Mays Business School
Texas A&M University

Research Paper No. 2015-5

Approximations to optimal sequences in single-gripper and dual-gripper robotic cells with circular layouts

H. Neil Geismar
Kyung Sung Jung
Michael Pinedo
Chelliah Sriskandarajah
Approximations to optimal sequences in single-gripper and dual-gripper robotic cells with circular layouts

Kyung Sung Jung\textsuperscript{a}, H. Neil Geismar\textsuperscript{b}, Michael Pinedo\textsuperscript{c} & Chelliah Sriskandarajah\textsuperscript{b}
\textsuperscript{a} Warrington College of Business Administration, University of Florida, Gainesville, FL 32611, USA E-mail:
\textsuperscript{b} Mays Business School, Texas A&M University, College Station, TX 77843, USA E-mail:
\textsuperscript{c} Leonard N. Stern School of Business, New York University, New York, NY 10012, USA E-mail:

Accepted author version posted online: 28 Jul 2014. Published online: 21 Dec 2014.

To cite this article: Kyung Sung Jung, H. Neil Geismar, Michael Pinedo & Chelliah Sriskandarajah (2014): Approximations to optimal sequences in single-gripper and dual-gripper robotic cells with circular layouts, IIE Transactions, DOI: 10.1080/0740817X.2014.937019

To link to this article: http://dx.doi.org/10.1080/0740817X.2014.937019

PLEASE SCROLL DOWN FOR ARTICLE
Approximations to optimal sequences in single-gripper and dual-gripper robotic cells with circular layouts

KYUNG SUNG JUNG1, H. NEIL GEISMAR2, MICHAEL PINEDO3,* and CHELLIAH SRISKANDARAJAH2

1Warrington College of Business Administration, University of Florida, Gainesville, FL 32611, USA
E-mail: ksjung@ufl.edu
2Mays Business School, Texas A&M University, College Station, TX 77843, USA
E-mail: ngeismar@mays.tamu.edu
E-mail: chelliah@mays.tamu.edu
3Leonard N. Stern School of Business, New York University, New York, NY 10012, USA
E-mail: mpinedo@stern.nyu.edu

Received July 2013 and accepted May 2014

This article considers the problems of scheduling operations in single-gripper and dual-gripper bufferless robotic cells in which the arrangement of machines is circular. The cells are designed to produce identical parts under the free-pickup criterion with additive intermachine travel time. The objective is to find a cyclic sequence of robot moves that minimizes the long-run average time required to produce a part or, equivalently, that maximizes the throughput. Obtaining an efficient algorithm for an approximation to an optimal k-unit cyclic solution (over all k ≥ 1) is the focus of this article.

The proposed algorithms introduce a new class of schedules, which are referred to as epi-cyclic cycles. A polynomial algorithm with a 5/3-approximation to an optimal k-unit cycle over all cells is developed. The performed structural analysis for dual-gripper cells leads to a polynomial-time algorithm that provides at worst a 3/2-approximation for the practically relevant case in which the dual-gripper switch time is less than twice the intermachine robot movement time. A computational study demonstrates that the algorithm performs much better on average than this worst-case bound suggests. The performed theoretical studies are a stepping stone for researching the complexity status of the corresponding domain. They also provide theoretical as well as practical insights that are useful in maximizing productivity of any cell configuration with either type of robot.

Keywords: Robotic cells, single-gripper robots, dual-gripper robots, cyclic solutions, approximation algorithms

1. Introduction

Automated material handling systems are extensively used in modern manufacturing in order to improve productivity. One such implementation is a robot-served manufacturing cell, or robotic cell, which is used in a wide variety of industries under various configurations (Miller and Walker, 1990; Asfahl, 1992; Dawande et al., 2007; Jung, 2013). After a difficult 2009 and 2010, the robotics industry has strongly rebounded. Robot sales in both 2011 and 2012 have hit new all-time highs with annual sales exceeding 150,000 units and dollar volume surpassing $25 billion (World Robotics, 2012, 2013). The two largest markets are Japan and the Republic of Korea, with Germany and the United States close behind. The fastest growing market for robots is, not surprisingly, China. The major industries driving these increases are automotive, electronics (computers, televisions, and medical/optical instruments), rubber and plastics, food and beverage, solar power systems, and metal products.

Semiconductor manufacturing is an important application area for robotic cells (Venkatesh et al., 1997; Geismar et al., 2004; Mönch et al., 2011). Dawande et al. (2007) provide descriptions of a variety of other real-world implementations of robotic cells, including electroplating, textiles, glass, cosmetics, and lawn tractors. The optimal use of these cells requires efficient operating schedules, and finding such schedules leads to a variety of challenging combinatorial optimization problems. Good operating schedules can improve productivity by 3 to 5%, without requiring any additional equipment.

A typical robotic cell consists of m machines $M_i, i = 1, 2, \ldots, m$, an input buffer (I) that holds unprocessed parts, an output buffer (O) for completed parts, and a robot that performs all material-handling functions (loading,
unloading, and transferring parts from one machine to the next). Each part must be processed on each machine. The processing order through the machines is the same for all parts, similar to a flow shop. A machine can hold only one part, and the cell has no buffers for intermediate storage. Figure 1(a) shows a five-machine robotic cell with a circular layout and a dual-gripper robot. Figure 1(b) exhibits a real-world implementation at a Korean company of a three-machine robotic cell with a circular layout and a dual-gripper robot (FANUC Robot Model M-16/B/20, Korea Fanuc Corporation) for manufacturing oil pumps.

A robot has either a single-gripper or a dual-gripper effector at the end of its arm. A single-gripper robot can hold only one part at any given time. Because the cells do not contain internal buffers for storage of parts between processes, a single-gripper robot cannot move a part from machine $M_i$ to $M_{i+1}$ if machine $M_{i+1}$ is occupied; this is called blocking in the classic flow shop scheduling literature (Pinedo, 2012). Blocking does not occur in a dual-gripper robotic cell since the robot can hold two parts simultaneously; thus, a part can be moved from machine $M_i$ to $M_{i+1}$ even if machine $M_{i+1}$ is occupied. In a typical operation, a robot with an empty dual-gripper may pickup a part at $M_i$, move its arm to $M_{i+1}$, use its empty gripper to unload the finished part from $M_{i+1}$, rotate the "wrist" at the end of its arm, and load the other part onto $M_{i+1}$. In most implementations, the time required for this switch of grippers is significantly less than the time needed for the robot to travel between machines (Geismar et al., 2006). Thus, the decision of having a dual-gripper is based on a trade-off between a gain in throughput and an increase in cost and complexity.

Many examples of circular cells with dual-gripper robots can be found in a variety of industries. In semiconductor manufacturing they are usually referred to as cluster tools, each consisting of several processing stages. Processes performed by these tools include photolithography, low-pressure chemical vapor deposition, dielectric etching, photo-resist cleaning, rapid thermal processing, and chemical–mechanical planarization (Rostami et al., 2001; Rostami and Hamidzadeh, 2002; Kim et al., 2003). Companies that use these technologies include Brooks Automation, Inc. (Rostami et al., 2001), Jusung Engineering Co., Ltd. (Kim et al., 2003), Applied Materials, Inc. (Venkatesh et al., 1997), and FSI International, Inc. (Geismar et al., 2004).

Xerox Corporation is a typical company that uses circular robotic cells; each one of its robotic cells is served by a dual-gripper 2105B Unimate robot and produces copper fuser rolls for its 9200 duplicator in its Rochester, New York, facility. The cells contain two CNC center-drive, double-ended lathes, plus brazing, grinding, broaching, and turning machines (Miller and Walker (1990), pp. 4–27). Mold-makers, Inc. (Germantown, WI), uses several cells to make plastic molds for the plastic injection and blow molding industries via wire electrical discharge machining (MGŠ News, 2007).

Robotic cells are classified into three types based on the part pickup criterion used by the robot (Dawande et al., 2007): a no-wait cell, an interval cell, or a free-pickup cell. In no-wait cells, as soon as machine $M_i$ completes processing a part, that part must be removed from $M_i$ and transferred to machine $M_{i+1}$. Examples of this production environment include steel manufacturing and plastic molding, where the products must maintain certain temperatures between processes. The reader may refer to Levner et al. (1997), Agnetis (2000), and Dawande et al., (2007) for results concerning no-wait cells. The hoist scheduling problem on an electroplating line is an example of an interval robotic cell in which each machine has a specific time interval during which a part must be removed from the machine after completing its processing (Lei and Wang, 1994; Dawande et al., 2010). In free-pickup cells there is no limit on the time that a part may reside at a machine after its processing.

Three travel-time metrics for the robot’s intermachine movements have been studied for each part pickup criterion: additive, constant, and Euclidean. For additive travel-time cells, the time required for the robot to travel between any two adjacent machines is a constant, denoted by $\delta$. When traveling between any two non-adjacent machines, the robot’s path passes each machine in between, so its

Fig. 1. (a) A five-stage dual-gripper robotic cell in a circular layout and (b) an automated manufacturing system for oil pumps (manufactured by a Korean Company).

\[\text{Dual-Gripper Robot}\]

\[M_1\]

\[M_2\]

\[M_3\]

\[M_4\]

\[M_5\]

\[M_6(O)\]

\[M_6(I)\]
travel time is the sum of the times required to traverse these segments in between the machines. In constant travel-time cells, introduced by Dawande et al. (2002), the time required to travel between any two machines, adjacent or non-adjacent, is a constant value. Brauner et al. (2003) define Euclidean travel-time cells in which the only restriction on the intermachine travel times is that they obey the triangle inequality and are symmetric in the sense that the travel times between any two machines are the same in both directions.

Two types of machine layouts are considered in the literature: a linear layout, in which the machines I and O are arranged in a straight line or a semicircle, and a circular layout, in which the machines I and O are arranged in a circle such that I and O are adjacent (see Fig. 1(a)). In a circular cell the direction of robot travel between two locations can be clockwise or counterclockwise, whichever is shortest. For example, the robot in Fig. 1(a) travels from the output buffer O to machine M1 by moving counterclockwise (passing I, time 2δ) because clockwise travel (traversing machines M2, M4, M5, and M6, time 5δ) would require more time. This flexibility in robot movements can significantly improve the productivity of a cell; however, the sequencing of operations becomes more challenging, even for single-gripper cells (Brumson, 2001; Rajapakshe et al., 2011). Furthermore, schedules that are optimal in linear cells are not necessarily optimal in circular cells (Geismar, Sethi, Sidney, and Sriskandarajah, 2005), so different techniques are required to find efficient robot sequences.

The vast majority of publications on robotic cell scheduling consider only one-unit cycles (defined explicitly in Section 3) because they are commonly used in industry, given that they are easier to implement. However, cycles optimal in the realm of one-unit cycles are not necessarily optimal over all k-unit cycles, k ≥ 1 (Brauner and Finke, 2001; Dawande, Geismar, and Sethi, 2005). The increase in complexity through the inclusion of k-unit cycles has limited the number of results published for this case, even though k-unit cycles may lead to higher throughput. To fill this void, we derive one-unit and two-unit schedules of robot actions that produce parts at throughput rates that are guaranteed to be within fixed percentages of optimal k-unit cycles’ throughput.

We study throughput optimization problems in free-pickup cells with additive travel times, with these cells having circular layouts and producing identical parts. Because we allow all possible combinations of production parameter values, we need to consider both single-gripper and dual-gripper robotic cells. The contributions of this article are of a theoretical nature as well as of a practical nature.

1. Algorithmic and complexity results: The main theoretical contribution is to provide insights into the schedules that maximize the productivity of both single-gripper and dual-gripper circular robotic cells for any combination of parameters. Our results indicate that several special cases are polynomially solvable. The complexity status of the remaining cases are open, but for them we provide polynomial-time algorithms with performance guarantees. We also show that lower bounds for two-unit cycles apply to k-unit cycles for any k ≥ 1. Our theoretical results are based on a new class of schedules, called epicyclic cycles, which we believe will be useful in other scheduling environments as well.

2. Performance of dual-gripper cells under practically relevant conditions: Our structural analysis for dual-gripper cells leads to a polynomial-time algorithm that provides a one-unit or a two-unit cycle that is guaranteed to have at least 66% of the optimum throughput (over all k-unit cycles, k ≥ 1) for the practically relevant condition that the dual-gripper switch time is less than twice the intermachine robot movement time. Computational experiments demonstrate that, on average, our schedules have a productivity higher than 91% of the theoretical optimum.

This article is organized as follows. Section 2 reviews the relevant literature. Section 3 provides the necessary notation and definitions. Section 4 presents lower bounds that are important for our analysis. Section 5 describes classes of robotic schedules (cycles) and the special cases for which they provide straightforward solutions. Section 6 analyzes the more general case. Section 7 provides a counterpoint to our theoretical work by describing a computational study that analyzes our approximations’ average results; i.e., those that are most likely to be realized in practice. Section 8 presents conclusions and makes recommendations for future research.

2. Literature review

A cyclic schedule in a robotic cell specifies the repetitive execution of a fixed sequence of robot moves that produces k ≥ 1 parts. Intuitively, a k-unit cycle is a sequence of robot moves in which exactly k raw parts are taken from the input buffer I, exactly k completed parts are dropped at the output buffer O, and the cell returns to its initial state at the end of this sequence. Precise definitions of cyclic schedule and cycle are provided in the next section. In practice, their repetitive nature makes these cyclic schedules easy to implement and control, so they are the primary way of describing the operation of a robotic cell.

