Available online at www.sciencedirect.com
SCIENCE DIRECT®
Discrete Applied Mathematics 145 (2005) 334–348
DISCRETE APPLIED MATHEMATICS
www.elsevier.com/locate/dam

Robotic cell scheduling with operational flexibility

M. Selim Akturk*, Hakan Gultekin, Oya Ekin Karasan

Department of Industrial Engineering, Bilkent University, Bilkent, Ankara 06533, Turkey

Received 14 May 2002; received in revised form 14 January 2004; accepted 19 February 2004

Abstract

In this paper, we study the problem of two-machine, identical parts robotic cell scheduling with operational flexibility. We assume that every part to be processed has a number of operations to be completed in these two machines and both machines are capable of performing all of the operations. The decision to be made includes finding the optimal robot move cycle and the corresponding optimal allocation of operations to these two machines that jointly minimize the cycle time. We prove that with this definition of the problem 1-unit robot move cycles are no longer necessarily optimal and that according to the given parameters either one of the 1-unit robot move cycles or a 2-unit robot move cycle is optimal. The regions of optimality are presented.

Keywords: Robotic cell; Cyclic scheduling; Automated manufacturing

1. Introduction

In order to be successful in today's highly competitive world, increasing productivity is an essential factor for manufacturing systems. Recent technological improvements opened new perspectives for industries. As a result, many industries are making use of automation, and hence the use of robots in industry is increasing rapidly. Robots are installed in order to reduce labor cost, to increase output, to provide more flexible production systems and to replace people working in dangerous or hazardous conditions [3]. An important application of robots in manufacturing is their use as material handling devices in robotic cells, where a robotic cell contains two or more robot-served machines [10]. There are no buffers at or between the machines. For the complexity of unlimited buffer space problem, we refer to Hurink and Knust [9]. Logendran and Sriskandarajah [11] study three different robotic cell layouts: robot-centered cells (where the robot movement is rotational), in-line robotic cells (where the robot moves linearly) and mobile-robot cells (generalization of in-line robotic cells and robot-centered cells). Sethi et al. [12] proved that for a 2-machine robotic cell producing a single part type, the optimal solution is a 1-unit cycle, where an n-unit cycle can be defined as a robot move cycle which produces exactly n units and ends up with the same state of the cell as the starting state. Hall et al. [8] proved that for three machine cells, producing single part-types, the repetition of 1-unit cycles dominates more complicated policies that produce two units. Crama and Klundert [6] established the validity of the conjecture of Sethi et al. [12] that "1-unit cycles yield optimal production rates for 3-machine robotic flowshops". Brauner and Finke [2] proved that 1-unit cycles do not necessarily yield optimal solutions for cells of size four or large. Reviews of the literature related to cyclic scheduling problems in robotic cells without buffers can be found in Crama et al. [4] and Sriskandarajah et al. [13].

In the current literature, the allocation of the operations to each machine is assumed to be constant and for given processing times the optimum robot move cycle minimizing the cycle time is to be determined. In some manufacturing

* Corresponding author.
E-mail addresses: arturk@bilkent.edu.tr (M.S. Akturk), ghakan@bilkent.edu.tr (H. Gultekin), karasan@bilkent.edu.tr (O.E. Karasan).

0166-218X/$ - see front matter
doi:10.1016/j.dam.2004.02.012

operations such as chemical electroplating this assumption is meaningful and these operations mostly require no-wait constraints (see for example [1]). However, in flexible manufacturing systems (FMS) the processing stations are predominantly CNC machines and they possess *operation flexibility* by definition. Operation flexibility can be defined as the ability to interchange the ordering of several operations for each part type [3]. Therefore, assuming that processing times are fixed on each CNC machine may not accurately represent the capabilities of the CNC machines and limits the number of alternatives unnecessarily for these systems. In this study we assume that each part has a number of different operations to be completed. Henceforth, the problem to be considered is to allocate the operations to the machines and is to find the robot move cycle that corresponds to this allocation of operations in order to jointly minimize the cycle time or in other words, maximize the long run average throughput rate.

For this problem definition, we prove that the optimal solution is not necessarily a 1-unit cycle as in the case of Sethi et al. [12] and show that the optimal solution could be a 2-unit cycle for some parameter input. We present regions of optimality for the potentially optimal robot move cycles.

The remainder of the paper is organized as follows. In the next section we present the problem definition and 1-unit robot move cycles. The operation allocation problem is described in Section 3. In Section 4, sensitivity analysis on parameters is made and in Section 5, we comment on the results and suggest new research directions.

2. Preliminaries and problem definition

This section reviews some standard terminology from the literature and introduces several definitions particular to this paper. Also in this section we give a formal definition of our problem and introduce our systematic approach in cycle time calculations.

We consider an in-line robotic cell of two identical machines which repeatedly produces one type of product. There is no buffer between the machines. The layout of the cell can be seen in Fig. 1. Each of the identical parts has a number of operations to be performed. The processing times are assumed to be identical for every operation in both of the machines. Consistent with the existing literature (see [12]), the loading and unloading times and the travel time of the robot from one station to a consecutive station are all assumed to be constant. Furthermore, the robot travel times satisfy the triangular equality (see [6]).

In this study, we consider the robotic cell to be composed of CNC machines and a material handling robot. The CNC machines possess operational flexibility and process flexibility by definition. Browne et al. [3] defines process flexibility as the ability to handle a mixture of operations. That is, if a part requires different operations such as drilling, milling, etc., process flexibility states that, one CNC can handle all of these operations. This is achieved by the tool magazines of these CNC machines which are capable of changing the tools easily and with very small setup times. On the other hand, operational flexibility is defined as the ability to interchange the ordering of several operations for each part type. That is, the processing sequence of operations required for a part type can be changed. In this study we assume that the operations constituting each part may be processed in any order. These two types of flexibilities make it possible to allocate every operation to any one of the two machines. As the allocation of the operations changes, the processing times on the machines also change accordingly. As a result, in our study the processing times are not fixed but rather are decision variables. We will try to find the processing times on both of the machines by allocating the operations to the machines and finding the robot move cycle which will jointly minimize the cycle time.

The following definitions on robot activities and n-unit robot move cycles are borrowed from Crama and Van de Klundert [5].

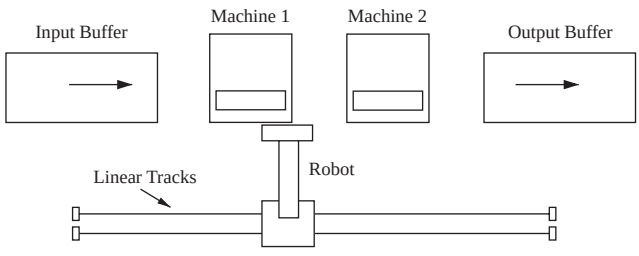

Fig. 1. Inline robotic cell layout.

Definition 1. A_i is the robot activity defined as; robot unloads machine i, transfers part from machine i to machine $i+1$, loads machine $i+1$. The machines are numbered as 1 and 2, the input buffer is numbered as 0 and the output buffer is numbered as 3. So we have activities A_0, A_1, and A_2.

Definition 2. An n-unit robot move cycle is the move cycle in which starting with an initial state of the system, the robot performs each activity exactly n times and ends up with the initial state of the system again. Note that, in an n-unit robot move cycle exactly n parts are produced.

Let us assume that each part has p required operations to be performed on either machine. We will use the following parameters and decision variables:

$t_i =$ Processing time of operation i on either machine 1 or 2, where $i \in [1 \dots p]$.
$P =$ Total processing time required to finish the production of a part, that is the sum of the processing times of all the p operations, $t_1 + t_2 + \cdots + t_p$.
$\varepsilon =$ The load (or unload) time of workstations by the robot.
$\delta =$ Time taken by the robot to travel between two consecutive stations.
$T =$ Long run average cycle time of any robot move cycle to produce *one* part.

In other words, using the classification scheme of Hall et al. [8], we can define our problem as IRC2 $|\, k{=}1,\ \delta_i{=}\delta,\ \varepsilon_i{=}\varepsilon \,|\, T$ with operational flexibility.

