ and $\gamma^N = L - \alpha^N a$
       {here $\gamma^N$ satisfies $b > \gamma^N > a-b$}
15:    Return ($\alpha^N, \beta^N, \gamma^N$)
16: **end if**
17:
18: $\lambda = \frac{(\beta^N+1)(2L+a+b)-\gamma^N}{(\beta^N+1)(1+\alpha^N+2\beta^N)}$
19: Return $\lambda$
```

Remark 2 If $\gamma^N \neq (\beta^N+1)(a-b)$ till line 9 of Algorithm 1, then the best profile is unique. Otherwise, there are multiple solutions which have the profiles $(\lfloor L/a\rfloor - 2i, i)$ with $i = 0,1,\ldots,\lfloor L/(a+b)\rfloor$.

Remark 3 The triplet $(\alpha^N, \beta^N, \gamma^N)$ is of size $O(\log L)$ since $\alpha^N, \beta^N, \gamma^N < L$. Computing the euclidean division of L by a or $a+b$ demands $O(\log L)$ operations on bits, and hence the overall complexity for obtaining the triplet $(\alpha^N, \beta^N, \gamma^N)$ is $O((\log L)^2)$, since one has to compare γ^N and $(\beta^N+1)(a-b)$. The mean cycle time λ is obtained in $O((\log L)^2)$, and hence the complexity of Algorithm 1 is $O((\log L)^2)$.

Example 4 We give an example for all forms of profiles that can occur for $\mathcal{C}^N$.

(a) Previous example: $a = 5$, $b = 3$, $L = 43$: $\mathcal{C}^N = \mathcal{C}(0,5,3)$, $\lambda = 8.5$
(b) $a = 5, b = 3, L = 31$: $\mathcal{C}^N = \mathcal{C}(1,3,2)$, $\lambda = 8.69$
(c) $a = 10, b = 9, L = 112$: $\mathcal{C}^N = \mathcal{C}(11,0,2)$, $\lambda = 20.08$
(d) $a = 10$, $b = 9$, $L = 111$. Here $\gamma^N = (\beta^N+1)(a-b)$ and we get multiple solutions: $\mathcal{C}^N = \mathcal{C}(11,0,1)$, $\mathcal{C}(9,1,2)$, $\mathcal{C}(7,2,3)$, $\mathcal{C}(5,3,4)$, $\mathcal{C}(3,4,5)$, $\mathcal{C}(1,5,6)$, $\lambda = 20.0$

Since Algorithm 1, giving the best cycle for class $\mathcal{L}$, is polynomial in the input size with complexity $O((\log L)^2)$, we obtain directly the following result.

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

So far we have imposed certain restrictions on $W_1 = [a, L, b]$ (3) and the element sequence Z (4) for the construction of our cycles $\mathcal{C}(\alpha,\beta)$. One may start the cycle anywhere in the form (5), where the profile (α,β) is just satisfying the inequality (1) as given in Definition 2. For our cycles $\mathcal{C}(\alpha,\beta)$, we have taken first all a's and then all (ba)'s. We could have chosen as well to begin with all (ba)-terms and then the a's, or any other order of β terms (ba) and α terms a and also $\overline{a}$. Let us denote W_1' and Z' such more general sequences, associated with the profile (α,β). Compare also with Example 2, where different windows in $\mathcal{C}(\alpha,\beta)$, following W_1, also take this more general form. We may now relax the conditions, imposed on the cycle $\mathcal{C}(\alpha,\beta)$, as follows:

(Relax 1) Start with any window W_1' and sequence Z' with the profile (α,β). Observe that one can always find a window W_1' in the generic form (5). Simply start with any 'a' that is preceded by a 'b'.
(Relax 2) Construct a cycle $\mathcal{C}'(\alpha,\beta)$ from the sequence $Z', Z' \ldots$ by inserting suitable idle times, not necessarily applying the left-shifted placement strategy.

We want to make sure that the cycle $\mathcal{C}(\alpha,\beta)$ cannot be improved, *i.e.*, its mean cycle time cannot be reduced by considering these more general cycles $\mathcal{C}'(\alpha,\beta)$. We define the larger class $\mathcal{L}' = \{\mathcal{C}'(\alpha,\beta) : (\alpha,\beta) \text{ is a profile}\}$.

Proposition 3 *$\mathcal{C}^N$ is also the best cycle in the class $\mathcal{L}' \supseteq \mathcal{L}$.*

Proof (Relax 1) We define idle times ϵ_i in the initial window W_1' according to Rule 1. Then we obtain a feasible cycle $\mathcal{C}'(\alpha, \beta)$, using the left-shifted placement strategy. All results, in particular the mean cycle time, will be the same, since Rule 1 is independent of the order of the terms (ba), a, and $\overline{a}$. We know now that our cycles $\mathcal{C}(\alpha, \beta)$ are equivalent to such cycles $\mathcal{C}'(\alpha, \beta)$.