In one of the first analytical studies of robotic cells, Sethi et al. (1992) develop one-unit cycles for free-pickup cells with single-gripper robots, two or three machines, additive travel time, and linear layout. Crama and van de Klundert (1997) develop a polynomial-time algorithm for finding an optimal one-unit cycle for single-gripper robotic cells with an arbitrary number of machines and additive travel time under linear layout, and Dawande et al. (2002) do the same for cells with constant travel times. For linear single-gripper cells with either travel-time metric, the problem of finding an optimal k-unit cycle over all k ≥ 1 remains open.
Rajapakshe et al. (2011) prove that the problem of finding an optimal one-unit cycle in single-gripper robotic cells with a circular layout and additive travel time is NP-hard. A detailed discussion of the complexity of various robotic cell problems can be found in Dawande et al. (2007). Other reviews of the literature on scheduling robotic cells include Hall (1999), Crama et al. (2000), Dawande, Geismar, Sethi, and Sriskandarajah (2005), and Brauner (2008).

The problem of scheduling operations in dual-gripper robotic cells is more complicated than in single-gripper robotic cells, even in the two-machine case. In the first work in this area, Su and Chen (1996) consider a cell with two machines producing identical parts and describe five different one-unit cycles. Sethi et al. (2001) show that there are 52 possible one-unit cycles for two-machine cells. They study robotic cells with a circular layout that produce identical parts under free-pickup and additive travel time. Drobouchevitch et al. (2006) show that a particular dual-gripper cycle is optimal among all one-unit cycles under certain conditions that are common in practice.

A few studies develop polynomial algorithms that produce cycles whose throughput is proven to be within a certain percentage of the optimum for all \( k \)-unit cycles, \( k \geq 1 \), in their particular environment. Geismar et al. (2007) and Geismar et al. (2008) develop polynomial algorithms that provide a 9/7-approximation (resp., a 3/2-approximation) for single-gripper (resp., dual-gripper) cells with constant travel time and a 10/7-approximation for single-gripper cells with additive travel time in a linear layout. Rajapakshe et al. (2011) derive a 5/3-approximation to the optimum one-unit cycle time for single-gripper circular cells with additive travel time. In contrast, ours is the first work to find approximations with performance guarantees to optimal \( k \)-unit cyclic solutions in robotic cells with circular layout, additive travel time, and free-pickup, for either single-gripper or dual-gripper robots.

Table 1 summarizes the complexity status of optimization problems in single-gripper and dual-gripper robotic cells with free-pickup; thus, it shows how this current study fills a gap in the theoretical framework. The table also provides performance bounds available in the literature for those problems that are either NP-hard or open with regard to complexity status.

3. Notation and definitions

Our notation for describing robotic cells follows Dawande et al. (2007) and Rajapakshe et al. (2011).

\( M_1, \ldots, M_m \): machines in the robotic cell listed in their processing order (see Fig. 1(a) for \( m = 5 \)).

\( I \): the input buffer, also denoted \( M_0 \) and referred to as a machine (see Fig. 1(a)).

\( O \): the output buffer, also denoted \( M_{m+1} \) and referred to as a machine (see Fig. 1(a)).

\( p_i \): the processing time of a part on machine \( M_i \), \( 1 \leq i \leq m \).

\( \delta \): the time taken by a robot to travel between two adjacent machines, regardless of whether operations that are executed on adjacent machines require different grippers or the same gripper. The travel time between any two machines \( M_i, M_j, 0 \leq i, j \leq m + 1 \) is min\( (|i - j|\delta, (m + 2) - |i - j|\delta) \). The travel time between two machines does not depend on whether or not the robot is carrying a part.

\( \epsilon \): the time taken by the robot to load (unload) a part onto (from) any machine where to load a part onto \( M_{m+1} \) means to drop-off a part at \( O \), and to unload a part from \( M_0 \) means to pickup a part at \( I \).

\( M_i^l \): the robot activity load a part onto machine \( M_i \), \( i = 1, 2, \ldots, m + 1 \).

\( M_i^u \): the robot activity unload a part from machine \( M_i \), \( i = 0, 1, \ldots, m \).

\( \theta \): the gripper switch time for a dual-gripper robot; i.e., the time from the moment one gripper has unloaded a machine until the moment the second gripper is positioned to load the same machine.

\( w_{ij} \): the duration of the robot waiting time at \( M_i \); note that \( w_0 = w_{m+1} = 0 \), since there is no processing at \( I \) or \( O \). For \( k \)-unit cycles, \( w_{ij} \) is the waiting time before unloading the \( r \)th part to be completed on \( M_i \) during the cycle for \( r = 1, \ldots, k \).

| Table 1. Complexity map of the problem of obtaining optimal one-unit cycles and \( k \)-unit cycles with free-pickup (P: Polynomially solvable; NPHO: Ordinarily NP-Hard) |
|-------------------------------------------------|-------------------------------------------------|------------------------|------------------------|
| **Distance** | **Single-gripper cell** | **Dual-gripper cell** | **One-unit cycle (\( \theta \leq \delta \))** | **One-unit cycle (\( \theta \leq \delta \))** |
| **metrics** | **One-unit cycle** | **\( k \)-Unit cycle** | **P** | **Open, [3/2]** |
| Constant | P | Open, [9/7] | Geismar et al. (2006) | | |
| Additive | Dawande et al. (2002) | Geismar et al. (2008) | Open | |
| linear | P | Open, [10/7] | Geismar et al. (2007) | Geismar et al. (2008) |
| Additive | Crama and van de Klundert (1997) | Open, this article | | |
| circular | NPHO, [5/3] | 5/3-approximation | Drobouchevitch et al. (2006) | |
| | Rajapakshe et al. (2011) | | P | |

Best known approximation bounds are given in square brackets.
**Definition 1.** A k-unit activity sequence is a sequence of robot moves that loads and unloads each machine exactly k times.

It follows that a k-unit activity sequence produces k completed parts. We have the following necessary criteria for an activity sequence to be feasible.

1. The robot cannot be instructed to load (resp., unload) an occupied (resp., unoccupied) machine.
2. A dual-gripper robot cannot be instructed to unload (resp., load) a machine if both of its grippers are occupied (resp., empty); e.g., a feasible activity sequence cannot have a subsequence \((M'_i, M'_j, M'_k)\) (resp., \((M'_i, M'_j, M'_k)\)).
3. The robot can load a part onto \(M_i\) only if that part’s most recent processing was on \(M_{i-1}\), \(i = 1, \ldots, m + 1\).

Let the function \(f(M'_i, r)\) denote the completion time of the \(r\)th execution of activity \(M'_i\) (Crama and van de Klundert, 1997). Given a feasible infinite sequence of activities and a compatible initial state, we can define its long-run average throughput or, simply, throughput to be

\[
\mu = \lim_{r \to \infty} \frac{r}{f(M'_{m+1}, r)}.
\]

Intuitively, this quantity represents the long-term average number of completed parts placed into the output buffer per unit of time.

**Definition 2.** A k-unit cycle \(S(k)\) is the performance of a feasible k-unit activity sequence that leaves the cell in exactly the same state as its state at the beginning of that sequence.

We omit the argument \((k)\) when the number of units produced by the cycle is apparent by the context. Since the start and end states of a k-unit cycle are the same, we can repeat it indefinitely in order to obtain a k-unit cyclic sequence. The throughput of a k-unit cyclic sequence is therefore the throughput of the corresponding k-unit cycle.

In turn, the throughput of a k-unit cycle is the reciprocal of its per unit cycle time (see Definition 3), so minimizing per unit cycle time is equivalent to maximizing throughput. For clarity, we use \(S'\) (resp. \(S''\)) to denote a cycle that applies to a single-gripper (resp., dual-gripper) cell.

**Definition 3.** A robotic cell that repeatedly executes a k-unit cycle \(S(k)\) of robot moves is operating in steady state if there exists a constant \(T(S(k))\) and a constant \(N\) such that for every \(M'_i, i = 1, \ldots, m + 1\), \(f(M'_i, \tau + k) = f(M'_i, \tau) = T(S(k)), \tau \in \mathbb{Z}^+, \tau > N\). \(T(S(k))\) is called the cycle time of \(S(k)\). The per unit cycle time of k-unit cycle \(S(k)\) is \(T(S(k))/k\); i.e., the average time required to produce one unit.

Online Appendix A provides additional definitions used for the analysis in this article.

### 4. Lower bounds on per unit cycle times

Because our intent is to prove the optimality or the guaranteed approximation bound of particular cycles over both single-gripper and dual-gripper k-unit cycles (for all \(k \geq 1\)) under various conditions, we start with lower bounds for both single-gripper and dual-gripper robotic cells. We cannot completely separate the analyses for these two robot types because all single-gripper cycles are also feasible for dual-gripper cells. (Of course, dual-gripper cycles requiring gripper switching operations cannot be implemented in single-gripper cells.) The following lower bound on the per unit cycle time in single-gripper robotic cells is derived in Dawande, Geismar, Sethi, and Sriskandarajah (2005):

\[
\frac{T(S'(k))}{k} \geq \max \left\{ \frac{(m + 2)\delta + 2(m + 1)\epsilon}{k}, \sum_{i=1}^{m} \min \{p_i, \delta\}, p_{\max} + 4\delta + 4\epsilon \right\}, \quad (1)
\]

where \(p_{\max} = \max_{1 \leq i \leq m} \{p_i\}\).

Geismar et al. (2008) develop the following lower bounds for the per unit cycle time in dual-gripper robotic cells:

\[
\frac{T(S''(k))}{k} \geq (m + 2)\delta + 2(m + 1)\epsilon
\]

\[
+ \sum_{i=1}^{m} \min \{\delta, p_i, \theta, p_i + \frac{\theta - \delta}{2}\}
\]

\[
+ \min\{0, \theta - \delta\}. \quad (2)
\]

When \(\delta \geq \theta\), bound (2) becomes

\[
\frac{T(S''(k))}{k} \geq (m + 1)(\delta + 2\epsilon + \theta)
\]

\[
+ \sum_{i=1}^{m} \min \{0, p_i - \frac{\theta + \delta}{2}\}. \quad (4)
\]

Geismar et al. (2006) establish lower bound (5) under the condition that \(\theta \leq \delta \leq p_{\min}\), where \(p_{\min} = \min_{1 \leq i \leq m} \{p_i\}\).

\[
\frac{T(S''(k))}{k} \geq \max \{(m + 2)\delta + 2(m + 1)\epsilon + m\theta, p_{\max} + 2\epsilon + \theta\}. \quad (5)
\]

### 5. Special cases of single-gripper and dual-gripper cycles

For both single-gripper and dual-gripper cycles, we describe some special cycles and the cases for which either they are optimal or they provide effective and efficient approximations to the optimum per unit cycle time. Each result refers to optimality over k-unit cycles, \(k \geq 1\), in both single-gripper and dual-gripper cells. We show that, in general,
single-grippers are preferred if $\theta \geq 2\delta$, and dual-grippers are preferred if $\theta \leq 2\delta$. Proofs for this section’s results are given in Online Appendix B.2.

### 5.1. Single-gripper cycles

We now define four one-unit cycles for single-gripper cells. Note that symbols for unloading parts from machines are not required to specify single-gripper cycles. Each of the first two is optimal under certain conditions (Lemmas 1 and 2). The last two are used to guarantee bounded approximations to the optimum per unit cycle time (Lemmas 3 and 4).

**Forward cycle $S'_1$:** In cycle $S'_1 = (M'_1, M'_2, \ldots, M'_m, M'_{m+1})$ (Fig. 2(a)), the robot picks up a part from the input buffer $(M_0)$, moves to $M_1$, loads $M_1$, waits at $M_1$ during the processing of that part, unloads $M_1$, and then moves to $M_2$. It follows this process at each machine in ascending order and finishes the cycle by returning to $M_0$ after loading the output buffer $(M_{m+1})$. Note that it has full waiting at each machine (see Online Appendix A for the definition of full waiting). The cycle time for $S'_1$ is

$$T(S'_1) = (m + 2)\delta + 2(m + 1)\epsilon + \sum_{i=1}^{m} p_i.$$  

**Lemma 1.** If $p_{\text{max}} \leq \delta \leq \theta$, then cycle $S'_1$ is optimal.

**Reverse cycle $S'_2$:** In cycle $S'_2 = (M'_m, M'_{m+1}, M'_m, M'_{m-1}, \ldots, M'_2, M'_1)$ (Fig. 2(b)), the robot unloads a part from the input buffer $(M_0)$, carries it to $M_1$, and loads $M_1$. It then travels to $M_m$, unloads $M_m$, and carries that part to the output buffer $(M_{m+1})$. It repeats the following sequence for $i = m - 1, m - 2, \ldots, 1$: travels to $M_i$, unloads $M_i$, carries the part to $M_{i+1}$, loads $M_{i+1}$. After loading $M_2$, the robot completes the cycle by traveling to the input buffer $(M_0)$. The robot has partial waiting at each machine (see Online Appendix A for a definition of partial waiting). The cycle time of $S'_2$ is

$$T(S'_2) = \max \{(3m + 4)\delta + 2(m + 1)\epsilon, \ p_{\text{max}} + 4\delta + 4\epsilon\}.$$  

**Lemma 2.** If $\theta \geq 4\delta + 2\epsilon$ and $p_{\text{max}} \geq \min((m + 2)\delta + 2m\epsilon + (m - 1)\theta, 3m\delta + 2(m - 1)\epsilon)$, then cycle $S'_2$ is optimal.