Sethi et al. [12] proved that there are two feasible 1-unit robot move cycles:

$S1 \equiv$ This can be represented by the sequence of robot activities $A_0A_1A_2$. Initially the system is empty and the robot is in front of the input buffer. After the listed activities are performed, the robot returns to the input buffer. Sethi et al. [12] provided the cycle time of $S1$ as

$$6\varepsilon + 6\delta + a + b,$$

where a is the processing time of the part on the first machine and b is the processing time of the part on the second machine. In our notation, a is the sum of a specific subset of operation times from $\{t_1, t_2, \dots, t_p\}$ and $b = P - a$.

$S2 \equiv$ This cycle is represented as $A_0A_2A_1$. Initially, only the second machine is loaded and the robot is in front of the input buffer. Then the robot sequences these activities and returns to the input buffer. The cycle time of $S2$ as given in [12] is

$$4\varepsilon + 4\delta + max\{2\varepsilon + 4\delta, a, b\}$$

or equivalently

$$6\varepsilon + 8\delta + max\{0, a - (2\varepsilon + 4\delta), b - (2\varepsilon + 4\delta)\},$$

where a and b are as defined above.

Now let us consider an example which will shed light on our assumptions of process and operational flexibility.

Example. Assume that we have five operations to be completed with corresponding operation times, $t_1 = 13$, $t_2 = 17$, $t_3 = 10$, $t_4 = 5$ and $t_5 = 5$. So $P = t_1 + t_2 + t_3 + t_4 + t_5 = 50$. Take $\varepsilon = 1$ and $\delta = 2$. If we assumed, as done in the current literature, that there is no process and operational flexibility, we should state that some operations must be processed on the first machine while the remaining ones must be processed on the second machine in all repetitions of the production cycle. For such a case, considering all allocation possibilities of these operations to two machines, allocating operation 1 and 3 to the first (second) machine and the remaining ones to the second (first) machine gives the minimum cycle time using $S2$. Thus, let us assume that $a = t_1 + t_3 = 23$ and $b = t_2 + t_4 + t_5 = 27$. In such a case, the cycle time would be:

$$T = 6 \times 1 + 8 \times 2 + max\{0, (23 - 10), (27 - 10)\} = 39.$$

However, if we consider the process and operational flexibility of the machines, then the allocation of these operations to the machines need not be the same in all repetitions of the robot move cycle. Consider repetitions of cycle $S2$. Assume that in the first repetition of the cycle we allocated the operations as before so that $a = 23$ and $b = 27$ and in the second repetition we did just the opposite. Consider allocating tasks 1 and 3 to the second machine instead of the first and the remaining ones to the first machine. Thus, now we have $a = 27$ and $b = 23$. Assume that we use these two different

allocations of operations alternatingly. Then, the long run average cycle time for $S2$ becomes:

$$T = \frac{12 \times 1 + 16 \times 2 + max\{0,(23-10),(23-10)\} + max\{0,(27-10),(27-10)\}}{2} = 37$$

which is less than 39. The Gantt Charts shown in Fig. 2 helps comparing the standard and the operational-assignment cases. The bold dashed lines show the end of the first repetition and start of the second repetition of the $S2$ cycle. As it is observed from the Gantt Chart for the standard case, the first and second repetitions of the $S2$ cycle are identical. The first repetition ends at time 39 and the second repetition ends at time $39 \times 2 = 78$. However, two repetitions are not identical for the operational assignment case. In particular, the first repetition takes 35 time units while the second repetition takes 39 time units. In this example, the total processing times of each machine are the same in both cases. By allowing different allocations in each repetition, we can decrease the total idle time of the first machine from 14 to 10 and the robot from 17 to 15 time units, respectively, in each cycle. As a result, the long run average cycle time is decreased by 2 time units, which corresponds to $\frac{2}{39} = 5.1\%$ increase in the throughput. This example clearly shows that fixing the processing times on each CNC machine is a major hindrance limiting the better of the alternatives.

In the current practice, the processing time of every part on every machine has to be known in advance so as to calculate the cycle times. Since, in this study, we considered processing times as decision variables, in order to find the cycle times we need to know the allocation of operations to the two machines. The following methodology will help us in finding the cycle times of the robot move cycles.

As the above example suggests, with process and operational flexibility, the processing times on the first and the second machines may differ from repetition to repetition of the robot move cycle. It is not known how many such different repetitions are required to get the minimum cycle time for each robot move cycle. Thus, let us assume that we have k such different repetitions, called k-allocation type cyclic production. A k-allocation type cyclic production is simply a repetition of identical k-unit move cycles. In each k-unit move cycle, each one of the k parts has a differing processing time on the first and hence the second machine. Consider any one of these repeated k-unit move cycles. There are exactly k parts produced. More formally, let $O = \{1,\ldots,p\}$ be the index set of the operations of each part to be produced. For the processing of the jth part within any k-unit move cycle, where $j \leqslant k$, assume that the operation set is partitioned into $O^j \subseteq O$ and $O - O^j$ where O^j $(O - O^j)$ is the set of operations that is allocated to the first (second) machine for this part. Thus, the processing time of the jth part on the first machine is $P_{kj} = \sum_{i \in O^j} t_i$. Fig. 3 schematizes these concepts.

One can easily conclude that, since we have p operations to be allocated for each part type and the order of the operations is not important, there are exactly 2^p different possible allocation types.

More formally we can define the k-allocation type cyclic production which will be used to find the long run average cycle time of the robot move cycles as follows:

Definition 3. We say that a given robot move cycle uses *k-allocation types* if there is a total of k distinct partitions of the operations to the two machines. In a cyclic manner, the first k parts in the queue have differing allocation types and after the kth one the cycle returns to the beginning and the next k parts have the same allocation types as the previous k.

According to this definition we can conclude that the problems considered in the literature so far are robotic cell scheduling problems with one-allocation type move cycles where the partition of operations is fixed.

We shall now proceed to present the cycle times of 1-unit robot move cycles $S1$ and $S2$ in our notation. Let $T_{S1(k)}$ be the long run average cycle time (to produce one part) of $S1$ using k-allocation types. Let $T_{S2(k)}$ be defined in a similar fashion for move cycle $S2$. Then, with our notation, the cycle time of robot move cycle $S1$ found by Sethi et al. [12] becomes:

$$T_{S1(1)} = 6\varepsilon + 6\delta + P.$$

However, since the cycle time of $S1$ does not depend on P_{kj} but instead on P, we conclude that there is no allocation problem for $S1$ and $T_{S1(k)} = T_{S1(1)}$ $\forall k > 1$ as well.

The cycle time of $S2$ provided by Sethi et al. [12] is for one-allocation type and with our notation it becomes:

$$T_{S2(1)} = 6\varepsilon + 8\delta + max\{0, P_{11} - (2\varepsilon + 4\delta), (P - P_{11}) - (2\varepsilon + 4\delta)\}.$$

For two-allocation types, the cycle time is computed as follows:

Let w_{ij} be defined as the waiting time of the robot in front of machine j for the part which has ith allocation type to be processed. Since we have two-allocation types, it is enough to consider the first and the second part production only. Initially, the robot is in front of the input buffer just starting to unload part 2; machine 1 (M1) is empty and machine 2 (M2) is loaded with part 1 (the start of the first repetition of $S2$). Robot unloads a part from the input buffer, transfers to