(Relax 2) Let us take the cycle $\mathcal{C}'(\alpha, \beta)$ with the specific idle times $\epsilon_i = 0$ for $i = 1, 2 \ldots \beta$ and $\epsilon = \gamma$ in window W_1'. Then any other feasible cycle $\mathcal{C}'$, not necessarily using the left-shifted placement strategy, is based by definition on the same sequence $Z', Z', \ldots$ However, in $\mathcal{C}'(\alpha, \beta)$, all corresponding a's are placed at least as early as in $\mathcal{C}'$. This means that this particular cycle $\mathcal{C}'(\alpha, \beta)$ is at least as good as $\mathcal{C}'$. □

If we want to specify completely the cycles $\mathcal{C}(\alpha, \beta)$, we also have to define the idle times ϵ_i in the starting window. Of particular interest are the two placements introduced in Sect. 2.1. In the *left-shifted placement* version, $\epsilon = \gamma, \epsilon_1 = \cdots = \epsilon_\beta = 0$ and we have an integer placement of the tasks. The corresponding cycles are denoted by $\mathcal{C}_s(\alpha, \beta)$. In the *balanced* or *equilibrated* version, with cycles called $\mathcal{C}_e(\alpha, \beta)$, idle times are such that $\epsilon_i = \epsilon = \frac{\gamma}{\beta+1}$ for all i.

So far, we have only considered cycles that are characterized by a single profile (classes $\mathcal{L}$ and $\mathcal{L}'$). There, $\mathcal{C}^N$ is the best cycle (Theorem 1, Proposition 3). However, general cycles will have varying profiles in different windows. See the following illustrative example.

Example 5 Set $a = 2$, $b = 1$, $L = 6$ and consider the cycle C (in brackets { }, all idle times are indicated):

$\{ababab(2)b(2)baaaab(1)b(1)b(1)ba(1)a(1)\}$
$\{ababab(2)b\ldots\}$.

The windows of C have in succession the following profiles:

(0, 2); (1, 1); (2, 0); (3, 0); (3, 0); (3, 0); (3, 0); (2, 0); (1, 1).

One has $\mathcal{C}^N = \mathcal{C}(0, 2)$ with $\lambda(\mathcal{C}^N) = 3.0$ and $\lambda(\mathcal{C}) = \frac{36}{9} = 4.0$. Hence, $\mathcal{C}^N$ is dominating this particular 4-profile cycle $\mathcal{C}$.

It will be shown in Sect. 3 that the mono-profile cycle $\mathcal{C}^N$ is also outperforming all possible multi-profile cycles.

3 Optimal solution of the identical coupled task scheduling problem

We continue to assume that $a > b$ and $L \geq a + b$. We know already that $\mathcal{C}^N$ is the best mono-profile cycle. Before turning to general multi-profile cycles, we first summarize the key facts of the cycles $\mathcal{C}(\alpha, \beta)$ and in particular of $\mathcal{C}^N$ and their corresponding letter patterns Z. The final goal of this section is to show that the cycle $\mathcal{C}^N$ of Algorithm 1 is in fact an optimal cycle for the identical coupled task problem.

3.1 Tight mono-profile cycles

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:

$$\begin{aligned} &\max\ M = \alpha + 2\beta \\ &\text{s.t.}\quad L = \alpha a + \beta(a+b) + \gamma \\ &\qquad\ \alpha, \beta, \gamma \geq 0 \qquad \text{integer} \end{aligned} \tag{13}$$

This is true since for every optimal solution (α, β) of (13), the cycle $\mathcal{C}(\alpha, \beta)$ with $\beta \geq 1$ and $\alpha \geq 2$ can be transformed to $\mathcal{C}^N$ without ever decreasing the number of elements $\alpha + 2\beta$ (*i.e.,* using uniquely the dominance rules *[iii]* and *[iv]* in Lemma 2).

Denote by M^* the optimal value of the integer program (13). Then each window $W = [a, L, b]$ of $\mathcal{C}^N$ with profile (α^N, β^N) contains the maximum possible number of elements $M^* + 2 = 2 + \alpha^N + 2\beta^N$. We call such windows *tightly packed* and say that a cycle is *tight* if all its windows are tightly packed.

For instance, the coupled task, marked 1 in Fig. 2, forms a tightly packed window.

We also call a profile *tight* if it is an optimal solution of (13). Our cycles $\mathcal{C}(\alpha, \beta)$, whose profile (α, β) is tight, are tight cycles, and the cycle $\mathcal{C}^N$ is the best tight cycle in this set.

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

$$\alpha = M^* - 2\beta; \quad \max\left\{0, \left\lceil \frac{M^* a - L}{a - b} \right\rceil\right\} \leq \beta \leq \left\lfloor \frac{M^*}{2} \right\rfloor. \tag{14}$$

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.

