Available online at : http://josi.ft.unand.ac.id/

Jurnal Optimasi Sistem Industri

| ISSN (Print) 2088-4842 | ISSN (Online) 2442-8795 |

Article Type

Achieving Quasi-Complete Balance in U-shaped Assembly Lines Using Idle Time Elimination Methodologies

Zening Wu *, Wenqian Luo

Faculty of Engineering, The University of Sydney, Camperdown, Australia

*Corresponding Author: zewu0754@uni.sydney.edu.au

DOI: 10.25077/josi.v24.n1.p1-17.2025 **Submitted:** December 4, 2024 **Accepted:** March 3, 2025 **Published:** June 30, 2025

ABSTRACT

The problem of balancing the U-shaped production line is a well-known NP-problem in mass production, where the main objective is to allocate tasks efficiently to the workstations while minimising idle time and balancing the workload throughout the production line. The study introduces two new methods to address these issues and increase the efficiency of line balancing: Operators Arrangement for U-shaped line Balancing (OAUB) and Layout Design for U-shaped line Balancing (LDUB). OAUB optimizes the allocation of workers by strategically adjusting the assignment of tasks, ensuring that idle time is minimised and that the use of workers is maximised. By contrast, LDUB alleviates idle time by filling in the gaps between tasks with independent, low-processing tasks from non-critical paths, thus ensuring a more efficient use of available time during the transitions. Implementation of these methods has shown promising results and significantly reduced the idle time of the production processes. Specifically, OAUB achieved a reduction of only 0.9 units of idle time, while LDUB decreased idle time by 3 units. These results are particularly important from a sustainability point of view, as they contribute to reducing waste through better use of resources and increased production efficiency. Unlike traditional approaches, which are mainly focused on minimizing the number of workstations or the reduction of cycle times, our methodology provides a more integrated approach to balancing the workload between workstations. The practical implications of these methods are significant as they are applicable to the real-world production environment and offer effective and efficient solutions to achieve near complete alignment on U-shaped assembly lines, thus improving overall productivity and sustainability in an environment of mass production.

Keywords: U-shaped line balancing, OAUB, LDUB, idle time elimination, production efficiency

INTRODUCTION

Since its popularization in 1913, the assembly line has become the cornerstone of mass production [1]. Designing an assembly line, regardless of its configuration, requires assigning tasks to workstations while adhering to the precedence relations among tasks [2]. The transition from straight assembly lines to U-shaped assembly lines was driven by the implementation of lean manufacturing systems and the adoption of the "just-in-time" principle, which focuses on reducing waste and improving efficiency. A distinctive feature of a U-shaped line is its ability to group tasks from both the front and back sequences of the assembly belt into a single workstation. This configuration not only improves operational flexibility but also supports sustainability goals by reducing the number of workstations required, optimizing space utilization, improving material flow, and reducing transportation waste, all of which are core principles of lean manufacturing. Building on these practical advantages, research into the USLB problem has

expanded steadily over the past three decades. The remainder of this review traces that evolution and synthesizes current knowledge to provide a holistic view of the field's development.

The concept of U-shaped line balancing (USLB) was first introduced by Miltenberg and Wijngaard [3], who proposed heuristic algorithms to achieve balance in U-shaped line layouts. They defined the line balancing problem as the assignment of tasks to workstations with the objectives of minimizing the number of workstations for a given cycle time or minimizing the cycle time for a given number of workstations. Fundamentally, the primary goal of line balancing is to minimize the total idle time across all workstations, thereby enhancing efficiency [4]. Expanding on this groundwork, Miltenburg [5] later explored the mixed-model U-line balancing problem, where tasks from multiple products are assigned to workstations. In this context, the mixed-model sequencing problem becomes a crucial factor in optimizing balancing performance. Specifically, sequencing—the order in which different models are produced—affects the load distribution across workstations. Variations in processing times and task precedence can cause imbalances, leading to overloading or underloading of certain workstations. Miltenburg's work highlighted that effective sequencing, in conjunction with balancing, plays a critical role in minimizing these imbalances and achieving more efficient production flows. This insight reinforced the significance of both line balancing and sequencing in ensuring the success of mixed-model production.

To address computational challenges associated with larger problems—typically those involving more than 21 tasks—Urban [6] developed an integer programming formulation. By appending a "phantom" network to the original precedence network, this method visualized the flexibility of U-shaped lines in allowing tasks to be performed both backward and forward from the start or end points. This innovative approach enabled the solving of large-scale USLB problems using commercially available, general-purpose software. Building on the growing body of research into the structural and operational advantages of U-shaped lines, the advantages of U-shaped assembly lines have been extensively discussed by researchers, including fewer workstations required, increased flexibility, improved balancing and material handling, enhanced visibility and information flow, and better quality control [3], [7]-[9]. However, these benefits come with limitations, such as the need for more skilled labor and increased coordination efforts, as operators may need to work across multiple tasks and manage the flow of products between both sides of the U-shaped layout. Beyond single U-shaped line problems, Miltenburg [10] introduced the concept of the U-line facility problem, developing a dynamic programming algorithm to balance multiple U-shaped lines. As research progressed from deterministic to more realistic production environments, the study of stochastic USLB problems began with Guerriero and Miltenburg [11], who developed a recursive algorithm capable of solving practical-sized instances with stochastic task times. Further research into the stochastic U-shaped balancing problem, for example, was conducted by Erel [12], and it is also the first research to implement the heuristic procedure for the stochastic U-shaped balancing problem.

Historically, most research has focused on achieving single objectives, such as minimizing cycle time or the number of workstations. Recognizing the limitations of this approach, Gökçen and Agpak [13] were the first to employ multi-criteria decision-making for simultaneous multi-objective optimization in U-shaped lines. They emphasized the pursuit of "satisfactory" solutions over "optimal" ones. The growing complexity of modern production environments has made multi-objective approaches increasingly relevant, as they allow for a more balanced focus on key factors such as balancing efficiency, resource utilization, and cost, which are all critical to achieving sustainable and cost-effective production. Given the difficulty decision-makers face in determining precise multi-objective values due to uncertainties, Toklu and Özcan [14] introduced a fuzzy goal programming model to tackle simple U-line balancing problems with multiple goals, allowing decision-makers to prioritize objectives hierarchically. Subsequent studies, such as Widyadana's [15], expanded on this by developing a three-

 DOI: 10.25077/josi.v24.n1.p1-17.2025

goal model in manufacturing settings, where temporary workers are recruited during a peak demanding period.

To further exploit the flexibility of U-shaped lines, novel layout strategies have been proposed. Yegul et al. [16] developed the two-sided U-line. As their study shows, one side of line is viewed as a straight line where operators are allocated as the normal straight layout, whereas the other side of line is treated as a U-shaped line. Kucukkoc and Zhang [17] introduced the parallel U-shaped configuration, which parallelizes two separate U-shaped lines and allows operators to perform tasks across multi-line stations, crossover stations, and regular stations, thereby improving workstation flexibility. Case studies confirm the advantages of parallel U-shaped lines in real-world applications. For example, Kucukkoc and Zhang demonstrated that balancing two parallel U-shaped lines together, rather than independently, significantly reduces the number of required workstations and improves resource utilization. Such configurations allow production lines to more effectively adapt to fluctuating demands by sharing tasks across lines, resulting in cost savings and increased operational efficiency.

