

Designing a Labor Optimization Model for Wesco

by

Andrea Vega Rodriguez

and

Sutherland Hart Wiggins

Submitted on May 8th, 2026 in Partial Fulfillment of the
Requirements for the Degree of Master of Applied Science in Supply Chain Management at the
Massachusetts Institute of Technology

Abstract

Labor allocation in warehouse distribution centers is a persistent operational challenge: new orders arrive continuously throughout a shift, worker availability fluctuates, and even small mismatches between labor supply and demand result in delayed fulfillment, overtime costs, and missed customer commitments. This capstone presents a mixed integer linear programming (MILP) model designed to align same-day task demand with available labor supply across the six functional departments of a Wesco distribution center. Given the tasks currently on the warehouse floor and the workers available, the model determines the optimal number of workers to allocate to each department by optimizing six hierarchical objectives: completing urgent, same-day tasks before deferrable ones, ordering remaining work by priority code, customer due date, and time waiting on the warehouse floor, and minimizing total labor cost only after task urgency is fully resolved.

The model was validated at two scales: a small-scale set confirmed that all constraints and objective logic behaved as designed, and a full-scale set of 41,124 tasks solved 452,484 decision variables across 327,334 constraints in under two seconds. At full scale, the model successfully allocated available workers across all six departments, completing 26,702 same-day tasks with only 12.4 total overtime hours required. Beyond same-day allocation, the model identifies forward departmental staffing needs as the known backlog drains over the planning horizon, giving managers a baseline for the days ahead. In all cases, the model recommends how many workers each department needs rather than prescribing individual worker assignments, preserving manager discretion over who moves and where. More broadly, this model gives Wesco's warehouse managers a consistent, data-driven tool to allocate labor by department rather than by intuition alone.

Capstone Advisor: Milena Janjevic
Title: Research Scientist



Table of Contents

1	Introduction	1
1.1	Motivation	1
1.2	Problem Statement and Key Questions	1
1.3	Project Goals and Expected Outcomes	2
2	State of the Practice	2
3	Methodology	3
3.1	Phase 1: Data Collection and Preparation	3
3.2	Phase 2: Iterative Model Development	4
3.3	Sets and Indices	4
3.4	Parameters	5
3.5	Decision Variables	5
3.6	Objective Function	6
3.7	Constraints	7
3.8	Model Validation	8
4	Results and Discussion	8
4.1	Model Assumptions	9
4.2	Model Output by Validation Set	9
4.3	Model Implications	11
5	Recommendations	11
	References	13

1 Introduction

Wesco is a Fortune 500 company specializing in the distribution of electrical, industrial, and communications products to commercial and industrial customers. They operate more than 300 warehouse locations across the United States, which vary significantly in size and operational purpose. Some serve as large, high-capacity distribution centers (DC) built to handle full scale fulfillment operations, while others are smaller, decentralized sites strategically positioned close to customers with limited operations. The company employs over 10,000 people nationwide to support this extensive and varied distribution network (Wesco, 2025).

1.1 Motivation

Labor planning is one of the most challenging and critical levers for operational efficiency and cost control in a warehouse or distribution center. Labor availability is inherently variable; unplanned absences and partial shift attendance mean the headcount a manager expects at the start of a shift is not always the one they finish with. Compounding this, task demand fluctuates constantly as new orders arrive throughout the day, continuously shifting the priority of existing work. Shifting of existing work requires changes to staffing plans, and this constant need to rebalance workers across departments has historically been a reactive decision, relying on manager judgment rather than a data-driven approach. Even small mismatches between labor supply and operational demand can create disruptions such as delayed orders, overtime costs, or deferred workloads. Wesco has experienced these costly disruptions firsthand. Recognizing the operational strain and opportunity costs this creates, the company is motivated to explore data-driven approaches that can intelligently balance workforce allocation to task demand while managing cost. Better alignment between labor supply and operational demand will enhance service reliability, improve warehouse productivity, and strengthen the company's competitive position.

1.2 Problem Statement and Key Questions

Our sponsor finds it difficult to align warehouse labor staffing by department, a term they use to describe distinct functional areas in the warehouse, such as Receiving, Inventory, Cutting, Picking, Shipping, and Auditing. Each of these departments requires adequate staffing that aligns not only with the tasks on the floor at the start of a shift, but throughout the shift as new orders arrive and priorities change. Even with some visibility into current demand, translating that into optimal staffing decisions across departments remains a challenge. The dynamic nature of both task demand and workforce availability, compounded by the varying capabilities of workers across departments, makes it challenging to allocate the right number of workers to the right departments at the right times, resulting in disruptions to service quality and cost. These challenges persist regardless of whether total headcount is sufficient, as having enough workers in the warehouse does not guarantee they are allocated where demand requires them most.

