Modeling and Optimization of Multi-Dolly Material Handling System in General Assembly Lines

by

Chaoye Pan

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Mechanical Engineering)
in The University of Michigan
2013

Doctoral Committee:

Professor Jun Ni, Chair
Professor Yavuz Bozer
Professor Jack Hu
Guoxian Xiao, General Motors Co.

To my Family

ACKNOWLEDGEMENTS

During the past 7 years of academic study, I have learned many things; some things I realized early while other things I have only recently come to understand. Among them, the most important things I have learned are persistence, humbleness and diligence. I would not have been able to make any of these advancements without the help of many individuals, to whom I am deeply indebted and with whom I am very grateful.

I would like to express my most sincere gratitude to my advisor, Professor Jun Ni, for his continuous support, guidance and encouragement. Without him, I would never have had the chance to begin, continue, or complete my research. I also appreciate the assistance provided by Dr. Guoxian Xiao, who has taught me how to link my research with real environment. My interaction with Dr. Xiao benefited me both professionally and personally.

I am grateful to Professor Yavuz Bozer for devoting his precious time to reviewing this dissertation and providing me with detailed and useful feedback. Finally, I want to thank Professor Jack Hu for agreeing to join my committee in lieu of the unexpected absence of my original committee member.

Finally, I dedicate this work to my parents, wife and parents in law for the support and encouragement that they have given me through the toughest time.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

1.1 Motivation and background

Material handling (MH) is an essential component of manufacturing. Even though material handling is generally viewed as a “non-value added” operation, if it cannot deliver the right parts at the right time, it often leads to significant production losses. It is estimated that the material handling typically accounts for over 20% cost in manufacturing systems (Askin & Standridge, 1993; Asef-Vaziri & Laporte, 2005).

The goal of this thesis is to support efficient real time MH decision making in automotive manufacturing systems. MH in our thesis deals with delivering parts for vehicle assembly. This kind of operation can also be called “part delivery”, “part feeding”, or “part replenishment”.

Traditional part delivery practice in automotive manufacturing has three emphases: 1) A planning tool (such as a simulation model) is used to either generate the delivery schedule beforehand or level the production sequence to balance the handling process based on statistical data (such as average part utilization, mean time between failures). 2) The basic dispatching rule is one dolly per trip with first-come-first-serve routing policy (FCFS), since only bulk parts are discussed in this research, one dolly here indicates one type of part. Multi-dolly trains currently exist, that deliver multiple dollies per trip; however their principal dispatching rule is based on a one dolly per trip policy. 3) Drivers are assigned to working zones with specific line-side buffers, who will be only respond to part replenishment requests within their working zones. This allows tracking of responsibilities, and reducing the errors in delivery process. Typically trial and error methods are used for zoning formation.

These three emphases from traditional part delivery system lead to three main problems: 1) Due to the usage of historical data, it is difficult to deal with dynamic uncertainties such as real time machine failure or starvation. 2) One dolly per trip and FCFS policy results in overstaffing, and causes high MH costs. 3) It would be difficult to assign drivers to form proper working zones, if the number of drivers is close to the minimum number to meet the requirements such as system throughput and average driver utilization.

Traditional material handling systems are somewhat like an open loop system without control strategies to deal with feedback or disturbance. To overcome these problems, we will use multi-dolly MH system to improve the part delivery process. Multi-dolly trains hold the potential to reduce MH costs by reducing the number of drivers and the number of delivery trips without sacrificing the production throughput. In order to utilize multi-dolly material handling systems properly, three basic questions need to be answered: 1) How to develop an efficient model, to estimate the flow of material in the assembly system? 2) How to dispatch the drivers to fulfill production requirements with minimum MH efforts? 3) How to properly form working zones with desired system performance? These three questions are far more difficult than they appear.

First, how can we develop an efficient model to estimate the flow of material. Let us briefly describe a production system in automotive manufacturing. A simple example of a production system is shown in Figure 1-1.

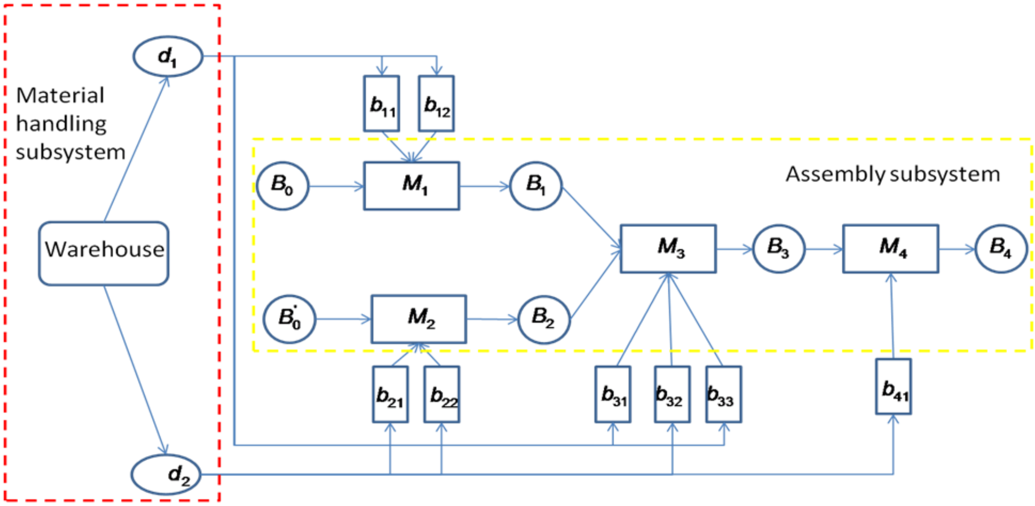

Figure 1-1: A simple example of a production system

The rectangles marked with M_i represent machines on which jobs are processed or assembled. The circles represent in-process buffers where work-in-process products and parts are held; rectangles marked with b_{ij} represent line-side buffers, each storing one type of part for assembly. The ellipses represent drivers who deliver parts from the central warehouse to the line-side buffers. Drivers are only responsible for the part requested within their working zones.

In a production system in automotive manufacturing, the machines and the in-process buffers form the assembly subsystem; and the central warehouse with the drivers form the material handling subsystem. These two subsystems are connected by the line-side buffers.

Accurate analysis of such systems is difficult due to the following characteristics:

1) Randomness and nonlinearity: Unreliable machines and finite buffers can cause uncertainties and nonlinearity.
2) Synchronous dependent machines: Failure of the conveyor or any machine within a section will lead to the failure of the whole section.
3) Coupling system dynamics: The whole system is strongly coupled by the buffers with finite capacities. Any perturbations at one section will propagate to upstream and downstream sections. Together with (2), any local event or state change has a global influence.
4) Asynchronism among sections: Different sections may have different cycle times which will lead to asynchronous behavior between them.

More generalized assembly system may include re-work operations, which will address the production quality issue. According to Driels and Klegka (1991), Re-work operations can be classified as terminal re-work (all rework activities are concentrated at the end of the production), or distributed re-work (some re-work activities will occur after each assembly process), which will further increase the system complexity by changing the production sequence and dynamics. Besides, quality issue is beyond our research scope. Therefore, we will not consider re-work operations in the main research, but regard re-work operations as an extension to currently investigated production systems in future work.

Since the production system does not satisfy assumptions of queuing networks for processing time or routing probabilities (more details will be discussed in literature review section 1.2), consequently, results from queuing network models cannot be used. Even more general models in queuing network cannot be applied directly to describe this system to obtain analytical solutions, since the time intervals between events or state change do not follow the exponential distribution. Furthermore, our sections are asynchronous with different cycle times, so they do not follow typical queuing network assumptions. With part delivery system, the overall system modeling may become more difficult.

Second, how do we allocate and dispatch drivers with minimum MH efforts. This requires the investigation of dispatching and routing policies that are suitable for our multi-dolly material handling (MH) systems. MH in an automotive production system is important, because inappropriate dispatching schedules of drivers may lead to delayed parts delivery, which means certain machines cannot receive an adequate supply of parts on time and can lead to starvation. The consequence of starvation is immense, the whole section will shut down due to the lack of timely parts supply, which costs a large amount of money and leads to great loss of productivity. The simplest methods to reduce or avoid starvation are to either have a large number of drivers, or a large line side buffers to hold many parts. While the disadvantages are obvious, the large number of drivers indicates very low utilization and efficiency. Limited space in a general assembly (GA) system cannot allow huge line side buffers. Thus, proper dispatching and routing policies for multi-dolly material handling systems becomes necessary.

For the purpose of online material handling dispatching, we will limit our research to a deterministic production sequence, while we assume that our machine failure is stochastic. A multi-dolly material handling system with deterministic production sequence implies that the daily log (sequence) of each machine in the manufacturing facility is known with a determined sequence of production. It is a practical assumption, since for mass production we need to know the actual production sequence beforehand, which is essential for supply chain management as well as production control. So if we can estimate the time information from general assembly system, we can then calculate the time for replenishment of line side buffers via the MH system. This provides us with the ability to optimize dispatching and routing policies.

From the discussion above, we know that the analysis of driver's dispatching & routing policies is essential in our MH system. To be more specific, we need to investigate how the number, type and order of parts carried for a specific trip will directly or indirectly influence the productivity and MH cost. However, because of system complexity, it is difficult for us to balance the MH efforts and GA productivity. Therefore, a systematic approach will be needed to evaluate and analyze the impact on the general assembly system by MH with different dispatching policies and routing rules. With this model, we will have a clear understanding of how to optimize the MH system in automotive manufacturing.

Third, the requests of drivers' zoning assignment usually come from production management consideration of assigning responsibility to specific drivers. In most cases, a driver is only responsible for a few pre-defined parts in order to track the delivery responsibility and reduce errors in handling process. It means that this driver will respond to certain part requests within his/her working zone and ignore others. Thus the problem becomes how to divide all parts into subsets (zones) and properly assign all these subsets to corresponding drivers to achieve minimum MH efforts. The challenges in solving the drivers' zoning assignment (DZA) problem come from two aspects. 1) The problem has its root as a complex combinatorial problem. We have to face huge time, state and event spaces for problem solving. Since a typical automotive general assembly system with material handling includes several hundred kinds of parts and about a dozen drivers, as a result, zoning problems have a large solution space (e.g., the number of alternative assignments in a typical system containing 300 parts and 12 drivers is about $\frac{12^{300}}{300!}$≈1.19×10^{315}) and the number of alternative assignments increases exponentially with the number of parts. 2) The complex system dynamics and the randomness in the production process make the problem even harder since the system throughput cannot be precisely and analytically expressed. Thus feasibility checking is essential, but the checking process would be time consuming due to the system complexity. Therefore, finding a systematic way of assigning grouped parts to corresponding drivers is important.

Surprisingly, little literature investigating the drivers' zoning assignment (DZA) problem has been written. Pan et al. (2008) used a meta-heuristic method Particle Swam Optimization (PSO) to solve this problem, however, the feasible solutions were highly dependent on the initial zoning formation. No methods other than a trial-and-error approach have been reported to address this problem in practice.

Overall, multi-dolly train systems provide an additional delivery option in a production system, which possibly leads to higher efficiency of drivers and lower MH costs. However, by adding this option, the resulting MH system becomes more complicated and various problems can arise. In our study, major problems are as follows:

1) Modeling of general assembly system with material handling

The internal blocking mechanism and asynchronous characteristics make it difficult to analytically model the general assembly system with MH. To model the general assembly system with MH in terms of timing information, accuracy and efficiency become challenging.

2) Dispatching and routing control of multi-dolly material handling systems

An optimal dynamic dolly building strategy, including dispatching and routing rules, for part delivery needs to be carefully designed to fulfill complicated flow of material requests in automotive manufacturing systems.

3) Drivers' zoning assignment (DZA) optimization

Due to a large solution space and a large number of alternative assignments, a systematic zoning optimization method is needed to properly assign subsets of grouped parts to corresponding drivers.

1.2 Literature Review

1.2.1 Discrete Event Dynamic System (DEDS) Modeling

An automotive GA system is a typical discrete event dynamic system (DEDS). The name “discrete event dynamic systems” is widely known to designate systems whose behavior can be completely characterized by the knowledge of starting and ending times of events.

DEDS covers flexible manufacturing systems, telecommunication networks, multiprocessor operating systems, railway networks, traffic control systems, logistic systems, intelligent transportation systems, computer networks, multi-level monitoring and control systems, and so on. Since its wide application, both industry and academic communities have become more and more interested in techniques to model, analyze and control DEDS (Baccelli et al., 1992).

Simulation is one of the most important approaches for DEDS, because of its flexibility, time compression, physical scaling and risk avoidance (Banks et al., 2005; Harding & Popplewell, 2000). The event scheduling scheme, also known as the next-event time advance approach (Banks et al., 2005; Law & Kelton, 2000) is a general simulation approach for DEDS (Cassandras & Lafortune, 1999). In this approach, a simulation clock and an event list are introduced. A timing routine is invoked to determine which event in the event list will occur next (Cassandras & Lafortune, 1999). Dozens of successful software packages have been developed based on this approach, such as Arena, AutoMod, ProModel, etc. However, this approach is difficult to be applied to our automotive production system with a large list of events, which leads to low efficiency and long search times. In addition, this approach reveals nothing about the underlying interactions and system behaviors.

Event relationship graph (ERG) is another simulation technique for DEDS. Schruben et al., (2000) presented a linear programming formulation for a single-server queueing system. This formulation has solutions representing the system trajectory of the single-server queue. Event relationship graph modeling has certain advantages in terms of simplicity and efficiency in simulation, and can solve simple DEDS problems. However, for very complicated DEDS with random events, such as random machine failures in a general assembly system, the linear programming formulation is very difficult to apply.

Parallel and distributed simulation (Bertsekas & Tsitsiklis, 1997; Das, 2000; Fujimoto, 2001) is also a relevant approach for simulating manufacturing systems. An obvious benefit of parallel simulation is that computational times can be reduced by dividing a large simulation task into many sub-tasks that can be executed concurrently (Das, 2000). However, dividing a task is challenging, which is only suitable for loosely coupled systems with weak interactions and is not directly applicable to closely coupled general assembly and material handling systems.

For the analytical modeling, exact analytical expression only exists for the two-machine one-buffer system, or the system with infinite buffer capacity, or without buffers (Dallery & Gershwin, 1992; Li at el., 2006). However, an automotive general assembly system with material handling is considerably larger and more complicated. Furthermore, time intervals for each car entering the system, time of machine state change, as well as time of each material delivery trip do not follow exponential distributions, which is the key assumption in analytical queueing networks, Markov chains and Semi Markov chain models (Yao, 1994; Li at el., 2006). Therefore these approaches cannot be directly applied to the coupled production system problems.

For approximation methods, such as aggregation and decomposition methods, accuracy becomes a big concern when the scale of the system is expanded. Considering complicated general assembly systems with both serial lines and assembly lines (Gershwin, 1999; Bihan & Dallery, 2000), neither of these two approaches can directly provide real-time timestamp information for material handling dispatching. Buzacott and Shantikumar (1993), and Altiok (1997) investigated decomposition based on non-exponential machine systems, however, such approaches are typically computationally intensive, and are very difficult to extend to practical systems for real time multi-dolly material handling dispatching.

Modeling and analysis of manufacturing systems have attracted significant research attention during the last 50 years, resulting in substantial effort being devoted to performance evaluation, continuous improvement, customer demand satisfaction, etc., for general assembly systems and material handling systems (Bozer & Yen, 1996; Johnson & Brandeau, 1996; Zhao et al., 2010; Yan et al., 2010; Govind et al., 2011). In most of these studies, general assembly systems and material handling systems are analyzed independently. The study of highly coupled assembly operations and material handling is almost neglected in the literature. Some work, intended to address this issue has been done (e.g., Yan et al., (2010) in automotive assembly, Govind et al., (2011) in semiconductor manufacturing), however, the focuses of these researches are mainly on system design and planning; thus the methods are not readily applicable for the purpose of real-time multi-dolly material handling dispatching.

In this thesis, a modified max-plus algebra modeling approach is utilized for DEDS with coupled general assembly and material handling systems. A substantial amount of research has been devoted to the field of applying max-plus algebra in queueing and traffic networks (Alexopoulos et al., 2007; Becker & Lastovetsky, 2010).