Over the decades, a multitude of algorithms have been developed to find optimal solutions for USLB. Reviews by Sivasankaran and Shahabudeen [18] highlight various approaches: simulated annealing algorithms, ant colony algorithm, shortest path algorithms, and imperialistic competitive algorithms, each aiming to minimize workstations and balance workloads. The genetic algorithm, first applied to USLB by Ajenblit and Wainwright [19], proposes six different sub-algorithms for addressing variations such as idle time and workload for each workstation. Kim et al. [20] further advanced this area by applying endosymbiotic evolutionary algorithms to the mixed model U-shaped assembly line balancing problem and its sequencing problem in which each task time is given. As the scope of USLB research expanded to include more practical and technologically advanced implementations, some researchers have incorporated scheduling tools into USLB. Avikal et al. [21] innovatively integrated the critical path method into heuristics for task assignment, providing evidence that U-shaped lines can enhance labor productivity.

Nilakantan and Ponnambalam [22] were pioneers in considering robotic line balancing within U-shaped configurations, using particle swarm optimization algorithms to improve production rates in automotive lines. Building on this, Li et al. [23] tackle the cobot-assisted U-shaped line under a budget constraint, formulating three mixed-integer models that jointly select cobot types and minimise cycle time. Zhang et al. [24] advance the mixed-model U-shaped robotic assembly line balancing and sequencing problem (MURALBSP), introducing a hybrid multi-objective dragonfly algorithm (HMODA) that balances energy use and makespan. In line with Industry 4.0, Mao et al. [25] present one of the first formulations for lines equipped with multiple cobot types, while Nourmohammadi et al. [26] integrat balancing and scheduling in cobot lines to optimise station count, cycle time, and energy-related costs. Most recently, under the Industry 5.0 advent, Nourmohammadi et al. [28] introduce constraint-programming models for U-shaped layouts that minimise stations, cycle time, and total operational cost within a unified framework.

Despite three decades of advances in layout design, algorithm development, and robotic integration, most USLB studies still settle for near balancing rather than achieving complete balance. It is particularly evident that contemporary research continues to prioritise ever-more sophisticated algorithms that (i) minimise the number of workstations for a given cycle time (Type 1) or (ii) minimise cycle time for a fixed workstation count (Type 2), whereas objectives such as maximising line efficiency (Type E) remain comparatively under-explored [29]. This algorithm-centric bias is understandable: USLB is NP-hard, so exact optimization quickly becomes intractable as problem size grows, prompting widespread adoption of heuristics, metaheuristics, and other

approximation techniques that yield high-quality—but not necessarily globally optimal—solutions within practical computation times [30]. Consequently, existing methods mainly reduce idle time indirectly by shrinking cycle time or workstation count, yet none guarantee an even distribution of task time across all stations. Addressing this gap, the present study introduces idle-time-elimination methodologies that target quasi-complete balance by either assimilating idle time through optimized workforce allocation or filling it with additional assignable tasks; under appropriate settings, these principles can achieve a fully balanced workload across every workstation.

The contributions of this paper are multifaceted. It is the first to approach the U-shaped line balancing problem from a systems perspective rather than relying solely on traditional algorithmic development, providing practical guidance for achieving complete balance. Two novel approaches are introduced, offering both theoretical foundations and real-world applicability. In practice, managers can adopt these methodologies to achieve quasi-complete balance with a controlled degree of unevenness, enabling practical solutions across various manufacturing scenarios.

The rest of the paper is organized as follows: Section 2, Methods, delineates the U-shaped assembly-line balancing problem and introduces two novel idle-time-elimination techniques—Operator Arrangement for U-shaped line Balancing (OAUB) and Layout Design for U-line Balancing (LDUB)—devised to achieve quasi-complete balance along the line. Section 3, Result and Discussion, evaluates these techniques through a detailed illustrative case, demonstrating their efficacy in realizing near-complete balance. Finally, Section 4, Conclusion, synthesizes the principal findings and identifies avenues for future research, including further refinement of the proposed methods and exploration of their broader industrial applications.

METHODS

Problem Formulation of the U-Shaped Line Balancing Problem

Driven by the principles of Just-In-Time (JIT) production, many manufacturing facilities have increasingly adopted U-shaped line layouts over traditional straight-line configurations [3], [31]. The ULBP can be elaborated as follows. Consider a product disassembled into its smallest unit parts that need to be reassembled into a finished product through a number of processing tasks, denoted by n. For each processing task i (where i = 1, 2, 3, ..., n), the corresponding task time is represented by t_i.

Each processing task must adhere to specific precedence constraints (e.g., the assembly of the shell part must occur after the correct placement of internal components). These relationships are defined by a precedence network, as illustrated in Figure 1. The tasks are to be assigned to an unknown number of workstations, denoted by m (where j = 1, 2, 3, …, m), with j indicating a specific workstation. A key distinction between U-shaped and straight-line layouts

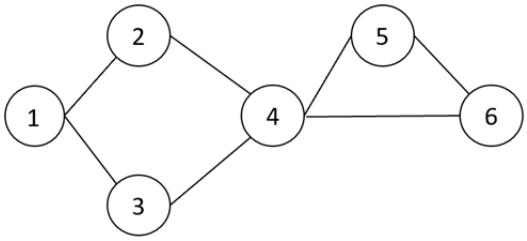

Figure 1. A precedence network example

 DOI: 10.25077/josi.v24.n1.p1-17.2025

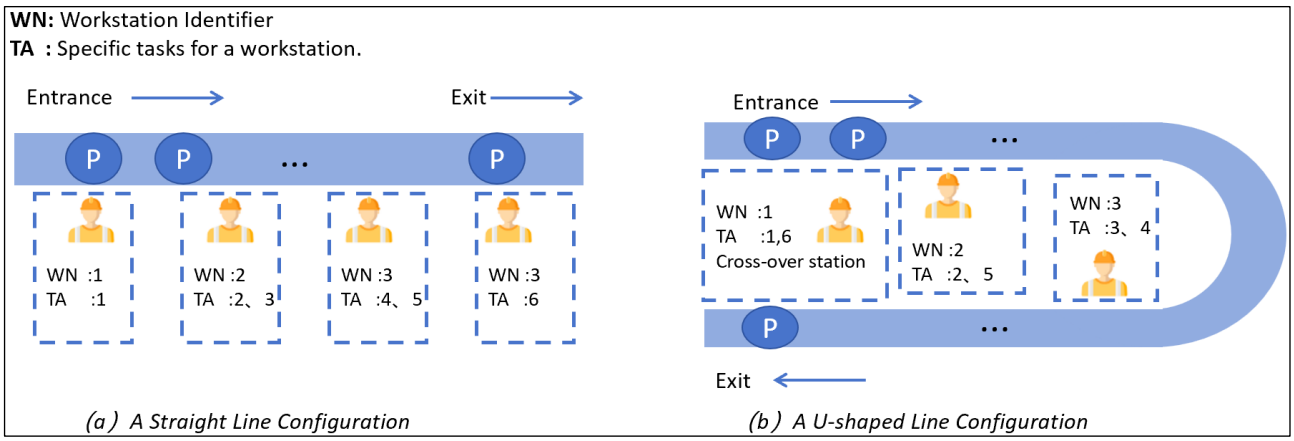

Figure 2. Comparing Straight and U-shaped Line Layouts

is that the U-shaped line allows for task assignments both forward and backward simultaneously from the starting and ending points of the precedence network. In contrast, the straight-line layout can only assign tasks in the forward direction from the starting node. Although this bidirectional flexibility often reduces the number of workstations required, it can lead to increased operator movement and pose challenges in high-mix, low-volume production environments, where task variability may be more pronounced. Figure 2 compares these layouts, with part (a) depicting the straight-line layout and part (b) representing the U-shaped line layout, both relative to the precedence relationships in Figure 1.