To develop an effective labor optimization model, we must first understand:

- The current labor planning process and its constraints, including how staffing decisions are made today.
- The availability and quality of operational data across warehouse departments.
- How to classify tasks by urgency to correctly prioritize labor allocation.
- How to maintain recommendations' feasibility in the hectic DC environment by striking a compromise between operational viability and model accuracy.

1.3 Project Goals and Expected Outcomes

Our solution will deliver a labor optimization model designed to align same-day task demand with available labor supply. Given the tasks currently on the warehouse floor and the workers available, the model will determine the optimal number of workers to allocate to each department, prioritizing fulfillment based on the priority hierarchy established with our sponsor. As new orders arrive or workforce availability changes, managers will be able to re-run the model to ensure labor is always deployed most appropriately. Rather than prescribing individual assignments, the model will highlight each department's tasks and the number of workers needed to complete them, giving managers the information they need to make informed reallocation decisions. In doing so, the model provides a data-driven foundation for staffing allocation decisions, giving Wesco a consistent and repeatable tool that is grounded in warehouse realities and built for managers to trust and act on.

2 State of the Practice

The central problem our capstone is solving is the development of a labor optimization model that better matches labor supply with same-day task demand to ensure Wesco has the right number of workers allocated to each department.

In preparation for this capstone, we reviewed key literature to understand various ways managers attempt to solve labor scheduling problems. The literature reveals that labor scheduling problems are typically handled using either rule-based approaches like priority rules or dispatching heuristics, or mathematical optimization models such as mixed integer linear programming (MILP). While rule-based methods are simple to implement and understand, they rely on managers' judgment and often struggle to handle multiple competing objectives or complex constraint sets systematically. In contrast, mathematical optimization models can evaluate all feasible staffing options and identify solutions that best meet defined objectives across multiple criteria simultaneously (Pinedo, 2022). To inform our model design, we reviewed prior research on workforce scheduling, shift planning, and resource allocation in different industries. This section synthesizes key findings and explains why a MILP-based framework is best suited to address our Wesco's problem.

Our review began with the work of Ernst, Jiang, Krishnamoorthy, and Sier (2004), *Staff scheduling and rostering: A review of applications, methods and models*, published in the *European Journal of Operational Research*. Their review examines staff scheduling across different applications and demonstrates how different environments, like warehousing, involve multiple constraints that often compete with one another. The emphasis on treatment of constraints is directly relevant to our work because of the complexity that emerges when translating such requirements into mathematical formulation. Therefore, a critical consideration is how these constraints are represented in our model, whether as hard constraints that must always be satisfied or as soft constraints that allow flexibility with associated penalties. It is the decisions we make when adding these constraints that increase the complexity of the model and the quality of the solution.

Building on the foundation provided by Ernst et al. (2004), we examined Malik's (2012) thesis "Decision Support Tool for Dynamic Workforce Scheduling In Manufacturing Environments," where the author emphasizes the dynamic nature of labor scheduling and the need for effective systems to remain responsive due to frequent changes in demand across shifts, task priorities, and varying worker availability. The author's decision framework highlights how effective schedules must continuously adapt and evolve in response to those conditions, while still respecting precedence relationships and other operational constraints that must be completed in the correct sequence. Although Malik's scenarios are situated in manufacturing, Wesco

faces parallel challenges in warehousing, where fluctuating customer demand cascades into shifting tasks and corresponding adjustments in workers’ assignments. This parallel underscores the relevance of Malik’s framework to our context, as both domains require labor schedules that adapt to shifts in demand and resource availability.

To examine model formulation structure and notation, we reviewed Aykin’s (2000) paper *A Comparative Evaluation of Modeling Approaches to the Labor Shift Scheduling Problem*, published in the *European Journal of Operational Research*. Aykin demonstrated that different shift scheduling models in literature could be unified under a single mathematical framework. Rather than requiring distinct formulations for different scheduling scenarios, the generalized MILP structure accommodates various constraint types and objective functions through parameter adjustments, meaning constraints can be activated or relaxed without redesigning the entire model. Equally important, Aykin’s analysis revealed that model complexity does not guarantee better solutions, and that simpler formulations often perform better and can be solved more efficiently. This directly shaped our approach to model design: we focus on keeping the model flexible and simple rather than overly detailed from the start. Our model begins with basic constraints and adds more complexity only when needed based on testing or feedback from our sponsor.