The first uses of max-plus algebraic system theory in the modeling and analysis of DEDS can be dated back to the early 1960s (Giffler, 1960; Cunninghame-Green, 1961). The idea is that the system dynamics are represented as a set of linear recursions consisting of two types of algebraic operators, namely {max} for maximization and {+} for addition. Solving the recursions provides information about the system performance. An account of the pioneering work on max-plus algebraic system theory for DEDS was given in (Cunninghame-Green, 1979). Related work had been done by Gondran (1984). In the late eighties, the early topic attracted new interests. Cohen et al. (1985) considered a certain class of decision-free systems, where all places had only one output transition and one input transition. They used the state equations to develop system transfer-matrix representations for stability and observability, etc. Related work also included (Cohen et al., 1989).

For recent work, Krivulin (1995) addressed the issue of system blocking and provided the scaling equation with manufacturing blocking and communication blocking. Gaubert (1995) introduced (max +) automata as non-machinery autonomous max-plus linear system with finitely valued dynamics, i.e., $\mathrm{x(k) = A \otimes x(k-1)}$, where A takes its values in a finite set {A1,…,An}, which gives us the inspiration of using switching mechanism to break the synchronization of traditional max-plus linear algebra by replacing matrices under different perturbations, such as random failure or starvation.

From previous work, we can see that there exists a remarkable similarity between the basic operations of the max-plus algebra (maximization and addition) and the basic operations of conventional algebra (addition and multiplication). As a consequence, many concepts and properties of conventional algebra have a max-plus analogy. This analogy allows us to translate many concepts, properties and techniques from a conventional linear system to a max-plus linear system. Besides, max-plus algebra has many advantages: (1) it yields timing equations directly from the system configuration and hence there is no need to first derive a Petri net or a digraph equivalent of the system; and (2) a change in the system configuration only affects the interconnection matrices and hence does not require deriving the entire set of equations.

However, the traditional max-plus linear system modeling method cannot be applied to systems with uncertainties and nonlinearity. Therefore, a modified max-plus modeling approach is necessary to model systems in a flexible fashion to directly handle non-deterministic events and scalable system configuration. The goal of our research is intended to contribute to this end.

1.2.2 Dispatching and Routing Control of Multi-Dolly Material Handling Systems

The multi-dolly material handling system in this thesis, to some degree, resembles the Milk Run (MR) system in lean manufacturing, which is used to deliver parts to multiple machines (Bozer & Ciemnoczolowski, 2013; Ciemnoczolowski & Bozer, 2013). However, since route and frequency are fixed in MR systems, dynamic disturbances (e.g., random failure) may not be reflected promptly. Besides, MR systems do not perform well outside of lean environments. Different from MR systems, in which dispatching policy is trivial, the dolly building problem in our thesis focuses on the dispatching and routing policy that determines the number and types of parts to deliver and the delivery routes at the beginning of the trip.

Traditional part delivery systems in an automotive assembly plant are treated as a static system. Under the assumption that hourly part consumption amounts are forecasted based on the information of the daily production sequence every morning, drivers feed parts according to the fixed hourly part-delivery plan. Traditional approaches for handling the part delivery problem focus on how to optimize the production sequence by leveling consumption rates of parts. Monden (1983) described nicely the operation of a parts-delivery system for an automotive assembly system, by maintaining a smooth production sequence in terms of production volume and model mix. Okamura and Yamashani (1979) proposed a heuristic method for determining a production sequence by minimizing the probability of conveyor line stoppage. This well-known method was developed and applied by Toyota Motor Company for leveling the consumption rates of parts (Monden, 1998) and several variants of it have been presented since then. Inman and Bulfin (1991) presented a leveling algorithm with a polynomial complexity in time. Sumicharast and Clayton (1996) surveyed the sequencing procedures and compared them based on their ability to achieve desired production targets.

However, even if automotive manufacturers try to optimize the production sequence, part consumption rates cannot be nearly constant, as investigated by (Inman et al., 1997). Automotive manufacturing system has thousands of parts, translating into millions of possible configurations. The mixed model sequencing problem becomes highly constrained because leveling one part's consumption will worsen others. Hence, when the number of parts required by an assembly system is very large, and there are many other parts with low consumption due to the intermittent usage, it is difficult to achieve the leveling of part consumption rates all the time.

Therefore, we treat the part delivery problem from a different perspective, i.e., how to dispatch and route the drivers rather than how to level the production sequence. The most related work is automated guided vehicle (AGV) dispatching (Vis, 2006). Qiu and Hsu (2002) provided a comprehensive survey for existing dispatching and routing algorithms for AGV. Among them, a multi-load AGV system most resembles ours. Hirao and Tamaki (2002) addressed dispatching policies for a single, multi-load AGV system, where all loads originate from a single machine. However, they focused on unit-load AGVs, where multiple types of loads were viewed as a unit-load, which moved only one unit-load at a time.

For all the AGV dispatching policies, hybrid push and pull strategies draw our special attention. According to Qiu and Hsu (2000), its ability to forecast and penalty prediction would be very promising for our part feeding problem.

Hodgson and Wang (1991) studied the hybrid push and pull (pp) control strategies for a parallel multistage system. The results showed that the hybrid policy required lower total inventory and achieved higher supply reliability. Yim and Linn (1993) developed an efficient petri-net based simulation model to analyze a flexible manufacturing system (FMS) with push and pull dispatching rules. Huang and Kusiak (1998) addressed a push-pull approach which could reduce in-process inventory and shorten lead times. Their approach was verified in an industrial case study. Inspired by corresponding AGV dispatching, we propose several dispatching policies. With the knowledge of request time information of part replenishment generated from the modified max-plus modeling, it is possible to forecast MH dynamics and reduce MH related cost.

When the requests and number of parts are sent to drivers dynamically, we need to form routes for drivers, which will lead to our second question in dolly building: In what sequence will these parts be delivered? First-come-first-serve (FCFS) is a commonly used routing sequence strategy (Pinedo, 2002). However, it is far from the best solution. To further analyze the problem, we show that, under some specific conditions, there exists a transformation schema for a one-to-one mapping between our routing problem and vehicle routing problem (VRP), which allows us to take advantage of standard approaches from the VRP. The VRP area has been intensively studied in the literature. We refer to (Magnanti, 1981; Solomon & Desrosiers, 1988) for a comprehensive survey of the VRP.

In order to integrate both dispatching and routing processes to support real time MH systems, the question is how to quantify the performance impact from an MH system on a GA system. First, we need a standard to evaluate the performance. Since it is possible to convert all the MH related manufacturing activities into cost, overall cost would be a reasonable parameter to evaluate the performance.

Commonly speaking, MH system cost involves many aspects, which is viewed from an activity based costing perspective (Brimson , 2002) including:

Labor cost:

Wages of all operators that are assigned tasks, fringe benefits and possibly overtime premiums.

Activity cost:

1. Transportation costs
2. Loading and unloading activity costs
3. Set-up activity costs
4. Work in process storing activity costs
5. Material handling system monitoring and control activity costs
6. Production stoppage costs

7. Inventory holding costs
8. Waiting and idle time costs

For simplicity, we will only focus our attention on three areas: labor cost, material handling transportation cost, and production loss due to starvation. Therefore, system performance can be quantitatively presented. Impacts on general assembly system can be compared and analyzed in a systematic way in order to achieve our goal of supporting a MH system with minimum cost.

Again, most of the related research of material handling dynamics has addressed AGV systems. Egbelu (1987) calculated the minimum MH cost in a manufacturing system based on the loaded traveling times and empirical estimates of the unloaded ones. Tanchoco et al. (1987) employed a queuing theory-based computer model for material handling dynamics. Wysk et al. (1987) used a spreadsheet analysis to address the same problem. Their approaches provided initial estimates for the number of AGVs, which might be further refined by simulation. For the problem of determining the AGV fleet size, Sinriech and Tanchoco (1992) developed a multi-criteria optimization model exploiting the trade-off between investment costs and system throughput, and proposed the use of decision tables relating to the investment cost, the number of AGVs and their utilization, as well as the corresponding “conflicting” costs. Maxwell and Muckstadt (1982) studied the design of an efficient horizontal unit-load MH system when the production rate of each manufacturing resource was constant and known. Bozer and Srinvasan (1991) presented a partitioning algorithm for the design of single vehicle loops, in an effort to distribute the workload evenly among the AGVs in the MH system. Tompkins et al. (1996) and Loannou (2007) summarized relevant literature in the AGV system vehicle management area, as well as the integration of the MH system design and facility layout design problems. Finally, Loannou (2007) proposed an integrated model for concurrent layout and MH system design, in which the MH cost was calculated.

1.2.3 Drivers’ Zoning Assignment (DZA) Optimization

Meta-heuristic based aggregation zoning is an important issue in many traditional applications, such as political districting, plant location, health care zoning, and travel demand forecasting. It is similar to our drivers’ zoning assignment (DZA) problem. Different aggregation zoning algorithms have been proposed in the literature, with emphasis on the optimization of search algorithms, such as Tabu Search, Simulated Annealing (SA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO) and others (Kirkpatrick et al., 1983; Glover, 1989; Dueck & Scheuer, 1990; Gorilli et al., 1999).

Particle Swarm Optimization (PSO) algorithm is an adaptive algorithm. A population of individuals adapts by returning stochastically toward previously successful regions in the search space, and is influenced by the successes of their topological neighbors. Based on the results of a variety of prior research, it has been possible to efficiently find a global optimum in a simultaneous poly-modal function with high precision (Kennedy & Eberhart, 2001).

Ant Colony Optimization (ACO) is another meta-heuristic for solving hard combinatorial optimization problems. The first ACO algorithm, ant system (AS), was proposed by Colorni & Maniezzo (1991) as a means of solving the travelling salesman problem (TSP). Based upon observations of real ant colonies, it was found that over a fixed time period, the shorter paths between the nest and the food source were likely to be travelled more often than the longer paths.

Simulated Annealing (SA) is a stochastic local search method analogous to physical annealing, the process of melting and then slowly cooling a solid so that the substance reaches its lowest energy state. By accounting the likelihood of a particular molecular configuration at a given temperature, Langevin et al. (1993) introduced simulated annealing, a probabilistic search procedure for solving combinatorial optimization problems. Morse (1997) extended simulated annealing with an ad hoc introduction of a variable penalty multiplier.

Some comparison results among different optimization search algorithms were demonstrated in (Ricca & Simeone, 2000; Pan et al., 2008). Among them, PSO shows its advantages of simplicity (easy to implement and few parameters to adjust), fast convergence with lower dimension and high accuracy. However, Pan et al. (2008) addressed the convergence issue of PSO for our DZA problem, which was highly dependent on its initial zoning assignment. This means a proper initial zoning assignment is essential for the success of PSO based search to insure that a global minimum instead of a local minimum is found in the DZA problem.

To serve this end, three classes of problems related to our initial zoning assignment research are reviewed. The first class of problems is the assignment problem (AP) (Pentico, 2007; Burkard et al., 2009). In the AP problem, the cost of assigning a task to a worker is additive. But the system throughput, which depends on the whole assignment of line-side buffers to drivers in the DZA problem, is not additive. We cannot apply AP algorithms to solve the DZA problem because the additivity is necessary in AP algorithms.

The second class is the Bin-Packing Problem (BPP), which is NP-complete (Johnson et al., 1974). Although the DZA problem seems similar to the BPP, BPP algorithms are not suitable to solve the DZA problem. The reason is that in BPP, the size of the bins and the space of room storage are known, while in the DZA problem the bin sizes cannot be obtained due to complexity on throughput and driver utilization.

The third class of problems, the Parallel Machine Scheduling (PMS) problem (Mokotoff, 2001), which is in a sense regarded as the dual problem to the BPP (Coffman et al., 1978), can give us some insights into the methods to solve our problem. The PMS problem, where the makespan is minimized, is also NP-complete (Garey et al., 1979). PMS problems may be modified in order to establish our solutions for the initial zoning, which may not be related to the PMS problem at first glance. Though the main difficulty of PMS is its calculation complexity and low convergence efficiency (Yan et al., 2010), it would be a great choice to form the initial zoning assignment to serve for the PSO based zoning optimization.

1.3 Research Objectives

After a review of previous work on all the challenges related to our research, the following tasks are proposed:

1) To model a real time production system efficiently, the traditional max-plus linear system will be modified and extended to obtain an analytical model of DEDS. Based on the proposed method, the timing information of coupled general assembly and MH systems will be analyzed to forecast MH behaviors, and reduce MH time & costs.
2) To improve the real time multi-dolly MH process, dispatching policies based on forecasting and penalty prediction will be implemented. The mapping procedure between our MH system and vehicle routing problem with time window (VRPTW) shall be investigated in order to integrate proper routing strategy to achieve minimum MH cost.
3) To form a proper initial zoning, the fixed zoning version of the DZA problem will be formulated, which can effectively assign line-side buffers to drivers. Then a meta-heuristic algorithm PSO will be implemented to solve the DZA optimization problem to achieve minimum MH cost.

1.4 Outline of the Dissertation

In Chapter 2, the modeling of a general assembly system using max-plus algebra is investigated. Basic concepts of max-plus algebra are introduced, a model of 2 machines with finite buffers model is studied and extended to a multi-machine and multi-buffer system. Using max-plus algebra, a switching mechanism is proposed for the GA system with material handling. Numerical experiments and validation of the method are provided.

In Chapter 3, a framework for online multi-dolly material handling system is introduced. Existing dispatching policies are discussed. Several policies based on forecasting and penalty predictions are proposed. A mapping procedure between our multi-dolly MH system and a vehicle routing problem with time window (VRPTW) is introduced, and an integrated model considering both dispatching and routing is proposed. Numerical results demonstrate the effectiveness and robustness of our approach.

In Chapter 4, the zoning optimization problem is introduced and formulated. We will discuss the similarity between PMS and our DZA, and adopt an existing PMS algorithm, as well as a backtracking method to determine the proper initial zoning configuration. After a proper initial zoning is formed, PSO is investigated and implemented for our zoning assignment problem. A numerical example on a practical scale will be shown for validation and demonstration.

Finally, the contributions and recommendations for future work of the doctoral research are summarized in Chapter 5.

CHAPTER 2
MODELING OF GENERAL ASSEMBLY SYSTEM WITH MATERIAL HANDLING USING MAX-PLUS ALGEBRA

2.1 Introduction

A typical general assembly (GA) system in a high volume automotive plant has hundreds of machines. An analytical description of the entire system is difficult to obtain, because of the internal blocking mechanism, asynchronous nature of the general assembly system, and the coupling relationship with MH system. Besides, dolly trains are gradually used to replace forklifts, which enable multiple dollies to be delivered per trip, which potentially reduces the number of drivers without sacrificing the throughput; however it will further increase our difficulty in analysis.

Therefore, we narrow our research scope on material handling system with the assumptions that production sequence and part demand are known, so when we estimate the timing information dynamically from the general assembly system model, detailed timing information for MH system will be available. Our main focus in this chapter will be to accurately and efficiently obtain timing information from the production system.

Previous literature reviews indicate max-plus algebra might be a good choice to model complex discrete event dynamic systems (DEDS), such as general assembly systems in our research. However, two difficulties need to be addressed, which are finite buffers and random failures.

In this chapter, we will first introduce the model of the general assembly system. Secondly, we will extend the max-plus linear system to model the integrated general assembly system coupled with the material handling. A procedure of generating part request list is developed, and a numerical example is used to validate our modeling approach.

2.2 Modeling Of General Assembly System

2.2.1 System Description

In this section, we will first introduce the system and our assumptions. The detailed modeling of two subsystems (i.e., general assembly and material handling subsystems) is described in Fig. 1-1 in section 1.1. Before we present the detailed system description, the nomenclatures used in this chapter are introduced:

$M(m,s)$	the machine index, m stands for the number of machine, s represents which section it is in
$B(s)$	the downstream buffer capacity in section s
$\hat{S}_i(M(m,s))$	the perfect starting time for car i on M(m,s) (processing without any stops) (min)
$S_i(M(m,s))$	the real starting time for processing car i on M(m,s) (min)
$F_i(M(m,s))$	the finishing time for car i on M(m,s) (min)
$D_i(M(m,s))$	the departure time for car i on M(m,s) (min)
$P_i(M(m,s))$	the whole processing time for car i on M(m,s) (min)

$CT(s)$	cycle time on section s (min)
$FA_j(s)$	the time of failure j on section s (min)
$RE_j(s)$	the time of repair j on section s (min)
$ST_j(s)$	the time of parts starvation on section s (min)
$RP_j(s)$	the time of parts replenishment on section s (min)
$INV(p)$	current inventory level of part p
$RON(p)$	re-order number of part p
$AVE(p)$	average utilization of part p
$ROQ(p)$	replenishment quantity of part p
$OUT_i(j)$	time instance of driver i starting j-th trip (min)
$BACK_i(j)$	time instance of driver i coming back from j-th trip (min)

A typical automobile general assembly (GA) system consists of many sections and there are buffers with finite capacities between sections. Each section consists of a sequence of assembly machines with one single conveyor. The conveyor transfers cars from machine to machine where operators/robots finish assembly tasks during a required period. GA in a high volume plant has thousands of different parts, which need to be assembled onto a car. Drivers deliver these parts from the docking area to the corresponding line-side buffers. When a part has been consumed, the remaining number in the line-side buffers will be updated. A signal, requesting replenishment of the part will be sent to the material dispatching center when the part quantity goes down below a certain level.