For each operator, the total processing time per product assigned must not exceed a predefined cycle time C. This assumption treats operators as having consistent performance levels and does not account for human-centric factors such as skill variation, fatigue, or learning effects. We introduce a decision variable x_{ij}, which indicates whether task i is assigned to workstation j:

$$x_{ij} = \begin{cases} 1, & \text{if task } i \text{ is assigned to workstation } j, \\ 0, & \text{otherwise;} \end{cases} \tag{1}$$

Our primary objective is to eliminate idle time and ensure that the workload among operators is evenly distributed. However, holistic line optimization may also involve factors like ergonomic considerations, operator skill levels, and product variability, which can further influence how workloads are assigned. Denoting the number of operators at j workstation as n_j, the target function is formulated as:

$$C - \left(\frac{\sum_{i=1}^{n} t_i \cdot x_{ij}}{n_j}\right) = 0 \tag{2}$$

Subject to:

$$x_{ij} \in \{0,1\},\ \forall i \in \{1,2,3,\ldots n\},\ \forall j \in \{1,2,3,\ldots m\} \tag{3}$$

Idle Time Elimination Methodologies

Traditionally, USLB has focused on finding optimal solutions to the Type 1 problem (minimizing the number of workstations for a given cycle time), the Type 2 problem (minimizing the cycle time for a given number of workstations), or mixed-type problems. However, these approaches often face implementation constraints, such as handling the NP-hard complexity of large instances, managing uncertainties in task times, or integrating real-world

factors like operator training and equipment availability. Despite extensive research, existing literature has not provided a methodology that completely eliminates idle time. To address this gap, we propose two methodologies aimed at achieving a balanced state in U-shaped assembly lines by eliminating idle time through strategic arrangement of operators and layout design, rather than solely relying on traditional workstation and task assignments.

This study assumes that task times are fixed and unaffected by factors such as operator fatigue or proficiency levels. Automation can readily satisfy these conditions, significantly reducing the influence of labor variability. We define the balancing challenge under these assumptions as the Quasi-Complete U-shaped Line Balancing Problem (QULBP). The primary objectives of QULBP are:

1. Idle time is reduced, achieving complete balance.
2. The solution is practical and applicable in real-world production settings.

Operators Arrangement for U-shaped Line Balancing (OAUB)

The first methodology, termed Operators' Arrangement for U-line Balancing (OAUB), focuses on eliminating idle time by optimizing the allocation of operators or robots across workstations. The methodology consists of the following steps:

Step 1: Assign tasks to workstations using an appropriate algorithm to reduce problem complexity.

Step 2: Determine an acceptable level of deviation d for the number of operators required to achieve complete balancing. This deviation accounts for practical considerations in production environments, such as scheduling flexibility and resource availability. The value of d should be set based on the organization's operational constraints. For instance, an organization may set $d = 0.2$ if it can accommodate a 0.2-unit variation in operator allocation from perfect staffing levels. To refine the choice of d, Monte Carlo simulations can be employed, randomly varying d and analyzing the resulting balance status. By merging simulation outcomes with the organization's specific constraints (e.g., operator availability), managers can determine a tailored d that upholds efficiency while remaining operationally feasible.

Step 3: Allocate a specific number of operators or robots to each workstation based on processing times. Let n be the number of operators assigned to the workstation with the minimum total processing time T_{min}. The total processing time for workstation j is denoted as T_j. The number of operators N_j assigned to workstation j is calculated as:

$$N_j = round\left(\frac{T_j}{T_{min}} \times n\right) \tag{4}$$

$$\left|\frac{T_j}{T_{min}} \times n - N_j\right| \leq d \tag{5}$$

Subject to:

$$T_j = \sum_{i=1}^{n} t_i \cdot x_{ij} \tag{6}$$

$$x_{ij} = \begin{cases} 1, \ if \text{ task } i \text{ is assigned to workstation } j, \\ 0, \ otherwise; \end{cases} \tag{7}$$

$$x_{ij} \in \{0,1\}, \ \forall i \in \{1,2,3,\ldots n\}, \ \forall j \in \{1,2,3,\ldots m\} \tag{8}$$

The *round(.)* function is used for rounding, and the smallest n that satisfies the constraint (5) is selected. Importantly, this minimum n must meet all N_j requirements, where N_j represents the number of operators for each workstation.

 DOI: 10.25077/josi.v24.n1.p1-17.2025

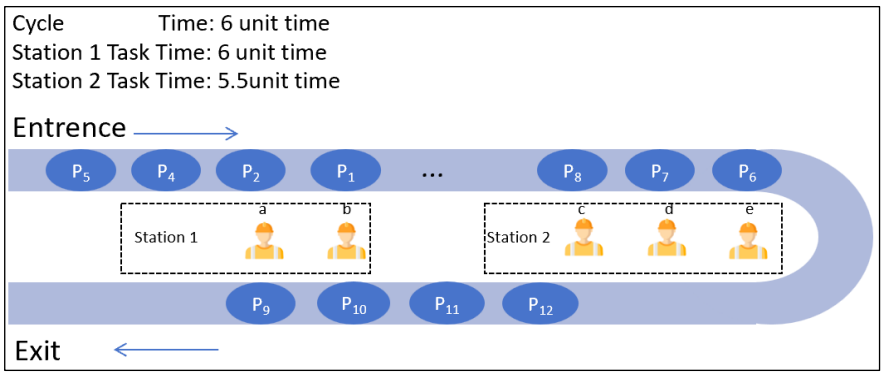

Figure 3. Operators' Arrangement for U-line Balancing Layout

Step 4: Deploy N_j operators or robots to each workstation j according to the calculated allocations. For example, if $N_1 = 2$ and $N_2 = 3$, operators are arranged accordingly, as illustrated in Figure 3.

In Figure 3, Station 1 functions as a crossover workstation where tasks are performed concurrently on both the front and back of the U-shaped line. Operators at the same station employ an interleaved task approach to maximize efficiency. For instance, Operator 'b' is responsible for Products 1 and 10. Upon completing tasks for these products, Operator 'b' transitions to tasks for Products 4 and 12. Similarly, Operator 'a' proceeds to tasks on Products 5 and 11 after completing their initial assignments on Products 2 and 9. This interleaving approach ensures continuous workflow. Nevertheless, implementing interleaved tasks and crossover stations can present practical limitations in workforce coordination and specialized skill requirements, which may be mitigated through targeted training programs and task rotation strategies.

Layout Design for U-shaped Line Balancing (LDUB)

While the OAUB methodology provides valuable insights into the QULBP and demonstrates the potential to eliminate all idle time, it presents significant challenges regarding schedulability. Specifically, OAUB requires:

- Increased Operator Requirements: Achieving complete balance necessitates more than one operator per workstation.
- Need for Collaborative Skills: Operators must be trained to collaborate effectively, as tasks are performed alternately, requiring coordination.
- Requirement for a Longer Assembly Line: The increased number of operators implies a longer assembly line, necessitating substantial initial investment in equipment and space.