Collectively, these works informed our approach to implement an iterative MILP in complementary ways. Pinedo (2022) provided foundational context on scheduling methodologies, Ernst et al. (2004) highlighted the criticality of constraint treatment, Malik (2012) reinforced the need for responsive scheduling in dynamic environments, and Aykin (2000) provided a clear and practical mathematical formulation structure that we adapted to fit our sponsor’s warehouse requirements.

3 Methodology

This section explains our approach to developing a MILP-based labor optimization model for our sponsor’s warehouse operations. We begin by justifying our selection of a MILP framework based on findings from our literature review, then describe our two-phase implementation approach. Phase 1 focuses on collecting and preparing data required for model inputs, while Phase 2 details our iterative process for building the MILP model in Gurobi Optimizer. Our objective is to create a model that considers upcoming priority demand tasks by department, calculates the number of workers required in each department to meet that demand, and provides managers with a clear staffing allocation recommendation.

As noted in Section 2, State of the Practice, we selected a MILP-based optimization approach for our labor allocation problem. While rule-based methods are straightforward to implement, they struggle to systematically handle multiple competing objectives and complex constraints such as those present in our sponsor’s multi-department warehouse operations. MILP provides a framework to evaluate all feasible combinations of staffing and task assignments and identify optimal solutions while accommodating the varied constraint types identified (Pinedo, 2022). Following Aykin’s (2000) methodology, MILP allows us to progressively refine the model’s structure and constraints without a complete redesign.

3.1 Phase 1: Data Collection and Preparation

As input into our MILP model, we utilize datasets provided by our sponsor covering three operational streams: inbound, internal, and outbound warehouse operations. Each dataset contains departmental task records detailing task name, number, quantity, and task duration. To build a complete and accurate model, we worked closely with our sponsor to fill in a few key data elements. First, the existing datasets lacked priority codes, customer due dates, and drop-in dates, all of which are used by the model to determine which

tasks should be worked and in what order. Where these fields were absent, we applied data proxies: ship date, our sponsor’s internal target date to ship an order, is used as a substitute for customer due date, the date by which the customer expects delivery. Assigned date is used as a substitute for task drop-in date. The drop-in date represents the moment a task becomes available to the warehouse floor, when the work drops into the building and is ready to be processed. The assigned date, used here as a proxy, represents the moment a worker claims a task in our sponsor’s warehouse management system, marking when the task enters an individual worker’s queue rather than when the work physically arrives at the facility or when it begins. For the Receiving department specifically, neither a ship date nor a customer due date was available, so assigned date is used as a substitute. While functional, these proxies introduce risk to priority ordering, as they do not reflect true task urgency, and will be replaced when the sponsor gets the appropriate data. We then consolidated and standardized the three operational datasets into a single unified input structure incorporating the data proxies described above. This process was repeated twice, once for the small-scale validation set and once for the full-scale validation set, the details of which are discussed in Section 3.8.

We worked with Wesco to obtain a comprehensive priority score to classify tasks by urgency: a priority code below 30 is treated as high priority and “non-movable,” meaning they must be worked that day. This threshold was established based on early conversations with the sponsor for this model only, as it is site-specific and will differ across warehouse locations. Tasks that arrive past their due date are also treated as non-movable, as they are already late upon arrival. The remaining tasks are classified as “movable” as they are lower-priority and eligible to be deferred. Among movable tasks, priority is determined first by priority code, with customer due date and drop-in date serving as a tiebreaker. For example, consider two movable tasks with the same priority code and the same duration. Once non-movable tasks have been scheduled and capacity remains, the model must determine which movable task to pull into the current day first. It does so by comparing customer due dates, scheduling the task with the earlier due date first. Only if two tasks share the same priority code and customer due date does the model fall back to drop-in date, prioritizing the task that has been waiting in the warehouse longer.

3.2 Phase 2: Iterative Model Development

The MILP model in Gurobi is developed through progressive refinement. Building the model iteratively allows us to validate outputs against the sponsor’s expertise before introducing additional complexity into the model. The current iteration introduces a multi-objective framework that prioritizes task completion by urgency, scheduling non-movable tasks first, followed by movable tasks ordered by priority score, due date, and age in queue.

3.3 Sets and Indices

Sets define the elements the model operates over, and indices are used to reference individual members within each set. The model is defined over three sets that establish the scope of the optimization: departments, tasks, and planning days.

- G : set of departments (groups); index $g \in G$
- J : set of tasks (jobs); index $j \in J$
- T : set of planning days, $T = \{1, 2, \dots, \bar{T}\}$; index $t \in T$

3.4 Parameters

Parameters are fixed inputs that define the operating environment of the model. They are known before the model runs and remain constant throughout the optimization.