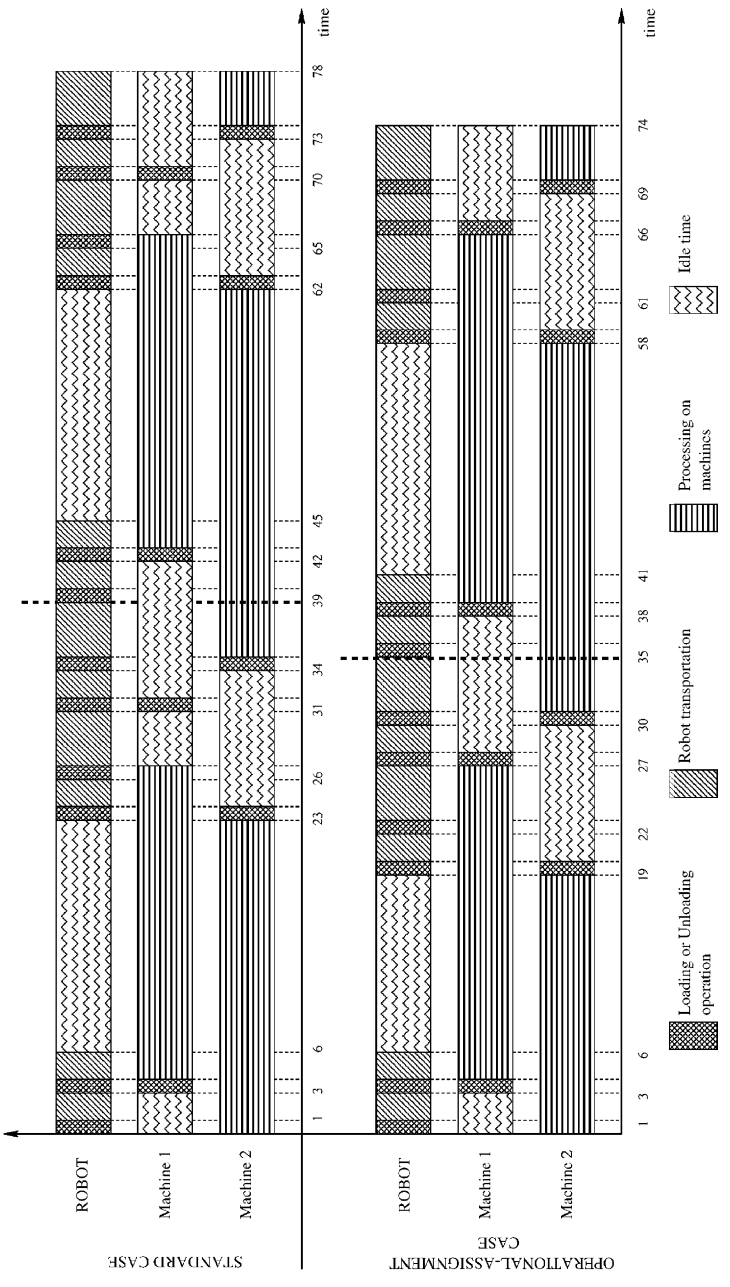

Fig. 2. Gantt chart for the standard and operational-assignment cases.

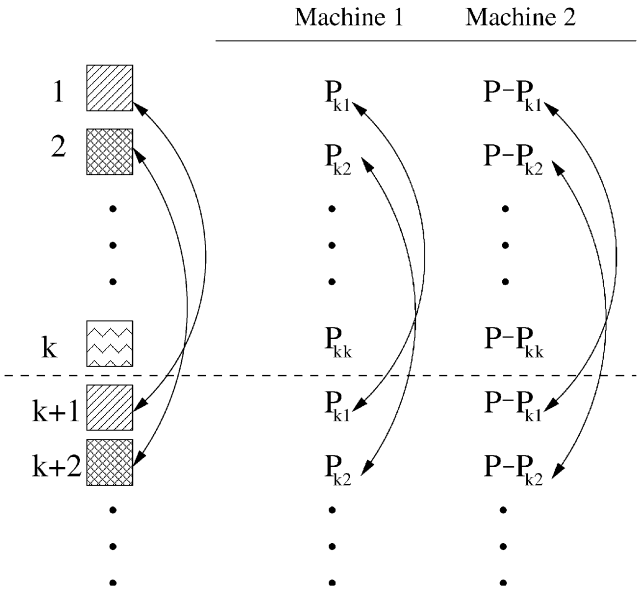

Fig. 3. Different allocation of k parts to the machines.

M1 and loads it $(\varepsilon+\delta+\varepsilon)$, moves to M2 (δ), waits if necessary for the machine to finish the operation (w_{12}), unloads M2, transfers the part to output buffer, drops the part and returns back to M1 $(\varepsilon+\delta+\varepsilon+2\delta)$, waits if necessary (w_{21}), unloads the part from M1, transfers it to M2 and loads M2 $(\varepsilon+\delta+\varepsilon)$, and returns back to input buffer (2δ) (the end of the first repetition and the start of the second repetition of $S2$). The operations that the robot make in the second repetition are exactly the same as the first one, the only difference being in the processing times of the machines. Thus, we have w_{11} instead of w_{21} and w_{22} instead of w_{12} in the second repetition. The average cycle time for two-allocation $S2$ is simply found by summing up these values and dividing by 2:

$$\frac{12\varepsilon+16\delta+w_{11}+w_{12}+w_{21}+w_{22}}{2},$$

where

$$w_{11}=max\{0,P_{21}-(2\varepsilon+4\delta+w_{22})\},$$

$$w_{22}=max\{0,P-P_{22}-(2\varepsilon+4\delta)\},$$

$$w_{21}=max\{0,P_{22}-(2\varepsilon+4\delta+w_{12})\},$$

$$w_{12}=max\{0,P-P_{21}-(2\varepsilon+4\delta)\}.$$

Then,

$$w_{11}+w_{22}=max\{0,(P-P_{22})-(2\varepsilon+4\delta),P_{21}-(2\varepsilon+4\delta)\}, \quad \text{and}$$

$$w_{12}+w_{21}=max\{0,(P-P_{21})-(2\varepsilon+4\delta),P_{22}-(2\varepsilon+4\delta)\}.$$

Thus, the cycle time for two-allocation $S2$ is

$$T_{S2(2)}=6\varepsilon+8\delta+(max\{0,(P-P_{22})-(2\varepsilon+4\delta),P_{21}-(2\varepsilon+4\delta)\})/2 +(max\{0,(P-P_{21})-(2\varepsilon+4\delta),P_{22}-(2\varepsilon+4\delta)\})/2. \quad (1)$$

Let us now proceed to find the cycle time for k-allocation type $S2$. Consider the lth repetition of the robot move cycle within a k-allocation cyclic production $S2$, where $l\leqslant k$. Initially, the second machine is loaded with a part which has $(l-1)$th allocation type. The robot activity sequence is $A_0A_2A_1$. Thus, the cycle time can be found as: $6\varepsilon+8\delta+w_{(l-1)2}+w_{l1}$ where

$$w_{(l-1)2}=max\{0,(P-P_{k(l-1)})-(2\varepsilon+4\delta)\} \quad \text{and} \quad w_{l1}=max\{0,P_{kl}-(2\varepsilon+4\delta+w_{(l-1)2})\}.$$

After a few manipulations the cycle time for the lth repetition can be simplified as:

$$6\varepsilon + 8\delta + max\{0, (P - P_{k(l-1)}) - (2\varepsilon + 4\delta), P_{kl} - (2\varepsilon + 4\delta)\}.$$

If $l = 1$, then $(l-1)$th allocation is the kth allocation type due to the cyclic nature of cycles. Thus, if we add up the cycle times for all repetitions of the cycles in a k-allocation type $S2$, the total time to complete all cycles becomes:

$$\begin{aligned} 6(k)\varepsilon + 8(k)\delta &+ max\{0, P_{k1} - (2\varepsilon + 4\delta), (P - P_{kk}) - (2\varepsilon + 4\delta)\} \\ &+ max\{0, P_{k2} - (2\varepsilon + 4\delta), (P - P_{k1}) - (2\varepsilon + 4\delta)\} \\ &+ max\{0, P_{k3} - (2\varepsilon + 4\delta), (P - P_{k2}) - (2\varepsilon + 4\delta)\} \\ &\vdots \\ &+ max\{0, P_{kk} - (2\varepsilon + 4\delta), (P - P_{k(k-1)}) - (2\varepsilon + 4\delta)\}. \end{aligned} \tag{2}$$