The entire production system can be divided into two sub-systems: 1) general assembly subsystem, and 2) material handling subsystem. Although these two sub-systems have quite different logic and dynamic characteristics, they are coupled in terms of part requests. The assumptions of our general assembly system are as follows:

1) A machine can be blocked if it is up and its downstream buffer is full. The last machine is never blocked.

2) A machine can be either starved by upstream buffer or by line-side buffer parts. A machine is starved by the upstream buffer if it is up and its upstream in-process buffer is empty. It is starved by parts if it is up, the upstream in-process buffer is not empty, but one of the line-side buffers is empty. The first machine is never starved by an upstream buffer.

3) When failure or starvation happens, all other machines in the same section will stop and wait until the problem has been eliminated.

4) Mean time between failure (MTBF) and mean time to repair (MTTR) can be obtained from historic data.

5) There exist six basic states for machines:

- Blocked state – the downstream buffer is full, conveyor on that section cannot move.
- Starving by part – a certain part on a specific machine is below the minimum number to finish one assembly process on the machine.
- Starving by car – the upstream buffer is empty, after one cycle time, when all machines in this section turn to idle, the first one changes into car starving state automatically.
- Failed state – machine breaks down.

- Processing state – machine is working.

We want to evaluate the following performance measures in our model. These performance measures are very important to identify the system performance level and system dynamic behavior. Therefore, they can be used to validate model accuracy when comparing with real production data, which will be demonstrated in detail in Section *2.3*.

- *Throughput*. The average number of vehicles produced by the general assembly system per time unit in the steady-state of system operations.
- *Utilization of driver*. The average ratio of working time over the total time for driver.

As we mentioned in the introduction, the main goal of this research is to model the coupled general assembly and material handling systems using part requests as the linkage. A modified max-plus model is utilized for this purpose.

2.2.2 Basic Operations of the Max-Plus Algebra

The basic operations of the max-plus algebra are maximization and addition, which will be represented by $\oplus$ and $\otimes$ respectively (Krivulin, 1995):

$$x \oplus y = \max(x, y),\ x \otimes y = x + y,\ \text{ for any } x, y \in \mathbb{R}.$$

It is easy to see that these new operations have the following properties:

Associativity: $x \oplus (y \oplus z) = (x \oplus y) \oplus z$
 $x \otimes (y \otimes z) = (x \otimes y) \otimes z$

Commutativity: $x \oplus y = y \oplus x$, $x \otimes y = y \otimes x$

Distributivity: $x \otimes (y \oplus z) = (x \otimes y) \oplus (x \otimes z)$

Idempotency of Addition: $x \oplus x = x$

With $e = 0$, $\varepsilon = -\infty$, we further have

Null element: $x \oplus \varepsilon = \varepsilon \oplus x = x$

Identity element: $x \otimes e = e \otimes x = x$

Absorption Rule: $x \otimes \varepsilon = \varepsilon \otimes x = \varepsilon$

Clearly, in the max-plus algebra, these properties allow ordinary algebraic manipulation of expressions to be performed under the usual conventions regarding brackets and precedence. The plus operator $\otimes$ takes precedent over the max operator $\oplus$.

The scalar max-plus algebra can be extended to the max-plus algebra of matrices in a similar way. Specifically, for any square (n x n) matrices $A = (a_{ij})$ and $B = (b_{ij})$, the elements of the matrices $C = A \oplus B$ and $D = A \otimes B$ are calculated as:

$$c_{ij} = a_{ij} \oplus b_{ij} \text{ and } d_{ij} = \sum_{k=1}^{n} a_{ik} \otimes b_{kj} = \max_{k=1 \ldots n} (a_{ik} \otimes b_{kj})$$

Where $\sum\otimes$ denotes the iterated operation $\otimes$. Similarly the multiplication of a matrix by a scalar, as well as the operations of both matrix-vector multiplication and vector addition are observed. As in the scalar max-plus algebra, there are null and unit elements in the matrix algebra, defined respectively as:

$$\varepsilon = \begin{pmatrix} \varepsilon & \cdots & \varepsilon \\ \vdots & \ddots & \vdots \\ \varepsilon & \cdots & \varepsilon \end{pmatrix}, \qquad E = \begin{pmatrix} e & \varepsilon & \ldots & \varepsilon \\ \varepsilon & e & \varepsilon\ldots & \varepsilon \\ \ldots & \ldots & \ldots & \ldots \\ \varepsilon & \varepsilon\ldots & \varepsilon & e \end{pmatrix}$$

2.2.3 Max-Plus Linear State Space Models

DEDS with only synchronization and no concurrency can be modeled by a max-plus algebraic model of the following form (Krivulin, 1995):

$$x(k) = A \otimes x(k-1) \oplus B \otimes u(k)$$

$$y(k) = C \otimes x(k) \qquad (2.1)$$

With $A \in \mathbb{R}_{\varepsilon}^{n\times n}$, $B \in \mathbb{R}_{\varepsilon}^{n\times m}$, $C \in \mathbb{R}_{\varepsilon}^{l\times n}$, where m is the number of inputs, n is the number of variables, and l the number of outputs. The vector x represents the state, u is the input vector, and y is the output vector of the system. It is important to note in (2.1) the components of input, output, and state are event times, and the counter k in (2.1) is an event counter. Due to the analogy with conventional linear time-invariant systems, a DEDS can be modeled by (2.1) and will be called a max-plus linear time-invariant DEDS. Typical examples of such systems that can be modeled as max-plus linear DEDS are general assembly systems, and queuing systems. Now we give an example of how the behavior of a simple manufacturing system can be described by a max-plus linear model.

2.2.3.1 General assembly system with finite buffers

Suppose that the buffers in a general assembly system have limited capacity. Consequently, machines may be blocked due to full downstream buffer and starved because of empty upstream buffer. Consider a flow shop production system with n machines, and assume the buffer at the i-th machine, i=2,.., n, to have capacity B_i . If a blockage condition occurs after the completion of an assembly service, the i-th machine sees the buffer of the (i+1)st machine as full, resulting in the i-th machine not being able to be freed and therefore remains busy until the (i+1)st machine completes its current service and there is free space in its buffer. Under the condition of starvation, the buffer of the (i+1)st machine is empty, the (i+1)st machine will be idle until the i-th machine finishes its work. Obviously, for the last machine, the products leave the system right away upon their service completion, which cannot be blocked. In this thesis, we restrict our consideration to these types of manufacturing blockages and starvations, which are most commonly encountered in practice.

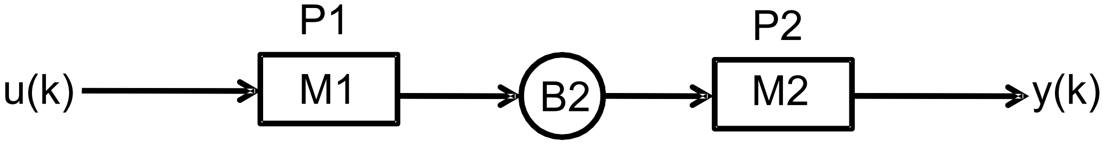

Figure 2-1: 2 machines with finite buffer capacity

A processing unit can only start working on a new product if it has finished processing the previous one. We assume that each processing unit starts working as soon as all parts are available. We define:

$u(k)$: time instance at which k-th raw product is fed to the system.

$x_i(k)$:time instance at which i-th machine starts working on the k-th product.

$y(k)$: time instance at which k-th product leaves the system.

Let us assume that transportation time *t1*=*t2*=*t3*=0 to simplify the model without losing the essence of the example. Let T_1be the time at which the k-th part arrives at M_1, $T_1 = u(k) + t_1 = u(k)$

Let T_2 be the time at which the (k-1)th part leaves M_1, which will happen until the (k-1)th part is finished and there is at least one empty space in buffer B_2.

$$T_2 = \max(x_1(k-1)+P1, x_2(k-B2-1))$$

Therefore,

$$x_1(k) = \max(T_1, T_2) = \max(u(k), x_1(k-1)+P1, x_2(k-B2-1))$$

Similarly we can derive:

$$\begin{aligned} x_2(k) &= \max(x_2(k-1)+P2, x_1(k)+P1) \\ &= \max(x_2(k-1)+P2, x_1(k-1)+2P1, u(k)+P1, x_2(k-B2-1)+P1) \end{aligned}$$

$$y(k) = x_2(k)+P2$$

Rewrite the equations in max-plus algebraic matrix notation, we obtain:

$$X(k) = \begin{bmatrix} P1 & \varepsilon \\ 2P1 & P2 \end{bmatrix} \otimes X(k-1) \oplus \begin{bmatrix} \varepsilon & e \\ \varepsilon & P1 \end{bmatrix} \otimes X(k-B2-1) \oplus \begin{bmatrix} e \\ P1 \end{bmatrix} \otimes u(k)$$

$$y(k) = \begin{bmatrix} \varepsilon & P2 \end{bmatrix} \otimes X(k) \tag{2.2}$$

Where $X(k) = \begin{bmatrix} x_1(k) & x_2(k) \end{bmatrix}^T$, and $X(k-B2-1) = \varepsilon$ if system is at its warm-up stage, $k - B2 - 1 \leq 0$

Note that the difference between (2.1) and (2.2). Because of the starvation and blockage due to the finite buffer an additional term must be added:

$$A^1 \otimes X(k-B2-1)$$

1) Assume B2=0, which means there is no buffer between the two machines, therefore, we have:

$$X(k)=\begin{bmatrix} P1 & \varepsilon \\ 2P1 & P2 \end{bmatrix}\otimes X(k-1)\oplus\begin{bmatrix} \varepsilon & e \\ \varepsilon & P1 \end{bmatrix}\otimes X(k-1)\oplus\begin{bmatrix} e \\ P1 \end{bmatrix}\otimes u(k)$$

$$=(\begin{bmatrix} P1 & \varepsilon \\ 2P1 & P2 \end{bmatrix}\oplus\begin{bmatrix} \varepsilon & e \\ \varepsilon & P1 \end{bmatrix})\otimes X(k-1)\oplus\begin{bmatrix} e \\ P1 \end{bmatrix}\otimes u(k)$$

$$=\begin{bmatrix} P1 & e \\ 2P1 & P1\oplus P2 \end{bmatrix}\otimes X(k-1)\oplus\begin{bmatrix} e \\ P1 \end{bmatrix}\otimes u(k)$$

$$\Leftrightarrow X(k)=(A\oplus A^1)\otimes X(k-1)\oplus B\otimes u(k)$$

2) Assume *B2*=∞, $X(k-B2-1)=\varepsilon$, in any condition, $A^1\otimes X(k-B2-1)$ term can be ignored.

3) Assume B2≥1, we can divide the model into two parts:

When k≤B2+1, $X(k)=A\otimes X(k-1)\oplus B\otimes u(k)$

When k> B2+1, $X(k)=A\otimes X(k-1)\oplus A^1\otimes X(k-B2-1)\oplus B\otimes u(k)$

Using recursion, we can calculate all the time instances for each machine i starting operation on the k-th raw product. In addition we can also calculate the time when the k-th raw product leaves the system.

2.2.3.2 Extension to 'n' machine system

Traditionally, a flow shop system contains a single line of machines in which all jobs share the same processing order on these machines, each job visits all the machines, and each job may visit each machine only once.

In the previous example, we showed how to model a two machines system with infinite buffer and limited buffer. It is not difficult to understand that the dynamics can be generalized by the ordinary scalar equations.

1) For infinity buffer capacity and 'n' buffers:

$$x_1(k) = \max(x_1(k-1) + P1, u(k))$$

$$x_i(k) = \max(x_i(k-1) + P_i, x_{i-1}(k) + P_{i-1})$$

$$X(k) = A \otimes X(k-1) \oplus B \otimes u(k)$$

$$y(k) = C \otimes x(k)$$

$$A = (a_{ij}) = \begin{cases} P_i & i = i \\ a_{i-1,j} + a_{i-1,i-1} = \sum_{k=j}^{i} P_k + P_j, & i \geq j \\ \varepsilon & i \end{cases}$$

$$B = (b_{ij}) = \begin{cases} e & i = 1 \\ \sum_{k=1}^{i} P_i & i = 2, \ldots, n \end{cases}$$

$$C = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon & \cdots & \varepsilon & P_n \end{bmatrix} \tag{2.3}$$

2) For limited buffer capacity and ‘n’ buffers:

$$x_1(k) = \max(x_1(k-1) + P_1, u(k), x_2(k - B2 - 1))$$

$$x_i(k) = \max(x_i(k-1) + P_i, x_{i-1}(k) + P_{i-1}, x_{i+1}(k - B_{i+1} - 1))$$

$$x_n(k) = \max(x_n(k-1) + P_n, x_{i-1}(k) + P_{n-1})$$

$$X(k) = A \otimes X(k-1) \oplus A^1 \otimes X(k - B2 - 1) \oplus \cdots \oplus A^{n-1} \otimes X(k - B_n - 1) \oplus B \otimes u(k)$$

$$y(k) = C \otimes X(k) \tag{2.4}$$

Where A, B, C are the same as the infinity buffer capacity model:

$$A^l = (a_{ij}) = \begin{cases} e & j = l+1, i = l \\ \sum_{k=1}^{i-1} P_k, & j = l+1, i > l \\ P_i & otherwise \end{cases}$$

For $B_i = 0, X(k) = (A \oplus A^1 \oplus \cdots \oplus A^{n-1}) \otimes X(k-1) \oplus B \otimes u(k)$

2.2.3 Switching Max-Plus Linear System

Dynamic property of DEDS determines the system possibility of "switching" in different "modes". A mode refers to a set of required synchronization processes and scheduled event orders in this section. Let "switching" variable $z(k)$ represent a general assembly system mode at event step k, where k is the event counter. The max-plus linear state space model of the system can be represented as

$$X(k) = A_{z(k)} \otimes X(k-1) \oplus B_{z(k)} \otimes u(k) \tag{2.5}$$

Therefore, a "switching" max-plus system modeling may be developed. A "switching" can be seen as a mechanism where the system modes can be changed due to certain scenarios such as breaking of synchronization or changing of event orders. The "switching" variable $z(k)$ is defined as $z(k) = \phi(x(k-1), l(k-1), u(k), v(k))$. It is a function of the previous state $x(k-1)$, the previous mode $l(k-1)$, the input variable $u(k)$ and control variable $v(k)$. The control variable $v(k)$ corresponds to a specific system property and behavior, i.e., processing sequence, event order, or scheduled maintenance.