**Odd-even cycle $S'_3$:** In this cycle (Fig. 2(c)), the robot first loads the odd-numbered machines in increasing order and then loads the even-numbered machines in increasing order (Rajapakshe et al., 2011). The cycle and its cycle times are

For $m$ even: $S'_3 = (M'_1, M'_3, M'_2, \ldots, M'_m, M'_2, M'_m, M'_1)$, then loads the even-numbered machines in increasing order (Rajapakshe et al., 2011). The cycle and its cycle times are

$$T(S'_3) = \max(2(m + 2)\delta + 2(m + 1)\epsilon, (m + 2)\delta + 2m\epsilon + p_{\text{max}}).$$

For $m$ odd: $S'_3 = (M'_1, M'_3, M'_2, \ldots, M'_m, M'_2, M'_m, M'_1)$,

$$T(S'_3) = \max(2(m + 2)\delta + 2(m + 1)\epsilon, (m + 2)\delta + 2m\epsilon + p_{\text{max}}).$$

where $p_{\text{max}} = \max_{1 \leq i \leq \frac{m+1}{2}} \{p_{2i-1}\}$, and $\tilde{p}_{\text{max}} = \max_{1 \leq i \leq \frac{m+1}{2}} \{p_{2i}\}$.

Combining the two expressions for cycle $S'_3$ above leads to

$$T(S'_3) \leq \max(2(m + 2)\delta + 2(m + 1)\epsilon, (m + 2)\delta + 2m\epsilon + p_{\text{max}}).$$

**Loop cycle $S'_4$:** For a single-gripper robotic cell, Rajapakshe et al. (2011) define loop cycle $S'_4$ (Fig. 2(d)) for a given subset of machine indices, $E$. A loop cycle is a hybrid of the forward cycle $S'_3$ and the reverse cycle $S'_2$. The following two rules are applied in the construction of the cycle. Let $E = \{j_1, j_2, \ldots, j_{s-1}, j_s\}$, where $j_i \in \mathbb{Z}$, $i = 1, \ldots, s$, $j_1 = 0$, $j_s = m + 1$, and $j_1 < j_2 < \cdots < j_s$.

**Rule A:** For any subset of consecutive indices $\{j_p, j_p + 1, j_p + 2, \ldots, j_q - 1, j_q\} \subseteq E$ (i.e., $j_{p+r} = j_p + r$ for $r = 1, 2, \ldots, q - p$), the sequence of machine loadings is $M'_{j_p + 1}, M'_{j_p + 2}, \ldots, M'_{j_q}$.

**Rule B:** Otherwise, for any subset of two non-consecutive indices (i.e., $j_{p+1} \neq j_p + 1$), the sequence of machine loadings is $M'_{j_{p+1}}, M'_{j_{p+1}+1}, \ldots, M'_{j_{p+1}+1}$.
Subset $E$ specifies the type of waiting (full or partial) that the robot has at each machine. For example, consider a six-machine cell for which $E = \{0, 1, 2, 5, 6, 7\}$. Rule A is applied to the sequences of consecutive indices $\{0, 1, 2\}$ and $\{5, 6, 7\}$, generating $(M'_1, M'_2)$ and $(M'_6, M'_7)$. For $\{2, 5\}$, Rule B is used, which generates $(M'_3, M'_4, M'_5)$. Therefore, $S'_i = (M'_1, M'_2, M'_3, M'_4, M'_5, M'_6, M'_7)$; see Fig. 2(d).

Let $D_2 = \{j : p_j \geq 2\delta\}$ and $\bar{D}_2 = \{j : p_j < 2\delta\}$. An efficient cycle should have partial waiting at machines in $D_2$ and either full waiting or zero partial waiting at those in $\bar{D}_2$. This is accomplished by specifying subset $E$ so that $E = D_2 \cup \bar{D}_2$. Rajapakse et al. (2011) show that when $|E| > (m+2)/2$, the travel time in cycle $S'_i$ is $[5(m+2) - 4|E|]2\delta$, so

$$T(S'_i) \leq (m + 2)\delta + 2(m + 1)\epsilon + 4|D_2|\delta + \sum_{i \in \bar{D}_2} p_i.$$  

Rajapakse et al. (2011) use this bound to establish lower bounds that are used in the proofs of the following two lemmas.

Lemma 3 addresses the only parameter combination for which our best approximation exceeds 3/2.

**Lemma 3.** If $\theta > 2\delta$ and $p_{\text{max}} \geq (m + 2)\delta + (m - 1)\epsilon$, then the best of cycles $\{S'_i, S'_i, S'_i\}$ provides a $5/3$-approximation.

Lemma 4 establishes an approximation bound that complements Lemma 3.

**Lemma 4.** If $\theta > 2\delta$ and $p_{\text{max}} \leq (m + 2)\delta + (m - 1)\epsilon$, then the better of cycles $\{S'_i, S'_i\}$ provides a $3/2$-approximation.

**Remark 1.** As a consequence of Claim 3 in Online Appendix B.1, which states that single-gripper one-unit cycles dominate all dual-gripper cycles if $\theta > 2\delta$, no practitioner would use a dual-gripper robot under this condition.

The results obtained in this section are summarized in Table 2.

### 5.2. Dual-gripper cycles

We now define three cycles for dual-gripper cells. Each visits machines $M_0, \ldots, M_{m+1}$ once in ascending order. The first is a two-unit cycle; the others are one-unit cycles. As in the single-gripper case, each of the first two is optimal under certain conditions (Lemmas 5 and 6), and the third is used to guarantee bounded approximations to the optimum per unit cycle time (Lemma 7).

**Dual forward cycle $S''_m$:** In the initial state of cycle $S''_m$, all machines and the robot are empty, and the robot is at $M_0$. The robot picks up two parts (say $X_r$ and $X_{r+1}$) from $M_0$, then visits each machine in ascending order. At each machine, $X_r$ is loaded, processed, and unloaded, and then $X_{r+1}$ is loaded, processed, and unloaded. After serving $M_m$, both parts are placed onto $M_{m+1}$, and the robot returns to $M_0$. This two-unit cycle and its cycle times are (Sethi et al., 2001)

$$S''_m = (M'_0, M'_1, M'_2, M'_2, M'_3, M'_4, M'_5, M'_6, M'_6, M'_6, M'_7, M'_7, M'_7, M'_8, M'_9, M'_9),$$

$$T(S''_m) = (m + 2)\delta + 2(m + 1)\epsilon + (m + 2)(\delta + 2 \sum_{i=1}^{m+1} p_i).$$

**Lemma 5.** If $\theta \leq \delta$ and $p_{\text{max}} \leq (\delta + \theta)/2$, then cycle $S''_m$ is optimal.

**Exchange cycle $S'''_m$:** In cycle $S'''_m$, the initial state has all machines occupied, and the robot is empty at $M_0$. The robot picks up a part (say $X_r$) from $M_0$ and moves to $M_1$. If necessary, the robot waits for part $X_{r-1}$ to be completed by $M_1$. The robot unloads part $X_{r-1}$ from $M_1$, rotates its grippers, and loads part $X_r$ onto $M_1$. Part $X_{r-1}$ is carried to $M_2$. If necessary, the robot waits for part $X_{r-2}$ to be completed by $M_2$, unloads part $X_{r-2}$ from $M_2$, rotates its grippers, and loads part $X_{r-1}$ onto $M_2$. The robot continues in this manner to $M_m$, then drops part $X_{m-1}$ onto $M_{m+1}$, and returns to $M_0$. This one-unit cycle and its cycle times are (Geismar et al., 2006)

$$S'''_m = (M'_0, M'_1, M'_2, M'_2, M'_3, M'_4, M'_4, M'_5, M'_6, M'_6, M'_6, M'_7, M'_7, M'_8, M'_9, M'_9),$$

$$T(S'''_m) = \max \{ (m + 2)\delta + 2(m + 1)\epsilon + m\delta, p_{\text{max}} + 2\epsilon + \theta \}.$$
Lemma 6. Cycle $S''_m$ is optimal under either of the following sets of conditions:

(a) $\theta < 4\delta + 2\epsilon$ and $p_{\text{max}} \geq (m + 2)\delta + 2m\epsilon + (m - 1)\theta$;
(b) $p_{\text{min}} \geq \delta \geq \theta$.

Cycle $S''_m$ is a special case of the Quick Cycle.

Quick cycle $S''$: Quick Cycle $S''_q$ is a one-unit cycle that is generated through the following procedure.

Begin

For $i = 1$ to $m$

if $p_i \geq \theta$, assign machine $M_i$ the activity sequence $(M''_i, M''_i)$.

otherwise $(p_i < \theta)$, assign machine $M_i$ the activity sequence $(M'_i, M'_i)$.

Next $i$

End

Its cycle time is

$$T(S''_q) = \max \left\{ (m + 2)\delta + 2(m + 1)\epsilon + \sum_{i=1}^{m} \min\{p_i, \theta\}, p_{\text{max}} + 2\epsilon + \theta \right\}.$$  

Lemma 7. Cycle $S''_q$ provides a $3/2$-approximation under any of the following sets of conditions:

(a) $\delta < \theta \leq 2\delta$;
(b) $\theta \leq \delta$ and $p_{\text{max}} \geq (m + 2)\delta + (m - 1)\epsilon$;
(c) $\delta/3 \leq \theta \leq \delta$;
(d) $\theta \leq \delta$ and $r_2 \leq (m - 1)/3$, where $R_2 = \{i : p_i < \theta, 1 \leq i \leq m\}$ and $r_2 = |R_2|$.

Remark 2. Inequality (B.1) from the proofs of Lemma 7(c) and 7(d) (see Online Appendix B.2) may be generalized to state that cycle $S''_q$ provides a $(1 + (1 - \rho)/(1 + \rho))$-approximation, where $0 \leq \rho \leq \delta$, for $0 \leq \rho \leq 1$, and also a $(1 + (r_2 + 2)/(2m - r_2 + 2))$-approximation, if $\theta \leq \delta$.

The results of this section (see Table 3) demonstrate that we can generate a cycle whose throughput is at worst $5/3$ of the optimum for all cases except $\theta < \delta/3$, $(\delta + \theta)/2 < p_{\text{max}} < (m + 2)\delta + (m - 1)\epsilon$, $p_{\text{min}} < \delta$, and $r_2 > (m - 1)/3$. We design a class of dual-gripper cycles to resolve this case in the following section to complete our approximation scheme.

6. Approximation algorithms for optimum $k$-unit cycle times

We define the class of epicyclic cycles to provide approximations to optimal cycles for the open case described at the end of the previous section. To do so, we first separate the machines of the robotic cell into two disjoint subsets, based on a given two-unit cycle. One set contains those machines that are visited only one time by the robot. The second set—those machines that are visited at least twice—is further divided into maximal subsets of consecutive machines by Procedure Grouping (Section 6.1). These subsets are used to prove new lower bounds on the per unit cycle time over all $k$-unit cycles, $k \geq 1$. Section 6.2 defines Procedure Transform, which converts any two-unit cycle into an epicyclic cycle, and then proves that the set of epicyclic cycles dominates the set of two-unit cycles. Algorithm Epi-path (Section 6.3) finds the best epicyclic cycle and proves that its per unit cycle time is at worst $3/2$ that of the optimum per unit cycle time for $\theta \leq \delta$ and $p_{\text{max}} \leq (m + 2)\delta + (m - 1)\epsilon$. Algorithm QCYCLE (Section 6.4) finds an approximation to an optimal $k$-unit cycle for any cell by combining cycles defined in Section 5 and the set of epicyclic cycles. Section 6.4 then states the performance guarantee for Algorithm QCYCLE, which is the main result of the article. Proofs of theoretical results of this section (Lemmas 8, 9, and 10; Corollaries 1 and 2; Theorem 1) are given in Online Appendix D.

6.1. Lower bounds: procedure grouping

We derive lower bounds for two-unit cycles and then extend these to $k$-unit cycles, $k \geq 1$. Let cycle $S''_q$ be a two-unit
cycle for a problem instance $I$. Without loss of generality, we assume that cycle $S'_j^U$ starts with activity $M_0^U$.

Geismar et al. (2006) show that a two-unit cycle $S'_j^U$ can be represented by a sequence of two consecutive loadings and unloadings of machine $M_i$ of the form $S'_j^U = (M_{i_1}^U, \sigma_1, M_{i_2}^U, \sigma_2, M_{i_1}^U, \sigma_3, M_{i_2}^U, \sigma_4)$, for any $i \in \{0, 1, \ldots, m + 1\}$. (Online Appendix A presents the technical details behind this paragraph's development.) Activity $M_{i_j}^U$ ($M_{i_j}'^U$) denotes the $r$th loading (unloading) of machine $M_i$, $r = 1, 2$. The $\sigma_i$, $h = 1, \ldots, 4$, are permutations of other activities that together make $S'_j^U$ a feasible cycle. Let $A(\sigma_i)$ be the set of activities in permutation $\sigma_i$, $h = 1, 2, 3, 4$. How many visits the robot makes to $M_i$ and the sequence of actions it performs during each visit are determined by which of the sets $A(\sigma_i)$ are empty. Each distinct combination of empty $A(\sigma_i)$s, $h = 1, 2, 3, 4$, is called a case. There are 10 feasible cases and corresponding sets $U_i$, $s = 0, \ldots, 9$, such that machine $M_i$ belongs to Case $s$ if and only if $i \in U_s$. We may also state “machine $M_i$ belongs to set $U_s$” when, technically, it is index $i$ that belongs to $U_s$. For our purposes, we need only Cases 5 to 9 (for completeness, Cases 0 to 4 are described in Online Appendix A).

1. For Case 8, the robot visits $M_i$ twice. During each visit it loads $M_i$, rotates its grippers, then loads $M_i$. Hence, a two-unit cycle can be defined by $A(\sigma_1) = A(\sigma_2) = \emptyset$ and $S'_j^U = (M_{i_1}^U, \sigma_1, M_{i_1}^U, \sigma_2, M_{i_2}^U, \sigma_3, M_{i_2}^U, \sigma_4) = (M_{i_1}^U, M_{i_1}^U, \sigma_1, M_{i_2}^U, M_{i_2}^U, \sigma_3, M_{i_2}^U, \sigma_4)$. (We use the schedule’s cyclicity to present it in a more intuitive form.) This case is preferred for machines with large processing times.