Example 6 List of tight profiles

(a) $a = 10, b = 9, L = 80$:
$(\alpha, \beta, \gamma) = (0, 4, 4), (2, 3, 3), (4, 2, 2), (6, 1, 1), (8, 0, 0)$
$\mathcal{C}^N = \mathcal{C}(0, 4, 4)$, optimal value of (13) is $M^* = \alpha + 2\beta = 8$.
(b) $a = 5, b = 3, L = 100$:
$(\alpha, \beta, \gamma) = (0, 12, 4), (2, 11, 2), (4, 10, 0)$, $\mathcal{C}^N = \mathcal{C}(0, 12, 4)$, optimal value of (13) is $M^* = \alpha + 2\beta = 24$.

The number of tight profiles may be quite large and has the upper bound $(\lfloor M^*/2 \rfloor + 1)$. This bound is actually reached for the instance (a) of Example 6.

One can expect that tight cycles will play an important role in the construction of the optimal cycle for the cyclic identical coupled task problem, simply because all its windows contain the maximum number of elements.

3.2 Block decomposition of a general cycle

Let us now consider an arbitrary cycle $\mathcal{C}$. There might be the possibility to obtain very good (even optimal) cycles, for which only pieces of different tight cycles are somehow patched efficiently together. To analyze such *irregular*, *i.e.,* multi-profile cycles and finally to discard them, we shall generalize the concept of the letter patterns Z that generate the cycles $\mathcal{C}(\alpha, \beta)$. We know that Z contains exactly $2(M^* + 1)$ elements if the profile is tight.

Definition 4 Let $\mathcal{C}$ be an arbitrary cycle. We define a *block* B of $\mathcal{C}$ as a sequence of exactly $2(M^*+1)$ consecutive elements. The length $|B|$ is given by the sum of a's and b's and all the idle times contained in B, starting with its first element and including the idle time that follows its last element. The blocks following exactly the sequence of a's, b's, and idle times of a tight cycle, are called *tight*; all others are non-tight. A block B is *short* if its length satisfies $|B| \leq 2L + a + b$ and *long* if $|B| > 2L+a+b$. The *gain* of block B is defined as $\Delta(B) = (2L + a + b) - |B|$. We say that a block B' *dominates* block B if their gains satisfy $\Delta(B') \geq \Delta(B)$.

We want to write a cycle $\mathcal{C}$ as a sequence of blocks. We assume an integer number of complete blocks since one can always, if necessary, take a suitable multiple of the cycle $\mathcal{C}$.

Example 5 (continued) The cycle $\mathcal{C}$ of Example 5 contains 18 elements. The cycle $\mathcal{C}^N = \mathcal{C}(0, 2)$ defines the number of elements of a block: $2(M^* + 1) = 2(1 + \alpha^N + 2\beta^N) = 10$. Thus one can take five copies of $\mathcal{C}$ to obtain an equivalent cycle that consists of exactly 9 complete blocks. It is easy to see that all 9 blocks are long.

As a first result, we show that one cannot switch directly, without passing through non-tight blocks, from one tight block to another tight block with a different profile. Two tight blocks with different tight profiles just do not fit together. The exact statement is as follows.

Lemma 3 *All neighboring tightly packed windows of a feasible cycle have necessarily the same profile. Therefore, also neighboring tight blocks must have the same profile.*

Proof Consider two neighboring tightly packed windows. Then we have two consecutive a-terms that are only separated by a certain number x of b's, see as an illustration Fig. 2 (the boxes marked 1 and 2 are neighboring tightly packed windows and $x = 1$). 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 Proposition 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. □

With the generality of Definition 4, we have to distinguish three forms of tight blocks. A block may start with an 'a' (a_1) preceded by a 'b' or a certain number of a's or it may start with a certain number of leading b's ($b\,.\,ba_1$). Let b_1 be the b-part of a_1. Let a_2 be the last a preceding b_1, and let b_2 denote its corresponding b-part. The elements belonging to the block are denoted between brackets. For a tight profile (α, β, γ), we have in general $(\beta + 1)$ non-intrinsic idle times (compare with Rule 1). We indicate below in brackets the only idle time (denoted φ) that influences the block length.

(a) Standard form:

$$b\{a_1 \ldots a_2(\varphi)b_1 \ldots b_2\}a \qquad \text{block length } 2L + a + b - \varphi \tag{15}$$

(b) Shifted form Type I:

$$ba\,.\,a\{a_1 \ldots a_2(\varphi)b\,.\,bb_1 \ldots b_2a\,.\,a\}a \qquad \text{block length } 2L + a + b - \varphi \tag{16}$$

The sequence of b's $b\,.\,b$ and the two sequences of a's $a\,.\,a$ have the same number of elements.

(c) Shifted form Type II:

$$ab\,.\{b\,.\,ba_1 \ldots ba\,.\,a\,.\,a_2(\varphi)b_1 \ldots ab\,.\}b\,.\,b_2a \qquad \text{block length } 2L + a + b - \varphi \tag{17}$$

Again, sequences $b\,.\,b\,.\,b$, $a\,.\,a\,.\,a_2$, and $b\,.\,b\,.\,b_2$ have the same number of elements.