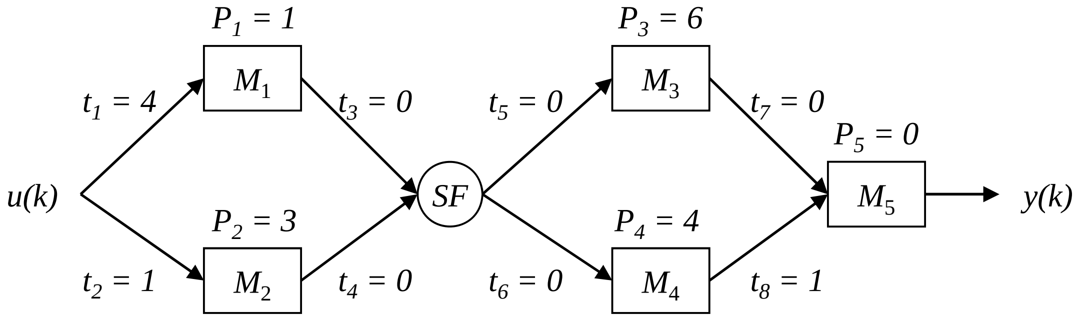

Figure 2-2: Assembly line with concurrency operation

Figure 2-2 demonstrates an example with five machines M_1, M_2, M_3, M_4 and M_5. Raw products arrive at time $u(k)$ and then are fed to M_1 and M_2. The intermediate products need additional processing at M_3 and M_4, and finally meet at M_5 and leave the system. In our example, "switch" operation SF is added in the system. SF determines where the intermediate products will go to after they come out of M_1 or M_2. It is assumed that the system logic is always to feed the slower machine between M_3 and M_4 first, which is M_3 in this case. It can be derived that the system equations for x_1 and x_2 are:

$$x_1(k) = \max(x_1(k-1) + P_1, u(k) + t_1)$$
$$x_2(k) = \max(x_2(k-1) + P_2, u(k) + t_2)$$

If $x_1(k-1) + P_1 \leq x_2(k-1) + P_2$ (i.e., M_1 finishes first), then the intermediate product of M_1 will be directed to M_3 and intermediate product of M_2 will be directed to M_4, since M_3 is slower than M_4. Similarly, we obtain:

$$x_3(k) = \max(x_1(k) + P_1, x_3(k-1) + P_3) = \max(x_1(k-1) + 2P_1, x_3(k-1) + P_3, u(k) + P_1 + t_1)$$
$$x_4(k) = \max(x_2(k) + P_2, x_4(k-1) + P_4) = \max(x_2(k-1) + 2P_2, x_4(k-1) + P_4, u(k) + P_2 + t_2)$$
$$x_5(k) = \max(x_3(k) + P_3, x_4(k) + P_4 + t_8) = \max(x_1(k-1) + 2P_1 + P_3,\ x_2(k-1) + 2P_2 + P_4 + t_{8,}\ x_3(k-1) + 2P_3, x_4(k-1) + 2P_4 + t_8, u(k) + P_1 + t_1 + P_3,\ u(k) + P_2 + t_2 + P_4 + t_8)$$

The system mode is determined by the "switching" variable, which is defined by:

$$\begin{bmatrix} z_1(k) \\ z_2(k) \end{bmatrix} = \begin{bmatrix} x_1(k) + P_1 \\ x_2(k) + P_2 \end{bmatrix} = \begin{bmatrix} \max(x_1(k-1) + 2P_1, u(k) + P_1 + t_1) \\ \max(x_2(k-1) + 2P_2, u(k) + P_2 + t_2) \end{bmatrix} =$$

$$\begin{bmatrix} \max(x_1(k-1) + 2, u(k) + 5) \\ \max(x_2(k-1) + 6, u(k) + 4) \end{bmatrix}$$, and

$$\mathrm{z(k)} = \begin{cases} 1; & \text{if } \mathrm{z_1(k) \le z_2(k)} \\ 2; & \text{if } \mathrm{z_1(k) > z_2(k)} \end{cases}$$

Therefore, for the 1st mode $\mathrm{x_1(k-1) + P_1 \le\ x_2(k-1) + P_2}$, the system matrices are:

$$\mathrm{A_{(1)}} = \begin{bmatrix} 1 & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & 3 & \varepsilon & \varepsilon & \varepsilon \\ 2 & \varepsilon & 6 & \varepsilon & \varepsilon \\ \varepsilon & 6 & \varepsilon & 4 & \varepsilon \\ 8 & 11 & 12 & 9 & \varepsilon \end{bmatrix}, \qquad \mathrm{B_{(1)}} = \begin{bmatrix} 4 \\ 1 \\ 5 \\ 4 \\ 11 \end{bmatrix}$$

Similarly, we can derive the system matrices for the 2nd mode $\mathrm{x_1(k-1) + P_1 \ge}$ $\mathrm{x_2(k-1) + P_2}$:

$$\mathrm{A_{(2)}} = \begin{bmatrix} 1 & \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & 3 & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & 6 & 6 & \varepsilon & \varepsilon \\ 2 & \varepsilon & \varepsilon & 4 & \varepsilon \\ 8 & 12 & 12 & 9 & \varepsilon \end{bmatrix}, \qquad \mathrm{B_{(2)}} = \begin{bmatrix} 4 \\ 1 \\ 4 \\ 5 \\ 10 \end{bmatrix}$$

It is noted that $\mathrm{z_1(k)}$ and $\mathrm{z_2(k)}$ are the time instances at which $\mathrm{M_1}$ and $\mathrm{M_2}$ finish their assembly tasks in cycle k. It is clear that the 1st mode corresponds to $\mathrm{M_1}$ finishing first, and the 2nd mode corresponds to $\mathrm{M_2}$ finishing first. Therefore $\mathrm{z(k)}$ determines two typical max-plus linear models in two modes.

The use of “switching” max-plus linear systems in modeling discrete event dynamic systems offers some new opportunities. In regular max-plus linear systems, one can only model synchronization in a fixed order. The use of “switching” max-plus linear systems provides the possibility to include concurrency operation and non-deterministic interruption events. Typical non-deterministic events in production systems include machine random failures and starvations caused by delayed material delivery.

Modeling of machine down time with “switching” max-plus linear model

1) Predetermined machine down time (e.g., preventive maintenance schedule, predetermined starvation)

Preventive maintenance is one of the predetermined machines down time events aimed at the prevention of breakdowns and failures. It is designed to preserve and enhance equipment reliability by replacing worn components before they actually fail.

Modeling of preventive maintenance can be easily done using a “switching” max-plus linear system. When we decide the scheduled machine maintenance criteria, (e.g., machine processing time, number of products assembled, or reliability level, etc.), control variables can be used to switch the modes, maintenance time can be added into the corresponding machine process, which turns out to be a nominal processing time, and converts the A,B,C matrices respectively.

Example: Preventive maintenance on two machines with infinite buffer capacity:

We define when machine M1 produces every 100 intermediate products, certain machine parts need to be replaced to prevent its actual failure.

We can set: $z(k) = \begin{cases} 1; & if\ (k\ \mathrm{MOD}\ 100) \neq 0 \\ 2; & if\ (k\ \mathrm{MOD}\ 100) = 0 \end{cases}$

From previous calculation we get: $A_{(1)} = \begin{bmatrix} P1 & \varepsilon \\ 2P1 & P2 \end{bmatrix}, B_{(1)} = \begin{bmatrix} t1 \\ P1 \end{bmatrix}, C_{(1)} = \begin{bmatrix} \varepsilon & P2 \end{bmatrix}$

If the preventive maintenance time is PM1 for machine 1, then when maintenance is triggered, the nominal processing time can be calculated as $P1^{*} = P1 + PM1$. Correspondingly, we have

$$A_{(2)} = \begin{bmatrix} P1+PM1 & \varepsilon \\ 2P1 & P2 \end{bmatrix}, B_{(1)} = \begin{bmatrix} t1 \\ P1+PM1 \end{bmatrix}, C_{(1)} = \begin{bmatrix} \varepsilon & P2 \end{bmatrix}$$

2) Random failure

Machine random failures cannot be predicted precisely and will cause system uncertainties. Using a “switching” max-plus modeling technique, we first model the general assembly system without any random downtimes and obtain $\mathrm{x(k)}$ vector for each process and machine. Second, we define the “switching” variable when random failures happen. $\mathrm{FS_i(j)}$ denotes the j-th random failure start time at machine i and $\mathrm{FF_i(j)}$ denotes the j-th random failure finish time at machine i. A control variable $\mathrm{v(k)}$ can be defined for each $[\mathrm{x_i(k)}, \mathrm{x_i(k+1)}]$ as:

$$\mathrm{v_i(k)} = \begin{cases} 0; & \text{for } \mathrm{FS_i(j) \geq FF_i(j)} \\ 1; & \text{otherwise} \end{cases}$$

Therefore the “switching” variable can be defined as:

$$\mathrm{z(k)} = \begin{cases} 1; & \text{if } \prod_{\mathrm{i=1}}^{\mathrm{n}} \mathrm{v_i(k)} = 0 \\ 2; & \text{if } \prod_{\mathrm{i=1}}^{\mathrm{n}} \mathrm{v_i(k)} \neq 0 \end{cases}$$

where n is the total number of machines. If $\mathrm{z(k)} = 1$, then no random failures at step k. If $\mathrm{z(k)} = 2$, then processing time for machine i can be modified as:

$P_i^* = P_i + \sum_{j=1}^{m}(FF_i(j) - FS_i(j))$, where m denotes the number of random failures during the time period $[x_i(k), x_i(k+1)]$.

For the two-machines-infinite-buffer example, we assume M_1 has a random failure in time period $[x_1(k), x_1(k+1)]$, therefore we can derive system matrices for both modes:

$$A_{(1)} = \begin{bmatrix} P_1 & \varepsilon \\ 2P_1 + t_2 & P_2 \end{bmatrix}, \quad B_{(1)} = \begin{bmatrix} t_1 \\ P_1 + t_1 + t_2 \end{bmatrix}, \quad C_{(1)} = [\varepsilon \quad P_2 + t_2]$$

and

$$A_{(2)} = \begin{bmatrix} P_1^* & \varepsilon \\ 2P_1^* + t_2 & P_2 \end{bmatrix},$$

$$B_{(2)} = \begin{bmatrix} t_1 \\ P_1^* + t_1 + t_2 \end{bmatrix},$$

$$C_{(2)} = [\varepsilon \quad P_2 + t_2]$$

where $P_1^* = P_1 + FF_1(1) - FS_1(1)$.

At the beginning, system equations with no random failures can be derived in mode 1 using $A_{(1)}, B_{(1)}$ and $C_{(1)}$ to obtain $x(k)$, $k = 1, \dots, N$. Where N is the total number of products. And $x(k) = [x_1(k) \quad x_2(k) \dots \ x_n(k)]^T$, where n is the total number of machines, here in this example, n=2. Whenever a random failure happens, the mode is switched, and the corresponding P_i^* will be updated in $A_{(2)}, B_{(2)}$ and $C_{(2)}$. Finally, the general assembly system equations and matrices can be updated in real time.

2.3 Modeling of Material Handling System with Predetermined Production Sequence

After our discussion of DEDS modeling of the general assembly system using a max-plus linear system, we need to take the material handling system into consideration. In general assembly system, different types of parts are stored in line-side buffers, located next to the assembly machines. Since the sizes of the line-side buffers are limited, we cannot store infinite number of parts. Therefore parts should be delivered to the assembly machines when parts are consumed to a certain level, called re-order point (RP). Typically, re-order points are defined in terms of the time remaining for parts to last for assembly. For example, a 15min re-order point means that the remaining parts would last for 15 min of production. We can calculate the re-order points as: $RON(p)$ = $AVE(p)$ $\times$ $\text{Reorder Time} / CT(s)$

Where $RON(p)$ is the re-order number of part p, $AVE(p)$ is the average utilization of part p, which can be obtained from historical production data.

Whenever $INV(p) <= RON(p)$, a part replenishment request will be sent out.

If any driver is available, the driver will start to deliver part p immediately,

If all drivers are in the trip of feeding, unresponsive requests will be put on the request list according to first-come-first-serve (FCFS).

It is assumed in this section that all production sequences are known, and therefore, corresponding part consumption can be predicted. Furthermore, for any assembly machine, each part consumption corresponding to different type of car is known. This information is attached to each car, and is called the part consumption tag. Therefore, every part consumption tag can be traced with each car. According to the tag, operators working on an assembly machine can easily identify which part should be used. Therefore, a part consumption tag matrix for each car can be obtained. Figure 2-3 demonstrates an example of a part consumption tag matrix. For example, car 1 will consume 1 piece of part1, 0 piece of part2, 1 piece of part3, etc.

$$\begin{array}{cccccccccccc}
 & \textit{part1} & \textit{part2} & \textit{part3} & \textit{part4} & \textit{part5} & \textit{part6} & \textit{part7} & \textit{part8} & \textit{part9} & \\
car\,1 & 1 & 0 & 1 & 1 & 0 & 1 & 2 & 1 & 1 & \ldots \\
car\,2 & 1 & 1 & 0 & 0 & 2 & 0 & 1 & 0 & 0 & \ldots \\
car\,3 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & \ldots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ldots \\
cari & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 2 & 1 & \ldots \\
cari\ +1 & 1 & 2 & 0 & 1 & 2 & 1 & 1 & 1 & 1 & \ldots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ldots \\
carN\ -2 & 1 & 1 & 0 & 2 & 1 & 0 & 1 & 0 & 2 & \ldots \\
carN\ -1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & \ldots \\
carN & 1 & 0 & 1 & 2 & 1 & 0 & 2 & 1 & 0 & \ldots
\end{array}$$

Figure 2-3: Parts consumption matrix

Since the car production sequence and the part consumption are pre-determined, the initial number of part p, $RON(p)$, $AVE(p)$ are all known, so the occurrence of part replenishment requests can be calculated beforehand; for example, for part 34, we can accurately calculate the part replenishment requests which will be sent for the production of car 204, car 416, car 610… Furthermore, parts will be consumed in a pre-determined sequence regardless of the condition of general assembly system. From max-plus based general assembly system model, we can calculate the time instance $X_i(k)$, given parameter (i, k). $X_i(k)$ can provide us the timing information for any car k entering machine i. Therefore, the time instance of a part replenishment request can be calculated based on a predetermined part consumption matrix:

M(j): re-order number of part j,

Acc(k,j): accumulative usage of part j, when car k uses part j,

Initial(j): initial quantity of part j,

ROQ (j): Replenishment quantity of part j,

For part request, *Index (k,j)*

*Initial(j)-Acc(k,j)+N*ROQ(j)== M(j)* ,where N=0,1,2,3,… (2.6)

Given $X_i(k)$, the starting time of car k entering machine i, we have a mapping relationship between car production and part consumption, which can help us calculate the time instances of part replenishment requests.

Rearranging the time instances $X_i(k)$ of part replenishment requests in ascending time order, we can have a predetermined replenishment requests sequence, for example car34-part 21, car78-part 132, car 149-part 56…. (car34-part 21 means when car 34 consumes part 21 to finish its assembly task, it will cause part 21 to send out a replenishment request, which occurs earlier than car 78-part 132's request been sent out according to the example sequence). Similarly all these rearranged $X_i(k)$ remain valid until starvation or random failure occurs. Whenever these kinds of interruptions happen, using "switching" max-plus model, we can obtain updated $X_i^{'}(k)$ by repeating the above part replenishment request generating procedure. With real time information, all the estimated time instances $X_i(k)$ should be accurate. Figure 2-4 demonstrates the procedure of rearranging part replenishment requests and generating a request list.

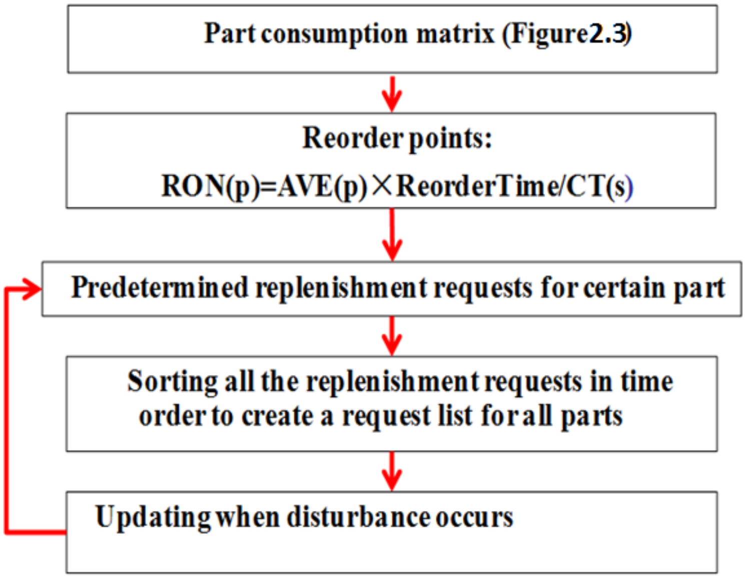
Part consumption matrix (Figure2.3)
Reorder points:
RON(p)=AVE(p)×ReorderTime/CT(s)
Predetermined replenishment requests for certain part
Sorting all the replenishment requests in time order to create a request list for all parts
Updating when disturbance occurs

Figure 2-4: Request list generating process

2.4 Numerical Experiments

In this case study, the schematic layout of a general assembly system with material handling is shown in Figure 2-5, which is based on an automotive assembly plant. The system consists of 8 sections, between which are buffers $(N_1, \ldots, N_7)$ with finite capacities.