Then we get the cycle time to produce one part by dividing this cycle time by k:

$$\begin{aligned} T_{S2(k)} = 6\varepsilon + 8\delta &+ (\tfrac{1}{k})\, max\{0, P_{k1} - (2\varepsilon + 4\delta), (P - P_{kk}) - (2\varepsilon + 4\delta)\} \\ &+ (\tfrac{1}{k})\, max\{0, P_{k2} - (2\varepsilon + 4\delta), (P - P_{k1}) - (2\varepsilon + 4\delta)\} \\ &+ (\tfrac{1}{k})\, max\{0, P_{k3} - (2\varepsilon + 4\delta), (P - P_{k2}) - (2\varepsilon + 4\delta)\} \\ &\vdots \\ &+ (\tfrac{1}{k})\, max\{0, P_{kk} - (2\varepsilon + 4\delta), (P - P_{k(k-1)}) - (2\varepsilon + 4\delta)\}. \end{aligned} \tag{3}$$

In addition to the 1-unit robot move cycles, $S1$ and $S2$, Hall et al. [8] define two more robot move sequences. They are not feasible robot move cycles by themselves, since, the starting and the ending states depend on the preceding and the following robot movements and unless we know the preceding and the following robot movements, we cannot represent these cycles by the help of robot activities.

$S_{12} \equiv$ The transition movement of the robot from performing cycle $S1$ to $S2$. The robot uses cycle $S1$ during processing of part i on the first machine, and cycle $S2$ during processing on the second machine.

$S_{21} \equiv$ The transition movement of the robot from performing cycle $S2$ to $S1$, in which, the robot uses cycle $S2$ during processing of part i on the first machine, and cycle $S1$ during processing on the second machine.

Hall et al. [8] show that we can represent any robot move cycle by the four robot move sequences: $S1$, $S2$, S_{12} and S_{21}. For example, a 3-unit robot move cycle can be represented as $S1S_{12}S_{21}$ and in terms of the robot activities this is equivalent to the sequence $A_0A_1A_2A_0A_1A_0A_2A_1A_2$. The same cycle can also be represented as $S_{12}S_{21}S1$. The difference is the starting and the ending points of the cycle but both give the same cycle time. However, we are not entirely free in representing robot move cycles by these four sequences. For example, $S1$ cannot be followed by $S2$ since at the end of cycle $S1$, both machines are empty and the robot is in front of the input buffer in order to take a part and load machine 1. If $S1$ is followed by $S2$ then, while performing $S2$ the robot will try to unload machine 2 which is already empty and this violates the basic assumption of Crama et al. [6]. We can represent the set of all allowable moves by the transition graph shown in Fig. 4. Thus, we conclude that, $S1$ can be followed by either $S1$ or S_{12}, $S2$ can be followed by either $S2$ or S_{21}, S_{12} can be followed by either S_{21} or $S2$ and lastly, S_{21} can be followed by either S_{12} or $S1$.

Lemma 4 (Hall et al. [8]). *In any feasible robot move cycle the number of S_{12} sequences is equal to the number of S_{21} sequences.*

Proof. In Fig. 3, we see that robot move sequence S_{12} must be preceded by $S1$ or S_{21}, and followed by $S2$ or S_{21}. Since the schedule is cyclic, it follows that the number of transitions from $S1$ into $S2$ (using S_{12}) must be equal to the number of transitions from $S2$ into $S1$ (using S_{21}). □

Before proceeding with the next lemma, let us note that though neither S_{12} nor S_{21} are complete cycles, both sequences $S_{12}S_{21}$ and $S_{21}S_{12}$ are 2-unit complete robot move cycles resulting from the robot activity sequence $A_0A_1A_0A_2A_1A_2$.

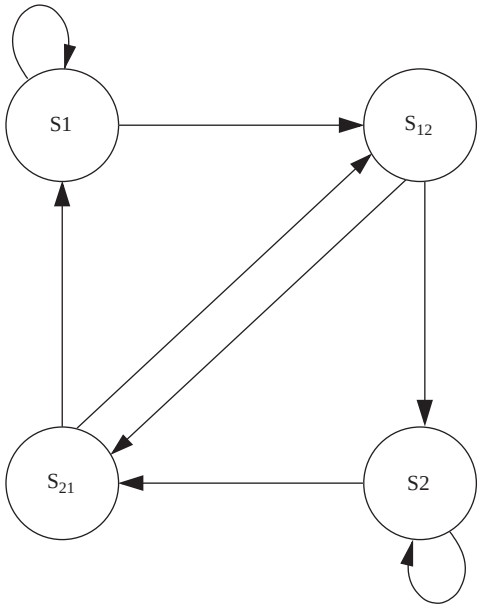

Fig. 4. Transition graph.

Lemma 5. *The cycle time of any n-unit robot move cycle is the summation of the cycle times of three robot move cycles* S_1, S_2 *and* $S_{12}S_{21}$ *in corresponding numbers that form the n-unit robot move cycle.*

Proof. As previously stated, any n-unit robot move cycle is a repetitive sequence of $S1$, $S2$, S_{12} and S_{21}. Let us first analyze $S1$. The four possible sequences of $S1$ with the preceding and following robot move cycles with corresponding activity sequences are:

$$S1S1S1 : \ldots A_2(A_0A_1A_2)A_0 \ldots,$$

$$S1S1S_{12} : \ldots A_2(A_0A_1A_2)A_0 \ldots,$$

$$S_{21}S1S1 : \ldots A_2(A_0A_1A_2)A_0 \ldots,$$

$$S_{21}S1S_{12} : \ldots A_2(A_0A_1A_2)A_0 \ldots .$$

In all the cases the preceding activity of cycle $S1$ is A_2 and the following one is A_0. Then the initial and the final states of the system before and after performing cycle $S1$ are the same. Thus, if we remove the activities of $S1$, $(A_0A_1A_2)$, the remaining sequence will still correspond to a valid cycle. So we can remove the activities of all $S1$ cycles and consider the cycle time of the remaining activities and add the total cycle time of the removed $S1$ cycles at the end to find the cycle time of the n-unit robot move cycle.

By following a similar sequence of arguments, the same result can be achieved for cycle $S2$ also. Thus, we can remove cycles $S1$ and $S2$, consider the rest, and add up the cycle times of the removed cycles at the end. After we remove cycles $S1$ and $S2$, we are left with an alternating sequence of S_{12} and S_{21}. Then the cycle time of the n-unit robot move cycle can be found by the summation of the cycle times of $S1$, $S2$ and $S_{12}S_{21}$ cycles that form the n-unit robot move cycle. □

Lemma 6. *At least one of the cycles* $S1$, $S2$ *or* $S_{12}S_{21}$ *has a cycle time that is less than or equal to the cycle time of any given n-unit robot move cycle.*

Proof. Let us consider any n-unit robot move cycle. This cycle can be expressed as a vector (b_1, b_2, b_3), where b_1 (respectively, b_2, b_3) is the number of $S1$ (respectively, $S2$, $S_{12}S_{21}$) cycles used to form the n-unit robot move cycle. Note that $S1$ and $S2$ are 1-unit robot move cycles and $S_{12}S_{21}$ is a 2-unit robot move cycle. Suppose that T^n is the total cycle time of the n-unit robot move cycle to produce n parts. So the average cycle time to produce one part with this

robot move cycle is (T^n/n). Using Lemma 5 we have:

$$\frac{T^n}{n} = \frac{(b_1)(T_{S1(k)}) + (b_2)(T_{S2(k)}) + (b_3)(T_{S_{12}S_{21}(k)})}{b_1 + b_2 + b_3},$$

where $T_{S_{12}S_{21}(k)}$ is the long run average cycle time to produce one part with $S_{12}S_{21}$ using k-allocation types. From this equation we conclude that, the cycle time of any n-unit robot move cycle is a convex combination of the cycle times of $S1$, $S2$ and $S_{12}S_{21}$. Thus, the cycle time of any n-unit robot move cycle cannot be less than $min\{T_{S1(k)}, T_{S2(k)}, T_{S_{12}S_{21}(k)}\}$. □

Lemmas 5 and 6 together state that the solution to IRC2 $|\, k = 1, \delta_i = \delta, \varepsilon_i = \varepsilon \,|\, T$ is one of the three robot move cycles: $S1$, $S2$ or $S_{12}S_{21}$. In Theorem 10, we will extend this result and present regions of optimality for each of these three cycles. Prior to that, we shall prove Theorems 7 and 9 together which will help us in finding the optimal number of allocation types to be used along with $S2$ and $S_{12}S_{21}$ cycles, respectively.