2. For Case 9, the robot visits $M_i$ only once. The robot arrives at $M_i$ with two parts, loads one, waits for its entire processing, unloads it, rotates grippers, loads the second part, waits for its entire processing, and unloads it: $S'_j^U = (M_{i_1}^U, M_{i_1}^U, M_{i_2}^U, M_{i_2}^U, \sigma_4)$. This case is preferred for machines with small processing times.

If $i \in U_9$, then the robot arrives at $M_i$ holding two parts and leaves $M_i$ holding two parts. Hence, neither $i - 1$ nor $i + 1$ can belong to $U_9$, because this case requires that the robot arrives and departs the machine holding exactly one part. For this reason, a machine with a large processing time adjacent to one with a small processing time poses a scheduling challenge.

The following two cases serve as junctions between machines in Case 8 and machines in Case 9. This is because the robot can arrive at a machine in Case 5 with two parts and leave with one, whereas the robot can arrive at a machine in Case 7 with one part and leave with two.

1. For Case 5, the robot visits $M_i$ twice. In the first visit, the robot loads $M_i$, waits for it to process the part, unload $M_i$, rotates its grippers, then loads $M_i$. The robot only unload $M_i$ during its second visit: $S'_j^U = (M_{i_1}^U, M_{i_1}^U, \sigma_1, M_{i_2}^U, \sigma_3, M_{i_2}^U, \sigma_4)$.

2. Case 7 is a mirror image of Case 5. In the first visit, the robot unload $M_i$, rotates its grippers, loads $M_i$, waits for it to process the part, then unloads $M_i$. The robot only loads $M_i$ during its second visit. For this case, $S'_j^U = (M_{i_1}^U, M_{i_1}^U, M_{i_2}^U, \sigma_2, M_{i_2}^U, \sigma_3)$.

Case 6 is often used for a machine with a small processing time that is between two machines with large processing times.

1. For Case 6, the robot visits $M_i$ twice, and during each visit it loads $M_i$, waits for it to process the part, then unload $M_i$. Hence, $S'_j^U = (M_{i_1}^U, M_{i_1}^U, \sigma_2, \sigma_2, M_{i_2}^U, \sigma_2, M_{i_2}^U, \sigma_4)$.

Lower bounds (derived in Geismar et al. (2006)) on the robot’s residence time at a machine and on the robot’s movement time to and from that machine for each of the above cases are shown in Table 4, along with which sets $A(\sigma_i)$ are empty and the robot’s actions during each visit.

The robot visits $M_0$ (resp., $M_{m+1}$) the same number of times that it visits $M_1$ (resp., $M_{m}$). If $M_1$ (resp., $M_{m}$) is in Case 9, then the robot carries two parts from $M_0$ to $M_1$ (resp., $M_{m}$ to $M_{m+1}$) after its only visit to $M_0$ (resp., $M_{m}$). Otherwise, it visits $M_0$ (resp., $M_{m+1}$) twice, unload one raw part (resp., loading one completed part) each time.

Given a two-unit cycle, the following procedure forms subsets of machine indices $\{0, 1, \ldots, m, m + 1\}$ that are useful in proving the results of this section.

Procedure grouping: By starting with $M_0$ and scanning the machines in a forward pass until reaching $M_{m+1}$, we group the machines into non-exhaustive disjoint sets 

| Cases | Sets | Residence times | Movement times | Characteristics | Robot actions |
|-------|------|-----------------|----------------|----------------|---------------|
| Case 5 | $U_5$ | $4\theta + \theta + p_i$ | $2\delta$ | $A(\sigma_1) = A(\sigma_2) = \emptyset$ (or $A(\sigma_3) = A(\sigma_4) = \emptyset$) | $M_0^U, M_1^U, M_2^U - M_4^U$ |
| Case 6 | $U_6$ | $4\theta + 2p_i$ | $2\delta$ | $A(\sigma_1) = A(\sigma_2) = \emptyset$ | $M_0^U, M_1^U, M_2^U - M_4^U$ |
| Case 7 | $U_7$ | $4\theta + 2p_i$ | $2\delta$ | $A(\sigma_1) = A(\sigma_2) = \emptyset$ (or $A(\sigma_3) = A(\sigma_4) = \emptyset$) | $M_0^U, M_1^U, M_3^U, M_4^U$ |
| Case 8 | $U_8$ | $4\theta + 2p_i$ | $2\delta$ | $A(\sigma_1) = A(\sigma_2) = A(\sigma_3) = \emptyset$ | $M_0^U, M_1^U, M_3^U, M_4^U$ |
| Case 9 | $U_9$ | $4\theta + 2p_i$ | $2\delta$ | (or $A(\sigma_1) = A(\sigma_3) = A(\sigma_4) = \emptyset$) | $M_0^U, M_1^U, M_3^U, M_4^U$ |
subsets $\mathcal{M}_j$, $j = 1, 2, \ldots, t$, of consecutive machines, $\mathcal{M}_j = \{M_{j_1}, M_{j_1+1}, \ldots, M_{j_t+1}\}$, such that

(i) the subsets $\mathcal{M}_j$, $j = 1, 2, \ldots, t$, are arranged so that their first machine indices are in ascending order. That is, $0 \leq j_1 < j_2 < \cdots < j_t < j_{t+1} = m$;

(ii) the robot passes across each interval connecting a pair of adjacent machines in subset $\mathcal{M}_j$ at least twice; and

(iii) the robot passes across the interval of machines $M_{j_1+1}, \ldots, M_{j_t+1}$; i.e., those in no subset $\mathcal{M}_j$, where $j = 1, \ldots, t$, and $M_{j_t+1} \equiv M_j$, only once.

In the example in Fig. 3, $t = 2$, $\mathcal{M}_1 = \{M_2, M_3, M_4\}$, and $\mathcal{M}_2 = \{M_5, M_6, M_7, M_{10}\}$.

The definition of $\mathcal{M}_j$ by Procedure Grouping implies the following Properties.

$P_1$: The cardinality of $\mathcal{M}_j$ satisfies $|\mathcal{M}_j| \geq 2$. Let $n_j \equiv |\mathcal{M}_j| - 1$. The travel time across the machines in $\mathcal{M}_j$ is at least $3n_j \delta$, and the residence time across the machines in $\mathcal{M}_j$ is at least $4(n_j + 1) \epsilon$, $j = 1, 2, \ldots, t$. If $i_0 = 1$, then the lower bound on the residence time in $\mathcal{M}_1$ is $(4n_1 + 2) \epsilon$. If $i_0 + n_1 + n_i = m + 1$, then the lower bound on the residence time in $\mathcal{M}_i$ is $(4n_i + 2) \epsilon$.

$P_2$: If $M_i \in \mathcal{M}_j$, $j \in \{1, \ldots, t\}$, then the robot visits $M_i$ at least twice, so $i \not\in U_0$.

$P_3$: Machines $M_{j_1+1}, M_{j_1+2}, \ldots, M_{j_t+1}$, $j = 1, 2, \ldots, t - 1$, which are located between the sets $\mathcal{M}_j$ and $\mathcal{M}_{j+1}$, are each visited once during cycle $S''_t$ and therefore belong to $U_0$.

$P_4$: Machines $M_{j_t+2}, M_{j_t+3}, \ldots, M_{i-1}$, which are located between $M_{j_t+1}$ and $M_i$, and machines $M_{i+1}, M_{i+2}, \ldots, M_m$, which are located between $M_{i+1}$ and $M_{m+1}$, are each visited once during cycle $S''_t$ and therefore belong to $U_0$.

The smallest possible $\mathcal{M}_j$ has $n_j = 1$: $\{M_{j_1}, M_{j_1+1}\}$. Since $M_{j_1-1}, M_{j_1+2} \in U_0$, the robot arrives at $M_{j_1}$ with two parts and leaves $M_{j_1+1}$ with two parts, and $M_{j_1}$ and $M_{j_1+1}$ are empty at the start and at the end of the robot's serving of these two machines. Hence, $M_j \in U_0 \cup U_6$ and $M_{j_t+1} \in U_6 \cup U_7$. Because the assignment of both to $U_6$ is dominated by the assignment of both to $U_0$, we may ignore the assignment $<U_6, U_6>$. We have the following segment execution times for the remaining assignments:

$$\hat{T}(U_5, U_6) = p_{i_j} + 2p_{i_j+1} + 3\delta + 8\epsilon + \theta,$$

$$\hat{T}(U_6, U_7) = 2p_{i_j} + p_{i_j+1} + 3\delta + 8\epsilon + \theta,$$

$$\hat{T}(U_5, U_7) = p_{i_j} + p_{i_j+1} + 3\delta + 8\epsilon + 2\theta.$$

Hence, for $n_j = 1$, we may choose the assignment for which

$$\hat{T}(\mathcal{M}_j) = \min(p_{i_j}, p_{i_j+1}, \theta) + p_{i_j} + p_{i_j+1} + 3\delta + 8\epsilon + \theta. \quad (6)$$

The following lemma and corollary use the above discussion of two-unit cycles to develop lower bounds on the per unit cycle time of any $k$-unit cycle ($k \geq 1$). The lemma's lower bound applies to two-unit cycles with a given collection of sets $\mathcal{M}_j$, $j = 1, \ldots, t$. The corollary generalizes this lower bound to all $k$-unit cycles, $k \geq 1$, for any collection of $\mathcal{M}_j$ sets. We assume that the first $n_i$ of the $\mathcal{M}_j$ are the ones with $n_j = 1$. This assumption causes no loss of generality because the $\mathcal{M}_j$ sets' executions are independent of each other. For each $\mathcal{M}_j$, let $h_j \equiv \argmax_{1 \leq s \leq n_{j-1}} \{p_{i_j+s}\}$. Let $n = \sum_{j=1}^t n_j$.

**Lemma 8.** If $\theta \leq \delta$ and $p_{\text{max}} \leq (m+2)\delta + (m-1)\epsilon$, then for given sets $\mathcal{M}_j$, $j = 1, \ldots, t$, the following lower bounds apply to the cycle time of any two-unit cycle $S''_t$:

$$T(S''_t) \geq L_1 \equiv (m+2n+2)\delta + (m+1)\epsilon$$

$$+ \sum_{j=1}^t \sum_{s=0}^{n_j} 2 \min(p_{i_j+s}, \theta) + \sum_{s \not\in J} \{p_{i_j+s}\} \quad (7)$$

$$T(S''_t) \geq L_2 \equiv \sum_{j=1}^t \{p_{i_j} + p_{i_j+1}\} + \sum_{j=1}^t \{p_{i_j+h_j}\}$$

$$+ (m+2)\delta + (4m-4n + 6t - 2t_1 + 4)\epsilon$$

$$+ (m-n+3t-3t_1+2)\theta + \sum_{s \not\in J} \{2p_s\}. \quad (8)$$

Lower bounds (7) and (8) of Lemma 8 depend on sets $\mathcal{M}_j$, $j = 1, \ldots, t$, which are assumed to be given. To find lower bounds over all two-unit cycles, we must find the collection of $\mathcal{M}_j$ sets that minimizes the maximum of lower bounds (7) and (8). The following corollary states that the resulting lower bound applies to $k$-unit cycles for all $k \geq 1$.

**Corollary 1.** If $\theta \leq \delta$ and $p_{\text{max}} \leq (m+2)\delta + (m-1)\epsilon$, then either cycle $S''_k$ is optimal over all $k$-unit cyclic solutions, or the following lower bound applies to the per unit cycle time $T(S''_k)/k$ of any $k$-unit cycle $S''_k$:

$$\frac{2T(S''_k)}{k} \geq \min \{L_1, L_2\}. \quad (9)$$

**Remark 3.** Corollary 1 and the logic of its proof imply that a cycle that provides a 3/2-approximation to an optimal two-unit cycle also provides a 3/2-approximation to an optimal $k$-unit cycle, for all $k \geq 1$.

Corollary 2 demonstrates that cycle $S''_k$ provides a 3/2-approximation for many cells on which Procedure Grouping has been run.

**Corollary 2.** Let $\theta \leq \delta$ and $p_{\text{max}} \leq (m+2)\delta + (m-1)\epsilon$. Given sets $\mathcal{M}_j$, $j = 1, \ldots, t$, if $n \geq (m+2)/6$, then cycle $S''_k$ provides a 3/2-approximation.
6.2. Epicyclic cycles: procedure transform

The results of Section 5 demonstrate that the dual-gripper cycles \( S_0'' \) and \( S_q'' \) (including its special case \( S_q'' \)) are very effective for many implementations with \( \theta \leq 2\delta \). We combine the characteristics of these two cycles to form a new class of cycles called *epicyclic cycles*, which we show provide effective approximations for cells with \( \theta \leq \delta \). This process requires care and precision because, although \( S_0'' \) and \( S_q'' \) each instruct the robot to visit each machine once in ascending order, \( S_0'' \) is a two-unit cycle in which all machines belong to Case 9, and \( S_q'' \) is a one-unit cycle in which all machines belong to either Case 6 or Case 8. Recall that a machine in Case 9 cannot be adjacent to one in Case 8.

We now specify how a two-unit epicyclic cycle is created from any two-unit cycle on which Procedure Grouping has been run. By demonstrating how any two-unit cycle can be converted to an epicyclic cycle whose throughput is no worse than the original cycle, we prove that the set of epicyclic cycles dominates the set of two-unit cycles. We distinguish epicyclic cycles by denoting them and their associated sets and machine indices with overlines.