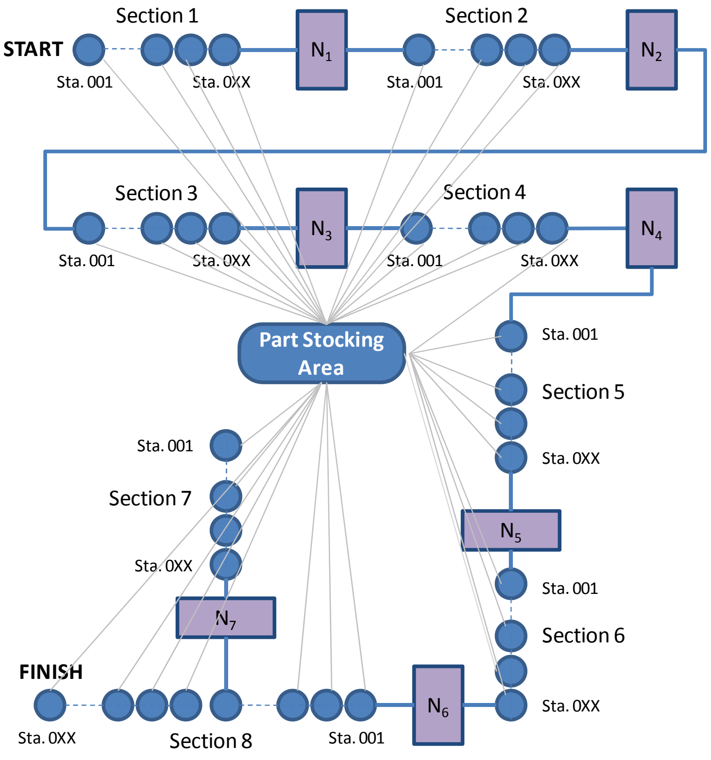

Figure 2-5: 8 section system layout

Each section consists of a sequence of assembly machines and a conveyor. The conveyor transfers cars from machine to machine where operators finish assembly tasks during a specific time period. In a real system with a continuous conveyor, a cycle time is defined as the time difference between a car enters a machine and leaves it. If the operator does not finish a job during a cycle time, a cord needs to be pulled, and the conveyor stops. Although the conveyors in a real system are moving continuously, we assume in our study that a conveyor transfers a job from a machine to the next machine only when all machines (of the section) complete their work. This assumption introduces little inaccuracy in our model though the production dynamics essentially are the same - the system will stop if a machine does not finish work in time. In a section, there are no buffers between machines. Operations are synchronous within each section. For simplicity, and because of the nature of synchronization, we assume that each machine of the same section has the same cycle time.

For manual operations, there are two types of failures. In real failures, tools or things of a similar nature are broken. In equivalent failures, operators do not finish assembly tasks within a cycle time. We call these over-cycled operation failures, because they have the same effect as a broken tool failure: the conveyors stop and all other machines in the same section are left waiting.

Typically, thousands of different parts, with distinctive part numbers, need to be assembled onto a car. A part is stored at a line-side buffer, it may or may not be used for every car, depending on the options that the car may have. In the simulation model, a predetermined production sequence is given. According to a predetermined production sequence, when a part has been consumed, the remaining part count at the line side buffer will be updated. When it reaches a certain level (re-order point), the operator should send a signal for requesting the delivery of the part.

In this example, we focus on the current production environment - one dolly per trip. In this situation, even if there is more than one request coming from the assembly operators, a driver can only deliver one dolly per trip, we will introduce the multi-dolly MH system in the next chapter.

Parts are consumed based on a predetermined production sequence, and are replenished as requested by the operators. The scale of this practical system includes 8 sections, more than 160 machines, more than 300 line side buffers and dozens of drivers as illustrated in Figure 2-5.

The simulation setup includes 20 replications with 10,000 minutes running time and 2,000 minutes warm-up time. In order to verify the effectiveness of the integrated modeling method, we compare the average throughput results with a detailed simulation model using the Arena software package. We use 12 groups of test cases with variety of different parameters (cycle times, buffer capacity, initial buffer size, drivers' travel speed, etc.). Table 2-1 shows the *t*-tests for the comparison. All *t*-tests return $h = 0$, i.e., failures to reject the null hypothesis at the 1% significance level.

Case #	Arena	Max-plus	Difference (%)	p-value
1	37.34	37.18	-0.43	0.36
2	39.17	39.19	0.051	0.41
3	35.05	34.75	-0.86	0.91
4	34.98	34.78	-0.57	0.67
5	37.65	37.53	-0.31	0.15
6	39.85	39.86	0.025	0.2
7	35.55	35.38	-0.48	0.55
8	35.19	35.04	-0.42	0.32
9	39.79	39.65	-0.35	0.41
10	36.31	36.16	-0.47	0.67
11	34.43	34.4	-0.087	0.18
12	35.91	35.69	-0.61	0.72
mean			-0.37	

Table 2-1: Cases of throughput comparison

The accuracy of the method is further evaluated by comparing the simulation results with the actual production data. The following production data are collected to validate the results from our analysis.

- Throughput of the tested production system: Throughput is the number of cars assembled per hour in steady state operations. It represents the production capacity and efficiency.
- Average driver utilization: The utilization of drivers provides the percentage of their time on delivery or waiting for dispatching, which illustrates the accuracy level of the model with respect to the real system.

	Actual	**Sequenced Max-plus**
Throughput (jobs/hour)	27.12	27.25
Driver #	**Driver Utilization %**	**Driver Utilization %**
1	66.56	66.82
2	63.44	63.58
3	86.87	87.12
4	76.34	76.84
5	81.41	81.76
6	84.28	84.58
7	78.56	79.14
8	85.32	84.58
9	77.85	79.14
10	99.13	99.82
11	86.32	86.73
12	86.31	86.91
13	72.67	73.01
14	69.13	69.42
15	64.67	64.33
16	54.31	54.61

Table 2-2: Throughput and average driver utilization results comparison (30 min re-order points)

In addition, the case study also demonstrates the efficiency of the modified max-plus method. We simulate the system with 10,000 minutes running time and 2,000 minutes warm-up time, while the actual CPU running time using our method is within 3 seconds. On the other hand, Arena software for the same practical system takes about 387 seconds for one simulation replication. These results are obtained on a laptop with 2.8 GHz CPU and 4GB RAM. It is also noticed that considering this 167 machines system, the matrix $\boldsymbol{A}$ can be of size 167×167. Although the sizes of system matrices are large, high computational efficiency can still be achieved.

In summary, these results have validated and confirmed that the integrated modeling based on the modified max-plus algebra is accurate in predicting the dynamic characteristics of general assembly system with MH. The modeling approach of general assembly system with MH focuses on four main challenges: (1) uncertainties of the system, (2) large number of events, (3) large state space, and (4) coupling system dynamics. The model is demonstrated to be efficient, accurate and can provide a platform for further material handling optimization.

2.5 Conclusions

In this chapter, a systematic modeling approach using a max-plus linear system was studied. We derived the mathematical representation of a system containing a finite buffer capacity as well as manufacturing blockage & starvation. The model in our example considered the case of a two-machine-one-buffer system, and was extend to multiple machines and multiple buffers systems. A "switching" max-plus linear system was proposed to deal with uncertainties and asynchronous behaviors. With these features, a general assembly system can be modeled, timing information can be directly generated from known system configurations. Coupled MH system activities can be estimated through a predetermined production sequence and a part consumption matrix. A request list generating process was presented. Numerical experiments showed the practicality and accuracy of the modeling approach compared to actual experimental data. With the knowledge of part requests timing information, further online material handling dispatching and routing policies became possible.

CHAPTER 3
REAL TIME DISPATCHING AND ROUTING CONTROL WITH PREDETERMINED PRODUCTION SEQUENCE

3.1 Introduction

From the previous chapter, we know that if the knowledge of a predetermined production sequence is known, the dynamic part replenishment requests can be estimated based on the timing information obtained from the general assembly system. Since we can estimate when and where starvation events may happen, more effective dispatching and routing policies become possible to bring out the full potential of multi-dolly material handling systems.

Obviously, single dolly material handling systems will need more frequent numbers of trips that lead to higher driver workload. Hence, if we could deliver more than one dolly per trip, we could reduce the number of trips for a driver. As we mentioned, dolly trains enable us to deliver multiple parts per trip, but at the same time, advanced delivery processes will lead to a more complicated material handling system. In this way, two questions need to be answered, how many and what types of parts need to be carried by the dolly trains when the driver leaves the docking area (dispatching policy), and in what sequence these parts need to be delivered (routing strategy).

Knowledge of part replenishment requests connects the general assembly system with the material handling system. With this knowledge, we can establish a framework of multi-dolly material handling system. Time instances of products assembling can be calculated as we discussed in Chapter 2. Upon disturbance interrupts, all time instances shall be updated accordingly based on the switching mechanism of the modified max-plus modeling technique. Combined with predetermined production sequence, a request list can be generated as discussed in Section 2.3. Based on the knowledge of existing and predicted time instances of part replenishment requests, we have a great advantage in estimating and predicting MH activities and optimizing them in our MH dispatching processes.

Besides, we know many factors will influence system performance, such as dispatching policies, routing strategies, production loss due to starvation, etc. In this chapter, we will use cost of these factors (listed above) as a quantitative parameter to evaluate their impact on system performance. With all the timing information obtained from an updated request list, it is possible for us to find a systematic way to achieve the minimum MH related cost considering all the factors we have investigated.

In this chapter, assumptions, notation, and existing dispatching policies are discussed. Several dispatching policies are proposed based on dynamic part replenishment requests using forecasting and penalty prediction model. A simple model will be presented to show how different policies may affect the MH system as well as the general assembly system. A mapping procedure between the parts routing problem and vehicle routing problem with time window (VRPTW) is discussed. An integrated model considering both dispatching and routing to improve system performance with minimum MH related cost is proposed. Research results are shown at the end of the chapter.

3.2 Dispatching Policies with Predetermined Production Sequence

3.2.1 Notations

Before we present the detailed system description, the nomenclatures used in this chapter are introduced:

$OUT_i(j)$ time instance of driver i starting j-th delivery trip from docking area(min)

$BACK_i(j)$ time instance of driver i coming back to docking area from j-th delivery trip (min)

$S_i(M(m,s))$ the real starting time for processing car i on M(m,s) (min)

$Driver_i(p)$ Time for driver i to deliver part (p) from docking area to location of line-side buffer p (min)

$Driver_i(p_k, p_l)$ single trip time of driver i moving from location of line-side buffer p_k to location of line side buffer p_l (min)

A_i^w the driver i's starting time for trip w (min)

Ta_i^w driver's arrival time at location i for trip w (min)

$\mathrm{F_w}$ accumulated cost for trip w ($)

3.2.2 Drivers' Assignment Rule

First we need to clarify the driver assignment process. We use the rule of longest waiting time, which means the dispatching center always sends requests to the earliest available drivers in the waiting queue: $\min_{i=1,2,\ldots,V} \{BACK_i(w)\}$ for delivery trip w, where $BACK_i(w) = \sup\{BACK_i(j) : \forall j < w\}$, $BACK_i(j)$ is the time instance at which driver i returns to the docking area (we define the docking area as location “0”) after delivery trip j, w is the number index for current trip, V is the total number of drivers.

3.2.3 Proposed Multi-Dolly Dispatching Policies

Now, with the knowledge of a predetermined production sequence, a dynamic part request list can be developed based on updated timing information, which is obtained from the general assembly system, this will provide us advantages in developing optimized MH dispatching policies.

It is noted that in this section, we will use the simplest first-come-first-serve (FCFS) routing strategy to determine the delivery sequence of chosen requests for a specific trip. Since a dispatching policy is our primary concern in this section, routing strategies will be discussed later.

3.2.3.1 Re-order points (RP)

It is the same as the basic RP policy in section 2.3, except that drivers are allowed to deliver more than one dolly per trip. Two possible situations are illustrated here:

1) Assuming the part request signal is triggered when the corresponding driver is in an idle state, the driver departs the docking area immediately: $OUT_i(j+1)$ in Figure 3-1.

2) If several part requests have been put onto the request list, after completing one trip and returning to the docking area, the driver should depart with the

current requested parts immediately. If the number of replenishment requests on the list is greater than the maximum allowed by the dolly trains, then take the maximum number in FCFS order, for example: $OUT_i(j+2)$ in Figure 3-1.

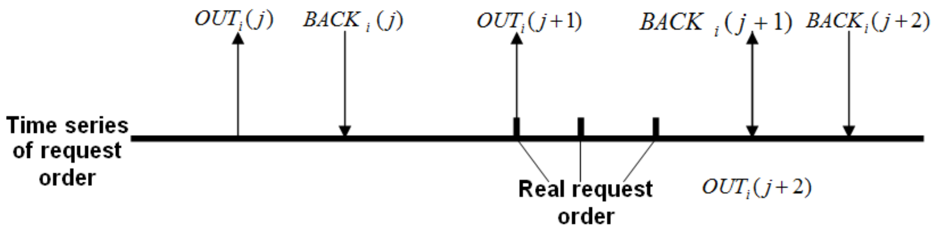

Figure 3-1: Illustration of re-order points (RP) policy

3.2.3.2 Double threshold (DT)

DT policy is mainly based on two possible lists, one for part requests, and the other for warning. In our study, we set the *request threshold* at 15 min (remaining parts will last for 15 min production), and the *warning threshold* at 30 min (remaining parts will last for 30 min production). Parts on the *request* list have higher priority than those on the *warning* list. Two situations are possible:

1) When a warning signal is triggered while the corresponding driver is idle, the driver keeps waiting, until the first part request occurs, and then the parts on the warning list will be added to this driver.
2) If there are several part requests and warnings already on the lists, the driver should first pick up all the part requests. If the number is less than the maximum number, current parts on the warning list can be added to the driver. Similarly, if the number is greater or equal to the maximum allowed dolly trains, the maximum number of dollies will be used for this trip.

3.2.3.3 Fixed Moving Window (FMW)

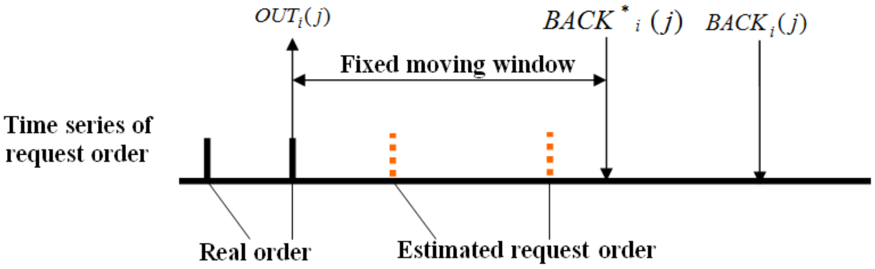

Figure 3-2: Illustration of Fixed Moving Window (FMW) policy

FMW dispatching policy is based on the request list for both existing and estimated requests, since the round trip travel time for a driver to deliver specific parts in FCFS is a constant, the whole round trip travel time can be treated as a fixed moving window on a time scale:

$$\text{MovingWindowWidth} = \text{Driver}_i(p_f) + \sum \text{Driver}_\text{i}(p_k, p_n) + \text{Driver}_\text{i}(p_l) \quad (3.1)$$

where requests p_l, p_f are the first and last existing part requests on the request list, $BACK^*{}_i(j)$ is the driver's estimated time back to docking area based on existing requests, as shown in Figure 3-2. Checking all the parts covered by this specific driver, we can pick the existing requests, and then calculate the corresponding time $S_i(M(m,s))$ of the estimated requests, based on the predetermined part replenishment sequence.

If $S_i(M(m,s)) - OUT_i(j) \le MovingWindowWidth$, it means the estimated part replenishment requests might also need be added to the driver. Two possible situations are illustrated:

1) When the part request signal is triggered while the corresponding driver is in idle state, the only moving window is obtained and we can check the estimated part requests within the window and add them to the driver.
2) After the driver returns from a trip, if one or more part requests have been put onto the request list, the moving window (the round trip travel time) can be calculated. If more estimated part requests are obtained within this moving window, they should be added to the driver. When the number of requests is equal to or greater than the maximum allowed for the dolly trains, then take the allowed maximum number as shown in Figure 3-2.

3.2.3.4 Dynamic Moving Window (DMW)

The key idea of DMW is similar to FMW. We need to build up the moving window and check whether more parts should be added to the driver. The main difference is that in FMW, a fixed moving window is decided by the round trip travel time to particular locations of line-side buffers. To fully utilize the advantages of the request list, we propose a dynamic moving window method. Simply speaking, when we add the first estimated part replenishment request within the moving window, a new window is generated accordingly and the new window width is the line-side buffer to line-side buffer travel time in addition to the one way travel time back to docking area. If a new estimated request is added to the driver, the dynamic window will expand until the maximum number of allowed dolly trains is reached, or no more estimated requests can be obtained within the window.