3. Solution procedure for operation allocation problem

With Theorems 7 and 9 we shall find the optimal number of allocation types for robot move cycles $S2$ and $S_{12}S_{21}$. The main result of the paper will be provided with Theorem 10 stating that the optimal solution to the problem of robotic cell scheduling with operational flexibility is one of the three robot move cycles and this theorem will provide intervals where each of the robot move cycles are optimal. Also in this section we show that finding the optimal allocation of operations with even a fixed number of allocation types is not an easy task.

Before proceeding with Theorem 7 which finds the optimal number of allocation types to be used with robot move cycle $S2$, let us present an observation about the optimal allocations to be used with $S2$.

Let us recall that:

$$\begin{aligned} T_{S2(k)} = 6\varepsilon + 8\delta &+ (\tfrac{1}{k})\, max\{0, P_{k1} - (2\varepsilon + 4\delta), (P - P_{kk}) - (2\varepsilon + 4\delta)\} \\ &+ (\tfrac{1}{k})\, max\{0, P_{k2} - (2\varepsilon + 4\delta), (P - P_{k1}) - (2\varepsilon + 4\delta)\} \\ &+ (\tfrac{1}{k})\, max\{0, P_{k3} - (2\varepsilon + 4\delta), (P - P_{k2}) - (2\varepsilon + 4\delta)\} \\ &\qquad\vdots \\ &+ (\tfrac{1}{k})\, max\{0, P_{kk} - (2\varepsilon + 4\delta), (P - P_{k(k-1)}) - (2\varepsilon + 4\delta)\}. \end{aligned} \tag{4}$$

Define Max_i as the ith max term in this equation. In the ideal situation, it is possible to allocate the operations to the two machines such that the sum of the operation times that are allocated to the first machine is equal to the sum of the operation times allocated to second machine. In other words, we can assign $P_{ki} = P/2,\ \forall i \in [1 \ldots k]$. In this case, independent of the number of allocation types used, this partitioning gives the minimum cycle time which simplifies to:

$$T_{S2(k)} = \begin{cases} 6\varepsilon + 8\delta & \text{if } \dfrac{P}{2} \leqslant 2\varepsilon + 4\delta, \\ 4\varepsilon + 4\delta + \dfrac{P}{2} & \text{if } \dfrac{P}{2} \geqslant 2\varepsilon + 4\delta. \end{cases}$$

If such a partitioning is not attainable, another situation in which the cycle time does not depend on the number of allocation types occurs when we can partition the tasks so that each machine has a total operation time of at most $2\varepsilon + 4\delta$, i.e. it is possible to have $P_{ki} \leqslant 2\varepsilon + 4\delta$ and $P - P_{ki} \leqslant 2\varepsilon + 4\delta$ for all i, then each Max_i term vanishes in (4). Thus for this case also, independent of the number of allocation types the cycle time is

$$T_{S2(k)} = 6\varepsilon + 8\delta.$$

If the previously mentioned situations are not attainable, we have:

Theorem 7. *Under the robot move cycle $S2$, using two allocation types is better than using any number of allocation types. That is $T_{S2(2)} \leqslant T_{S2(k)}\ \forall k \neq 2$.*

Proof. Observe that since in Eq. (4) P_{ki} appears in Max_i while $P - P_{ki}$ appears in Max_{i+1}, there is a cross relation between Max_i and Max_{i+1} which will be used frequently in the following case analysis.

We need to analyze the following sub-cases comparing each partition against the value of $2\varepsilon + 4\delta$.

1. If there is a possible partition such that both $P_{ki} \geqslant 2\varepsilon + 4\delta$ and $P - P_{ki} \geqslant 2\varepsilon + 4\delta$ for all $i \in [1 \dots k]$, then analyzing the cross relation between Max_i and Max_{i+1} in (4), the best allocation would simply be to set $P_{ki} = P - P_{k(i-1)}\ \forall i \in [2 \dots k]$ and $P_{k1} = P - P_{kk}$. The reasoning is simple. Let us consider Max_i and Max_{i+1} and assume the opposite so that $P_{ki} \neq P - P_{k(i-1)}$. Assume w.l.o.g that $P_{ki} \geqslant P - P_{k(i-1)}$. Also we assumed at the beginning that $P_{ki} \neq P - P_{ki}$. Then, either $P_{ki} \geqslant P/2$ or $P - P_{ki} \geqslant P/2$. Assume w.l.o.g that $P_{ki} \geqslant P/2$. Then, $Max_i = P_{ki} - (2\varepsilon + 4\delta)$ and Max_{i+1}, depending on $P_{k(i+1)}$, is at least $P - P_{ki} - (2\varepsilon + 4\delta)$. Thus $Max_i + Max_{i+1} \geqslant P - (2\varepsilon + 4\delta)$. But on the other hand, by setting $P_{ki} = P - P_{k(i-1)}$ we have $Max_i + Max_{i+1} = P - (2\varepsilon + 4\delta)$. Summing up for all i and dividing by k to find the cycle time to produce one part does not change the result. This completes the proof.
   Note that this is readily attained by letting the set $(O - O^i)$ be equal to the set O^{i+1}. But then in Max_{i+2} we have $P - P_{k(i+1)}$ and $P_{k(i+2)}$ and minimum is attained when $P_{k(i+2)} = P - P_{k(i+1)} = P - P + P_{ki} = P_{ki}$. Thus the minimum cycle time is attained when $P_{ki} = P_{k(i+2)}\ \forall i$ and $P_{k(i+1)} = P - P_{ki}\ \forall i = 0, \dots, k-1$ where $P_{k0} = P_{kk}$. This is nothing but using a two-allocation $S2$ and the long run average cycle time to produce one part with this allocation can be found as follows:
   Let $P_{ki} = \alpha$ for $i = 1, 3, 5, \dots, k-1$. Therefore, $P_{k2} = P_{k4} = \cdots = P_{kk} = P - \alpha$. In other words, what we have is actually a two-allocation type cycle with $P_{21} = \alpha$ and $P_{22} = P - \alpha$.

$$Max_i = \begin{cases} \alpha - (2\varepsilon + 4\delta) & \text{for } i \text{ odd}, \\ (P - \alpha) - (2\varepsilon + 4\delta) & \text{for } i \text{ even}, \end{cases} \quad 1 \leqslant i \leqslant k.$$

   The total time to produce k parts is

$$6(k)\varepsilon + 8(k)\delta + \sum_{i=1}^{k/2} (\alpha - (2\varepsilon + 4\delta)) + \sum_{i=1}^{k/2} (P - \alpha - (2\varepsilon + 4\delta))$$

$$= 6(k)\varepsilon + 8(k)\delta + \frac{k}{2}(\alpha - (2\varepsilon + 4\delta)) + \frac{k}{2}(P - \alpha - (2\varepsilon + 4\delta)).$$

   Dividing by k to find the cycle time to produce one part we obtain:

$$T_{S2(k)} = T_{S2(2)} = 6\varepsilon + 8\delta + \frac{P - (4\varepsilon + 8\delta)}{2}.$$