In two-unit epicyclic cycle \( S_f \), once the robot visits one machine of a subset \( M_j \), it visits each machine in subset \( M_j \) exactly twice before it moves to any machines not in \( M_j \). Because the robot cycles through each \( M_j \) while performing cycle \( S_f \), we call the robot activities in each such subset an *epicycle*. (The term *epicycle* comes from the Ptolemaic geocentric model of the solar system in which small circles (epicycles) were added to planets’ orbits to explain anomalies such as retrograde motion.) An *epicyclic cycle* is a two-unit cycle composed of epicycles and the robot activities at the remaining machines, each of which is visited only once (and, hence, belongs to Case 9). For simplicity, we call these *E-cycles*. We illustrate E-cycles via an example:

**Example A:** Consider cycle \( \overline{S_f} \) for \( m = 13 \) and \( t = 2 \), where \( \overline{S_f} = (M_6, M_9, M_2, M_1, M_4, M_5, M_7, M_6, M_3, M_4, M_2, M_1, M_3, M_4, M_1, M_5, M_6, M_7, M_9, M_2, M_1, M_4, M_5, M_7, M_6, M_9, M_2, M_1, M_4, M_5, M_7, M_6, M_9, M_2, M_1, M_4, M_5, M_7, M_6, M_9, M_2, M_1, M_4, M_5, M_7, M_6, M_9, M_2, M_1, M_4, M_5, M_7, M_6, M_9) \) (see Fig. 3). In cycle \( \overline{S_f} \), there are two epicycles, which are formed by the robot’s activities in machine sets \( \{M_3, M_9, M_6\} \) and \( \{M_5, M_6, M_8, M_9, M_{10}\} \). In each epicycle, the first and the last machine belong to Cases 5 and 7, respectively, and each remaining machine \( M_i \) in the epicycle is assigned as if it were in Quick Cycle \( S_q'' \); it belongs to Case 6 if \( p_s < \theta \) and to Case 8 if \( p_s \geq \theta \). All other machines (those in no epicycle) belong to Case 9. Similar assignments hold in all E-cycles, except for epicycles with \( \bar{n}_j = 1 \), which are explained below. We now define Procedure Transform to create E-cycles.

**Procedure Transform:** Procedure Transform assumes that Procedure Grouping, based on some two-unit cycle \( S_f \), has created sets \( M_j \), \( j = 1, \ldots, t \), of consecutive machines. It converts cycle \( S_f \) into an E-cycle \( \overline{S_f} \). E-cycle \( \overline{S_f} \) begins as an exact copy of \( S_f \) with \( \overline{M_j} = M_j, \overline{n}_j = n_j \), and \( \bar{i} = i_j \), for \( j = 1, \ldots, t \), and \( \bar{i} = \bar{t} \). Each of its four steps sequentially...
addresses each set $\overline{M}_j$. Step 1 of Procedure Transform applies only to sets $\overline{M}_j$ with a special structure. Step 2 considers the first machine ($M_i$) of each $\overline{M}_j$, and Step 3 considers the last machine ($M_i+\bar{\delta}$). Step 4 addresses the internal machines $M_i+1, \ldots, M_i+\bar{\delta}−1$ of each $\overline{M}_j$. Note that for each $j = 1$ to $\bar{t}$, $\{M_i−1, M_i+\bar{\delta}−1\} \subseteq U_0$ implies that machines $M_i$ and $M_i+\bar{\delta}$ must each belong to $U_2 \cup U_3 \cup U_4 \cup U_5 \cup U_6 \cup U_7$ in cycle $S_F^j$.

**Step 1:** This step applies only to two special cases. The travel time across the machines in $\overline{M}_j$ is equal to $3\bar{n}_j\bar{\delta}$ in each case. In the first one, machine $M_i$ belongs to $U_5$. This implies that machine $M_i+\bar{\delta}$ belongs to $U_7$ and that all other machines in $\overline{M}_j$ belong to $U_6$. The second case is symmetric: $M_i$ belongs to $U_5$ and all other machines in $\overline{M}_j$ belong to $U_6$. For simplicity, we address only the first of these two possible configurations, though the algorithm’s actions are symmetric in the second.

To execute this sequence, the robot departs $M_i−1, (\bar{t}_j−1 \in U_0)$ with two parts, say $X_i$ and $X_{i+1}$, whose next processing is on $M_i$. The robot loads part $X_i$ onto $M_i$, waits for its entire processing time $\bar{t}_j$, unloads it, then moves to $M_i+\bar{\delta}$. The robot performs these same actions at machines $M_i+1, \ldots, M_i+\bar{\delta}−1$, all while holding part $X_{i+1}$ in its second gripper. It then loads part $X_{i}$ onto $M_i+\bar{\delta}$, and returns to $M_i$ to perform the sequence load, wait, unload at each machine $M_i, \ldots, M_i+\bar{\delta}−1$ with part $X_{i+1}$. At $M_i+\bar{\delta}$, the robot unloads part $X_i$, rotates its grippers, loads part $X_{i+1}$, waits for its entire processing, then unloads part $X_{i+1}$.

We now make changes to $\overline{M}_j$ that will increase the cycle’s throughput. First, reduce the size of $\overline{M}_j$ to two machines ($\bar{n}_j = 1$) by moving machines $M_i, \ldots, M_i+\bar{\delta}−2$ from $U_5$ to $U_0$; this reduces the cycle’s execution time by $(\bar{t}_j−1)(2\bar{\delta}−\bar{\theta})$. The two remaining machines in $\overline{M}_j$ are relabeled $M_i$ and $M_i+\bar{\delta}$. We previously saw that $\bar{n}_j = 1$ implies three possible assignments: $<U_5, U_6>, <U_5, U_7>, \text{ or } <U_6, U_7>$. However, we also consider the possibility of having no $\overline{M}_j$ here (assignment $<U_0, U_0>$) or of expanding $\overline{M}_j$ so that $\bar{n}_j = 2$. This assignment is determined by **Subroutine Assign1**. For clarity, the details are provided after the description of Procedure Transform.

**Step 2:** If $p_i−1 + p_i ≤ 2\bar{\delta} + \bar{\theta}$ (excluding $\bar{n}_j = 1$ with assignment $<U_0, U_7>$), then machine $M_i$ is assigned to $U_5$. Otherwise, $p_i−1 + p_i > 2\bar{\delta} + \bar{\theta}$, so the following revision to $\overline{M}_j$ is made under two cases:

**Case 2(i):** $M_i−1 \in U_0 \text{ or } M_i−1 = I$ (i.e., $\bar{t}_j−1 = 0$, $j = 1, \ldots, \bar{t}$). Add machine $M_i−1$ to $\overline{M}_j$ so that $\overline{M}_j = \{M_i−1, M_i, M_i+1, \ldots, M_i+\bar{\delta}\}$; move $M_i$ to $U_6$ if $p_i ≥ \theta$, to $U_0$ otherwise. If $M_i−1 \neq I$, then move $M_i−1$ to $U_5$. Update $\bar{n}_j ← \bar{n}_j + 1$ and $\bar{t}_j ← \bar{t}_j−1$.

**Case 2(ii):** $M_i−1 \in \overline{M}_j−1$; i.e., $\bar{t}_j−1 = \bar{t}_j−1 + \bar{n}_j−1$, $j = 2, \ldots, \bar{t}$. In this case, we join the two sets $\overline{M}_j−1$ and $\overline{M}_j$ to create $\overline{M}_j = \overline{M}_j−1 \cup \overline{M}_j$. Update $\bar{n}_j ← \bar{n}_j−1 + \bar{n}_j + 1$. Re-index the machines of the new $\overline{M}_j$ so that $\overline{M}_j = \{M_i, \ldots, M_i+\bar{\delta}\}$.

**Step 3:** If $p_i−1 + p_i + 1 \leq 2\bar{\delta} + \bar{\theta}$ (excluding $\bar{n}_j = 1$ with assignment $<U_5, U_6>$), then machine $M_i+\bar{\delta}$ is assigned to $U_7$. Otherwise, $p_i−1 + p_i + 1 > 2\bar{\delta} + \bar{\theta}$, so a revision to $\overline{M}_j$ is made under two cases:

**Case 3(i):** $M_i+\bar{\delta}+1 \in U_0 \text{ or } M_i+\bar{\delta}+1 = O$ (i.e., $\bar{t}_j + \bar{n}_j + 1 = m + 1$). Add machine $M_i+\bar{\delta}+1$ to $\overline{M}_j$ so that $\overline{M}_j = \{M_i, M_i+1, \ldots, M_i+\bar{\delta}, M_i+\bar{\delta}+1\}$; move $M_i+\bar{\delta}$ to $U_6$ if $p_i+\bar{n}_j ≥ \theta$, to $U_0$ otherwise. If $M_i+\bar{\delta}+1 \neq O$, then move $M_i+\bar{\delta}+1$ to $U_7$. Update $\bar{n}_j ← \bar{n}_j + \bar{n}_j + 1$.

**Case 3(ii):** $M_i+\bar{\delta}+1 \in \overline{M}_j−1$, i.e., $\bar{t}_j + \bar{n}_j + 1 = \bar{t}_j+1$. In this case, we join the two sets $\overline{M}_j$ and $\overline{M}_j−1$ to create $\overline{M}_j = \overline{M}_j−1 \cup \overline{M}_j$. Update $\bar{n}_j ← \bar{n}_j + \bar{n}_j + 1$.

Repeat Step 3 on the revised $\overline{M}_j$ until no more revision is needed; i.e., until $p_i+\bar{n}_j + p_i+\bar{n}_j + 1 \leq 2\bar{\delta} + \bar{\theta}$ or $\bar{t}_j + \bar{n}_j = m$.

Clearly, each of the transformations of Steps 2 and 3 reduces the cycle’s execution time.

**Step 4:** At this point, all $\overline{M}_j$ begin with $M_i \in U_5 \cup \{I\}$ and end with $M_i+\bar{n}_j \in U_7 \cup \{O\}$, $j = 1, \ldots, \bar{t}$, except for those with $\bar{n}_j = 1$ and assignment $<U_5, U_6>$ or $<U_5, U_7>$ ($\overline{M}_j, \bar{t}_j, \bar{n}_j$, and $\bar{t}_j$ now refer to their values after they have been adjusted by Steps 1 to 3, if appropriate). We complete $S_F^j$ by assigning the remaining machines $M_i$ in $\overline{M}_j$ to set $U_5$ if $p_i < \theta$, to set $U_0$ otherwise.

We now specify **Subroutine Assign1**. For clarity, this process of determining the assignment of Step 1, which depends on four combinations of machines’ $M_i$, $M_i+\bar{\delta}$ processing times, is described in algorithm form. The numbers in brackets refer to the corresponding rows of Table 5.

**Subroutine Assign1**

If $p_i \leq 2\bar{\delta} + \bar{\theta}$, $p_i+1 \leq 2\bar{\delta} + \bar{\theta}$, and $p_i + p_{i+1} \leq 2\bar{\delta}$ then remove $\overline{M}_j$ by changing both $M_i$ and $M_i+\bar{\delta}$ to $U_0$ [1]

Else If $p_i ≥ \theta$, $p_{i+1} ≥ \theta$, and $p_i + p_{i+1} ≥ 2\bar{\delta}$ then change $M_i$ to $U_5$ and keep $M_i+\bar{\delta}$ in $U_7$ [2]

Else If $p_i ≤ \theta$ and $p_{i+1} ≥ 2\bar{\delta} − \bar{\theta}$ then

If $p_i + p_{i+1} + p_{i+2} ≤ 2\bar{\delta} + 2\bar{\theta}$ then no further changes are made ($M_i \in U_6, M_i+1 \in U_7$, and $M_i+2 \in U_6$) [3]

Else ($p_i + p_{i+1} + p_{i+2} > 2\bar{\delta} + 2\bar{\theta}$) change $M_i$ to $U_5$, $M_i+1$ to $U_8$, and $M_i+2$ to $U_7 (\bar{n}_j = 2)$.

Otherwise ($\bar{n}_j ≥ 2\bar{\delta} − \bar{\theta}$ and $p_{i+1} ≤ \theta$) Then $p_i+1 + p_{i+1} + p_{i+2} ≤ 2\bar{\delta} + 2\bar{\theta}$ then keep $M_i−1 \in U_0$, and assign $M_i$ to $U_5$
and $M_{j+1}$ to $U_6$ [4].

Else ($p_{j-1} + p_j + p_{j+1} > 2\theta$)

change $M_{j-1}$ to $U_5$, $M_j$ to $U_8$, and $M_{j+1}$ to $U_7$ ($\bar{n}_j = 2$).

Execution times for these assignments (except for those with $\bar{n}_j = 2$) are compared in Table 5.

The reader may verify that cycle $\bar{S}_T$ formed by this new assignment of machines $M_{j+1}, \ldots, M_{j+\bar{n}_j-1}$, $j = 1, \ldots, t$, to set $U_6$ or set $U_8$ is feasible. For example, suppose $m = 7$ and cycle $\bar{S}_T$ has $t = 1$ with $M_1 = \{M_2, M_3, M_5, M_6\}$ and $p_1 < \Delta + \theta$, $p_2 > \Delta + \theta$, $p_3 \geq \theta$, $p_4 < \theta$, and $p_5 + p_6 < \Delta + \theta$. Since $p_2 > \Delta + \theta$, we revise $M_1$ to $\overline{M}_1 = \{M_1, M_2, M_3, M_4, M_5\}$ and assign machines $M_1, \ldots, M_t$ to sets in $\overline{S}_T$ as $U_5$, $U_8$, $U_6$, $U_7$, $U_6$, $U_8$, respectively, so $\overline{S}_T$ is feasible.