3.2.3.5 Penalty Based Strategy (PB)

For a certain trip w, we have several request candidates (e.g., E1, E2, E3), where Ei is the time at which replenishment request is sent out for part i. Candidates selection rules will be addressed later. If we select E1 and E2 for trip w, which means E3 will be sacrificed this time, then what is the penalty for not carrying E3? Will this penalty influence our decision for request selection? This is the starting point for a penalty-based policy.

For example, we pick up E1 and E2, and sacrifice E3 for trip w. Then we define:

$F_w = F_{w-1}$+transportation cost +production loss for trip w

PF_w: Penalty cost of not carrying E3 for trip w

$PF_w = d \times (Ta_0^w - L_3^{w+1} + l_{03})$

$F_w^* = F_w + PF_w$

Where l_{ij} is the time for driver to move between locations i and j, d is the unit time cost for starvation duration, L_i^w is the “part used up time” for request Ei in trip w. Repeating the above procedure, we can calculate F_w^* for E1, E3 without E2, and F_w^* for E2, E3 without E1. For all F_w^* , $\min(F_w^*)$ indicates the best selected combination for trip w. Now after we introduce the idea of PF_w, for trip w, select as many requests as the maximum number of dollies available, sorted in ascending time order. For the following requests, calculate corresponding PF_w, if PF_w>0, then this replenishment request should be added as a candidate in this trip w, otherwise, this request does not need to be considered. In the previous example, we have a two dolly model, E1 and E2 which are selected due to the maximum dolly number. If PF > 0 for E3, then E3 should be added and E1, E2 and E3 are the candidates for this trip.

3.2.4 Case Study for Dispatching Policies

The following example is used to demonstrate how different dispatching policies influence the dispatching sequence as well as the system performance. Figure 3-3 shows a three-machine production system without buffers between each other, and there are 4 kinds of parts used on them. Machine 1 uses part 1, Machine 2 uses part 2 and Machine 3 use both part 3 and 4. Assume that only one driver delivers all the required parts, and the driver is available at time instance 12 from the docking area 0 for the first delivery.

To estimate the impact of MH onto the production system performance, it is assumed that the delivery cost of a part is c=1/min, and the cost of starvation caused by a delayed part delivery is d=1000/min. Table 3-1 shows the original predetermined request time (Ei) and time limit to accomplish the request (Li). Table 3-2 shows the travel time between any two line-side buffer locations l_{ij}. For example, $l_{12} = 0.5$ min means that the travel time from line-side buffer 1 location to line-side buffer 2 location is 0.5 minutes, and 0 indicates the docking area location.

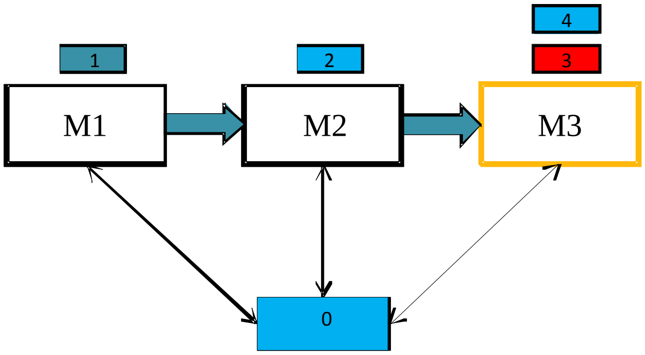

Figure 3-3: 3 machines, 4 parts model

i	Ei (min)	Li (min)
1	10	15
3	11	16
2	11.5	16.5
1	15	20
4	16	21
3	22	27
1	24	29

Table 3-1: Original request list

l_{ij} (min)	**0**	**1**	**2**	**3**	**4**
0	0	2	1.5	2	2.5
1	2	0	0.5	2	2
2	1.5	0.5	0	2	2
3	2	2	2	0	0
4	2.5	2	2	0	0

Table 3-2: Travel time l_{ij} between different locations

3.2.4.1 Case study using Re-order Points (RP)

Trip 1:

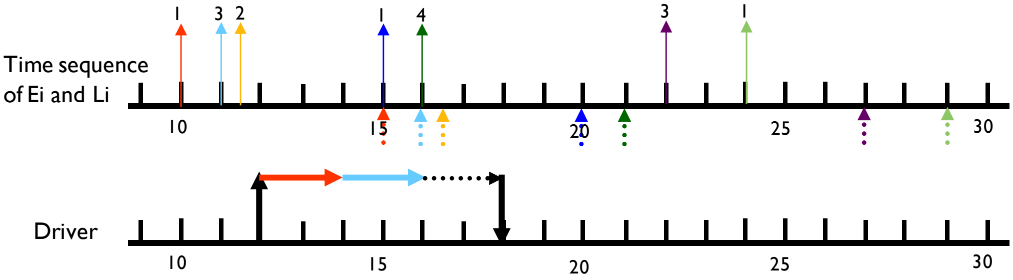

Figure 3-3: Time and delivery sequence of trip 1

Since we assume the driver is available at time instance 12, A^1 =12. According to RP policy and FCFS, we pick E1 and E3 for first trip. First we need to deliver part 1, the time instance for the driver to arrive at line-side buffer 1 location is Ta_1^1=$A_1^1 + l_{01} = 12 + 2 = 14$< ($L_1^1$=15). The driver departs from docking area and arrives at line-side buffer 1 location at time instance 14, which less than its L_1^1 time 15, so no starvation occurs. Next, we need to deliver part 3, the time instance for driver to arrive at line-side buffer 3 location is Ta_3^1=$Ta_1^1 + l_{13} = 14 +$ 2= (L_3^1=16). This can be calculated as: After unloading part 1, the driver heads for line-side buffer 3 and arrives at time instance 16, which is the same time as L_3^1=16, so still no starvation occurs. After delivering part 1, 3, the driver finishes the trip and returns to the docking area at time instance 18. Therefore, the only related cost is transportation cost: $F_1 = \mathrm{c} \times$ (time instance of the driver returns to docking area for trip 1$-A^1$) $= \mathrm{c} \times (18 - 12) = 6$, the cost for trip 1 is therefore 6 units.

Trip 2:

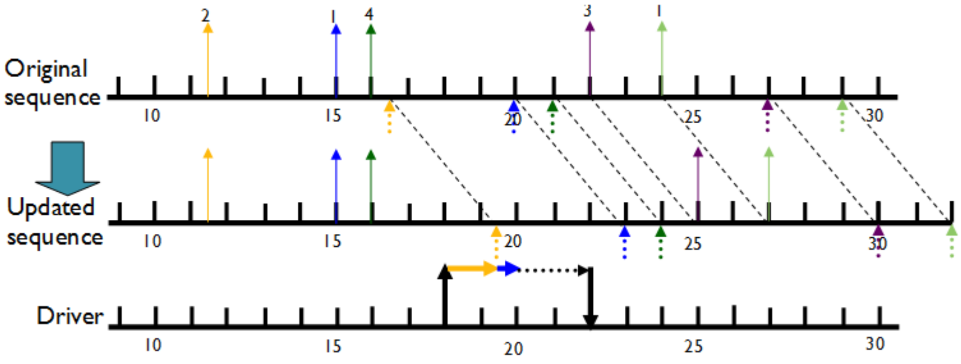

Figure 3-5: Time and delivery sequence of trip 2

We start our trip 2 at time instance A^2=18, and pick the earliest two requests E2 and E1, Ta_2^2=$\mathrm{A}^2 + \mathrm{l}_{02} = 19.5$> ($\mathrm{L}_2^2 = 16.5$), when the driver arrives at the location of lines-side buffer 2, the time instance is 19.5, which is larger than $\mathrm{L}_2^2 = 16.5$, which means starvation occurs. Using the “switching” max-plus model, time information and the request list are updated. We can see clearly from Table 3-3, all the remaining E and L (listed in brackets) are shifted by the starvation duration, which is 3 min. The driver finishes trip 2 and returns to the docking area at time instance 22. Ta_1^2=Ta_2^2+$\mathrm{l}_{12} = 19.5 + 0.5 = 20$<($\mathrm{L}_1^2 = 23$), $\mathrm{Ta}_0^2 = 22$, F_2=$\mathrm{F}_1 + 22 - 18 + \mathrm{d} \times 3 = 3010$ units. For trip 2, the cost includes two parts: transportation cost $\mathrm{c} \times (22 - 18) = 4$ and production loss $\mathrm{d} \times 3 = 3000$, which is dramatically larger than transportation cost.

i	Ei	Li
2	11.5	16.5(19.5)
1	15	20(23)
4	16	21(24)
3	22(25)	27(30)
1	24(27)	29(32)

Table 3-3 Updated request (in bracket) list vs original list

For the following trip, there are no new interesting points worth mentioning. We just list the brief process as follows:

Trip 3: Trip 3 starts at time instance $A^3 = 22$, E4 is picked, $Ta_4^3 = 24.5$>($L_4^3 = 24$), so there will be starvation for 0.5 minute, $Ta_0^3 = 26.5$, F_3=$F_2 + 4.5 + 500 = 3514.5$, the driver comes to docking are at 26.5 min, the accumulated MH cost is 3514.5 units.

Trip 4: Trip 4 starts at A^4 =26.5, E3 is picked, $Ta_3^4 = 29$<(L_3^4), no starvation $Ta_0^3 = 31.5$, F_4=$F_3 + 5 = 3519.5$ units

Trip 5: Trip 5 starts at $A^5 = 31.5$, E1 is picked, $Ta_1^5 = 33.5$>($L_1^5 = 32.5$), there will be starvation for 1 minute, $Ta_0^5 = 35.5$, F_3=$F_2 + 4 + 1000 = 4523.5$ units.

3.2.4.2 Case study using Fixed Moving Window (FMW)

The dispatching sequences for the first 3 trips are exactly the same as shown in 3.2.4.1. After trip 3, we have Ta_0^3=26.5, F_3=3514.5 units.

Trip 4:

So for trip 4 the driver starts the trip at time instance 26.5, $A^4 = 26.5$. According to FMW, E3 is picked first, which has the time window [26.5, 29], the following request 26.5<E1(27.5)<29 is within E3's time window, which indicates E1 needs to be added in trip 4.

Therefore trip 4 will end up with $Ta_0^4 = 33.5$, F_4=$F_3 + 7 = 3521.5$ units.

With FMW, we reduce the number of trips from 5 based on re-order points policy to 4, the corresponding overall trip time is 33.5 min and the total cost becomes 3521.5 units.

3.2.4.3 Case study using Penalty Based Strategy (PB)

Trip 1:

Delivery starts with A_1^1=12 and candidates E1, E3, E2

For E1, E3: Ta_1^1=14, Ta_3^1=16, Ta_0^1=18, $F_1 = 6$

$$PF_1 = 1000 \times (18 - 16.5 + 1.5) = 3000,\ F_1^* = F_1 + PF_1 = 3006$$

For E1, E2: Ta_1^1=14, Ta_2^1=14.5, Ta_0^1=16, $F_1 = 4$

$$PF_1 = 2500,\ F_1^* = F_1 + PF_1 = 2504$$

For E3, E2: Ta_3^1=14.5 , Ta_2^1=16.5, Ta_0^1=18, $F_1 = 6$

PF_1 =5000, $F_1^* = F_1 + PF_1 = 5006$

Check all the F_1^* value, we notice that *min(*$F_1^*) = 2504$ units, which indicates that E1, E2 is the best choice for trip 1.

Trip 2:

A_1^2=16, candidates: E3, E1, E4

For E3, E1: Ta_3^2=$18.5 > 16, Ta_1^2$=$20.5 < 22.5, Ta_0^1$=22.5, $F_2 = F_1 + 2500 + 6.5 = 2510.5$

$PF_2 = 1500$, $F_2^* = F_2 + PF_2 = 4010.5$

For E3, E4: Ta_3^2=8.5,Ta_4^2=18.5, Ta_0^2=21, F_2= $F_1 + 2500 + 5 = 2509$

PF_2 =500, $F_2^* = F_2 + PF_2 = 3009$

For E1, E4: Ta_1^2=18 ,Ta_4^1=20, Ta_0^1=22, F_2=10.5

PF_2 =9000, $F_2^* = F_2 + PF_2 = 9010.5$

min(F_2^*)=3009 units, which indicates that E3, E4 is the best choice for trip 2.

Trip 3:

A_1^3=21, candidates: E1, E3

Ta_1^3=$23 > 22.5$,Ta_3^3=$25 < 31$, Ta_0^3=27.5, F_3=F_3^*= F_2+500+6.5=3015.5

Trip 4:

A_1^4=27.5, candidates: E4

Ta_1^4=$29.5 < 32$, Ta_0^4=31.5, F_4=F_4^*= F_3+4=3019.5

In above simple model, we can compare the operation time, total cost, and delivery sequence with all these 3 dispatching rules.

Dispatching Policies	**Operation time**	**Cost**	**Delivery Sequence**
RP	35.5	4523.5	→[1,3]→[2,1]→ [4]→[3]→[1]
FMW	33.5	3521.5	→ [1,3]→[2,1]→[4]→[3,1]
PB	31.5	3019.5	→ [1,2]→[3,4]→[1,3]→[4]

Table 3-4: Comparison results of three dispatching policies

The comparison results using different dispatching policies, PR, FMW and PB, are illustrated in Table 3-4. Comparing to RP, our newly proposed dispatching methods FMW and PB decrease the operation time by 5.64% and 12.69% respectively, the cost by 22.15% and 33.25% respectively, and the number of trips from 5 to 4. Though it is only a simple case study, it shows how proper dispatching policies can create substantial improvement of a multi-dolly material handling system by reducing the operation time, cost and number of trips.

3.3 Integrated Dispatching and Routing Model

3.3.1 Assumptions and Definitions

After determining which parts need to be delivered for each trip based on a dispatching policy, the next step is to integrate the routing strategy to determine the optimal delivery sequences of all these dolly trains. Actually, in a single dolly

MH system, we will not encounter routing issues, since only one part should be delivered at a time.

In this section, we will adopt the concept of a penalty based (PB) dispatching policy from section 3.2 for our integrated model. Other than simply using FCFS strategy to determine routing sequences in the previous section, our goal in this section is to determine the optimal routing sequences with minimal handling related cost. Note that we assume the existence of a central docking area, where all drivers are located at the start of the shift and where they return at the end of the trip, see Figure 3-6. Let k be the index of the drivers.

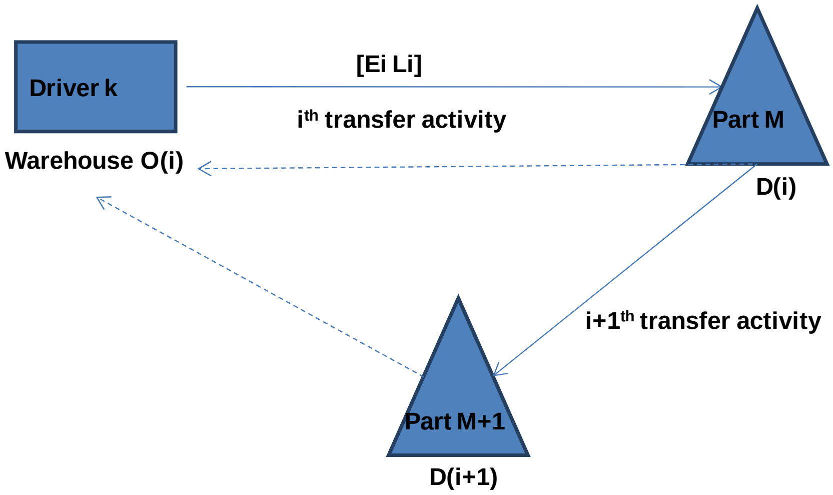

Figure 3-6: Illustration of single dolly delivery (i-th) move, and two dolly delivery (i-th,i+1-th) move

According to the respective schedule of delivery, E_i is the part replenishment request time for part M. Physically, time instance E_i must be earlier than the time when the driver starts i-th move, so the driver can pick up either E_i for single dolly or *E(i-th, i+1-th)* for a two dolly delivery, as shown in Figure 3-6. Therefore, we have the first time restriction E_i . Next we define the machine starvation time, which takes place when the time L_i is achieved, while the corresponding part replenishment request has not been finished. Thus, the second time restriction is L_i which is defined as the machine starvation start time due to part shortage. Consequently, there is a time window for each delivery trip i, given by the closed set [E_i, L_i] to avoid starvation. Finally, let us define l_i as the time that the part can be picked up from the docking location $o(i)$ without causing starvation, a parameter that can be defined based on E_i , L_i and the corresponding inter-resource distances.