While OAUB is cost-effective for large-scale production over extended periods, these constraints may limit its practicality in certain manufacturing settings. To address these concerns, we propose the Layout Design for U-Line Balancing (LDUB) methodology, which mitigates schedulability issues while aiming to eliminate idle time. LDUB involves identifying tasks on non-critical paths that can be performed independently and utilizing idle time by assigning these selected tasks to fill gaps in the workflow. The selected tasks, referred to as spare part manufacturing operations, should have relatively short processing times to fit within the available idle periods. Unlike OAUB, which relies on expanding the workforce for each station, LDUB balances workloads by distributing spare tasks to idle slots, thereby reducing the need for additional operators and minimizing coordination challenges. The steps for LDUB are:

Step 1: Draw the precedence network. LDUB favors employing diagrams to elucidate each individually operable process rather than consolidating multiple independently operable processes into a single, overarching process. To

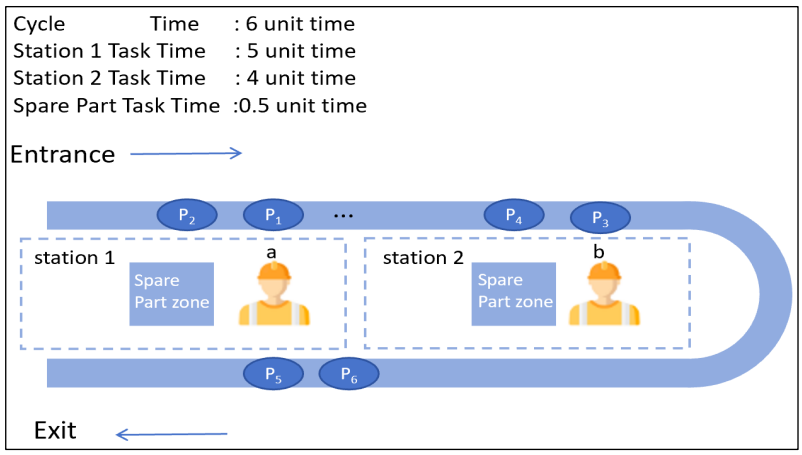

Figure 4. Design for U-line Balancing diagram

achieve a comprehensive and granular breakdown of assembly tasks, refer to the Work Breakdown Structure (WBS) methodology before drawing a precedence network [31].

Step 2: Select tasks that can be operated independently, and these tasks are referred to as spare part manufacturing operations. Identifying these spare tasks requires a careful examination of the precedence network. Specially, we prioritize spare part manufacturing operations with short processing time. Once we select these tasks, they are removed from the precedence network.

Step 3: Arrange tasks to workstations using an algorithm to reduce the complexity of the problem.

Step 4: Fill idle time with the selected tasks. The layout is illustrated in Figure 4.

As illustrated in Figure 4, the cycle time is established at 6 units. The processing time for Station 1 is 5 units, while Station 2 requires 4 units. Spare part manufacturing tasks each demand 0.5 units of time. In this layout, Station 1 functions as a crossover station. For example, an operator assembles Products 1 and 5, leaving 1 unit of idle time before the arrival of Products 2 and 6. During this idle period, Operator a can efficiently assemble two spare parts, each requiring 0.5 units of time. Similarly, after completing the assembly of Product 3, Operator b is able to undertake four spare part manufacturing tasks.

It is noteworthy that shorter tasks better fill these idle windows, because even if some idle time remains, it will not exceed the selected task duration. In contrast, longer tasks may not fit within the remaining idle slot at all, especially if their processing time surpasses the idle time itself. It is also noteworthy that when evaluating the number of spare parts being assembled, the sum of task time should not exceed the cycle time.

RESULTS AND DISCUSSION

This section presents two worked examples followed by a discussion of the results. To demonstrate these two approaches, we solve a modified version of Jackson's problem using the proposed methodologies. Additionally, we apply a heuristic approach developed by Avikal et al. [21], which is further enhanced by integrating the 'phantom' network concept proposed by Urban [6].

An Example of Operators' Arrangement for U-shaped Line Balancing (OAUB)

Step 1: Algorithmic Task Arrangement for Workstation Complexity Reduction

The implementation of this step is explained in Appendix. The resulting task assignments are summarized in Table 1 and illustrated in Figure 5.

 DOI: 10.25077/josi.v24.n1.p1-17.2025

Table 1. Computed Result of the Heuristic

Workstation	Assigned Tasks	Total Task Processing time	Idle Time	Accumulative Idle Time
1	*a*	59	1	1
2	*b, k′*	60	0	1
3	*j′, f*	60	0	1
4	*i′*	60	0	1
5	*c, g, h*	59	1	2
6	*e′*	60	0	2
7	*d*	23	37	39

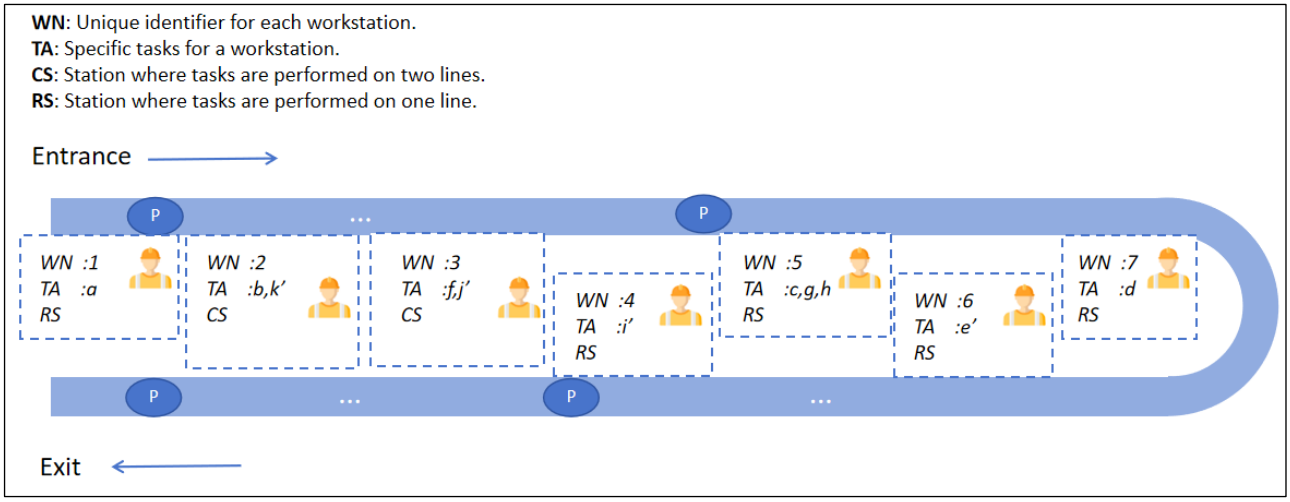

Figure 5. Design for U-line Balancing diagram

Step 2: Setting Tolerance Deviation

Despite the heuristic's effectiveness, there remains a cumulative idle time of 39 units, which could be even larger in different scenarios. To mitigate this, we establish a tolerance deviation d equals 0.23, indicating that we accept arrangement deviations less than 0.23 operators from a perfectly balanced U-shaped line. Proceeding with this deviation, we move to the next step.