Note that cycle $\overline{S}_T$ is the E-cycle with $t = 0$: $\{1, \ldots, m\} = U_6$. The reader may refer to Online Appendix C for applications of Procedures Grouping and Transform.

6.3. A 3/2-Approximation: algorithm epi-path

We show that cycle $\overline{S}_T$ (via Lemma 7 and Corollary 2) and the class of E-cycles provide a 3/2-approximation when $\theta \leq \Delta$ by forming a two-unit E-cycle $\overline{S}_T$ from an optimal k-unit cycle.

Theorem 1. If $\theta \leq \Delta$, then the better of one-unit cycle $S_T$ and two-unit E-Cycle $\overline{S}_T$ has a per unit cycle time that provides a 3/2-approximation to the optimal per unit cycle time. This bound is tight.

Define set $E$ to consist of all E-cycles. We now present an algorithm to find a cycle within $E$ that has the greatest throughput. We first prove a result that reduces the number of members of this set that must be considered.

Lemma 9. If $\theta \leq \delta$ and $p_{\text{max}} \leq (m + 2)\delta + (m - 1)\epsilon$, then there exists an E-cycle $\overline{S}_T$ that has the maximum throughput within $E$ whose total partial waiting during each epicycle $M_j$, $j = 1, \ldots, t$, is less than $2\delta$.

The following algorithm finds cycle $\overline{S}_T$, which is an E-cycle with the maximum throughput within $E$. The algorithm forms a network of $m + 1$ nodes and then finds a shortest path from node 1 to node $m + 1$. An arc from node $s$ to node $\ell$ represents the time required to execute all robot actions between the robot’s first arrival at machine $M_s$ and its first arrival at machine $M_\ell$, $s = 1, \ldots, m$; $\ell = s + 1, \ldots, m + 1$. Arcs beginning at node 1 also include the robot’s residence time at $I$ and its travel time from $I$ to $M_1$. Arcs ending at node $m + 1$ also include the robot’s residence time at $O$ and its travel time from $O$ to $I$, which completes the cycle. We limit the size of epicycles to $\bar{n}_j \leq (m + 2)/2$ because no E-cycle with $\bar{n} \geq (m + 2)/2$ can have higher throughput than cycle $\overline{S}_T$.

Algorithm Epi-path: Form an acyclic digraph $G$ with $m + 1$ nodes and the following arcs:

1. The arc connecting node $s$ to node $s + 1$ has length $2p_\delta + 4\epsilon + \theta$, $s = 1, \ldots, m$, which represents $s \in U_9$. For $s = 1$ (resp., $s = m$), $\delta + 2\epsilon + \theta$ is added to the arc length to account for the movement from $I$ to $M_1$ (resp., $O$ to $I$), the unloading at $I$ (resp., loading at $O$), and the gripper rotation at $I$ (resp., $O$).

2. For $p_\delta \leq \theta$ and $p_{j+1} \geq \Delta - \theta$, there is an arc connecting node $s$ to node $s + 2$ with length $\max\{2p_\delta + p_{j+1} + 4\delta + 8\epsilon + \theta, p_\delta + 2p_{j+1} + 2\delta + 6\epsilon + \theta\}$, which represents $s \in U_6$, $s + 1 \in U_7$, $s = 1, \ldots, m - 1$. The two arguments represent no positive partial waiting at $M_{j+1}$ and positive partial waiting at $M_{j+1}$, respectively. For $s = 1$ (resp., $s = m - 1$), $\delta + 2\epsilon + \theta$ is added to the arc length to account for the movement from $I$ to $M_1$ (resp., $I$ to $O$), the unloading at $I$ (resp., loading at $O$), and the gripper rotation at $I$ (resp., $O$).

3. For $p_\delta \geq \theta$ and $p_s + p_{j+1} \geq \Delta - \theta$, there is an arc connecting node $s$ to node $s + 2$ with length $\max\{p_\delta + p_{j+1} + 4\delta + 8\epsilon + 2\theta, 2p_\delta + p_{j+1} + 2\delta + 7\epsilon + 2\theta, p_\delta + 2p_{j+1} + 2\delta + 7\epsilon + 2\theta\}$, which represents $s \in U_3$, $s + 1 \in U_7$, $s = 1, \ldots, m - 1$. The three arguments represent positive partial waiting at neither machine, positive partial waiting at $M_{j+1}$ but not $M_{j+1}$, and positive partial waiting at $M_{j+1}$, respectively. For $s = 1$ (resp., $s = m - 1$), $\delta + 2\epsilon + \theta$ is added to the arc length to account for the movement from $I$ to $M_1$.
4. For $p_s \geq 2\delta - \theta$, $p_{s+1} \leq \theta$ and $p_{s+1} + p_s + p_{s+1} \leq 2\delta + 2\theta$, there is an arc connecting node $s$ to node $s + 2$ with length
\[
\max\{p_s + 2p_{s+1} + 4\delta + 8\epsilon + \theta, 2p_s + p_{s+1} + 2\delta + 6\epsilon + \theta, p_s + 3p_{s+1} + 2\delta + 7\epsilon + \theta\};
\]
which represents $s \in U_5$, $s + 1 \in U_6$, $s = 1, \ldots, m - 1$. The three arguments represent positive partial waiting at neither machine, positive partial waiting at $M_i$ but not $M_{i+1}$, and positive partial waiting at $M_{i+1}$, respectively. For $s = 1$ (resp., $s = m - 1$), $\delta + 2\epsilon + \theta$ is added to the arc length to account for the movement from $I$ to $M_i$ (resp., $I$ to $O$), the unloading at $I$ (resp., loading at $O$), and the gripper rotation at $I$ (resp., $O$).

5. The arc connecting node $s$ to node $\ell$, for $s = 1, \ldots, m - 2$, and $\ell = s + 3, \ldots, m$, $\max\{m + 1, s + 2 + [(m + 2)/2]\}$, represents an epicycle with $i_j = s$ and $l_j + n_j = \ell - 1$. This arc has length
\[
\max\left\{p_s + p_{\ell - 1} + 3(\ell - s - 1) + 1\right\} + 4(\ell - s)\epsilon
+ 2\theta + \sum_{q=s+1}^{\ell-2} 2\min\{p_q, \theta\},
\]
\[
\max\left\{p_s + 3\epsilon, p_{\ell - 1} + 3\epsilon, \max_{s+1 \leq q \leq \ell - 2}\{p_q\} + 4\epsilon + \theta\right\}
+ p_s + p_{\ell - 1} + 2\theta + (\ell - s)\delta + 2(\ell - s)\epsilon
+ \sum_{q=s+1}^{\ell-2} \min\{p_q, \theta\}.
\]
The second argument is the execution time if the robot has positive partial waiting at any machine within $\{M_i, \ldots, M_{i-1}\}$. If $s = 1$ (resp., $\ell = m + 1$) an extra $\delta + 2\epsilon + \theta$ is added to this formula to account for the movement from $I$ to $M_i$ (resp., $O$ to $I$), the unloading at $I$ (resp., loading at $O$), and the gripper rotation at $I$ (resp., $O$). Observe that we may omit these arcs if $p_{s-1} + p_s > 2\delta + \theta$ for $s \geq 2$, or if $p_{\ell} + p_{\ell+1} > 2\delta + \theta$ for $\ell \leq m - 1$.

6. The final set contains arcs that represent epicycles that begin at $I$ (resp., end at $O$). In these, the robot visits $I$ (resp., $O$) twice. They are used if $p_1 > 2\delta$ (resp., $p_m > 2\delta$), so $M_i \in U_8$ (resp., $M_m \in U_8$). These arcs connect node 1 to node $\ell$, $\ell = 3, \ldots, 2 + [(m + 2)/2]$ (resp., node $s$ to node $m+1$, $s = [(m + 2)/2] - 1, \ldots, m - 2$). Their lengths are the same as those of the previous class, minus $p_1 + 2\epsilon + \theta$ (resp., $p_\ell + 2\epsilon + \theta$), and they have no argument for partial waiting at $M_i \in U_8$ (resp., $M_{m-1} \in U_7$). The arc lengths do account for partial waiting at $M_i \in U_8$ (resp., $M_m \in U_8$):
\[
\hat{T}(I \to M_i)
= \max\left\{p_{\ell-1} + (3\ell - 2)\delta + (4\ell - 2)\epsilon + \theta
+ \sum_{q=1}^{\ell-2} 2\min\{p_q, \theta\}, p_{\ell-1} + \theta + \ell\delta + 2\ell\epsilon
+ \max_{1 \leq q \leq \ell - 2}\{p_q\} + 4\epsilon + \theta
+ \sum_{q=1}^{\ell-2} \min\{p_q, \theta\}\right\},
\]
\[
\hat{T}(M_i \to O)
= \max\left\{p_s + [3(m - s) + 1]\delta + [4(m - s) + 2]\epsilon
+ \theta + \sum_{q=s+1}^{m-1} 2\min\{p_q, \theta\}, p_s + \theta + (m + 1 - s)\delta
+ 2(m + 1 - s)\epsilon + \max_{s+1 \leq q \leq m-1}\{p_q\}
+ 4\epsilon + \theta + \sum_{q=s+1}^{m-1} \min\{p_q, \theta\}\right\}.
\]
As before, we may omit these arcs if $p_{s-1} + p_s > 2\delta + \theta$ for $s \geq 2$, or if $p_{\ell} + p_{\ell+1} > 2\delta + \theta$ for $\ell \leq m - 1$.

**Lemma 10.** If $\theta \leq \delta$ and $p_{\max} \leq (m + 2)\delta + (m - 1)\epsilon$, then Algorithm Epi-path finds cycle $S'_\epsilon$ in polynomial-time $O(m^2)$.

We can now state the following general result for $\theta \leq 2\delta$.

**Corollary 3.** If $\theta \leq 2\delta$, then either cycle $S'_\epsilon$ or cycle $S''_\epsilon$, given by Algorithm Epi-path provides a $3/2$-approximation to the optimum per unit cycle time over all k-unit cycles.

### 6.4. Main result: approximation Algorithm QCYLE for k-unit cycles

We now present an algorithm and the main result of the article for dual-gripper and single-gripper robotic cells. We use the facts that $S''_\epsilon$ is a special case of $S'_\epsilon$ and that $S'_\epsilon$ is an E-cycle to simplify the algorithm.

**Algorithm QCYLE**

**Input:** $m, \delta, \theta, \epsilon, p_i, i = 1, 2, \ldots, m$.

**Output:** A k-unit cycle, $k \in \{1, 2\}$

**Begin**

**Step 1:** If $\theta \geq \delta$ and $p_{\max} \leq \delta$, then output cycle $S'_\epsilon$. EXIT.

**Step 2:** If $\theta \geq 2\delta$, then output the best of cycles $S'_\epsilon$, $S''_\epsilon$, and $S'_\epsilon$. EXIT.
Step 3: If \( \theta \leq \delta \) and \( p_{\text{max}} \leq (m + 2)\delta + (m - 1)\epsilon \), then output the better of cycles \( S''_q \) and \( S'_{q,p} \), EXIT.

Step 4: Otherwise, output cycle \( S''_q \), EXIT.

End

As a consequence of the lemmas, we have the following result.

**Theorem 2.** For dual-gripper robotic cells, Algorithm QCYLE has the following performance guarantee to optimal \( k \)-unit cyclic solutions:

(a) a 5/3-approximation, if \( \theta > 2\delta \) and \( p_{\text{max}} \geq (m + 2)\delta + (m - 1)\epsilon \), and (b) a 3/2-approximation, otherwise.

When \( \theta \) is very large in a dual-gripper cell, the cell is in practice a single-gripper cell. As a consequence of Theorem 2 and the fact that single-gripper cycles and their lower bounds on optimal \( k \)-unit cycles dominate dual-gripper cycles and their lower bounds when \( \theta > 2\delta \) (see Remark 1 at the end of Section 5.1), we have the following result.

**Theorem 3.** For single-gripper robotic cells, Algorithm QCYLE provides a 5/3-approximation to optimal \( k \)-unit cyclic solutions.

### 7. Computational experiments

Our theoretical results provide bounds for the worst-case performance of our approximating cycles. We now examine how these cycles operate on average. We first briefly describe the primary objectives of our computational experiments. Section 7.1 gives a detailed statement of the test bed, and Section 7.2 provides the results.

**Objectives of the Study**

1. To test the performance of Algorithm QCYLE by finding its relative percentage above the maximum lower bound among those described in Section 4 and in Lemma 8.
2. To investigate how E-cycles improve the performance of Algorithm QCYLE.
3. To investigate how the arrangement of the machines’ processing times effects the value of E-cycles; i.e., to test whether E-cycles are more effective when the machines with larger processing times are bunched together, rather than being spread evenly throughout the cell.

#### 7.1. The test bed

We generate a realistic test bed based on Rajapakshe et al. (2011) to investigate our objectives. We consider seven different sizes of cells: \( m = 5, 10, 12, 15, 20, 25, 30 \). According to the theoretical results of Sections 4 to 6, the critical elements in our analysis are the maximum processing time \( p_{\text{max}} \) and the parameters \( \delta, \epsilon \), and \( \theta \). Consequently, we first specify three types of cells that are distinguished by the value of \( p_{\text{max}} \). Since we can always find the optimal solution if \( p_{\text{max}} \leq \min\{3m\delta + 2(m - 1)\epsilon, (m + 1)\delta + 2m\epsilon + (m - 1)\theta\} \), according to Lemmas 2 and 6, we only consider \( p_{\text{max}} \) values less than this amount.