2. Else if the only possibility is one partition having a total operation time of at least $2\varepsilon + 4\delta$ and the other one at most $2\varepsilon + 4\delta$, a similar analysis yields the same allocation as in the previous case optimum, i.e., $P_{ki} = P - P_{k(i-1)}\ \forall i \in [2 \dots k]$ and $P_{k1} = P - P_{kk}$. In order to find the cycle time again let $P_{ki} = \alpha$, for $i = 1, 3, \dots, k-1$ and $P_{k2} = P_{k4} = \cdots = P_{kk} = P - \alpha$. This is actually two-allocation type $S2$ with $P_{21} = \alpha$ and $P_{22} = P - \alpha$. Assume without loss of generality that $\alpha \geqslant 2\varepsilon + 4\delta$ and $(P - \alpha) \leqslant 2\varepsilon + 4\delta$, then each max term in (2) becomes:

$$Max_i = \begin{cases} \alpha - (2\varepsilon + 4\delta) & \text{for } i \text{ odd}, \\ 0 & \text{for } i \text{ even}, \end{cases} \quad 1 \leqslant i \leqslant k.$$

   The total time to produce k parts is $6(k)\varepsilon + 8(k)\delta + \sum_{i=1}^{k/2} (\alpha - (2\varepsilon + 4\delta))$. Dividing by k we obtain:

$$T_{S2(k)} = T_{S2(2)} = 6\varepsilon + 8\delta + \frac{\alpha - (2\varepsilon + 4\delta)}{2}. \qquad \square$$

Though, we manage to find the optimal allocation type in a cyclic production $S2$ using Theorem 7, finding the optimal allocation of operations to the two machines for $S2$ is not an easy job even when there is a fixed allocation type. More formally, we shall now show that the following decision problem is NP-complete.

S2 Operation Allocation for One-Allocation Type Decision Problem (Problem S2TAP).

Instance: A set of operations O such that $|O| = p$ with respective integer processing times $\{t_1, \dots, t_p\}$, loading/unloading time ε, transportation time δ, $P = \sum t_i$ and a real number K.

Question: Can we allocate operations to the two machines in such a way that the long run average cycle time $T_{S2(1)} \leqslant K$?

Theorem 8. *Problem S2TAP is* NP*-complete.*

Proof. S2TAP is in NP since whenever we are given a specific allocation of operations, we can readily find the corresponding long run average cycle time and hence decide if it is less than or equal to K.

We will show that S2TAP is NP-complete by reducing the 2-Partition problem to it. As a reminder the 2-Partition problem can be stated as (see [7]):

Instance: Given $A=\{a_1,\ldots,a_r\}$, a_i integer and $s(a_i)\in Z^+$ size of item i.

Question: Is there a partition of A into A' and $A\setminus A'(A'\subseteq A)$ such that $\sum_{i\in A'} s(i)=\sum_{i\in A\setminus A'} s(i)$?

Say we have an arbitrary instance of 2-Partition. From this we are going to construct a specific instance of S2TAP and show that S2TAP has a solution if and only if 2-Partition instance has a solution. Let $A=a_1,\ldots,a_r, s(a_i)\ i\in[1,\ldots,r]$ be the given instance of 2-Partition. We shall have r operations in our set O each corresponding to an item from the given set A. Each t_i will have processing time $s(a_i)$. Let $\varepsilon=(\sum_{i\in[1,\ldots,r]} s(a_i))/8$ and $\delta=(\sum_{i\in[1,\ldots,r]} s(a_i))/16$.

Claim. *S2TAP has a solution with these specifications and* $K=\frac{7}{4}\sum_{i\in[1,\ldots,r]} s(a_i)\Leftrightarrow$*2-Partition instance has a yes answer.*

For one-allocation type, $T_{S2(1)}=6\varepsilon+8\delta+max\{0,P_{11}-(2\varepsilon+4\delta),P-P_{11}-(2\varepsilon+4\delta)\}$ and $T_{S2(1)}\geqslant 6\varepsilon+8\delta=\frac{7}{4}\sum_{i\in[1,\ldots,r]} s(a_i)$. With given ε and δ values, the cycle time for a one-allocation $S2$ becomes $T_{S2(1)}=6\varepsilon+8\delta+max\{0,P_{11}-P/2,P-P_{11}-P/2\}$. Thus, the minimum cycle time $7/4P$ is attained if and only if $P_{11}=P-P_{11}=P/2$, if and only if 2-Partition has a yes answer. □

Now we can give the best allocation type for the robot move cycle, $S_{12}S_{21}$.

Theorem 9. *For 2-unit robot move cycle* $S_{12}S_{21}$ *using two allocation types is better than using any number of allocation types.*

Proof. The cycle time derivations of $S_{12}S_{21}$ for one and two allocation types and the proof of equality of the optimal cycle time for a k-allocation type $S_{12}S_{21}$ with the optimal cycle time for a two-allocation type $S_{12}S_{21}$ is given in the appendix. Then, in order to prove this theorem we will compare the cycle times of one-allocation type $S_{12}S_{21}$ with the cycle time of two-allocation type $S_{12}S_{21}$. Eq. (A.1) in the appendix gives the cycle time for one-allocation type $S_{12}S_{21}$. For this assume without loss of generality that $P_{11}\geqslant(P-P_{11})$. So we have two cases:

1. If the operations can be partitioned such that $P_{11}\leqslant 2\varepsilon+4\delta$ then the cycle time is $T_{S_{12}S_{21}(1)}=6\varepsilon+7\delta+P/2$.
2. Else, if $P_{11}\geqslant 2\varepsilon+4\delta$ then the cycle time is

$$T_{S_{12}S_{21}(1)}=6\varepsilon+7\delta+P/2+\frac{P_{11}-(2\varepsilon+4\delta)}{2}.$$

Note that this cycle time is at least $6\varepsilon+7\delta+P/2$.

For two-allocation type $S_{12}S_{21}$, the cycle time is given in Eq. (A.2) in the appendix. The trivial solution for the allocation problem is found by letting $P_{21}=(P-P_{22})=P$ thus, $(P-P_{21})=P_{22}=0$. In words, in the first cycle all of the operations are allocated to the first machine and in the second cycle all of the operations are allocated to the second machine. The cycle time for this solution is

$$T_{S_{12}S_{21}(2)}=\begin{cases}6\varepsilon+7\delta, & P\leqslant 2\varepsilon+4\delta,\\ 5\varepsilon+5\delta+P/2, & P\geqslant 2\varepsilon+4\delta.\end{cases}$$

Comparing using one-allocation type with using two-allocation types, one can easily verify that $T_{S_{12}S_{21}(1)}\geqslant 6\varepsilon+7\delta+P/2$ and $T_{S_{12}S_{21}(2)}\leqslant 6\varepsilon+7\delta+P/2$. Thus, we say that using two-allocation type always gives better results than using one-allocation type. □

Now we can present the main results of this paper, but first recall that, in Theorem 7 we achieved the optimal number of allocation types for $S2$ as two and the optimal allocation was reached by letting $P_{21}=P-P_{22}$ and $P-P_{21}=P_{22}$.

Theorem 10. *For* $IRC2\,|\,k=1,\delta_i=\delta,\varepsilon_i=\varepsilon\,|\,T$ *with the assumption of operational flexibility, either* $S1$ *or* $S2$ *or* $S_{12}S_{21}$ *gives the minimum cycle time among the robot move cycles that are reported in the literature and the following holds*:

1. *If* $0\leqslant P\leqslant\delta$ *then* $S1$ *gives the minimum cycle time*
2. *If* $\delta\leqslant P\leqslant 2\varepsilon+6\delta$ *then* $S_{12}S_{21}$ *gives the minimum cycle time*
3. *If* $P\geqslant 2\varepsilon+6\delta$ *then*

3.1. *If* $(P_{21} \geqslant 2\varepsilon + 4\delta) \wedge ((P - P_{21}) \leqslant 2\varepsilon + 4\delta)$ *then*
  3.1.1. *If* $P \leqslant P_{21} + 2\delta$ *then* $S_{12}S_{21}$ *gives the minimum cycle time*
  3.1.2. *Else* $S2$ *gives the minimum cycle time*
3.2. *Else* $S2$ *gives the minimum cycle time.*

Proof. Lemmas 5 and 6 jointly prove that for IRC2 $| k = 1, \delta_i = \delta, \varepsilon_i = \varepsilon | T$ with the assumption of operational flexibility, either $S1$ or $S2$ or $S_{12}S_{21}$ gives the minimum cycle time. Now let us consider each region given in the theorem separately. For $S2$ and $S_{12}S_{21}$, it is proved in Theorems 7 and 9, respectively, that using two allocation types is optimal.