Remark 4 Let us consider separately the tight profile of the form $(\alpha, 0)$. In the case of scheme (b), a_1 may be equal to a_2 and in case (c), the sequence might be $a\{b\,.\,ba_1\,.\,a(\varphi)\}b_1$ where $a_1\,.\,a$ and $b\,.\,b$ are both of length $M^* + 1$ (again $a_1 = a_2$). The block length is again the same.

All three forms of tight blocks have the same lengths, and these blocks are short. For the tight blocks above, one has the positive gain $\Delta(B) = \varphi$. We know that φ varies from 0 to γ for the tight profile (α, β, γ), depending on the position of the block within the cycle and the values of the idle times in the initial window. Suppose we have a tight cycle $\mathcal{C}(\alpha, \beta, \gamma)$. Then any sequence of $(\beta + 1)$ blocks (a complete cycle), where the first block has the idle time pattern R_1 in (9), gives in succession $\Delta = \varphi = \epsilon, \epsilon_\beta, \epsilon_{\beta-1} \ldots \epsilon_1$. In particular, the left-shifted version in the first block gives $\Delta = \varphi = \gamma$ and then β times $\varphi = 0$. Another choice is $\Delta = \varphi = \frac{\gamma}{\beta+1}$ for every block.

We shall use the cycle $\mathcal{C}_e^N$ in the remainder of this section. Then each block of $\mathcal{C}_e^N$, given in standard form, is a complete cycle with gain $\Delta = \frac{\gamma^N}{\beta^N+1}$ for every block. Now consider a general cycle $\mathcal{C}$ and one of its possible block decompositions. We want to compare the block lengths of $\mathcal{C}_e^N$ with those of cycle $\mathcal{C}$. Here, it is understood that $\mathcal{C}$ contains a sequence of complete blocks and we take the matching multiple of blocks $\mathcal{C}_e^N$ for the comparison. If we are lucky, all blocks of $\mathcal{C}$ are long. Then $\mathcal{C}_e^N$ dominates $\mathcal{C}$ since the domination is given block by block. This is the case in Example 5. However, the general situation is much more complicated. Some blocks of $\mathcal{C}$ may be shorter than the corresponding ones in $\mathcal{C}_e^N$, and others may be longer. A more careful analysis is required to complete the dominance proof of $\mathcal{C}_e^N$, and we will also have to consider certain subsequences of blocks for the comparison of their lengths.

3.3 Essential non-tight blocks

Let us now turn to non-tight blocks. We want to determine their status (short or long). According to the general definition, the element sequence could still be that of a tight block, but the idle time pattern is different. For instance, one may have extra idle times at the end between the a's in blocks of Type I. Note that in a general cycle $\mathcal{C}$, one might have such blocks, since they may eventually be useful in the transition to another tight block. The resulting blocks are either short or long.

However, we shall only be interested in lower bounds for the block lengths in $\mathcal{C}$. For that purpose, it is sufficient to consider a more restricted class of non-tight blocks.

Definition 5 A non-tight block is called *essential*, if its sequence of a's and b's does not follow that of a tight block.

In the analysis of essential non-tight blocks of some cycle $\mathcal{C}$, the tight profiles of the form $(\alpha = 0, \beta, \gamma)$ will play a special role. In this case, $\gamma < a$ and may exceed b. All other tight profiles $(\alpha > 0, \beta, \gamma)$ satisfy $\gamma < b$. The tight blocks connected with $(0, \beta)$ reduce to two forms:

(d) Standard form:

$$b\{a_1 b \ldots b a_2(\varphi) b_1 a \ldots a b_2\} a \qquad \text{block length } 2L + a + b - \varphi \tag{18}$$

(e) Shifted form:

$$\{b a_1 b \ldots b a_2(\varphi) b_1 a \ldots a\} b_2 a b \qquad \text{block length } 2L + a + b - \varphi \tag{19}$$

It is possible to construct from the shifted form (e) essential non-tight blocks that are short.

Example 7 Set $a = 5, b = 3$, and $L = 20$ and use the tight profile $(0, 2, 4)$. Consider the following feasible sequences: $ba_1baba_2(\varphi_1 = 4)b_1(\varphi_2 = 5)bab_2$ and $ba_1(\varphi_1 = 3)aba_2(\varphi_2 = 4)b_1abab_2$, which form essential non-tight blocks.

Both are short blocks, consisting as defined of $2(M^* + 1) = 10$ elements a and b and satisfying $\Delta = 4 - b = 1$. Note that the corresponding sequence, but derived from (d), yields a long essential non-tight block: $a_1baba_2(\varphi_1 = 4)b_1(\varphi_2 = 5)bab_2a$ with $\Delta = 4 - a = -1$.