3.4.1 Task Characteristics

The following parameters define the characteristics of each task, including its urgency, department assignment, duration, and timing fields used to determine priority ordering.

- $m_j \in \{0, 1\}$: 1 if task j is non-movable, 0 if movable ($j \in J$)
- $e_{jg} \in \{0, 1\}$: 1 if task j is processed in department g , 0 otherwise
- t_j : due date of task j , pre-processed so $t_j \leq \bar{T} + 1$ (days)
- d_j : task duration, the estimated time for a worker to complete task j (hours), $j \in J$
- P_j : overall priority score for task j
- o_j : warehouse drop date (period) of task j
- a_j : age in queue, computed as $|o_j|$, the number of days between the drop-in date of task j and today

3.4.2 Workforce

The following parameters define the labor environment within which the model operates, including shift length, worker utilization, available headcount, and cost structure.

- H : working hours per day / shift length (hours)
- u : usable fraction of a shift (e.g., 0.80)
- $W_t^{\max}$: maximum number of workers available on day t
- C_w : base hourly cost per worker (\$/hr)
- v : overtime pay multiplier (e.g., 1.5)
- C_{OT} : overtime hourly cost, $C_{OT} = C_w \cdot v$ (\$/hr)

3.5 Decision Variables

Decision variables are the unknowns the model solves for. Unlike parameters, these values are not known in advance and are determined by Gurobi through the optimization process.

- $x_{jgt} \in \{0, 1\}$: 1 if task j is completed in department g in period t , 0 otherwise
- $W_{gt} \in \{0, 1, \dots, W_t^{\max}\}$: number of workers assigned to department g in period t

- $S_j \in T \cup \{\tilde{T} + 1\}$: scheduled day of task j
- $OT_{gt} \geq 0$: number of overtime hours used for department g in period t

Together these four decision variables give the model full flexibility to determine not just how many workers each department needs, but when each task gets done and whether overtime is required to meet demand.

3.6 Objective Function

The model optimizes six hierarchical priorities sequentially. Each objective is solved in order, with its optimal value locked in as a constraint before the next objective is optimized.

$$\text{Priority 1 (Non-movable tasks must be done today): } \min \sum_{\substack{j \in J \\ m_j=1}} \frac{1}{P_j} S_j \quad (1)$$

$$\text{Priority 2 (Movable tasks must be done by priority): } \min \sum_{\substack{j \in J \\ m_j=0}} \frac{1}{P_j} S_j \quad (2)$$

$$\text{Priority 3 (Movable tiebreaker: earlier due date wins): } \min \sum_{\substack{j \in J \\ m_j=0}} t_j S_j \quad (3)$$

$$\text{Priority 4 (Movable tiebreaker: older warehouse age wins): } \min \sum_{\substack{j \in J \\ m_j=0, |o_j|>0}} \frac{1}{|o_j|} S_j \quad (4)$$

$$\text{Priority 5 (Fill today with movable tasks): } \max \sum_{\substack{j \in J \\ m_j=0}} x_{j,0} \quad (5)$$

$$\text{Priority 6 (Minimize total regular + overtime labor cost): } \min \sum_{g \in G} \sum_{t \in T} (W_{gt} C_w H u + OT_{gt} C_{OT}) \quad (6)$$

The ordering of these six priorities reflects the operational hierarchy established directly with Wesco. Non-movable tasks represent firm customer commitments that cannot slip under any circumstance, so their completion takes absolute precedence before any other consideration. Once non-movable demand is satisfied, movable tasks are ordered by priority code, due date, and age in queue, ensuring the remaining workload is sequenced by urgency rather than convenience. For Priorities 1 and 2, the priority code is inverted as $1/P_j$ because lower codes indicate higher urgency in Wesco's system, ensuring the most urgent tasks are weighted more heavily and scheduled earliest. Once non-movable demand is satisfied and movable tasks are ordered by urgency, the model maximizes worker utilization by filling remaining capacity with as many movable tasks as possible before turning to cost minimization.

This sequential approach is enforced by Gurobi's multi-objective framework. As a result, higher-priority objectives are never sacrificed to improve lower-priority ones, giving the model a mathematically guaranteed ordering rather than a trade-off between competing goals. In practice, this means non-movable tasks are always completed today before any movable task scheduling begins, movable tasks are then ordered strictly by priority code, with labor cost minimization as the final consideration. Once all tasks are scheduled, the model determines the minimum number of workers needed per department, favoring regular hours over overtime to minimize total labor cost.