1. If $0 \leqslant P \leqslant \delta$, the cycle times of $S1$, $S2$ and $S_{12}S_{21}$ (to produce one part) are as follows:

$$T_{S1(1)} = 6\varepsilon + 6\delta + P,$$

$$T_{S2(2)} = 6\varepsilon + 8\delta,$$

$$T_{S_{12}S_{21}(2)} = 6\varepsilon + 7\delta.$$

In this region $P \leqslant \delta$, therefore $T_{S1(1)} \leqslant 6\varepsilon + 7\delta$. Thus we conclude that in this region $S1$ gives the minimum cycle time.

2. If $\delta \leqslant P \leqslant 2\varepsilon + 6\delta$ then the cycle times are:

$$T_{S1(1)} = 6\varepsilon + 6\delta + P,$$

$$T_{S2(2)} = \begin{cases} 6\varepsilon + 8\delta & \text{if } (P_{21} \leqslant 2\varepsilon + 4\delta) \wedge ((P - P_{21}) \leqslant 2\varepsilon + 4\delta) \\ 6\varepsilon + 8\delta + P_{21} - (2\varepsilon + 4\delta) & \text{if } P_{21} > 2\varepsilon + 4\delta \end{cases}$$

$$T_{S_{12}S_{21}(2)} = \begin{cases} 6\varepsilon + 7\delta & \text{if } P \leqslant 2\varepsilon + 4\delta, \\ 5\varepsilon + 5\delta + \dfrac{P}{2} & \text{if } P \geqslant 2\varepsilon + 4\delta. \end{cases}$$

A simple comparison reveals that $T_{S_{12}S_{21}(2)}$ is the minimum for this range.

3. If $P \geqslant 2\varepsilon + 6\delta$ then the cycle times are:

$$T_{S1(1)} = 6\varepsilon + 6\delta + P,$$

$$T_{S2(2)} = \begin{cases} 6\varepsilon + 8\delta & \text{if } (P_{21} \leqslant 2\varepsilon + 4\delta) \wedge ((P - P_{21}) \leqslant 2\varepsilon + 4\delta), \\ 4\varepsilon + 4\delta + \dfrac{P}{2} & \text{if } (P_{21} \geqslant 2\varepsilon + 4\delta) \wedge ((P - P_{21}) \geqslant 2\varepsilon + 4\delta), \\ 5\varepsilon + 6\delta + \dfrac{P_{21}}{2} & \text{if } (P_{21} > 2\varepsilon + 4\delta) \wedge ((P - P_{21}) < 2\varepsilon + 4\delta), \end{cases}$$

$$T_{S_{12}S_{21}(2)} = 5\varepsilon + 5\delta + \frac{P}{2}.$$

Thus, in this region, if $(P_{21} > 2\varepsilon + 4\delta) \wedge ((P - P_{21}) < 2\varepsilon + 4\delta)$, $S_{12}S_{21}$ is better than $S2$ if and only if $P \leqslant P_{21} + 2\delta$. For all other cases, S2 is better than $S_{12}S_{21}$.

This completes the proof. □

In the next section, we will present sensitivity analysis on parameters and show how these regions behave for different parameter values ε and δ.

4. Sensitivity analysis

In this section we will try to figure out how the changes in the values of parameters such as loading/unloading time, ε, and transportation time, δ, affect the regions of optimality presented in Theorem 10. The following example is intended to depict the results obtained so far.

Example. Since there is no allocation problem for $S1$ and $S_{12}S_{21}$, their cycle times can be represented graphically as shown in Fig. 5. On the other hand, the best allocation for $S2$ was to let $P_{21} = (P - P_{22})$ and $(P - P_{21}) = P_{22}$.

1. If $P \leqslant 2\varepsilon + 4\delta$, then $(P_{21} \leqslant 2\varepsilon + 4\delta) \wedge ((P - P_{21}) \leqslant 2\varepsilon + 4\delta)$, thus the cycle time is $6\varepsilon + 8\delta$.

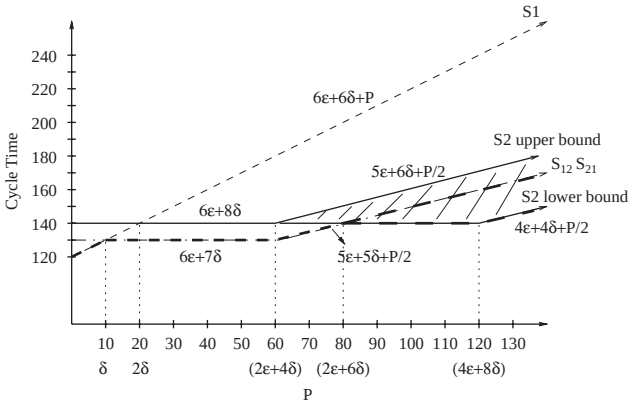

Fig. 5. Change of cycle times according to P.

2. If $P \geqslant 2\varepsilon + 4\delta$, the cycle time of $S2$ depends on the allocation, but we can still find upper and lower bounds for it. In the worst case we have only one task to allocate with processing time P and we cannot divide this into two machines. Thus, the upper bound is obtained by letting $P_{21} = (P - P_{22}) = P$ and thus $P_{22} = (P - P_{21}) = 0$, so the cycle time is $5\varepsilon + 5\delta + P/2$. On the other hand, in the best case the tasks can be partitioned equally, that is $P_{21} = (P - P_{22}) = P/2$ and $P_{22} = (P - P_{21}) = P/2$. Thus, if $P \leqslant 4\varepsilon + 8\delta$ then $T_{S2(2)} = 6\varepsilon + 8\delta$. Otherwise, if $P > 4\varepsilon + 8\delta$ then $T_{S2(2)} = 4\varepsilon + 4\delta + P/2$.

In summary, Fig. 5 shows how the cycle times of the three robot move cycles change with respect to total processing time. The boldly dashed lower envelope shows the optimal solution to the problem. As we can see, for $P \geqslant 2\varepsilon + 4\delta$ the cycle time of $S2$ may take any value between the upper and lower bounds shown as the dashed region depending on the allocation of operations. Thus, the solution changes according to the allocation of operations after the point where the cycle times of $S_{12}S_{21}$ and $S2$ intersect, that is, for $P > 2\varepsilon + 6\delta$ the solution can be anywhere between the two boldly dashed lines in the graph depending on the allocation of the operations. For $P \leqslant 2\varepsilon + 6\delta$, we know the exact solution and there is no allocation problem in this region.

Now let us consider how a change in the value of δ affects these regions. Let us first analyze the case where $\delta = 0$. This can be assumed when the transportation time is negligible with respect to processing times of the operations and loading/unloading times. For this setting the regions of Theorem 10 become:

1. If $0 \leqslant P \leqslant 2\varepsilon$ then $S2$ and $S_{12}S_{21}$ give the same cycle time
2. If $P > 2\varepsilon$ then
   - 2.1. If $(P_{21} \geqslant 2\varepsilon) \wedge ((P - P_{21}) \leqslant 2\varepsilon)$ then
     - 2.1.1. If $P_{21} = P$ then $S2$ and $S_{12}S_{21}$ give the same cycle time
     - 2.1.2. Else $S2$ gives the minimum cycle time
   - 2.2. Else $S2$ gives the minimum cycle time.