Step 3: Allocation of Operators or Robots to Workstations

Using Formula 4 from the OAUB methodology, we determine the smallest n and corresponding N_j values, as presented in Table 2. The minimum task time T_{min} is 23 units (see Table 1). Calculations reveal that the smallest n satisfying all N_j is 2, resulting in N_1 to N_6 each being 5 operators. Each workstation meets the condition that

Table 2. Number of Operators Required for Each Workstation

Workstation	Assigned Tasks	Number of Operators N_j	Time Required per Product (units)	Idle Time (units)	Cumulative Idle Time (units)
1	*a*	5	11.8	0.2	0.2
2	*b, k′*	5	12	0	0.2
3	*j′, f*	5	12	0	0.2
4	*i′*	5	12	0	0.2
5	*c, g, h*	5	11.8	0.2	0.4
6	*e′*	5	12	0	0.4
7	*d*	2	11.5	0.5	0.9

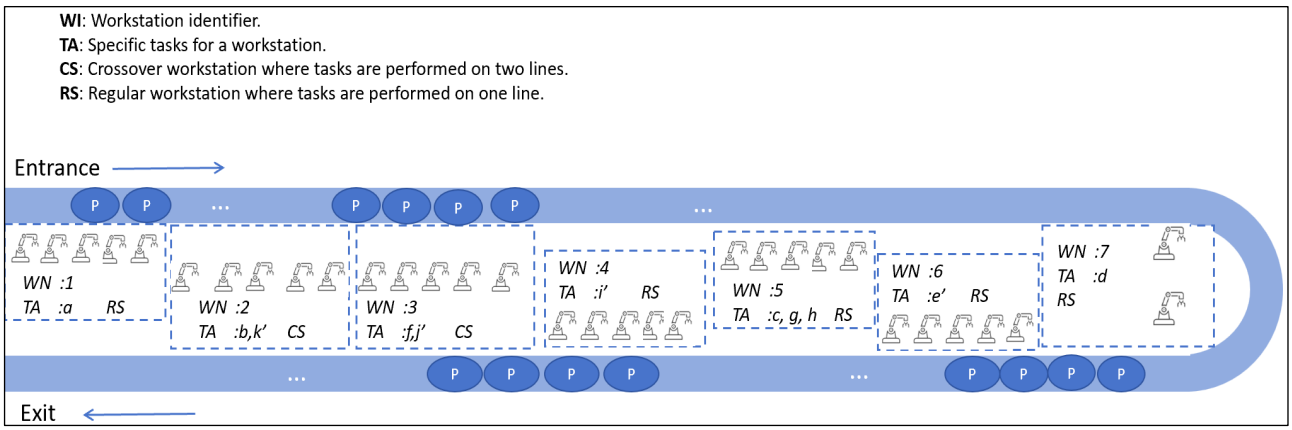

Figure 6. A Final Configuration Using the OAUB methodology

deviations are smaller than 0.23. For example, the optimal number of operators for Workstation 1 is 5.21; allocating 5 operators yields a deviation of only 0.21.

Step 4: Allocation of N_j Operators/Robots to Each Workstation

The computational results indicate that multiple operators or robots are assigned to certain workstations, as depicted in Figure 6. We adopt the interleaved task approach discussed earlier to ensure efficient utilization of resources. By implementing the OAUB methodology, which permits a deviation of 0.23 operators from a perfectly balanced line, we achieve a substantial reduction in idle time from 39 units to merely 0.9 units.

While the OAUB methodology has the potential to achieve complete balance with zero deviation, schedulability remains a critical consideration when addressing the QULBP. As illustrated in Table 3, attaining complete balance with zero deviation necessitates an impractically large number of operators, rendering it unfeasible for current manufacturing environments. This envisioned scenario might become feasible in future precision-operated robotic factories but exceeds the capabilities of current manufacturing settings. Nonetheless, the concept holds potential benefits for microscale cooperation. For example, at the electron or particle level, such strategies could enhance computational efficiency, particularly in quantum computing applications. Additionally, the proposed approach can be integrated into algorithms aiming for complete balance when computational bottlenecks arise.

An Example for Layout Design for U-shaped Line Balancing (LDUB)

Step 1: Draw the Precedence Network

The modified precedence network, which was used in the previous section, serves as the foundation for the LDUB method. This network is a graphical representation of tasks and their dependencies, showing the sequence in which tasks must be completed.

Table 3. The Number of Operators Required for Complete Balance

Workstation	Assigned Tasks	Number of Operators	Time Required per Product (units)	Idle Time (units)
1	*a*	59	1	0
2	*b, k'*	60	1	0
3	*j', f*	60	1	0
4	*i'*	60	1	0
5	*c, g, h*	59	1	0
6	*e'*	60	1	0
7	*d*	23	1	0

 DOI: 10.25077/josi.v24.n1.p1-17.2025

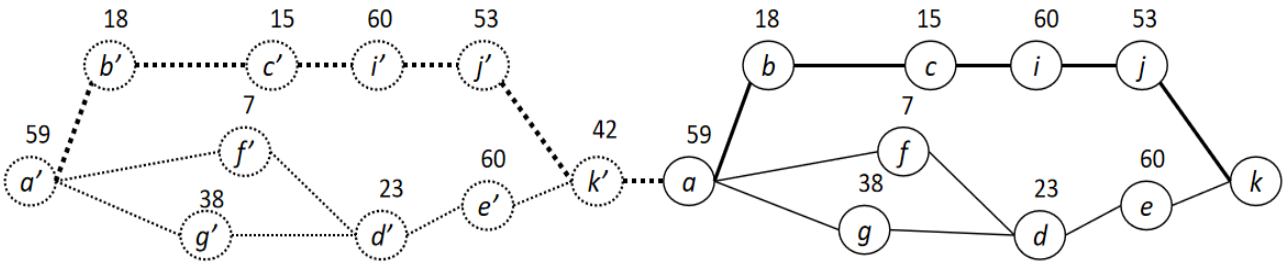

Figure 7. Modified Precedence Network after Removal of Independent Tasks

Step 2: Select Tasks That Can Be Operated Independently

In this step, we distinguish between independent and inseparable tasks. Independent tasks, typically related to the assembly of spare parts, can be isolated and performed separately without disrupting the overall assembly process. Meanwhile, inseparable tasks—though not always on the critical path—must remain linked due to interdependencies with other components or parts of the product. The guideline for identifying suitable tasks is:

- Bill of Materials (BOM) Review: Components below the top-level finished product (Level 0) or first-level assemblies are often eligible for separation. Tasks involving second-level or deeper subcomponents typically qualify as independent if they can be assembled in parallel.
- Work Breakdown Structure (WBS) Review: Aside from the first-level assembly tasks, tasks at the second or lower levels of the WBS can typically be performed independently, since they do not affect or block the primary assembly steps.

It is important to note that not all assemblies contain easily isolated tasks. Certain products may have no subcomponents, making it impossible to identify spare tasks. Once independent tasks are identified, those with shorter processing times should be selected first. For instance, in our example, task h is classified as independent and is thus removed from the precedence network, as shown in Figure 7.

Step 3: Arrange the tasks into Workstations

We then apply Avikal's [21] heuristic to arrange the remaining tasks into workstations, reducing the complexity of the task assignment. The results of this heuristic, as applied to the modified network after task h has been removed, are summarized in Table 4.