Consider now an essential non-tight block B in a general cycle $\mathcal{C}$. We denote again its first window $W_1 = [a_1, L, b_1]$ and last window $W_2 = [a_2, L, b_2]$, where a_2 is the last 'a' preceding b_1 (similar to (a), …, (e)) and b_2 may lie inside or outside B.

In general, we have the following result.

Lemma 4 *All essential non-tight blocks, not containing tightly packed windows with profile* $(0, \beta)$*, are long.*

The proof is tedious and technical and will be given in the Appendix.

Lemma 4 shows that the type of essential non-tight blocks in Example 7 is the only ones which are short.

3.4 Final block decomposition of a multi-profile cycle $\mathcal{C}$

We know already that $\mathcal{C}_e^N$ is the best mono-profile cycle. Let now a general *multi-profile cycle* $\mathcal{C}$ be given, *i.e.,* a cycle for which at least two windows have a different profile.

We take multiple copies of $\mathcal{C}$ to obtain a convenient sequence of elements for the comparison with $\mathcal{C}_e^N$.

First, we scan $\mathcal{C}$ from left to right, starting the search with any window, to detect the windows with tight profile of the form $(0, \beta)$. Each such window with elements left-shifted is itself a complete cycle. Then, we cut out, one at a time, any such window encountered in the remaining of the cycle and append it at the end of this first window with profile of the form $(0, \beta)$. We continue this process until all windows with profile $(0, \beta)$ are following one another in the contracted sequence.

We make sure, by taking multiple copies of $\mathcal{C}$, that the number of these windows is a multiple of $\beta + 1$ blocks and the remaining elements consists of an integer number of complete blocks. The cycle $\mathcal{C}$ can be started anywhere. Therefore, we can assume that the last block is an essential non-tight block.

We apply to the sequence of all windows with profile $(0, \beta)$ the left-shifted placement strategy to obtain replicas of the feasible tight cycle $\mathcal{C}_s(0, \beta)$. We know that cycle $\mathcal{C}_e^N$ dominates this cycle so that this part of the sequence can be dropped. It remains a sequence of complete blocks denoted by $\mathcal{C}'$.

If we can show that $\mathcal{C}_e^N$ is also dominating the block sequence $\mathcal{C}'$, then $\mathcal{C}_e^N$ would in fact dominate $\mathcal{C}$.

The sequence $\mathcal{C}'$ cannot consist of only tight blocks since $\mathcal{C}$ is a multi-profile cycle. Any transition from a tight block to another tight block with a different profile must go through some essential non-tight blocks (Lemma 3).

There is the following simple case:

Proposition 4 *If there are only essential non-tight blocks in $\mathcal{C}'$, then $\mathcal{C}_e^N$ is dominating $\mathcal{C}$.*

Proof In this case, all blocks in $\mathcal{C}'$ are long (Lemma 4). Hence, $\mathcal{C}^N$ is dominating $\mathcal{C}'$ and hence $\mathcal{C}$. □

For all other cycles, there is at least one block in $\mathcal{C}'$ following the sequence of a tight profile and a block following a different sequence. Let us contract $\mathcal{C}'$ even further to construct a sequence of blocks $\overline{\mathcal{C}}$ as follows:

(i) The sequence $\mathcal{C}'$ can start with essential non-tight blocks. If there are any, scan $\mathcal{C}'$ from left to right until arriving at a block that follows the element sequence of some tight profile. As explained earlier, such a block may be tight or not. We continue with the sequence of blocks, until we get for the first time to a block with a different profile. This subsequence is considered in isolation, independent of the actual cycle structure, and we apply the left-shifted placement strategy to all its elements. These shortened blocks follow now exactly the pattern of some tight cycle, followed by a single essential non-tight block.

(ii) Continue with the following essential non-tight blocks in $\mathcal{C}'$, until one gets to the next block that follows again the element sequence of some tight profile. Go to (i) and start another subsequence in exactly the same fashion. Continue until the sequence of blocks is exhausted.

Now suppose that a subsequence, constructed above in (i), exists in the modified block sequence $\overline{\mathcal{C}}$ of $\mathcal{C}$. It is of the form $B_1, \ldots, B_u; D$, where all blocks B_i are tight and belong to the same tight profile (α, β, γ) (Lemma 3) and D is an essential non-tight block. By construction, $\Delta(B_1) = \gamma$, then $\Delta(B_i) = 0$ for $2 \leq i \leq \beta + 1$ if there are that many blocks, and the total gain on the complete cycle is $\Delta(B_1, \ldots, B_{\beta+1}) = \Delta(B_1) + \cdots + \Delta(B_{\beta+1}) = \gamma$. All these complete tight cycles are dominated by $\mathcal{C}_e^N$ and can be eliminated from $\overline{\mathcal{C}}$.