Note that d denotes the travel time (e.g., $d_{o(i),d(i)}$is the travel time between the origin and the destination of delivery), ari defines the "actual arrival time" of a driver at the origin of delivery i, while the start time of delivery i is $ai = \max(E_i, ari)$, which indicates the driver can only start a delivery trip until he/she is ready and there is an existing request.

Condition 1. (Definition of l_i): A delivery i is feasible if:

$$E_i < l_i < L_i - d_{o(i),d(i)}$$

Condition 2. Consecutive deliveries i, i+1 are feasible if:

$$a_i + d_{o(i),d(i)} + d_{d(i),o(i)} < l_{i+1}$$

Based on the above two conditions, the sequences of the drivers deliveries can be easily formulated as a traditional vehicle routing problem with time windows (Solomon & Desrosiers, 1988). The one-to-one mapping between the MH problem (MHP) and the typical vehicle routing problem with time windows (VRPTW) is as follows:

(a) Customers of the VRPTW are the drivers' deliveries of the MHP.

(b) Time windows of the VRPTW are the $[E_i, L_i]$ of the MHP.

(c) Distances (time) of the VRPTW are the lengths (time) of deliveries of the MHP.

(d) Feasible sequences of customers in the VRPTW are defined solely by time windows and inter-machine distances, while feasible deliveries and sequences of deliveries in the MHP are defined by conditions 1 and 2 above.

Then, we are able to exploit the VRPTW method to solve the MHP and integrated the VRPTW model in manufacturing environments.

3.3.2 Multi-Dolly Material Handling Routing Optimization

Objective function:

Minimize: Material handling related cost

-Distance based transportation cost

-Penalty cost due to starvation caused by parts shortage

-Driver labor cost

Subject to:

- Maximum number of dolly trains
- Availability of drivers
- Driver routing and timing feasibility
- Windows and relationships for delivery time restriction

To formulate the problem of optimally routing sequence of MH we use the following variables:

a) The arrival–departure time to/from the origin/destination resource of delivery i, denoted by a_i and p_i for delivery i, respectively.

b) The travel time from location i to location j, d_{ij}

c) The sequence in which driver performs deliveries, x_{ij}^{k}

d) The activation index of the driver k, z_k.

e) Production loss due to starvation duration per unit time, *D*

f) Number of drivers in the multi-dolly MH system, *V*

g) Transportation cost, *C*

h) Driver labor cost, G

Variables (c) and (d) are defined as follows:

$$x_{ij}^{k} = \begin{cases} 1, & \text{if delivery j follows delivery i in the route of driver k} \\ 0, & \text{otherwise} \end{cases} \tag{3.2}$$

$$z_k = \begin{cases} 1, & if\ driver\ k\ is\ available \\ 0, & otherwise \end{cases} \tag{3.3}$$

The multi-dolly material handling routing problem can be expressed as follows:

$$\min \quad C \times \sum_{k=1}^{M} \sum_{i=1}^{N} \sum_{j=1}^{N} d_{ij} x_{ij}^{k} + D \times \sum_{i=1}^{N} (a_i - L_i)^{+} + G \times V \tag{3.4}$$

$$\text{Where,} \quad (a_i - L_i)^{+} = \max\{0, a_i - L_i\}$$

Subject to:

$$\sum_{i=1}^{n} \sum_{k=1}^{V} x_{ij}^{k} = 1, \quad \forall\, j = 2,3,\dots,n \tag{3.5}$$

$$\sum_{j=1}^{n} \sum_{k=1}^{V} x_{ij}^{k} = 1, \quad \forall\, i = 2,3,\dots,n \tag{3.6}$$

$$x_{ij}^{k} \leq z_k \quad , \quad \forall\, i,j = 2,3,\dots,n \tag{3.7}$$

$$\sum_{j=2}^{n} x_{1j}^{k} \leq 1, \quad \forall\, i = 2,3,\dots,V \tag{3.8}$$

$$\sum_{i=2}^{n} x_{i1}^{k} \leq 1, \quad \forall\, j = 2,3,\dots,V \tag{3.9}$$

$$\sum_{i=1}^{n} x_{iu}^{k} - \sum_{j=1}^{n} x_{uj}^{k} = 1, \quad \forall\, k = 2,3,\dots,V, \forall u = 2,3,\dots,n \tag{3.10}$$

$$a_j \geq (p_i + d_{ij}) - \left(1 - x_{ij}^{k}\right)M, \ \forall i,j = 1,2,\dots,n, \ \forall k = 1,2,..,V \tag{3.11}$$

$a_j \leq (p_i + d_{ij}) + (1 - \mathrm{x}_{\mathrm{ij}}^{\mathrm{k}})\mathrm{M},\ \forall \mathrm{i}, \mathrm{j} = 1,2, \ldots, \mathrm{n},\ \forall \mathrm{k} = 1,2, \ldots, \mathrm{V}$ (3.12)

$\mathrm{a}1 = 0,$ (3.13)

$\mathrm{x}_{\mathrm{ij}}^{\mathrm{k}} \in \{0,1\},\ \forall \mathrm{i}, \mathrm{j} = 1,2, \ldots, \mathrm{n},\ \forall \mathrm{k} = 1,2, \ldots, \mathrm{V}$ (3.14)

$\mathrm{z}_{\mathrm{k}} \in \{0,1\},\ \forall \mathrm{k} = 1,2, \ldots, \mathrm{V}$ (3.15)

The objective function (3.4) seeks to minimize the overall MH related costs. The first term of (3.4) reflects the cost of the transportations; the second term considers the penalty cost due to starvation; the third term indicates the labor cost.

Constraints (3.5) and (3.6) ensure that exactly one driver loads and unloads one dolly per move for either single dolly trip or multi-dolly trip. Constraint (3.7) guarantees that no deliveries are performed by inactive drivers. Constraints (3.8) and (3.9) account for the availability of drivers leaving and returning to the docking area. Constraint set (3.10) is the typical flow conservation equation that ensures the continuity of each driver route.

Constraints (3.11)–(3.15) are related to time restrictions and guarantee the feasibility of the schedule for each driver. In particular, constraints (3.11) and (3.12) ensure that, if deliveries i and j are consecutive in the schedule of driver k, then the arrival time at the beginning of delivery j equals the departure time from end point of delivery i, plus the travel time between these two line-side buffers locations. Note that we use artificial variable M, which refers to a large number. In the event that deliveries i and j are not performed by the same driver or are not consecutive, constraints (3.11) and (3.12) become inactive. Constraint (3.13) sets the first departure time from the docking area equal to zero, as all routes originate from the docking area. Constraints (3.14) and (3.15), enforce integrality for the x_{ij}^{k} and zk variables, respectively. As a result the model of (3.4)–(3.15) is a linear binary program, closely related to the typical formulation of the vehicle routing problem with time window (Solomon & Desrosiers, 1988).

3.4 Numerical Experiments for Dispatching and Routing Policies

We will use the same set of example as used in Section 2-4, which simulates an actual automotive assembly plant.

1) Multi-dolly policies throughput analysis

Dispatching Policies	**1dolly**	**2 dolly**	**3dolly**
Integrated	35.84±0.04	37.58±0.05	37.81±0.05
PB	35.84±0.05	37.51±0.06	37.60±0.07
FMW	35.82±0.05	37.50±0.05	37.53±0.06
RP	35.82±0.05	37.44±0.06	37.48±0.06

Table 3-5: Throughput comparison with different dispatching policies and dolly numbers

The results show that under the same working environment, multi-dolly MH system outperforms the single dolly system for all policies by over 5%, while proper policies can produce more improvement. Besides, the integrated policy outperforms the others, and demonstrates smaller variances, especially when the number of dolly trains increases.

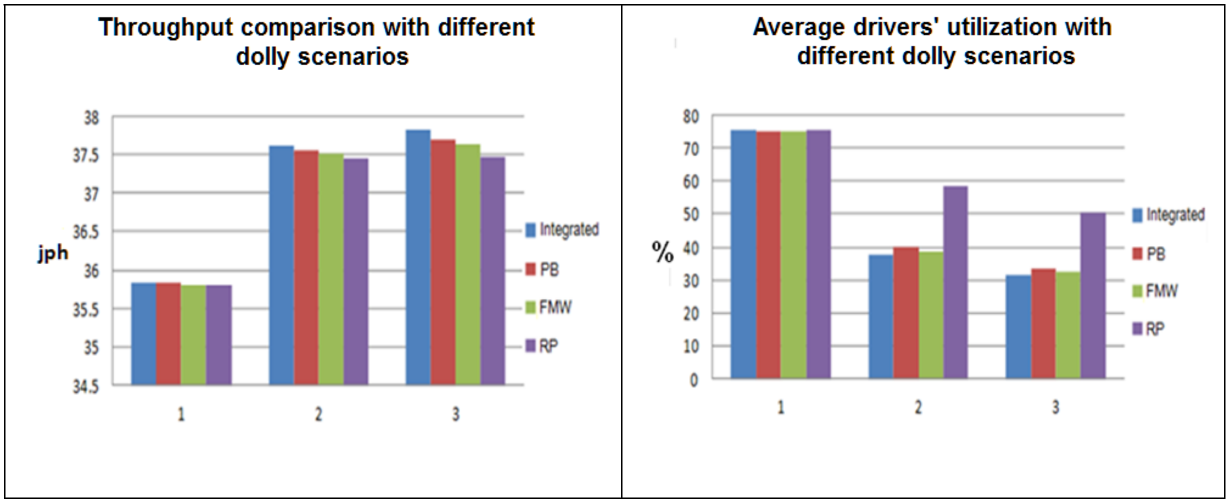

Figure 3-7: Throughput and average drivers' utilization with different dispatching policies and number of dolly trains

2) Average driver utilization analysis

Figure 3-7 shows that average driver utilizations for FMW, PB, and integrated policies are similar, while RP is substantially larger by over 40% in multi-dolly scenario, which demonstrates that these three proposed policies are all effective in reducing driver's workload in multi-dolly MH system. This is a promising sign that we can possibly reduce the number of drivers and the material handling related cost while still sustaining desired system throughput by applying proper dispatching and routing policies in multi-dolly MH system.

3.5 Conclusions

A framework of multi-dolly material handling system has been established, based on a dynamic request list to connect both the general assembly and the material handling systems. For dispatching policies, numerical tests showed that compared to currently used RP strategies, the proposed FMW, DMW and PB dispatching policies are more effective by reducing the number of trips, total trip time and cost. A mapping process between MH problem and VRPTW was established. Furthermore, an integrated model with both dispatching and routing has been proposed to support the production system with the minimum MH cost. The practical numerical case demonstrated the effectiveness of integrated modeling as well as other proposed dispatching policies.

CHAPTER 4
DRIVERS’ ZONING ASSIGNMENT OPTIMIZATION IN GENERAL ASSEMBLY SYSTEM WITH MATERIAL HANDLING

4.1 Introduction

In a real general assembly system with material handling, specific line-side buffers are assigned to certain drivers for parts delivery so that the responsibility of on-time parts delivery can be tracked. The purpose of this chapter is to find an assignment of drivers to working zones with specific line-side buffers, such that the desired system throughput is achieved with minimum MH related cost. The corresponding problem will be called the drivers’ zoning assignment (DZA). Although the manual assignment works fine when the number of drivers is relatively large, systematic methods are needed when the number of drivers is close to the minimum number (i.e., the minimum number of drivers necessary to maintain the desired throughput level), because the DZA problem becomes very complicated in these cases.

A drivers’ zoning assignment could be either fixed or flexible in a production system. A flexible zoning strategy is superior to a fixed zoning strategy in the case of timely parts delivery. However, fixed zoning strategy has its advantages in reducing errors in the delivery process and saving cost in driver cross training. We will start with fixed zoning in our initial zoning formation, and then consider both in later sections.

A schematic view of a production system in automotive manufacturing is given in Figure 4-1, the configuration is the same as described in section 1.1. To demonstrate the configuration of fixed zoning formation, we assign line-side buffers $b_{11}, b_{12}, b_{31}, b_{33}$ to driver 1, and line-side buffers $b_{21}. b_{22}, b_{32}, b_{41}$ to driver 2 to form 2 fixed zones A_1 and A_2. Drivers will be only responsible to the part replenishment requests within their assigned zones.

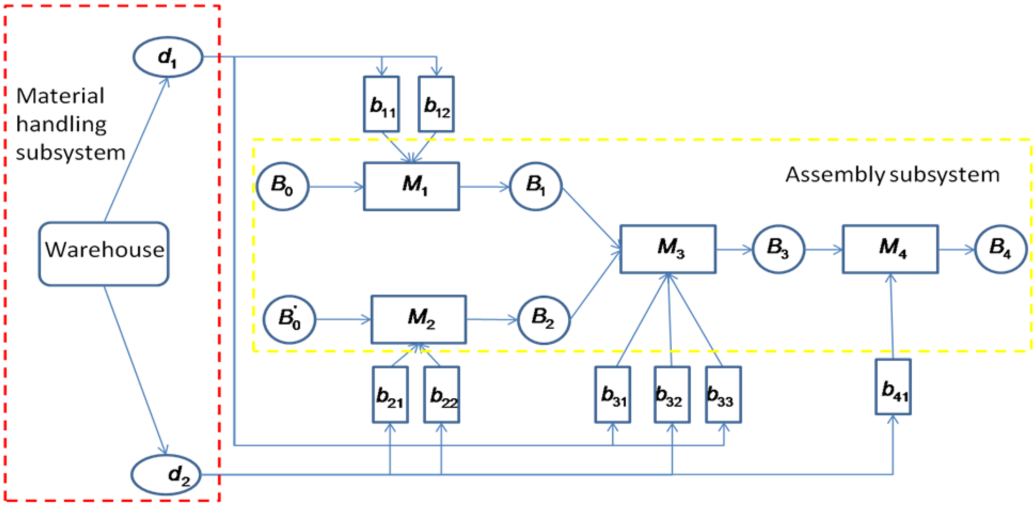

Figure 4-1: A schematic view of the production system with fixed zoning

As we discussed in the literature review section, Pan et al., (2008) proposed a meta-heuristic algorithm Particle Swan Optimization (PSO) based method to solve this problem. It has shown its effectiveness and efficiency in solving large scale DZA problems. However, the clear disadvantage is that it provides insufficient insight to the DZA problem. In addition, global convergence of PSO highly depends on the initial zoning. So in this chapter, we will first investigate the characteristics of DZA, find out a proper formation method of initial zoning by adopting the concept from Parallel Machine Scheduling (PMS) as discussed in Section 1.2.3. Based on corresponding initial zoning, PSO is implemented to solve more practical DZA problem in larger scale. A numerical example will be studied to validate the proposed zoning optimization.

4.2 Initial Zoning Investigation

4.2.1 System Formulation

In this section, the DZA problem will be mathematically formulated under fixed zoning assumption for initial zoning formation. First, details about the general assembly and material handling systems will be described. Then, some key problem definitions will be developed. Finally, the DZA problem will be formulated.

4.2.1.1 System Description

A typical general assembly system with material handling is shown in Figure 4-1. Before we present the detailed system description, the nomenclatures used in this research are introduced:

M Number of machines in the general assembly system. All machines are indexed as $m_i, i = 1, 2, ..., M$.

B_i In-process buffers upstream to machine m_i . N_i is the capacity of in-process buffer B_i , $i = 1, 2, ..., M - 1.$

b_i Number of line-side buffers attached to machine m_i. All line-side buffers attached to machine m_i are indexed as b_{ij}, $i = 1, 2, ..., M$, $j = 1, 2, ..., b_i$, capacity of line-side buffer b_{ij} is n_{ij}

TP Throughput of the production system

C Material handling related cost in the production system

RTT_{ij} Round trip time to finish part delivery from the docking area to line-side buffer b_{ij} and return to docking area

QTY_{ij} Replenishment quantity of line-side buffer b_{ij}

USG_{ij} Average usage rate (consumption rate) of line-side buffer b_{ij} in a cycle

π Delivery policy for driver part delivery

D Number of drivers in the material handling system. All drivers are indexed as d_i , $i = 1, 2, ..., D$.

A Assignment of line-side buffers to drivers, $A = (A_1, A_2, ..., A_D)$, where A_i is the set of line-side buffers assigned to driver d_i , $i = 1, 2, ..., D$.

All assumptions regarding the machines and buffers are the same as in Chapter 2. Assumptions about material delivery and zoning configuration are as follows:

1) Parts from central docking area are delivered to the line-side buffers of the general assembly system by drivers.

2) The capabilities of the drivers in MH system are identical and each line-side buffer is eligible to be assigned to any one of them. All line-side buffers must be assigned to the drivers and a driver's working zone is fixed and non-overlapping.