- **Type 1:** \( 2\delta \leq p_{\text{max}} < 4\delta \);
- **Type 2:** \( 4\delta \leq p_{\text{max}} < (m + 2)\delta + (m - 1)\epsilon \);
- **Type 3:** \( (m + 2)\delta + (m - 1)\epsilon \leq p_{\text{max}} < \min\{3m\delta + 2(m - 1)\epsilon, (m + 1)\delta + 2m\epsilon + (m - 1)\theta\} \).

For each type of cell, we consider various combinations of the number of machines consisting of small, medium, and large processing times, where we define these terms by

- Small: \( p_t < \frac{\delta}{3} \)
- Medium: \( \frac{\delta}{3} \leq p_t < 2\delta \)
- Large: \( 2\delta \leq p_t < p_{\text{max}} \)

For example, in a five-machine cell, Combination 2-1-2 (denoted by C2) implies two machines assigned to small processing times, one machine assigned to a medium processing time, and two machines assigned to large processing times. Table 6 lists six combinations for \( m = 5 \), nine combinations for \( m = 10 \), and 10 combinations each for the other values of \( m \). Note that none of these combinations fits the criteria for those with known optimal cycles that were developed in Lemmas 1 to 7.

The processing times are randomly generated from uniform distributions within their assigned intervals and are randomly assigned to the machines. Two parameters are fixed: \( \delta = 9 \) and \( \epsilon = 1 \). To test our analysis, the gripper

### Table 6. Machine combinations for testing

| Number of machines (m) | C1     | C2     | C3     | C4     | C5     | C6     | C7     | C8     | C9     | C10    |
|------------------------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| 5                      | —      | 2-1-2  | 1-2-2  | 2-2-1  | 1-3-1  | 1-1-3  | 3-1-1  | —      | —      | —      |
| 10                     | 3-3-4  | 5-1-4  | 1-5-4  | 5-4-1  | 1-8-1  | 1-1-8  | 8-1-1  | 7-2-1  | 7-1-2  | —      |
| 12                     | 4-4-4  | 6-1-5  | 1-6-5  | 6-5-1  | 1-10-1 | 1-1-10 | 10-1-1 | 8-3-1  | 8-1-3  | 8-2-2  |
| 15                     | 5-5-5  | 7-1-7  | 1-7-7  | 7-7-1  | 1-13-1 | 1-1-13 | 13-1-1 | 11-3-1 | 11-1-3 | 11-2-2 |
| 20                     | 6-7-7  | 10-1-9 | 1-10-9 | 10-9-1 | 1-18-1 | 1-1-18 | 18-1-1 | 14-5-1 | 14-1-5 | 14-3-3 |
| 25                     | 8-8-9  | 12-1-12| 1-12-12| 12-12-1| 1-23-1 | 1-1-23 | 23-1-1 | 17-7-1 | 17-1-7 | 17-4-4 |
| 30                     | 10-10-10| 15-1-14| 1-15-14| 15-14-1| 1-28-1 | 1-1-28 | 28-1-1 | 20-9-1 | 20-1-9 | 20-5-5 |
instances with a large percentage of smaller processing times, from the numerator and the denominator. For some instances of the smaller processing times generally are canceled proving the theoretical bounds, the terms that represent the times (but, oddly, not the maximum amount of small processing times (combinations C2, C8, C9, and C10) but, oddly, not the maximum amount of small processing times (C7). We suspect two causes for this trend: first, when proving the theoretical bounds, the terms that represent the sums of the smaller processing times generally are canceled from the numerator and the denominator. For some instances with a large percentage of smaller processing times, leaving these sums in the ratio would have reduced its value. Second, as the number of large processing times grows, the chance of having a processing time \(p_{\text{max}}\) near its maximum, ensuring that QCYCLE finds an optimal cycle, increases.

As expected, the preference for dual-gripper robot cycles is negatively correlated to the time required for a gripper switch. For \(\theta = 2, 14, 28, 42\), the percentages of instances in which the dual-gripper robot has higher throughput are 100, 86.21, 76.77, and 55.69%, respectively.

We now evaluate the contribution of E-cycles. Since E-cycles were designed for cases not covered by Lemma 7, we limit the gripper switch time to \(\theta < \delta/3\), i.e., \(\theta = 2\), and processing time combinations to \(C7-C10\), so that \(r_2 > 2(m-1)/3\). This yields a total of 10 \(\times\) 24 \(\times\) 3 = 720 instances. An E-cycle uniquely produces the best throughput among all others considered by QCYCLE in 269 of these instances (37.36%). The E-cycle’s relative percentage throughput advantage over the best non-E-cycle averages 8.02%, and this advantage steadily increases as \(m\) increases: 3.75, 7.42, 6.36, 8.93, 9.18, 10.22, and 10.35%. Hence, E-Cycles are a valuable contribution to practitioners who wish to efficiently schedule robotic cells.

Our last study tests our conjecture that the value of the best E-cycle depends not only on the cell’s processing times, but also on the relative locations of the machines to which those times are assigned. This characteristic distinguishes the class of E-cycles from most other cycles. Specifically, we hypothesize that E-cycles produce higher throughput if the machines with larger processing times are closer together, rather than evenly spread throughout the cell. Our reasoning is that a cell with the larger times together will have fewer epicycles and, hence, less travel time.

To ensure meaningful data, we limit this study to combinations C8–C10, \(m \in \{12, 15, 20, 25\}\), for a total of 12 \(\times\) 10 = 120 instances of machine processing times. For each such instance, the times are arranged in the two ways previously mentioned. Our results show that E-cycles produce on average 7.85% higher throughput in cells with the larger processing times close to one another. This result gives managers further insight into which environments are best suited to using this new type of scheduling sequence.

### 7.2. Computational results

The performance of QCYCLE for each instance is defined by QCYCLE’s result’s relative percentage above the lower bound for either a single-gripper or dual-gripper robot (whichever is better):

\[
\text{Performance} = \frac{T(\text{QCYCLE}) - \min\{\Omega', \Omega''\}}{\min\{\Omega', \Omega''\}},
\]

where \(\Omega'\) (resp., \(\Omega''\)) is the optimum per unit cycle time for a single-gripper (resp., dual-gripper) cell and \(T(\text{QCYCLE})\) is the per unit cycle time of the cycle generated by Algorithm QCYCLE.

Table 7 summarizes the average performance of QCYCLE for each combination. Across the 7800 instances, the average gap is 9.09%; this is significantly better than the worst-case guaranteed results. The total number of instances in which QCYCLE finds a provably optimal solution is 1325 (16.99%).

| Number of machines \((m)\) | C1 | C2 | C3 | C4 | C5 | C6 | C7 | C8 | C9 | C10 | Average (%) |
|---------------------------|----|----|----|----|----|----|----|----|----|-----|------------|
| 5  |     | 14.32 | 13.67 | 11.19 | 11.08 | 15.41 | 10.03 |    |    |    | 12.62    |
| 10 | 13.58 | 13.97 | 10.64 | 9.26 | 10.24 | 9.95 | 6.27 | 7.75 | 9.10 |    | 10.08    |
| 12 | 12.64 | 14.10 | 9.75 | 8.86 | 9.82 | 7.97 | 6.17 | 6.85 | 10.49 | 9.16 | 9.58    |
| 15 | 12.06 | 14.55 | 7.96 | 8.74 | 8.84 | 6.34 | 4.76 | 6.28 | 8.83 | 7.74 | 8.61    |
| 20 | 11.41 | 14.10 | 6.68 | 8.88 | 8.01 | 5.04 | 3.89 | 5.70 | 10.25 | 8.02 | 8.20    |
| 25 | 11.68 | 14.29 | 5.58 | 8.57 | 7.67 | 3.93 | 3.36 | 5.74 | 11.60 | 8.23 | 8.07    |
| 30 | 11.55 | 14.33 | 4.98 | 8.11 | 7.72 | 3.32 | 3.33 | 6.31 | 11.83 | 8.39 | 7.99    |

The performance of QCYCLE for each instance is defined by QCYCLE’s result’s relative percentage above the lower bound for either a single-gripper or dual-gripper robot (whichever is better):

\[
\text{Performance} = \frac{T(\text{QCYCLE}) - \min\{\Omega', \Omega''\}}{\min\{\Omega', \Omega''\}},
\]

where \(\Omega'\) (resp., \(\Omega''\)) is the optimum per unit cycle time for a single-gripper (resp., dual-gripper) cell and \(T(\text{QCYCLE})\) is the per unit cycle time of the cycle generated by Algorithm QCYCLE.

Table 7 summarizes the average performance of QCYCLE for each combination. Across the 7800 instances, the average gap is 9.09%; this is significantly better than the worst-case guaranteed results. The total number of instances in which QCYCLE finds a provably optimal solution is 1325 (16.99%).

On average, the approximation error decreases as the cell size \(m\) increases. This rate of decrease is fastest for those cells with only one machine with a small processing time and several machines with large processing times (combinations C3 and C6). Conversely, this rate is slowest, and at points negative, for those cells with a large number of small processing times (combinations C2, C8, C9, and C10) but, oddly, not the maximum amount of small processing times (C7). We suspect two causes for this trend: first, when proving the theoretical bounds, the terms that represent the sums of the smaller processing times generally are canceled from the numerator and the denominator. For some instances with a large percentage of smaller processing times, leaving these sums in the ratio would have reduced its value. Second, as the number of large processing times grows, the chance of having a processing time \(p_{\text{max}}\) near its maximum, ensuring that QCYCLE finds an optimal cycle, increases.

As expected, the preference for dual-gripper robot cycles is negatively correlated to the time required for a gripper switch. For \(\theta = 2, 14, 28, 42\), the percentages of instances in which the dual-gripper robot has higher throughput are 100, 86.21, 76.77, and 55.69%, respectively.

We now evaluate the contribution of E-cycles. Since E-cycles were designed for cases not covered by Lemma 7, we limit the gripper switch time to \(\theta < \delta/3\), i.e., \(\theta = 2\), and processing time combinations to \(C7-C10\), so that \(r_2 > 2(m-1)/3\). This yields a total of 10 \(\times\) 24 \(\times\) 3 = 720 instances. An E-cycle uniquely produces the best throughput among all others considered by QCYCLE in 269 of these instances (37.36%). The E-cycle’s relative percentage throughput advantage over the best non-E-cycle averages 8.02%, and this advantage steadily increases as \(m\) increases: 3.75, 7.42, 6.36, 8.93, 9.18, 10.22, and 10.35%. Hence, E-Cycles are a valuable contribution to practitioners who wish to efficiently schedule robotic cells.

Our last study tests our conjecture that the value of the best E-cycle depends not only on the cell’s processing times, but also on the relative locations of the machines to which those times are assigned. This characteristic distinguishes the class of E-cycles from most other cycles. Specifically, we hypothesize that E-cycles produce higher throughput if the machines with larger processing times are closer together, rather than evenly spread throughout the cell. Our reasoning is that a cell with the larger times together will have fewer epicycles and, hence, less travel time.

To ensure meaningful data, we limit this study to combinations C8–C10, \(m \in \{12, 15, 20, 25\}\), for a total of 12 \(\times\) 10 = 120 instances of machine processing times. For each such instance, the times are arranged in the two ways previously mentioned. Our results show that E-cycles produce on average 7.85% higher throughput in cells with the larger processing times close to one another. This result gives managers further insight into which environments are best suited to using this new type of scheduling sequence.
Single-gripper and dual-gripper robotic cells

8. Conclusions and recommendations for future work

This article studies both dual-gripper and single-gripper robotic cells under circular layout with additive travel time and free-pickup. By using lower bounds, we prove the optimality of certain $k$-unit cycles ($k \geq 1$) under various conditions and also provide an efficient polynomial algorithm for which the throughput is at least 60% of the optimum for single-gripper as well as dual-gripper robotic cells. For the practically relevant condition that the dual-gripper switch time is less than twice the intermachine robot movement time, we provide an efficient polynomial algorithm for which the throughput is guaranteed to be at least 66.67% of the optimum. Furthermore, using a comprehensive test bed of instances that may be encountered in practice, we demonstrate that our algorithm’s average throughput is better than 91.7% of optimum and that our new type of cycles, epicyclic cycles, make a major contribution to this efficiency. Our algorithm generates cyclic schedules that are easy to implement and control, so they are useful to practitioners. Therefore, our work provides new insights for scheduling operations to maximize productivity for either type of robotic cell.

Our structural analysis indicates the specific domains under which these problems appear to be hard. This may help future researchers to focus on those domains and to construct problem instances within them that are equivalent to known NP-hard decision problems. Thus, this analysis is a stepping stone for researching the complexity status of the long-standing open problem of finding an optimal $k$-unit cycle ($k \geq 1$). Furthermore, we expect epicycles to be useful in future research of multi-unit cyclic scheduling since they efficiently schedule transporters so that they wait at some machines but not at others, depending on the machines’ processing time requirements. Epicyclic cycles may have applications in multi-product flow shop environments with sequence-dependent setup times whose matrix has some special structure.

There are a number of directions in which this study may be extended.

1. Algorithmic improvement: This article provides a polynomial algorithm that is a 5/3-approximation to an optimal $k$-unit cycle, which is the first approximation result for dual-gripper robotic cells in this domain (i.e., free-pickup, circular layout, additive travel time). Therefore, improving the algorithm to obtain a better approximation is one direction for future research. The complexity status of our domain remains open.

2. Cells under linear layout: This article focuses on analysis of cells under circular layout for finding an optimal $k$-unit cycle. No study has sought an optimal $k$-unit cycle for free-pickup cells under linear layout with additive travel time. Neither the complexity status nor an algorithm with a performance guarantee has been established for dual-gripper cells. However, results are available for single-gripper cells (Crama and van de Klundert, 1997; Geismar et al., 2007).