3.7 Constraints

The model is governed by seven constraints that together ensure feasibility, enforce task urgency rules, and prevent the model from recommending staffing plans that could not be executed in practice.

- Each task done only once:

$$\sum_{g \in G} \sum_{t \in T} x_{jgt} \leq 1 \quad \forall j \in J \quad (7)$$

- Non-movable tasks only done on due date:

$$x_{jgt} = 0 \quad \forall j \in J \text{ with } m_j = 1, \forall g \in G, \forall t \in T \setminus \{t_j\} \quad (8)$$

- Task start day is the planning day the model assigns a task to begin, determined by the priority logic:

$$S_j = \sum_{g \in G} \sum_{t \in T} t \cdot x_{jgt} \quad \forall j \in J \quad (9)$$

- Movable tasks cannot start before their drop-in date:

$$S_j \geq \max(0, o_j) \quad \forall j \in J : m_j = 0 \quad (10)$$

- Capacity per department and period, utilization-adjusted:

$$\sum_{j \in J} d_j x_{jgt} \leq W_{gt} H u + O T_{gt} u \quad \forall g \in G, \forall t \in T \quad (11)$$

- Total workers across all departments cannot exceed building headcount:

$$\sum_{g \in G} W_{gt} \leq W_t^{\max} \quad \forall t \in T \quad (12)$$

- Minimum worker constraint:

$$W_{gt} \geq 1 \quad \forall g \in G, \forall t \in T \quad (13)$$

Together, these constraints ensure the model never produces an infeasible solution. Non-movable tasks are guaranteed to be completed today regardless of capacity, as the capacity constraint absorbs workload overflow through overtime rather than dropping tasks. Both regular and overtime hours are subject to the utilization factor u , reflecting that workers are assumed to be productive for 80% of a shift rather than its full duration. Overtime is only introduced after all available regular worker hours are fully utilized, as the cost minimization objective favors regular labor over overtime. Worker allocation is treated as a decision variable rather than a fixed input, allowing the model to determine the optimal distribution of available staff across departments based on existing task demand. The minimum worker constraint ensures every active department receives at least one worker, preventing the model from recommending a zero-staffing solution for departments with scheduled tasks. The building headcount constraint ensures the total workers deployed across all departments never exceeds the available workforce entered by the manager.

3.8 Model Validation

The model was validated in two stages using two datasets referred to throughout as the small-scale validation set and the full-scale validation set. The small-scale validation set comprised 75 tasks drawn from the same three client streams: inbound, internal, and outbound. These tasks were manually separated into their respective departments, and the dataset was intentionally small so that every decision the optimizer made could be verified by hand. The model successfully scheduled tasks by priority and recommended how many workers each department needed. It calculated overtime hours only when task demand exceeded available capacity, and it produced no infeasible solutions. Each decision variable, objective function, and constraint behaved as designed. With logic validated at small scale, we proceeded to the second stage of validation.

The second stage applied the same model to the full-scale validation set, comprising 41,124 unique tasks drawn from the same three client streams using February 24, 2026 as the reference date. Where the small-scale set had 75 tasks, this dataset reflects a volume far more representative of the warehouse scale. The objective was to confirm that the model remained feasible, solved efficiently, and continued to produce reliable outputs at that volume. Both the small-scale and full-scale validation sets rely on the proxy fields described in Section 3.1; accordingly, neither can confirm that the allocation recommendations the model produces for February 24, 2026, reflect the actual decisions managers made on the floor that day. Until true priority codes, customer due dates, and drop-in dates are available from the sponsor, there is no reliable basis for that comparison. The results of both stages are discussed in Section 4.

4 Results and Discussion

This section presents the model's outputs and discusses their operational implications for Wesco. Results are organized around two time horizons: Day 0 and the remaining planning period, Days 1 through potentially Day 9 depending on the volume of tasks in the system.

Day 0, representative of today, February 24, 2026, is the primary actionable output of the model. It gives managers an optimized worker allocation recommendation for today rather than leaving that to intuition, ensuring labor is deployed where task demand requires it most. The remaining planning period spans up to Day 9, though the backlog may clear sooner dependent on volume. Rather than a true optimization, this portion provides a directional forward staffing estimate based solely on tasks already in the system, assuming no new orders arrive. This forward look reflects the known backlog draining over time and is intended as a planning reference only, a baseline that will shift as new tasks enter the system throughout the week.