Step 4: Fill Idle Time with the Selected Tasks

In the final step, we address the idle time in workstations by assigning spare part assembly tasks. Workstations 5 and 7, which have enough idle time, can utilize this time for the assembly of spare parts. Specifically, Workstation 5 can assemble one spare part, while Workstation 7 can perform six spare part tasks in the Spare Part Zone (SPZ). The

Table 4. Results of the Heuristic in the Modified Network

Workstation	Assigned Tasks	Total Task Processing Time	Idle Time	Accumulative Idle Time
1	*a*	59	1	1
2	*b, k′*	60	0	1
3	*j′, f*	60	0	1
4	*i′*	60	0	1
5	*c, g*	53	7	8
6	*e′*	60	0	8
7	*d*	23	37	45

Table 5. Results of the Heuristic in the Modified Network

Workstation	Assigned Tasks	Total Task Processing Time	Idle Time	Accumulative Idle Time
1	*a*	59	1	1
2	*b, k′*	60	0	1
3	*j′, f*	60	0	1
4	*i′*	60	0	1
5	*c, g*	59	1	2
6	*e′*	60	0	2
7	*d*	59	1	3

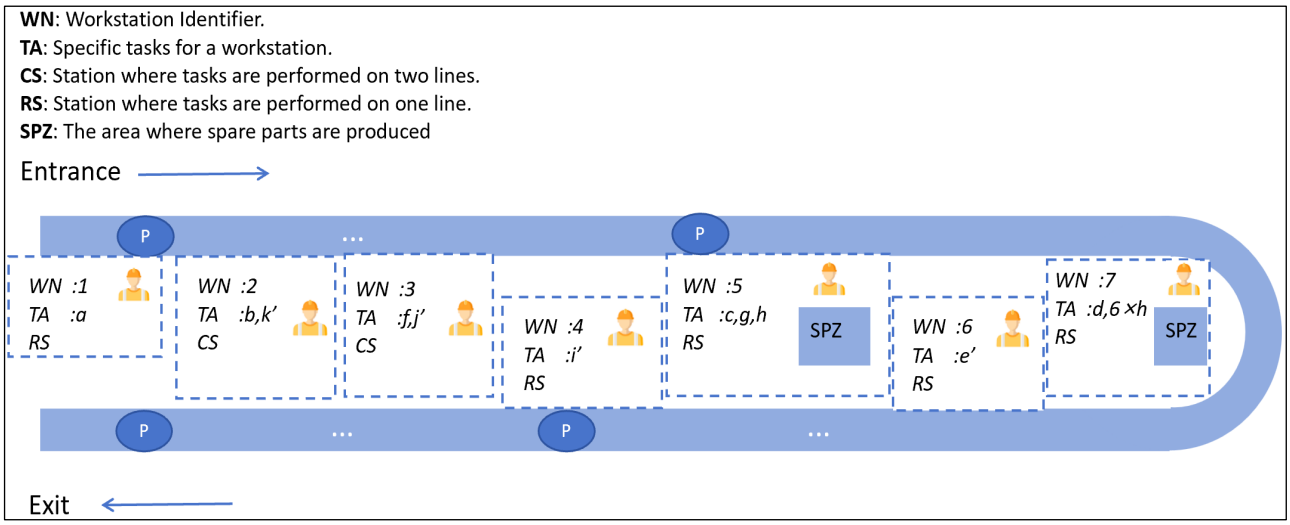

Figure 8. Configuration Using the LDUB Methodology

results after filling idle time with selected tasks are shown in Table 5, and the final workstation configuration is illustrated in Figure 8.

The application of the LDUB methodology provides greater flexibility compared to the OAUB methodology. However, a notable drawback of LDUB is the potential for unnecessary increases in spare part inventory due to the allocation of idle time for spare part assembly. To mitigate this, manufacturers should be cautious about overproduction—especially when demand for spare parts is low. A practical alternative is to use idle time for other beneficial activities, such as operator training or preventive maintenance. These tasks do not lead to excess inventory and can further optimize production processes. If there is a high demand for spare parts in another product line, and each task is brief or setup-free, those tasks could still be integrated into the Spare Part Zone without disrupting the main assembly flow. Hence, LDUB remains a practical way to address QULBP with fewer schedulability constraints, provided that inventory levels and workflow management are carefully monitored.

CONCLUSION

This study proposes two methodologies—OAUB and LDUB—that achieve quasi-complete balance in practical settings and complete balance in idealized scenarios. The OAUB methodology focuses on eliminating idle time through a systematic process that includes algorithmic task arrangement, setting an acceptable level of deviation, and allocating operators to stations. While this approach shows the potential for achieving complete balance, it demands a higher number of operators, enhanced collaborative skills, and a longer assembly line. Consequently,

 DOI: 10.25077/josi.v24.n1.p1-17.2025

OAUB is most suited to large-scale operations where operator capacity and line length are less constrained, but its scalability in resource-limited environments remains challenging. Conversely, the LDUB methodology simplifies the balancing process by identifying independent tasks from the non-critical path and filling idle time with spare part assembly tasks. Although LDUB reduces complexity and schedulability concerns, it poses the risk of generating excess inventory. Implementing demand-driven production strategies or using idle time for other beneficial tasks (e.g., training, maintenance) can help manage spare part accumulation effectively.

Both OAUB and LDUB underscore the importance of selecting an appropriate precedence network and integrating product design considerations, such as separable subassemblies, to streamline task assignments. This study demonstrates their practicality by adapting Avikal's [21] heuristic, though more specialized algorithms for managing the complexity of QULBP are still required. Future work should also explore heuristics and metaheuristics that address stochastic task times and operator-centric factors—such as fatigue and skill variation—and include pilot studies in environments with fluctuating demand and limited resources to validate these methods. Robotic assembly lines offer a particularly promising context for refining OAUB and LDUB, thanks to the precision of automated tasks. Meanwhile, micro-scale applications—such as quantum computing and high-throughput information systems—can benefit from idle time elimination on smaller, resource-sensitive scales. Specifically, the information transmission process can be viewed as a production process, where idle time may occur between a time-consuming task and subsequent steps.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the editors and reviewers for their insightful feedback, constructive suggestions, and careful scrutiny. Their dedication and generous investment of time have greatly improved the clarity and overall quality of this article.

CONFLICT OF INTEREST

The authors declare no conflict of interest regarding the publication of this paper.

FUNDING

The authors received no financial support for the research, authorship, and/or publication of this article.

References

[1] D. E. Nye, *America's Assembly Line*, Cambridge, MA, USA: MIT Press, 2015.

[2] M. Oksuz, K. Buyukozkan, and S. Satoglu, "U-shaped assembly line worker assignment and balancing problem: A mathematical model and two meta-heuristics," *Computers & Industrial Engineering*, vol. 112, pp. 246–263, 2017, doi: 10.1016/j.cie.2017.08.030.

[3] G. J. Miltenburg and J. Wijngaard, "The U-line line balancing problem," *Management Science*, vol. 40, no. 10, pp. 1378–1388, 1994, doi: 10.1287/mnsc.40.10.1378.

[4] A. L. Gutjahr and G. L. Nemhauser, "An algorithm for the line balancing problem," *Management Science*, vol. 11, no. 2, pp. 308–315, 1964, doi: 10.1287/mnsc.11.2.308.