3. Interval cells: The approximation to an optimal $k$-unit cycle for single-gripper or dual-gripper interval robotic cells has not been performed for either additive or constant travel time. Note that the complexity of finding an optimum $k$-unit cyclic solution in interval cells served by dual-gripper robots is NP-hard (Dawande et al., 2010).

4. Euclidean Travel-time Metric: Geismar, Dawande, and Sriskandarajah (2005) develop an approximation result to an optimal $k$-unit cycle for single-gripper robotic cells with free-pickup and Euclidean travel time. The complexity of finding an optimum $k$-unit cyclic solution ($k \geq 3$) in cells served by a single-gripper robot is a longstanding open problem. However, polynomial-time algorithms are available in no-wait cells for $k = 1$ (Levner et al., 1997) and $k = 2$ (Che et al., 2003). Optimal $k$-unit cycles in Euclidean cells with dual-gripper robots have not been studied for any part pickup criteria.

5. Multiple part-types problems: Studies with multiple part-types have been performed by Hall et al. (1997, 1998) and Sriskandarajah et al. (1998). These studies prove that this problem is NP-hard for single-gripper robots with additive travel times. Only one paper analyzes scheduling multiple-part-type dual-gripper cells for additive travel-time (Sriskandarajah et al., 2004). Devising efficient heuristics for multiple-part-type dual-gripper cells seems to be a promising direction for future research.

9. Supplemental Material

Supplemental data for this article can be accessed on the publisher’s website at www.tandfonline.com

References

Agnetis, A. (2000) Scheduling no-wait robotic cells with two and three machines. European Journal of Operational Research, 123(2), 303–314.

Asfahl, C.R. (1992) Robots and Manufacturing Automation, Second edition, Wiley & Sons, New York, NY.

Brauner, N. (2008) Identical part production in cyclic robotic cells: concepts, overview and open questions. Discrete Applied Mathematics, 156(13), 2480–2492.

Brauner, N. and Finke, G. (2001) Cycles and permutations in robotic cells. Mathematical and Computer Modeling, 34(5), 565–591.

Brauner, N., Finke, G. and Kubiak, W. (2003) Complexity of one-cycle robotic flow shops. Journal of Scheduling, 6(4), 355–371.

Brumson, B. (2001) Scara vs. Cartesian robots: selecting the right type for your applications, available at http://www.robotics.org/search_results.cfa?tab=2&searchField=cartesian, last accessed January 31, 2012.

Che, A., Chu, C. and Levner, E. (2003) A polynomial algorithm for 2-degree cyclic robot scheduling. European Journal of Operational Research, 145(1), 31–44.
Crama, Y., Kats, V., van de Klundert, J. and Levrer, E. (2000) Cyclic scheduling in robotic flowshops. *Annals of Operations Research*, 96(1-4), 97–124.

Crama, Y. and van de Klundert, J. (1997) Cyclic scheduling of identical parts in a robotic cell. *Operations Research*, 45(6), 952–965.

Dawande, M., Geismar, H.N., Pinedo, M. and Sriskandarajah, C. (2010) Throughput optimization in dual-gripper interval robotic cells. *IEEE Transactions*, 42(1), 1–15.

Dawande, M., Geismar, H.N. and Sethi, S.P. (2005) Dominance of cyclic solutions and some open problems in scheduling bufferless robotic cells. *SIAM Review*, 47, 709–721.

Dawande, M., Geismar, H.N., Sethi, S.P. and Sriskandarajah, C. (2005) Sequencing and scheduling in robotics cells: recent developments. *Journal of Scheduling*, 8(5), 387–426.

Dawande, M., Geismar, H.N., Sethi, S.P. and Sriskandarajah, C. (2007) *Throughput Optimization in Robotic Cells*. Springer, New York.

Dawande, M., Sriskandarajah, C. and Sethi, S.P. (2002) On throughput maximization in constant travel-time robotic cells. *Manufacturing and Service Operations Management*, 4(4), 296–312.

Drobouchevitch, I., Sethi, S.P. and Sriskandarajah, C. (2006) Scheduling dual-gripper robotic cell: one-unit cycles. *European Journal of Operational Research*, 171(2), 598–631.

Geismar, H.N., Chan, L.M.A., Dawande, M. and Sriskandarajah, C. (2008) Approximations to optimal $k$-unit cycles for single-gripper and dual-gripper robotic cells. *Production and Operations Management*, 17(5), 551–563.

Geismar, H.N., Dawande, M. and Sriskandarajah, C. (2005) Approximations algorithms for $k$-unit cyclic solutions in robotic cells. *European Journal of Operational Research*, 162(2), 291–309.

Geismar, H.N., Dawande, M. and Sriskandarajah, C. (2006) Throughput optimization in constant travel-time dual-gripper robotic cells with parallel machines. *Production and Operations Management*, 15(2), 311–328.

Geismar, H.N., Dawande, M. and Sriskandarajah, C. (2007) A (10/7)-approximation algorithm for an optimum cyclic solution in additive travel-time robotic cells. *IEEE Transactions*, 39(2), 217–227.

Geismar, H.N., Sethi, S.P., Sidney, J.B. and Sriskandarajah, C. (2005) A note on productivity gains in flexible robotic cells. *International Journal Flexible Manufacturing Systems*, 17(1), 5–21.

Geismar, H.N., Sriskandarajah, C. and Ramanan, N. (2004) Increasing throughput for robotic cells with parallel machines and multiple robots. *IEEE Transactions on Automation Science & Engineering*, 1(1), 84–89.

Hall, N.G. (1999) Operations research techniques for robotic systems, in *Handbook of Industrial Robotics*, volume II, Nof, S.Y. (ed.), Wiley, New York, NY, pp. 543–577.

Hall, N.G., Kamoun, H. and Sriskandarajah, C. (1997) Scheduling in robotic cells: classification, two and three machine cells. *Operations Research*, 45(3), 421–439.

Hall, N.G., Kamoun, H. and Sriskandarajah, C. (1998) Scheduling in robotic cells: complexity and steady state analysis. *European Journal of Operational Research*, 109(1), 43–63.

Jung, K.S. (2013) Analysis of three optimization problems in operations management. Ph.D. Dissertation, The University of Texas at Dallas, August 2013.

Kim, J.-H., Lee, T.-E., Lee, H.-Y. and Park, D.-B. (2003) Scheduling analysis of time-constrained dual-armed cluster tools. *IEEE Transactions on Semiconductor Manufacturing*, 16(3), 521–534.

Korea Fanuc Corporation. [http://www.fke.co.kr/](http://www.fke.co.kr/).

Lei, L. and Wang, T.J. (1994) Determining optimal cyclic hoist schedules in a single-hoist electroplating line. *IEEE Transactions*, 26(2), 25–33.

Levier, E., Kats, V. and Levit, V. (1997) An improved algorithm for cyclic flowshop scheduling in a robotic cell. *European Journal of Operational Research*, 97(3), 500–508.

MGS News (2007) [http://mgsnews.com/mgs-news3.htm#05](http://mgsnews.com/mgs-news3.htm#05)

Miller, R. and Walker, T. (1990) *FMS/CIM Systems Integration Handbook*, The Fairmont Press, Lilburn, GA.

Mönch, L., Fowler, J.W., Dauzére-Pérès, S., Mason, S.J. and Rose, O. (2011) A survey of problems, solution techniques, and future challenges in scheduling semiconductor manufacturing operations. *Journal of Scheduling*, 14(6), 583–599.

Pinedo, M. (2012) *Scheduling: Theory, Algorithms, and Systems*, Springer, New York, NY.

Rajapakse, T., Dawande, M. and Sriskandarajah, C. (2011) Quantifying the impact of layout on productivity: an analysis from robotic-cell manufacturing. *Operations Research*, 59(2), 440–454.

Rostami, S. and Hamidzadeh, B. (2002) Optimal scheduling techniques for cluster tools with process-module and transport-module residency constraints. *IEEE Transactions on Semiconductor Manufacturing*, 15(3), 341–349.

Rostami, S., Hamidzadeh, B. and Camporese, D. (2001) An optimal periodic scheduler for dual-arm robots in cluster tools with residency constraints. *IEEE Transactions on Robotics and Automation*, 17, 609–618.

Sethi, S.P., Sidney, J. and Sriskandarajah, C. (2001) Scheduling in dual-gripper robotic cells for productivity gains. *IEEE Transactions on Robotics and Automation*, 17(3), 324–341.

Sethi, S.P., Sriskandarajah, C., Sorger, G., Blazewicz, J. and Kubick, W. (1992) Sequencing of parts and robot moves in a robotic cell. *International Journal of Flexible Manufacturing Systems*, 4(3), 331–358.

Sriskandarajah, C., Drobouchevitch, I., Sethi, S.P. and Chandrasekaran, R. (2004) Scheduling multiple parts in a robotic cell served by a dual-gripper robot. *Operational Research*, 52(1), 65–82.

Sriskandarajah, C., Hall, N.G. and Kamoun, H. (1998) Scheduling large robotic cells without buffers. *Annals of Operations Research*, 76, 287–321.

Su, Q. and Chen, F. (1996) Optimal sequencing of double-gripper gantry robot moves in tightly-coupled serial production systems. *IEEE Transaction on Robotics and Automation*, 12(1), 22–30.

Venkatesh, S., Davenport, R., Foxhoven, P. and Nulman, J. (1997) A steady-state throughput analysis of cluster tools: dual-blade versus single-blade robots. *IEEE Transactions on Semiconductor Manufacturing*, 10(4), 418–424.

World Robotics. (2012) Executive summary 2012. [http://www.barra.org](http://www.barra.org) [pdf](http://www.barra.org/uploads/tx_zeifr/Executive_Summary_WR_2012.pdf). Accessed date: November 26, 2014.

World Robotics. (2013) Executive summary 2013. [http://www.worldrobotics.org](http://www.worldrobotics.org) [pdf](http://www.worldrobotics.org/uploads/tx_zeifr/Executive_Summary_WR_2013.pdf). Accessed date: November 26, 2014.

**Biographies**

Kyung Sung Jung is an Adjunct Assistant Professor in the Department of Information Systems and Operations Management at Warrington College of Business Administration, University of Florida. He received his Ph.D. from the University of Texas at Dallas on August 2013 and an MBA in Foster School of Business at University of Washington, Seattle. Before entering academics, he had worked at LG Electronics, Inc. as a Researcher in Mechanical Department at Mobile Handset R&D Center, Korea. He has two patents in the United States and 19 local patents in Korea for mobile phone technologies. His primary research interests are in addressing the operations problems in manufacturing and service operations management. He has published his research in *Annals of Operations Research*.

H. Neil Geismar is an Associate Professor in the Mays Business School at Texas A&M University. He has a Ph.D. degree from the University of Texas at Dallas on August 2013 and an MBA in Foster School of Business at University of Washington, Seattle. Before entering academics, he had worked at LG Electronics, Inc. as a Researcher in Mechanical Department at Mobile Handset R&D Center, Korea. He has two patents in the United States and 19 local patents in Korea for mobile phone technologies. His primary research interests are in addressing the operations problems in manufacturing and service operations management. He has published his research in *Annals of Operations Research*. 

M. Dawande was born in India. He received a B.Tech. degree in Electrical Engineering from BITS Pilani, India, an M.S.E degree in Mechanical Engineering from Texas A&M University, and a Ph.D. in Industrial Engineering from the University of Texas at Dallas in 2000. He is currently a Professor in the Department of Industrial and Systems Engineering at Texas Tech University. His research interests include robotic scheduling, manufacturing systems, and complex decision making.
profitability. His papers have appeared in a number of journals, including INFORMS Journal on Computing, Manufacturing and Services Operations Management, Production and Operations Management, SIAM Review, and IIE Transactions. He is a member of INFORMS and POMS.

Michael Pinedo is the Julius Schlesinger Professor of Operations Management in the department of Information, Operations and Management Sciences (IOMS) at the Stern School of Business at New York University. He received a Ph.D. in Operations Research from the University of California at Berkeley in 1978. His research focuses on the modeling of production and service systems and, more specifically, on the planning and scheduling of these systems. His research also has been focusing on operational risk in financial services. He has (co-)authored numerous technical papers on these topics. He is the author of several books, including ‘Scheduling: Theory, Algorithms and Systems’ and ‘Planning and Scheduling in Manufacturing and Services’. He is Editor of the Journal of Scheduling and Associate Editor of Management Science, Associate Editor of Naval Research Logistics, Department Editor of Production and Operations Management, Associate Editor of Manufacturing and Services Operations Management, Associate Editor of the Journal of Operational Risk, and a member of the editorial boards of Risk and Decision Analysis as well as of the International Journal of Shipping and Transport Logistics. He is a Fellow of INFORMS and POMS.

Chelliah Sriskandarajah is a professor of Operations Management and holds the Hugh Roy Cullen Chair at Mays Business School, Texas A&M University. He has a Ph.D. degree from the National Polytechnic Institute of Grenoble (France) in the field of Production Management. Prof. Sriskandarajah’s research interests lie in the general area of supply chain management, logistics, production planning and scheduling, and performance evaluation of production/service systems. He was inducted as a POMS fellow in April 2012, the most prestigious honor awarded by the Production and Operations Management Society. He has published over 100 scholarly articles in leading international journals. Over the years, his research has been supported by a number of sponsors including National Science Foundation (USA), Natural Sciences and Engineering Research Council (Canada), Manufacturing Research Cooperation of Ontario (Canada) and NATO. He currently serves as a Departmental Editor of Production and Operations Management Journal, an Associate Editor of Decision Science Journal, and a member of INFORMS, POMS, IIE, IEEE, and APICS. He served as the Associate Executive Director of the Production and Operations Management Society (POMS) from March 2004 to April 2012.