The model produces three outputs across both time horizons. First, the department-level summary reports the recommended number of workers per department, the number of movable and non-movable tasks scheduled, total task hours, and whether overtime is required and, if so, how many hours and at what cost. Second, the warehouse-level summary aggregates those figures across all departments, giving managers a single snapshot of total worker deployment, total task load, and total overtime cost. Third, both summaries are exported to Excel with two tabs: a task schedule listing every individual task with its assigned day, department, duration, due date, and movability classification so that each department can filter to their own work queue, and a department summary tab that rolls those tasks up into total task counts, total hours, workers assigned, overtime hours, and overtime cost per department so managers have a consolidated view to act on.

4.1 Model Assumptions

Several simplifying assumptions underlie the current model. Every task is assumed to be completable within a single day with no carryovers, and every task within the 10-day window must be assigned, meaning nothing can be skipped or deferred beyond the planning horizon. The model operates at the department level only, seeing headcount rather than individuals, and every worker is assumed to carry the same flat hourly rate. The available headcount entered by the manager as an input to the model is assumed to be accurate, as the model's recommendations are dependent on it. There is no overtime cap, meaning unlimited overtime can be assigned to any department by the hour, ensuring that the model always finds a feasible solution regardless of demand. Finally, the priority code threshold of 30 is for this model only, meaning any task at or below 30 is treated as non-movable and set manually rather than derived from the data. These assumptions hold across both validation sets. Any assumptions specific to a validation set are noted within that section.

4.2 Model Output by Validation Set

We ran the small-scale validation set with 20 available workers, scheduling all 75 tasks across the planning horizon. Through manual verification, the model correctly identified all tasks with a priority code at or below 30 as non-movable and assigned them to Day 0, while movable tasks were sequenced by priority score with due date and age in queue applied as tiebreakers. Worker allocation and overtime performed as designed, with overtime appearing only where regular capacity was insufficient. With the small-scale logic confirmed, we proceeded to the full-scale validation set.

We ran the full-scale validation set on February 24, 2026 with 230 available workers. The model solved the problem comprising 327,334 constraints, 452,484 decision variables, and 1.478 million active relationships across six sequential objectives, reaching optimality in under two seconds.

Of the 230 workers available on the warehouse floor, the model deployed 225 to direct labor across departments: Shipping received 92, Picking 53, Receiving 31, Cutting 29, Inventory 19, and Auditing 1. The remaining five workers represent indirect labor freed to support warehouse operations outside the customer order queue. The model makes no recommendation for how those other five workers are deployed, as its scope is limited to direct labor allocation.

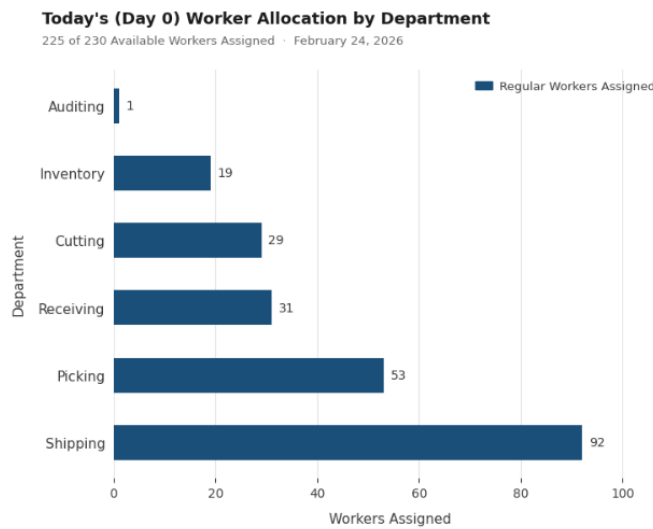


Figure 1. Day 0 worker allocation by department for February 24, 2026

Of the 26,702 tasks scheduled on Day 0, 22,648 (85%) were classified as non-movable and required completion today, while 4,054 (15%) were movable and eligible for deferral. Shipping dominated the task load, accounting for 11,242 tasks, the largest share of any department.