It remains a new sequence $\overline{\mathcal{C}}$ that contains no window with the tight profile $(0, \beta)$ and no complete tight subcycle. Hence, the block sequence $\overline{\mathcal{C}}$ consists of a certain number of subsequences $B_1, \ldots, B_u; D$ of length $u \leq \beta$. There, all tight blocks B_i have the same profile $(\alpha \neq 0, \beta, \gamma)$ and the subsequences are separated by further essential non-tight blocks. These essential non-tight blocks are all long and can be dropped (Lemma 4).

The last remaining block sequence $\overline{\mathcal{C}}$ is called the *final block decomposition* of $\mathcal{C}$. If we can show that $\mathcal{C}_e^N$ is also dominating $\overline{\mathcal{C}}$, then $\mathcal{C}_e^N$ would in fact dominate $\mathcal{C}$.

Now let us consider such a subsequence $B_1, \ldots, B_u; D$ with $u < \beta + 1$, which we call a *partial subsequence* in $\overline{\mathcal{C}}$. Note that the blocks of $\mathcal{C}_e^N$ are usually not dominating individually the blocks B_i since already $\Delta(B_1) = \gamma$, which may exceed the gain $\frac{\gamma^N}{\beta^N+1}$ of each of the blocks in $\mathcal{C}_e^N$.

Lemma 5 *Consider a partial subsequence $B_1, \ldots, B_u; D$ in the final block decomposition $\overline{\mathcal{C}}$. Then:*

$$\Delta(B_1, \ldots, B_u; D) \leq 0 \textit{ for } u < \beta$$

and

$$\Delta(B_1, \ldots, B_u; D) \leq \gamma \textit{ for } u = \beta.$$

Proof. See Appendix.

3.5 The main result

Let $\mathcal{C}$ be a general mono-profile or multi-profile cycle.

Theorem 2 *The cycle $\mathcal{C}_e^N$ dominates all other cycles.*

Proof We know already that $\mathcal{C}_e^N$ is dominating all mono-profile cycles and all cycles of the special form in Proposition 4. It remains the case where the final block sequence $\overline{\mathcal{C}}$ contains a certain number of partial subsequences.

Let us apply Lemma 5 to each of the partial subsequences $B_1, \ldots, B_u; D$. We get $\Delta(B_1, \ldots, B_u; D) \leq 0$ for $u < \beta$, and $\mathcal{C}_e^N$ is dominating this sequence since its gain is positive on each block. If $u = \beta$, we know that $\mathcal{C}_e^N$ dominates the complete tight cycle $B_1, \ldots, B_{\beta+1}$ which in turn dominates $B_1, \ldots, B_\beta; D$ since $\Delta(B_{\beta+1}) \geq 0$ and $\Delta(D) < 0$ (Lemma 4). Hence, $\mathcal{C}_e^N$ is performing better than $\overline{\mathcal{C}}$, which means that $\mathcal{C}_e^N$ is dominating $\mathcal{C}$.

This completes the proof of Theorem 2. □

Let us now return to a general cyclic identical coupled task problem, based on arbitrary a, L, and b. In Sect. 2, we have shown that the cases $a = b$ or $L < a + b$ can be solved in constant time $O(1)$. According to Algorithm 1 and Theorem 2, the case $a > b$ and $L \geq a + b$ is solved by $\mathcal{C}_e^N$ in $O((\log L)^2)$. For $a < b$, we get the same complexity by reversing the order. Altogether, we get the main result.

Theorem 3 *The cyclic identical coupled task problem, based on parameters a, L, and b is polynomially solvable with complexity $O((\log L)^2)$.*

As a further result, we also have determined the shortest optimal cycles. For the optimal profile (α, β, γ), it is the cycle $\mathcal{C}_e^N$ if $\alpha \neq 0$, consisting of a single block, and for $\alpha = 0$ the shortest optimal cycle is the triplet $(a, \frac{\gamma}{\beta+1}, b)$.

4 Conclusion

In Ahr et al. (2004), Table 2, the minimum 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 mean cycle times in the very special case $b = 1$, which is as follows:

$$\begin{array}{ll} \frac{a+1+2L}{2\frac{L+1}{a+1}} & \text{if } L \equiv -1 \mod (a+1) \\ \frac{a+1+L}{\lfloor \frac{L}{a+1} \rfloor + 1} & \text{otherwise} \end{array}$$

Also this formula is correct using (10). 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.

Appendix

Proof of Lemma 4 Let us start with some block in a cycle $\mathcal{C}$. We shorten this block even more by cutting it out of $\mathcal{C}$ and shifting the sequence as much as possible to the left. However, we leave all b's in the first window in their place. The resulting block B may be tight or is essential non-tight. Let B be non-tight and consider the three forms similar to (a), (b), and (c) in (15), (16), and (17).

Analog to (a), let B be of standard form

$$a_1 \ldots a_2(\varphi)b_1 \ldots b_2U, \tag{20}$$

where U is a string of a's and b's of length $|U|$, necessary to get a sequence of the required $2(M^* + 1)$ elements. The length of B is $|B| = 2L+a+b-\varphi+|U|$ or $\Delta(B) = \varphi - |U|$.