[5] D. Sparling and J. Miltenburg, "The mixed-model U-line balancing problem," *International Journal of Production Research*, vol. 36, no. 2, pp. 485–501, 1998, doi: 10.1080/002075498193859.

[6] T. L. Urban, "Optimal balancing of U-shaped assembly lines," *Management Science*, vol. 44, no. 5, pp. 738–741, 1998, doi: 10.1287/mnsc.44.5.738.

[7] J. L. Hartnett, D. H. Kropp, and K. Suzaki, "The new manufacturing challenge: Techniques for continuous improvement," *Academy of Management Review*, vol. 13, no. 3, p. 500, 1988, doi: 10.2307/258098.

[8] C.-H. Cheng, J. Miltenburg, and J. Motwani, "The effect of straight- and U-shaped lines on quality," *IEEE Transactions on Engineering Management*, vol. 47, no. 3, pp. 321–334, 2000, doi: 10.1109/17.865901.

[9] [9] R. J. Schonberger and E. M. Knod, *Operations Management: Continuous Improvement*, 5th ed., Burr Ridge, IL, USA: Irwin Professional Publishing, 1994.

[10] J. Miltenburg, "Balancing U-lines in a multiple U-line facility," *European Journal of Operational Research*, vol. 109, no. 1, pp. 1–23, 1998, doi: 10.1016/S0377-2217(97)00169-0.

[11] F. Guerriero and J. Miltenburg, "The stochastic U-line balancing problem," *Naval Research Logistics*, vol. 50, no. 1, pp. 31–57, 2003, doi: 10.1002/nav.10043.

[12] E. Erel, I. Sabuncuoglu, and H. Sekerci, "Stochastic assembly line balancing using beam search," *International Journal of Production Research*, vol. 43, no. 7, pp. 1411–1426, 2005, doi: 10.1080/00207540412331320526.

[13] H. Gökçen and K. Ağpak, "A goal programming approach to simple U-line balancing problem," *European Journal of Operational Research*, vol. 171, no. 2, pp. 577–585, 2006, doi: 10.1016/j.ejor.2004.09.021.

[14] B. Toklu and U. Özcan, "A fuzzy goal programming model for the simple U-line balancing problem with multiple objectives," *Engineering Optimization*, vol. 40, no. 3, pp. 191–204, 2008, doi: 10.1080/03052150701651642.

[15] G. Widyadana, "Multi-objective model for balancing U-type assembly line with permanent and temporary workers," *Teknologi Industri*, vol. 11, no. 1, pp. 33–42, 2009, doi: 10.9744/jti.11.1.33-42.

[16] M. Yegul, K. Agpak, and M. Yavuz, "A new algorithm for U-shaped two-sided assembly line balancing," *Transactions of the Canadian Society for Mechanical Engineering*, vol. 34, no. 2, pp. 225–241, 2010, doi: 10.1139/tcsme-2010-0014.

[17] I. Kucukkoc and D. Z. Zhang, "Balancing of parallel U-shaped assembly lines," *Computers & Operations Research*, vol. 64, pp. 233–244, 2015, doi: 10.1016/j.cor.2015.05.014.

[18] P. Sivasankaran and P. Shahabudeen, "Literature review of assembly line balancing problems," *International Journal of Advanced Manufacturing Technology*, vol. 73, nos. 9–12, pp. 1665–1694, 2014, doi: 10.1007/s00170-014-5944-y.

[19] D. A. Ajenblit and R. L. Wainwright, "Applying genetic algorithms to the U-shaped assembly line balancing problem," in *Proc. IEEE Int. Conf. Evolutionary Computation*, Anchorage, AK, USA, 1998, pp. 96–101, doi: 10.1109/ICEC.1998.699329.

[20] Y. K. Kim, J. Y. Kim, and Y. Kim, "An endosymbiotic evolutionary algorithm for the integration of balancing and sequencing in mixed-model U-lines," *European Journal of Operational Research*, vol. 168, no. 3, pp. 838–852, 2006, doi: 10.1016/j.ejor.2004.07.032.

[21] S. Avikal, R. Jain, P. K. Mishra, and H. C. Yadav, "A heuristic approach for U-shaped assembly line balancing to improve labor productivity," *Computers & Industrial Engineering*, vol. 64, no. 4, pp. 895–901, 2013, doi: 10.1016/j.cie.2013.01.001.

[22] J. M. Nilakantan and S. Ponnambalam, "Robotic U-shaped assembly line balancing using particle swarm optimization," *Engineering Optimization*, vol. 48, no. 2, pp. 231–252, 2016, doi: 10.1080/0305215X.2014.998664.

[23] Z. Li, M. Janardhanan, Q. Tang, and Z. Zhang, "Models and algorithms for U-shaped assembly line balancing problem with collaborative robots," *Soft Computing*, vol. 27, no. 14, pp. 9639–9659, 2023, doi: 10.1007/s00500-023-08130-y.

[24] B. Zhang, L. Xu, and J. Zhang, "Balancing and sequencing problem of mixed-model U-shaped robotic assembly line: Mathematical model and dragonfly algorithm-based approach," *Applied Soft Computing*, vol. 98, Art. no. 106739, 2020, doi: 10.1016/j.asoc.2020.106739.

[25] Z. Mao, Y. Sun, K. Fang, D. Huang, and J. Zhang, "Balancing and scheduling of assembly line with multi-type collaborative robots," *International Journal of Production Economics*, vol. 271, Art. no. 109207, 2024, doi: 10.1016/j.ijpe.2024.109207.

[26] A. Nourmohammadi, M. Fathi, and A. H. C. Ng, "Balancing and scheduling human–robot collaborated assembly lines with layout and objective consideration," *Computers & Industrial Engineering*, vol. 187, Art. no. 109775, Feb. 2024, doi: 10.1016/j.cie.2023.109775.

[27] A. Nourmohammadi, T. Arbaoui, M. Fathi, and A. Dolgui, "Balancing human–robot collaborative assembly lines: A constraint programming approach," *Computers & Industrial Engineering*, vol. 205, Art. no. 111154, Jul. 2025, doi: 10.1016/j.cie.2025.111154.

[28] L. Tiacci, "Mixed-model U-shaped assembly lines: Balancing and comparing with straight lines with buffers and parallel workstations," *Journal of Manufacturing Systems*, vol. 45, pp. 286–305, 2017, doi: 10.1016/j.jmsy.2017.07.005.

[29] E. Álvarez-Miranda and J. Pereira, "On the complexity of assembly line balancing problems," *Computers & Operations Research*, vol. 108, pp. 182–186, 2019, doi: 10.1016/j.cor.2019.04.005.

[30] [30] A. Fattahi, S. Elaoud, E. Sadeqi Azer, and M. Turkay, "A novel integer programming formulation with logic cuts for the U-shaped assembly line balancing problem," *International Journal of Production Research*, vol. 52, no. 5, pp. 1318–1333, 2014, doi: 10.1080/00207543.2013.832489.

[31] A. Lester, *Project Management, Planning and Control: Managing Engineering, Construction and Manufacturing Projects to PMI, APM and BSI Standards*, 7th ed., Oxford, U.K.: Elsevier, 2017.