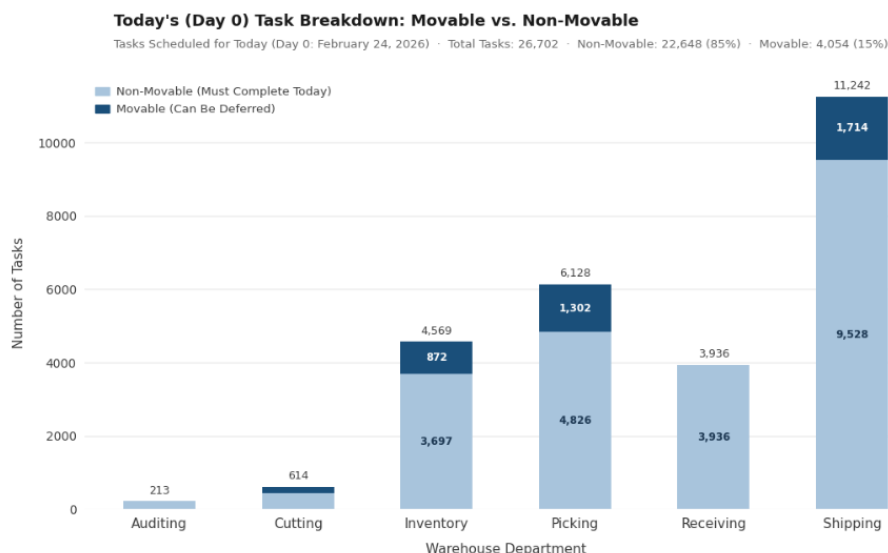


Figure 2. Day 0 task breakdown by department for February 24, 2026

One assumption worth noting in the context of the full-scale run is the treatment of inbound tasks. Because no ship date or customer due date exists for Receiving in Wesco's data, every inbound task is treated as non-movable and scheduled on Day 0. As can be seen in Figure 2, this is why Receiving appears entirely in the non-movable category with all 3,936 tasks worked today. Whether the concentration of all Receiving tasks on Day 0 reflects the true same-day nature of inbound work or is an artifact of the missing date fields is not yet confirmed and requires validation with Wesco.

To complete all 26,702 tasks on Day 0, the model required 12.4 overtime hours across four departments: Auditing 2.6 hours, Inventory 4.1 hours, Cutting 4.1 hours, and Picking 1.6 hours. Shipping and Receiving required no overtime. Because overtime is modeled as a continuous hourly variable rather than a full-shift commitment, the model determined in each case that absorbing residual task demand through fractional overtime hours was less expensive than assigning an additional worker to a full eight-hour shift. This behavior is governed by the no-overtime-cap assumption described in Section 4.1, which ensures the model always finds a feasible solution by absorbing demand overflow through additional hours rather than dropping tasks. This reflects the operational reality that a manager need not bring someone in for a full shift when 12.4 hours of overtime are sufficient to meet the demand for all non-movable tasks and the movable tasks the model was able to fit within Day 0's capacity.

To illustrate how the model's outputs translate to a single department, we examine Picking in detail. The model assigned 53 workers to fulfill 6,128 tasks, of which 4,826 were non-movable and 1,302 movable, with 1.6 hours of overtime required. In the Picking department alone, 27.9% of tasks (1,712 of 6,128) had already passed their ship date by the time they entered the queue. These tasks were declared "late" before a worker was assigned to fulfill them. This is likely overstated since ship date precedes the actual customer due date, and we expect this number to decrease once Wesco provides the true customer due date field. As noted in Section 3.8, without true priority codes, customer due dates, and drop-in dates there is no reliable basis for comparing the model's output against actual warehouse staffing decisions.

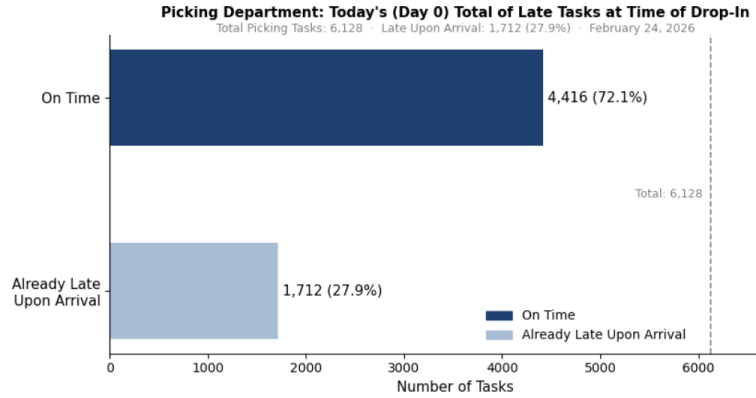


Figure 3. Picking department late tasks at time of drop-in for February 24, 2026

Having examined Day 0 in detail, we now turn to the forward staffing estimate described in Section 4. At the warehouse level, the model requires 43 workers on Day 1 (February 25, 2026), 46 on Day 2, and 41 on Day 3 to clear the remaining movable backlog. The slight increase from Day 1 to Day 2 reflects some deferred tasks becoming non-movable as their due dates approach, increasing urgency before the backlog is fully absorbed by Day 3. While this forward estimate will shift as new orders arrive, it gives managers a minimum staffing baseline: if no new work enters the system, these are the fewest workers needed to clear what is already on the floor.