Let $(t+1)a > \varphi \geq ta$ for some integer $t \geq 1$. Constraint (1) implies that both windows W_1 and W_2 cannot contain more than $(2 + M^* - t)$ elements. Hence, the string $(a_1 \ldots b_2)$ consists of at most $2(1 + M^* - t)$ elements. This means that U has at least $2t$ elements. Observe that the beginning of U of length $(t+1)a$ can only contain a's, implying that $|U| \geq (t+1)a + (t-1)b$.

Consider also the feasible block B' (in brackets {}):

$$a^t\{a_1 \ldots a_2(\varphi - ta)b^t b_1 \ldots b_2U'\}. \tag{21}$$

Then B' is dominating B, *i.e.*, $\Delta(B') \geq \Delta(B)$. The reason is that placing any 'b' between a_2 and b_1 reduces φ by a but since one gets an additional element in both windows W_1 and W_2, U will be shortened by two elements. The block B is long since $\Delta(B) \leq \varphi - |U| < (t+1)a - ((t+1)a + (t-1)b) \leq -(t-1)b \leq 0$.

One can, therefore, find the shortest blocks of form (20) if $\varphi < a$, since all others are dominated (21). In this case, if none of the windows W_1 and W_2 is tightly packed, U contains at least two elements. The block is long, since $\Delta(B) = \varphi - |U| < a - (a+b) = -b$. The shortest essential non-tight blocks are obtained, not surprisingly, if exactly one of the windows W_1 and W_2 is tightly packed and the other window contains one fewer element than the maximum, *i.e.*, $1 + M^*$ elements. Then U consists of a single element. For the particular form (20), one gets $|U| = a$, independent of what element, a or b, follows b_2. Hence, all blocks B are long for this case, even if the tight profile $(0, \beta)$ is present.

Analog to (b), the essential non-tight block B may have the shifted form (in brackets { })

$$B : a^s\{a_1 \ldots a_2(\varphi)b^s b_1 \ldots b_2U\} \tag{22}$$

or B is of the form (c) with leading b's

$$B : b^s a_1 \ldots a_2(\varphi)b_1 \ldots b_2 \ldots \tag{23}$$

where B consists of the first $2(M^* + 1)$ elements of this sequence.

Both sequences contain a block B' of type (a), shifting sequence (22) by a^s to the left and shifting (23) by b^s to the right. Therefore, the shortest block B is obtained whenever the block B' is shortest. Hence, we get the shortest block B, if $\varphi < a$, and if one of the windows W_1 and W_2 is tightly packed ($2 + M^*$ elements) and the other window contains

$(1 + M^*)$ elements, then the block is to be completed with a single additional element.

By assumption, there is no tight profile $(0, \beta)$. Thus $\varphi \le \gamma < b$. The extra element is at least of length b. Hence, these blocks B are long since $\Delta(B) \le \varphi - b \le \gamma - b < 0$.

This completes the proof of Lemma 4. □

According to this proof, all shortest essential non-tight blocks can be obtained from a tight block, in standard or shifted form, as follows: Keep the window W_1 tightly packed and drop one of the a's in window W_2. The other possibility is to drop one of the a's in W_1. Then the corresponding b-part in W_2 is also eliminated, and an 'a' is to be inserted instead to obtain a tightly packed window W_2. Then the block is completed by adding the next element.

Proof of Lemma 5 Let a partial subsequence $B_1, \ldots, B_u; D$, $u \le \beta$, be given. It is sufficient to establish the lemma for the shortest essential non-tight block D. We also consider the sequence $B_1, \ldots B_u, B_{u+1}$ where B_{u+1} is obtained by continuing feasibly the sequence of blocks $B_1, \ldots B_u$ by another block B_{u+1} with the same tight profile. Let us compare B_{u+1} with D to get a bound for $\Delta(D)$. Note that all b's in front of b_1 of B_{u+1} are fixed in the same fashion as for D. Therefore, it is possible to construct D from B_{u+1}. Let us consider the middle section of B_{u+1}, which takes the form

$$\begin{aligned} &\ldots ba \ldots aa_2(\varphi = 0)b_1 \ldots \text{ or} \\ &\ldots ba \ldots aa_2(\varphi = 0)b \ldots bb_1 \ldots \end{aligned} \tag{24}$$

The shortest block D has in its corresponding string $(a \ldots aa_2)$ in (24) either the same number of a's as B_{u+1} or one fewer 'a' and some idle time φ'. In any case, we have

$$\Delta(D) \le \varphi' - b. \tag{25}$$

1. As long as $\varphi' = \varphi = 0$, one has $\Delta(B_1, \ldots B_u; D) \le \gamma - b < 0$ since $\gamma < b$ for all tight profiles $(\alpha \ne 0, \beta, \gamma)$. If D contains all a's of the sequence $(a \ldots aa_2)$, then the corresponding idle time φ' satisfies of course $\varphi' = \varphi = 0$. This is in particular true if the window W_1 remains tightly packed in D or if the string $(a \ldots aa_2)$ contains only a_2.