[32] J. R. Jackson, "A computing procedure for a line balancing problem," *Management Science*, vol. 2, no. 3, pp. 261–271, 1956, doi: 10.1287/mnsc.2.3.261.

Appendix: OAUB Algorithmic Example

We consider a modified precedence network based on Jackson's problem (see Figure 9) [32]. In this network, tasks are represented by nodes, and the connecting lines indicate the precedence relationships extending from the starting point '*a*' to the endpoint '*k*'. The letters within the nodes denote task identities, and the numbers above the nodes represent task times.

To enhance task assignment flexibility, a 'phantom' network is appended to the original precedence network. This addition allows tasks to be arranged either in a forward direction from '*a*' to '*k*' through the original network or in a backward direction from '*k*″ to '*a*″ through the phantom network. The longest path in the network, known as the critical path, is highlighted in bold.

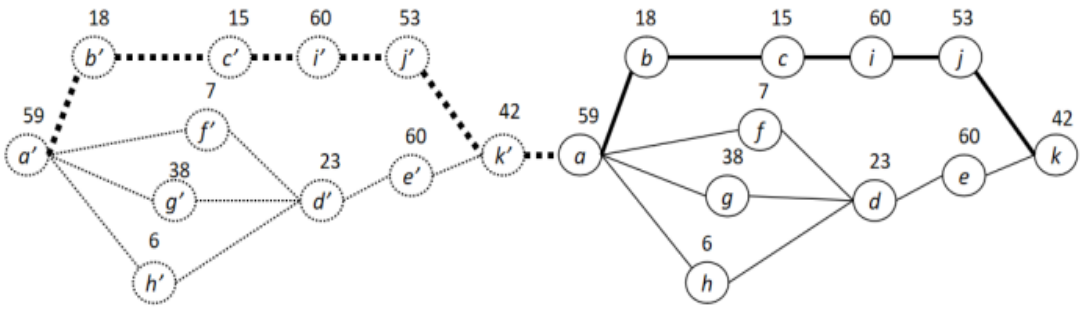

Figure 9. A Precedence Network with a 'Phantom' Network

Tasks on the critical path are prioritized and allocated to workstations first. Tasks on non-critical paths are only assigned if assigning critical path tasks alone would violate precedence relationships. Once tasks are assigned from

either network to a workstation, they are marked off the diagram. The notations used in executing the heuristic are as follows:

- C represents the cycle time;
- j represents the index of workstation;
- i represents the index of tasks, and tasks are {a, b, ..., k};
- $T_j(\cdot)$ represents the total task time for the workstation j when a set of tasks is assigned;
- SA represents the set of tasks is assigned to stations;
- SU represents the set of tasks is unassigned to stations;
- SCP represents the set of tasks on the critical path;
- SCP^{--} represents the set of tasks on non-critical path.

Illustration of the Computational Procedure

To simplify the problem, we apply Avikal's heuristic, which involves the following seven iterations:

- Initialization
  1. Set SU = {a, b, ...k, a', b', ..., k'} and j = 1.
  2. From Figure 5, identify SCP = {a, b, c, i, j, k, a', b', c', i', j', k'} and SCP^{--} = SU-SCP.
  3. Establish the cycle time C = 60 units.
- Iteration 1
  1. Create a new, empty workstation;
  2. Available tasks are {a} and {k'};
  3. Evaluate task time: $T_1(a)$ = 59 units and $T_1(k')$ = 42 units. Since $T_1(a) > T_1(k')$, assign task a to Workstation 1 (SA = {a});
  4. Mark off tasks {a} and {a'} from the network, updating SU = {b, c, ... k, b', c', ... k'};
  5. Increment j to 2.
- Iteration 2
  1. Create a new, empty workstation;
  2. Available tasks are as follows: {k'}, {b}, {f}, {g}, {h} {b, c}, {b, f}, {b, g}, {b, h}, {b, k'}, {f, g}, {f, h}, {f, k'}, {g, h}, {h, k'}, {b, c, f}, {b, c, h}, {b, f, h}, {f, g, h}, {f, h, k'}, {b, c, f, h};
  3. The combination {b, k'} yields a total task time of 60 units, matching the cycle time;
  4. Assign {b, k'} to Workstation 2 (SA={b, k'}) as it is on the critical path;
  5. Mark off the tasks {b}, {b'}, {k} and {k'} from the network, updating SU={c, d, ... j, c', d', ... j'};
  6. Increment j to 3.
- Iteration 3
  1. Create a new, empty workstation;
  2. Available tasks are as follows: {j'}, {e'}, {c}, {f}, {g}, {h}, {j', f}, {j', h}, {c, f}, {c, g}, {c, h}, {f, g}, {f, h}, {g, h}, {c, f, h}, {c, f, g}, {f, g, h};
  3. Both {e'}, {j', f}, and {c, f, g} have total task times of 60 units;
  4. Assign {j', f} to Workstation 3 (SA={j', f}) due to higher priority on the critical path;
  5. Mark off the corresponding tasks from the network, updating SU = {c, d, g, h, e, i, c', d', g', h', e', i'};
  6. Increment j to 4.
- Iteration 4
  1. Create a new, empty workstation;
  2. Available tasks are as follows: {e'}, {i'}, {g}, {h}, {c}, {c, g}, {c, h}, {g, h}, {c, g, h};

 DOI: 10.25077/josi.v24.n1.p1-17.2025

3. Tasks *{i'}* and *{e'}* both have task times of 60 units;
4. Assign task *{i'}* to Workstation 4 (*SA={i'}*) since it is on the critical path;
5. Mark off the corresponding tasks from the network, updating *SU = {c, d, g, h, e, c', d', g', h', e'}*;
6. Increment *j* to 5.

- Iteration 5
  1. Create a new, empty workstation;
  2. Available tasks are as follows: *{e'}, {g}, {h}, {c}, {c, g}, {c, h}, {g, h}, {c, g, h};*
  3. The combination *{c, g, h}* totals 59 units, slightly less than $T_4(e')$ =60;
  4. Since task *{c}* is on the critical path, we decide to assign tasks *{c, g, h}* to the workstation 5 *SA = {c, g, h};*
  5. Mark off the corresponding tasks from the network, updating *SU = {d, e, d', e'};*
  6. Increment *j* to 6.
- Iteration 6
  1. Create a new, empty workstation;
  2. Available tasks are as follows: *{d}, {e'}*;
  3. Task *{e'}* has a task time of 60 units;
  4. Assign *{e'}* to Workstation 6 (SA=*{e'}*);
  5. Mark off the corresponding tasks from the network, updating *SU = {d, d'};*
  6. Increment *j* to 6.
- Iteration 7
  1. Create a new, empty workstation;
  2. The only remaining task is *{d}* or *{d'};*
  3. Assign *{d}* to Workstation 7 (SA=*{d}*) and terminate the process.

AUTHORS BIOGRAPHY

Zening Wu is a Ph.D. student in the Faculty of Engineering and Information Sciences, University of Wollongong, Australia. He earned a Master of Project Management from the University of Sydney in 2025 and a Bachelor's degree in Industrial Engineering from Foshan University, China, in 2023. His research interests include optimization, systems control, human–robot collaboration and intelligent manufacturing.

Wenqian Luo is a Master of Project Management candidate in the Faculty of Engineering, University of Sydney, Australia. Her interests centre on optimization, resource planning and process management in complex engineering projects.