4.3 Model Implications

Prior to this model, each warehouse managed staffing decisions independently, and no standardized tool existed to guide labor allocation across the organization. With new orders arriving constantly and priorities shifting throughout the day, no manager can manually evaluate all possible staffing combinations in the time available. At the scale of a single facility processing thousands of orders daily across multiple shifts, even small misalignments between labor supply and task demand result in delayed customer orders and missed commitments. This model gives managers a data-driven tool to manage that complexity, ensuring the right number of workers are allocated to each department based on actual task demand. Because the model does not dynamically interface with Wesco's warehouse management system, managers should refresh the model throughout the day with updated task demand to ensure staffing recommendations remain accurate as conditions change. Each refresh ensures the model is working from current demand, prioritizing non-movable tasks before movable ones so that overtime hours, when required, are spent on work that must be completed that day rather than work that could wait. This represents a meaningful shift from reactive labor planning to one grounded in data that is designed to support managers' decision making, not replace it.

5 Recommendations

To get the most out of the model in its current form, we recommend the following approach for warehouse managers. At the start of each shift, the manager enters the number of available direct labor workers and runs the model to generate an optimized department-level staffing recommendation. The Excel output provides two views: a department summary showing how many workers each area needs and whether overtime is required, and a full task schedule showing which tasks are prioritized for today versus deferred. From there, the manager can identify any staffing imbalances, departments where demand exceeds current allocation,

and use their own judgment to decide which workers to move and where. As new orders arrive and conditions change throughout the shift, the manager re-runs the model with updated inputs to ensure recommendations always reflect current demand rather than a snapshot taken at the start of the day.

Our most immediate recommendation to improve model accuracy is for Wesco to obtain true customer due dates, task drop-in dates, and accurate priority codes across all departments. The model is currently running on proxy fields that approximate these inputs, and replacing them with true values will allow the model to classify and prioritize tasks based on actual customer urgency rather than internal substitutes. The effectiveness of the model's priority framework is dependent on Wesco instituting this change.

A behavioral change on the warehouse floor is required, though unlike the data recommendations above, this will be more difficult to resolve. Analysis of the data revealed that in the majority of cases, the assigned date and task completion date are within milliseconds of each other, indicating that workers are logging tasks into the system at the same moment they complete them rather than when the work actually begins. As a result, there is no visibility into how long tasks actually take workers to complete, meaning the task standards Wesco established and the model depends on cannot be verified. Our recommendation is to address this logging behavior, a training challenge that if resolved would improve data visibility for Wesco.

The natural next expansion of the model is worker-level intelligence. The current model operates at the head-count level and does not specify which workers should move across departments. Wesco's existing standard operating procedures (SOP), already associated with individual workers by user ID, provide a foundation for this expansion. Incorporating SOP eligibility into the model would allow reallocation recommendations to surface a list of qualified workers for each department, giving managers a shorter list to act on rather than the list of the full workforce.

In discussions with Wesco, we aligned on keeping the Day 1 and beyond output as a directional staffing estimate based on existing backlog rather than extending it into true demand forecasting. Predicting task demand that has not yet arrived is neither the scope of this work nor a direction Wesco intends to pursue. The optimization framework built here is designed to receive known demand as an input, meaning that as Wesco develops visibility into future workload, the model is ready to translate it directly into optimal staffing allocations.

This model gives Wesco managers a data-driven staffing allocation recommendation grounded in actual task demand, supporting the company's ability to deploy its workforce where it is needed most to fulfill customer orders. When labor is aligned with demand, orders ship on schedule and overtime is spent where it is actually needed. As data quality improves and the model matures, it provides a consistent and repeatable approach to labor planning across Wesco's more than 300 warehouse locations, replacing the varied and independent staffing decisions each site makes today.

More broadly, the labor-demand matching problem this model solves is not unique to Wesco. Any operation where task demand is dynamic, worker availability is variable, and misalignment results in delayed fulfillment and overtime costs can benefit from this framework.

References

- Aykin, T. (2000). A comparative evaluation of modeling approaches to the labor shift scheduling problem. *European Journal of Operational Research*, 125(2), 381–397.
- Ernst, A. T., Jiang, H., Krishnamoorthy, M., & Sier, D. (2004). Staff scheduling and rostering: A review of applications, methods and models. *European Journal of Operational Research*, 153(1), 3–27.
- Malik, R. (2013). *Decision support tool for dynamic workforce scheduling in manufacturing environments* (Master's thesis, Massachusetts Institute of Technology). MIT DSpace.
- Pinedo, M. L. (2022). *Scheduling: Theory, algorithms, and systems* (6th ed.). Springer.