2. Now let one 'a' be dropped from $(a \ldots aa_2)$. Then the corresponding idle time φ' may take positive values. Let the sequence of B_{u+1} in (24) be of the form

$$\ldots baa \ldots aa_2(\varphi = 0)b \ldots a(idle')bb \ldots bb_2, \tag{26}$$

where the sequence $(aa \ldots a)$ contains at least one 'a' and $idle'$ indicates the idle time in the given position.. The block D may be obtained from (26) by dropping the first 'a' in $(aa \ldots aa_2)$. Then the corresponding b-part is also eliminated and a new 'a' is started to obtain the sequence

$$\ldots b(idle{=}a)a \ldots aa_2(\varphi{=}0)b \ldots a(idle', \text{ new } a)b \ldots bb_2. \tag{27}$$

We get for D the left-shifted sequence

$$\ldots ba \ldots aa_2(\varphi' = idle')b \ldots aab \ldots bb_2. \tag{28}$$

The profile of all $B_i, i = 1, \ldots, u+1$, is in this case necessarily $(\alpha \ge 1, \beta, \gamma < b)$. By construction, there is only one non-zero (non-intrinsic) idle time in each B_i of value $idle' = \gamma$. A glance at (9) and Rule 1 shows that one can only have the idle time γ in position (26) at the end of the cycle, *i.e.,* $u + 1 = \beta + 1$. Since $\Delta(D) < 0$ (Lemma 4), we get in this case $\Delta(B_1, \ldots, B_\beta; D) \le \Delta(B_1, \ldots, B_\beta, B_{\beta+1}) = \gamma$. Hence, Lemma 5 holds.

The other possibility is to drop an 'a' in $(aa \ldots aa_2)$, different from the first 'a'. Then we get for D the shifted sequence

$$\ldots ba \ldots aa_2(\varphi' = a - b)b \ldots ab \ldots a \ldots b_2. \tag{29}$$

Observe that the tight profile of W_2 in B_{u+1} $(\alpha \ge 2, \beta, \gamma < b)$ has changed in D to the tight profile $(\alpha - 2, \beta + 1, \gamma + (a - b))$. A sequence bbb or $\overline{aa}b$ has become bab, increasing β by 1. Since by assumption $\alpha - 2 \ne 0$ or $\alpha \ge 3$, one has $\gamma + (a - b) < b$ and $\Delta(B_1, \ldots, B_u; D) \le \gamma + (a - b) - b < 0$ and the lemma holds.

This completes the proof of Lemma 5. □

References

Ageev, A. A., & 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., & Reinelt, G. (2004). An exact algorithm for scheduling identical coupled tasks. *Mathematical Methods of Operations Research (ZOR)*, *59*, 193–203.

Baptiste, Ph. (2010). A note on scheduling identical coupled tasks in logarithmic time. *Discrete Applied Mathematics*, *158*(5), 583–587.

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

Blazewicz, J., Ecker, K., Kis, T., Potts, C. N., Tanaŝ, M., & Whitehead, J. (2010). Scheduling of coupled tasks with unit processing times. *Journal of Scheduling*, *13*(5), 453–461.

Brauner, N., Crama, Y., Grigoriev, A., & van de Klundert, J. (2005). A framework for the complexity of high-multiplicity scheduling problems. *Journal of Combinatorial Optimization*, *9*, 313–323.

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

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

Duron, C. (2002). *Ordonnancement en temps-réel des activités des radars*. PhD thesis, Université de Metz.

Elshafei, M., Sherali, H. D., & Smith, J. C. (2003). Radar pulse interleaving for multi-target tracking. *Naval Research Logistics*, *51*, 72–94.

Farina, A., & Neri, P. (1980). Multitarget interleaved tracking for the phased radar array. *IEE Proceedings F*, *27*, 312–318.

Gupta, J. N. D. (1996). 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.

Karp, M. (1978). A characterization of the minimum cycle mean in a digraph. *Discrete Mathematics*, *23*(3), 309–311.

Lebacque, V. (2007). Théories et applications en ordonnancement: contraintes de ressources et tâches agrégées en catégories. PhD thesis, Université Joseph Fourier.

Levner, E., Kats, V., de Pablo, D. A. L., & Cheng, T. C. E. (2010). Complexity of cyclic scheduling problems: A state-of-the-art survey. *Computers & Industrial Engineering*, *59*, 352–361.

Milojevic, D. J., & Popovic, B. M. (1992). Improved algorithm for the interleaving of radar pulses. *IEE Proceedings F*, *139*, 98–104.

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

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

Shahani, A. K., Orman, A. J., Potts, C. N., & Moore, A. R. (1996). Scheduling for a multifunction phased array radar system. *European Journal of Operational Research*, *90*, 13–25.

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

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