Fig. 6 shows the regions of optimality for this case. Thus, we can conclude that, although in some regions $S2$ and $S_{12}S_{21}$ give the same cycle time, $S2$ gives the minimum cycle time in all cases. Also we may state that the cycle time of $S_{12}S_{21}$ constitutes an upper bound for the cycle time of $S2$. These results are quite logical since in cycle $S2$, instead of waiting in front of the machine to finish the operation of a part, the robot makes other operations and tries to fill this waiting time. Transportation times constitute a great deal of these operations and if they are small enough, the ability to balance the time of these extra operations with the processing time on the machine increases. On the other hand, in cycle $S1$ robot always waits in front of the machines. Thus, the total amount of transportation time of $S1$ is less than $S2$ but on the other hand, the waiting time in front of the machines is higher than $S2$. Cycle $S_{12}S_{21}$ is in between $S1$ and $S2$ in these perspectives. Thus, increasing the transportation time is in favor of $S1$ and $S_{12}S_{21}$ since in that case the robot is more likely to wait in front of the machine rather than go and make other operations.

Now let us analyze the loading/unloading time ε. Since in all of the cycles, there is equivalent loading and unloading operations, any change in ε does not change the general view of the graph in Fig. 5. However, one can easily observe that when ε is increased, this is in favor of $S_{12}S_{21}$ since it gives the minimum cycle time for $\delta \leqslant P \leqslant 2\varepsilon + 6\delta$ and as ε increases this region also increases.

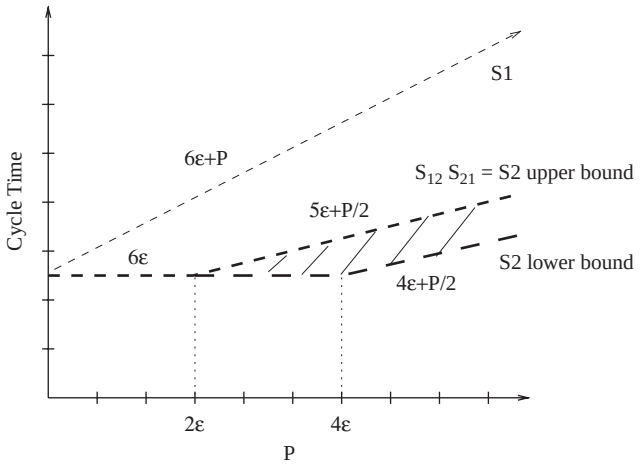

Fig. 6. Case for negligible transportation times.

5. Conclusion

In this paper, we considered the two machine identical parts robotic cell scheduling problem with operational flexibility. With this definition of the problem, each part has a number of operations to be processed and the problem is to allocate these operations to the machines and is to find the corresponding robot move cycle that jointly minimize the cycle time. The main result of the paper is given in Theorem 10 in which we proved that the optimal robot move cycle is not necessarily a 1-unit cycle as Sethi et al. [12] prescribes, but a 2-unit robot move cycle may also be optimal for some parameter inputs and we provided the regions of optimality for each robot move cycle. We also made sensitivity analysis on parameters of the problem such as loading/unloading time and transportation time in order to show how these regions of optimality change.

Possible topics for future research include the extension of this study to three machine robotic cells. Proving or disproving the validity of the conjecture about 1-unit cycles being optimal in three machine case with operational flexibility is important. Another topic is to consider the technological restrictions of CNC machines such as limited tool magazine sizes. Thus, a single CNC machine may not be capable of performing all of the required operations due to unavailability of the necessary cutting tools on the tool magazine. Consequently, certain operations must be processed on a specific CNC machine while others can be processed on both machines.

Appendix

Here we will make the cycle time derivations of the 2-unit robot move cycle $S_{12}S_{21}$.

The long run average cycle time for one-allocation type $S_{12}S_{21}$ can be derived as:

$$T_{S_{12}S_{21}(1)} = \frac{12\varepsilon + 14\delta + P_{11} + (P - P_{11}) + w_{11} + w_{12}}{2},$$

where $w_{11} = max\{0, P_{11} - (2\varepsilon + 4\delta + w_{12})\}$ and $w_{12} = max\{0, (P - P_{11}) - (2\varepsilon + 4\delta)\}$. In other words,

$$T_{S_{12}S_{21}(1)} = \frac{12\varepsilon + 14\delta + P + max\{0, P_{11} - (2\varepsilon + 4\delta), (P - P_{11}) - (2\varepsilon + 4\delta)\}}{2}. \quad (A.1)$$

We divided the cycle time by 2 since $S_{12}S_{21}$ is a 2-unit robot move cycle and one repetition of the cycle produces two parts.

Now consider the case when we have two different allocation types and the parts are processed according to these two types in turn:

$$T_{S_{12}S_{21}(2)} = \frac{12\varepsilon + 14\delta + P_{22} + (P - P_{21}) + w_{11} + w_{22}}{2},$$

where $w_{11} = max\{0, P_{21} - (2\varepsilon + 4\delta + w_{22})\}$ and $w_{22} = max\{0, (P - P_{22}) - (2\varepsilon + 4\delta)\}$. In other words,

$$T_{S_{12}S_{21}(2)} = \frac{12\varepsilon + 14\delta + P_{22} + (P - P_{21}) + max\{0, P_{21} - (2\varepsilon + 4\delta), (P - P_{22}) - (2\varepsilon + 4\delta)\}}{2}. \quad (A.2)$$

Now let us consider the cycle time for k-allocation type $S_{12}S_{21}$. Observe that, in a two-allocation type $S_{12}S_{21}$, initially, the machines are empty and the robot is waiting in front of the input buffer. Since this is a 2-unit cycle, exactly two parts are loaded and unloaded to each machine and at the end, identical to the initial state, the machines are again empty and the robot is waiting in front of the input buffer. Then, in a k-allocation type $S_{12}S_{21}$, this 2-unit cycle is repeated exactly $k/2$ times to produce all k parts with different allocation types. The objective is to find the optimal allocation for these parts to the machines. However, the optimal allocation at each repetition of the cycle must be the same as the optimal allocation of the tasks for a two-allocation type $S_{12}S_{21}$. This is because, a k-allocation type is a concatenation of $k/2$ two-allocation type $S_{12}S_{21}$. Thus, we conclude that the optimal cycle time for a k-allocation type $S_{12}S_{21}$ is the same as the optimal cycle time for a two-allocation type $S_{12}S_{21}$.

References

[1] A. Agnetis, Scheduling no-wait robotic cells with two and three machines, European J. Oper. Res. 123 (2) (2000) 303–314.
[2] N. Brauner, G. Finke, Final results on the one-cycle conjecture in robotic cells, Internal note, Laboratoire LEIBNIZ, Institut IMAG, Grenoble, France, 1997.
[3] J. Browne, J. Harhen, J. Shivnan, Production Management Systems, Addison-Wesley, New York, 1996.
[4] Y. Crama, V. Kats, J. Van de Klundert, E. Levner, Cyclic scheduling in robotic flowshops, Ann. Oper. Res. 96 (2000) 97–124.
[5] Y. Crama, J. Van de Klundert, Cyclic scheduling of identical parts in a robotic cell, Oper. Res. 45 (6) (1997) 952–965.
[6] Y. Crama, J. Van de Klundert, Cyclic scheduling in 3-machine robotic flow shops, J. Scheduling 4 (1999) 35–54.
[7] M.G. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman, New York, 1976.
[8] N.G. Hall, H. Kamoun, C. Sriskandarajah, Scheduling in robotic cells: classification, two and three machine cells, Oper. Res. 45 (3) (1997) 421–439.
[9] J. Hurink, S. Knust, Makespan minimization for flow-shop problems with transportation times and a single robot, Discrete Appl. Math. 112 (2001) 199–216.
[10] V. Kats, E. Levner, Cyclic scheduling in a robotic production line, J. Scheduling 5 (2002) 23–41.
[11] R. Logendran, C. Sriskandarajah, Sequencing of robot activities and parts in two-machine robotic cells, Internat. J. Production Res. 34 (1996) 3447–3463.
[12] S.P. Sethi, C. Sriskandarajah, G. Sorger, J. Blazewicz, W. Kubiak, Sequencing of parts and robot moves in a robotic cell, Internat. J. Flexible Manufacturing Systems 4 (1992) 331–358.
[13] C. Sriskandarajah, N.G. Hall, H. Kamoun, Scheduling large robotic cells without buffers, Ann. Oper. Res. 76 (1998) 287–321.