3) All drivers are idle at the central docking area at the beginning of the shift. The drivers follow a delivery policy for delivering parts. For simplicity, during the initial zoning formation, we will only use the re-order point (RP) and FCFS policy as described in chapter 2.

4.2.1.2 Problem definition

We need to find an assignment of the line-side buffers to the given number of drivers to form fixed working zones, which means no overlapping for different driver zoning assignments. Mathematically, an assignment A is to assign the set B of all line-side buffers into D disjointed zones, one for each driver. For assignment $A = (A_1, A_2, ..., A_D)$, $A_i \cap A_{i'} = \varnothing, \forall i, i' = 1, 2, ..., D, i \neq i'$ and $\bigcup_{i=1}^{D} A_i \subseteq B$, where B is the set of all line side buffers. For example, in the production system shown in Figure 4-1, the assignment of the line-side buffers to the drivers is $A = (A_1, A_2)$ since we only have 2 drivers, where $A_1 = \{b_{11}, b_{12}, b_{31}, b_{33}\}$ and $A_2 = \{b_{21}, b_{22}, b_{32}, b_{41}\}$.

Based on our system description and assumptions, the goal of drivers' zoning assignment (DZA) problem is to assign line-side buffers to drivers to form proper working zones such that the throughput is maintained above a desired level TP_0 with minimum material handling related cost, which can be formulated as follows.

Problem DZA: To find a zoning assignment $A=(A_1, A_2, ..., A_D)$ of line-side buffers B to D drivers in the production system:

$$
\begin{aligned}
&\text{Min } C^{\pi} \\
&s.t.: TP^{\pi} \geq TP_0 \\
&\quad \bigcup_{i=1}^{D} \mathrm{A}_i = \mathrm{B} \\
&\quad A_i \cap A_{i'} = \varnothing, \forall i, i' = 1, 2, ..., D, i \neq i'
\end{aligned}
\tag{4.1}
$$

Where π is the delivery policy, B is the set of all line-side buffers, *C* is the material handling related cost, which has three parts as discussed in section 3.3, i.e., production loss due to starvation penalty of line-side buffers, driver labor cost, and the transportation cost.

4.2.2 Algorithm Design for Initial Zoning in DZA

In this section, we will develop an algorithm to form initial zoning. First, some definitions will be given for algorithm development. Second, a necessary condition for the DZA problem to have a feasible solution will be discussed. Then, two structural characteristics of the DZA problem will be considered for algorithm design: (1) the similarity between the DZA problem and the Parallel Machine Scheduling (PMS) problem; and (2) the monotonicity property of the system throughput and the MH related cost with respect to the partial assignment. After that, we will design the algorithm to solve the initial zoning in DZA problem.

4.2.2.1 Definitions

Definition 4.1: Workload of a line-side buffer is the average service time for the line-side buffer per finished product in the steady state of the production system.

When the production system is in the steady state, in a period of time *T*, the expected total service time for $b_{ij} \in B$ is $W_{ij} \cdot RTT_{ij}$, where W_{ij} is the expected total number of trips for b_{ij} in the period of time *T*. Since the production system is in the steady state, total number of parts delivery for, b_{ij}, $W_{ij} \cdot QTY_{ij} = TP^{\pi}(A) \cdot T \cdot USG_{ij}$ holds, where $TP^{\pi}(A)$ is the system throughput under zoning assignment A and delivery policy π. Thus the total service time for b_{ij} in the period of T is $TP^{\pi}(A) \cdot T \cdot (USG_{ij} \cdot RTT_{ij} / QTY_{ij})$.

Therefore workload w_{ij} of buffer b_{ij} in the steady state can be derived as

$$w_{ij} = \frac{USG_{ij} \cdot RTT_{ij}}{QTY_{ij}} \text{ ,b}_{\text{ij}} \in \text{B} \tag{4.2}$$

Definition 4.2: Drivers d_i's workload is his/her average service time for parts delivery per finished product in the steady state, which is derived as

$$\Omega_i(A) = \sum_{b_{ij} \in A_i} \frac{USG_{ij} \cdot RTT_{ij}}{QTY_{ij}}, i = 1, 2, ..., D \tag{4.3}$$

Driver d_i's utilization Γ is his/her average service time for parts delivery per unit time in the steady state of the system.

$$\Gamma = TP^{\pi}(A) \cdot \Omega_i(A) \tag{4.4}$$

4.2.2.2 Necessary condition for feasible zoning assignment

In the general assembly system, we may not find a feasible assignment solution if there are too few drivers delivering parts for the line-side buffers. In this case, the number of drivers D is infeasible. Since there is no efficient way to identify the minimum number of drivers needed in the material handling system, $D_{\min}$ must be estimated so that there is no need to solve the problem with $0 \le D < D_{\min}$. It is clear that all driver utilizations in the steady state of the system are not beyond 1, so that we have $TP^{\pi}(A) \cdot \Omega_i(A) \le 1$, i=1, 2,…, D. It can be viewed as an implicit constraint in the DZA problem. Since we know $TP^{\pi}(A) \ge TP_0$, we derive $TP_0 \cdot \sum_{b_{ij} \in A_i} w_{ij} \le 1, \forall i = 1, 2, ..., D$

Therefore the optimal value of $D_{\min}$ can be obtained as follows.

$$
\begin{aligned}
&\min \quad \mathrm{D} \\
&s.t.: \sum_{b_{ij} \in A_i} w_{ij} \le \frac{1}{TP_0}, \forall i = 1, 2, ...D \\
&\qquad \bigcup_{i=1}^{D} \mathrm{A}_i = \mathrm{B} \\
&\qquad A_i \cap A_{i^{'}} = \varnothing, \forall i, i^{'} = 1, 2, ..., D, i \ne i^{'}
\end{aligned}
\qquad (4.5)
$$

Clearly, the above problem is an optimization problem which is NP-complete (Garey & Johnso, 1979). However, we can provide another lower bound which can be easily calculated. $TP_0 \cdot \sum_{b_{ij} \in B} w_{ij} \le D$, where B is the set of all line side buffers, the lower bound of $D_{\min}$ is

$$\tilde{D}_{\min} = \left\lceil TP_0 \cdot \sum_{b_{ij} \in B} w_{ij} \right\rceil \qquad (4.6)$$

It should be pointed out that $\tilde{D}_{\min} \le \tilde{D}_{\min}^{optimal}$, $\tilde{D}_{\min}^{optimal}$ is a tighter lower bound. However, $\tilde{D}_{\min}$ is preferable in practice, because it can be easily calculated and in many cases it is equal or much closer to $\tilde{D}_{\min}^{optimal}$. Therefore, for a fixed D number of drivers, $D \ge \tilde{D}_{\min}$ is a necessary condition for the DZA problem to have feasible solutions.

4.2.2.3 Similarity between DZA and PMS

In this section, we will address the similarities and differences between the DZA problem and the Parallel Machine Scheduling (PMS) problem. Let us first describe the PMS problem.

PMS is stated as: Is there a schedule of assigning n tasks with processing time $L_j > 0$ for the j-th task, j=1, 2,…, n, onto k parallel processors, such that the makespan (latest completion time of all tasks) is no more than a given time c_0?

Problem PMS: Does there exist a schedule $S = \{S_1, S_2, ..., S_k\}$

$$s.t.: \max_{1 \le i \le k} \sum_{j \in S_i} L_j \le c_0$$

$$\bigcup_{i=1}^{k} S_i = \mathrm{T}$$

$$\mathrm{S}_i \cap S_{i'} = \varnothing, \forall i, i' = 1, 2, ..., D, i \ne i' \qquad (4.7)$$

Where S_i is the set of tasks scheduled onto the i-th processor and T is the set of all tasks.

It is clear that the basic instances of the PMS problem are tasks. Thus, from the PMS problem point of view, we can rebuild its corresponding DZA problem as follows: The production system consists of a single reliable machine (reliable implies the machine's mean time between failures (MTBF) is infinite), line-side buffers and drivers. The system is shown in Figure 4-2. For every line-side buffer, the average usage rate, the replenishment quantity, and the round trip time are deterministic.

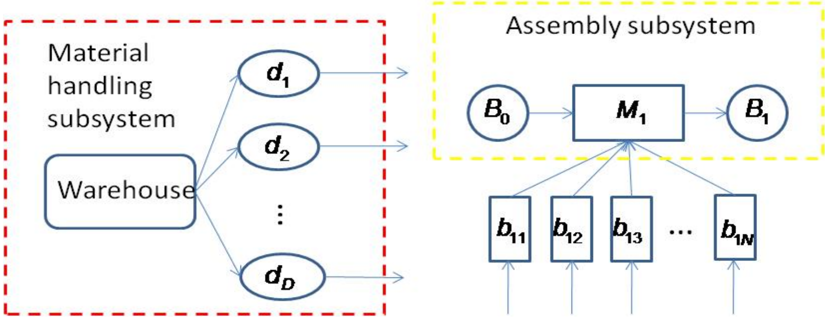

Figure 4-2: A single reliable machine with material handling system

The corresponding DZA problem is defined as a single reliable machine (infinite MTBF) with N line-side buffers and D drivers. We can link these two problems with following settings:

$$D = k, \tag{4.8}$$

$$N = n, \tag{4.9}$$

$$USG_{1j} = 1,\ j = 1, 2, ..., N \tag{4.10}$$

$$QTY_{1j} = 1,\ j = 1, 2, ..., N \tag{4.11}$$

$$RTT_{1j} = L_j,\ j = 1, 2, ..., N \tag{4.12}$$

$$P_1 = \max_{1 \le i \le k} \sum_{j \in S_i} L_j\ ,\ j = 1, 2, ..., N \tag{4.13}$$

$$TP_0 = \frac{1}{c_0} \tag{4.14}$$

$$RT = 0 \tag{4.15}$$

where $USG_{1j}, QTY_{1j}, RTT_{1j}$ are, respectively, the average usage rate, the replenishment quantity per trip, and the round trip time for line-side buffer b_{1j}, *j=1,2,…,N.* We also define the desired throughput TP_0, the cycle time of the machine P_1*,* and the re-order point RT in the material handling system.

Clearly, PMS can be regarded as a special case of the DZA problem with a single reliable machine. It is useful to adopt any existing PMS methods in solving our DZA problem. In the single-machine system in Figure 4-2, it is clear that drivers and line-side buffers can be, respectively, regarded as “parallel processors” and “tasks” from the PMS point of view, and *RTT* (average round trip time) as the “processing time” of the “task”. However, in a generalized production system, *RTT* cannot be regarded as the processing time anymore, because in general, frequencies of driver parts delivery are not identical for all line-side buffers.

However, the workloads of the line-side buffers as defined in 4.2.2.1 can be regarded as the “processing time”, from the PMS point of view. Since drivers' workloads are $\Omega_i(A) = \sum_{b_{ij} \in A_i} w_{ij}, i = 1,2,...,D$, which implies that in DZA, drivers and line-side buffers can be still viewed as “parallel processors” and “tasks” in the PMS problem, while the workload of the line-side buffer can be viewed as the “processing time” of the task on the processor.

From the discussion above, we can see that the DZA problem is harder than the PMS problem in two aspects: the PMS problem is a special case of the DZA problem and the feasibility checking of the DZA problem will be time-consuming due to system complexity. However, it is worthwhile to borrow the concepts of PMS with existing algorithms to solve our problem. The Longest Processing Time (LPT) algorithm, which was first developed by Graham (1966), is well-known and effective for solving the PMS problem. The essence of the LPT algorithm, sorting all tasks in decreasing order of their processing times and balancing total processing times on the processors, makes it an asymptotically optimal algorithm for the PMS problem (Coffman & Lueker, 1991). Similarly, the LPT algorithm can be helpful in sorting all line-side buffers in decreasing order of their workloads and assigning them to drivers to balance driver workloads in our DZA problem.

4.2.2.4 Monotonicity property of the throughput and MH related cost

In this subsection, we will investigate the monotonicity of the system throughput in the process of assigning line-side buffers to drivers, which can help accelerate our problem solving speed, like the "branch and bound" method (Balakrishnan et al., 1991).

Proposition 4.1: In the material handling system, for all line-side buffers, if the release times of all the parts are not delayed, then the arrival times of all finished products in-process buffer will not be delayed.

Lemma 4.1: In the material handling system, for all line side buffers, if the release times of all the parts are not delayed, then the throughput will not drop.

Lemma 4.2: In the material handling system, under partial assignment $A = (A_1, A_2, ..., A_D)$, drivers follow a delivery policy π. For another partial assignment $\hat{A} = (\hat{A}_1, \hat{A}_2, ..., \hat{A}_D), \hat{A}_i = A_i \setminus \{b_{kl}\},$ where $b_{kl} \in A_i$ and $A_j = \hat{A}_j, \forall j = 1, 2, ..., D, j \neq i$, ∃ a policy $\tilde{\pi}$ such that $\mathrm{TP}^{\pi}(\mathrm{A}) \leq \mathrm{TP}^{\tilde{\pi}}(\hat{\mathrm{A}})$

The above proposition and two lemmas are proved in (Yan et al., 2010)

Assumption: There exists an optimal delivery policy π^* for the part delivery, i.e., under any partial assignment A, $\forall\ \pi, \mathrm{TP}^{\pi^*}(\mathrm{A}) \geq \mathrm{TP}^{\pi}(\mathrm{A})$

Theorem 4.1: In the material handling system, under partial assignment $A = (A_1, A_2, ..., A_D)$, drivers follow a delivery policy π. For another partial assignment $\hat{A} = (\hat{A}_1, \hat{A}_2, ..., \hat{A}_D), \hat{A}_i = A_i \setminus \{b_{kl}\},$ where $b_{kl} \in A_i$ and $A_j = \hat{A}_j, \forall j = 1, 2, ..., D, j \neq i$, the throughput will not drop, i.e., $\mathrm{TP}^{\pi^*}(\mathrm{A}) \leq \mathrm{TP}^{\pi^*}(\hat{\mathrm{A}})$, while the material handling related cost will not increase, , i.e., $\mathrm{C}^{\pi^*}(\mathrm{A}) \leq \mathrm{C}^{\pi^*}(\hat{\mathrm{A}})$.

Proof: It is noted that different from comparing the throughput with different dispatching policy $\pi, \tilde{\pi}$ in Lemma 4.2, we are dealing with the same optimal delivery policy π^* for throughput comparison in Theorem 4.1. In terms of Lemma 4.2, there exists a feasible delivery policy $\tilde{\pi}$ such that $\mathrm{TP}^{\pi}(\mathrm{A}) \leq \mathrm{TP}^{\tilde{\pi}}(\hat{\mathrm{A}})$. Due to π^* is an optimal delivery policy, we have $\mathrm{TP}^{\widehat{\pi^*}}(\hat{\mathrm{A}}) \leq \mathrm{TP}^{\pi^*}(\hat{\mathrm{A}})$. Thus, $\mathrm{TP}^{\pi^*}(\mathrm{A}) \leq \mathrm{TP}^{\pi^*}(\hat{\mathrm{A}})$ holds, while under the same number of drivers, which indicates that $\mathrm{C}^{\pi^*}(\mathrm{A}) \leq \mathrm{C}^{\pi^*}(\hat{\mathrm{A}})$. It completes the proof.

Theorem 4.1 indicates that, under the optimal delivery policy, the throughput will not increase and the MH related cost will not drop if more additional line-side buffers are assigned to drivers, which implies that if a partial assignment is infeasible, all partial assignments derived by assigning more line-side buffers to the drivers based on the infeasible one remains infeasible. Therefore we do not need to proceed under this partial assignment. Obviously, the monotonicity property in Theorem 4.1 can largely reduce the search space to improve the search efficiency in the algorithm design.

4.2.2.5 Algorithm Design

In this part, we will present a two-phase sequential zoning assignment (SZA) algorithm to solve our DZA problem. It is note that adopting the concept from Yao et al. (2010), our algorithm can be viewed as an extension, however with the consideration of find a feasible solution with minimum MH related cost, rather than simply finding a feasible solution, our algorithm shall outperform the original one, which will be compared in numerical examples. Our algorithm design is based on two main phases: The first is the efficient search of feasible zoning assignments inspired by the PMS algorithms; the second is the use of backtracking techniques to avoid type I, II errors, inspired by the Nested Partitions Algorithm (Shi & Ólafsson, 2009), given the complicated system dynamics and the throughput evaluation noises.

In the first phase, we take advantages of the similarity between the DZA problem and the PMS problem (demonstrated in Section 4.2.3.3) in order to adopt the idea of PMS algorithms to solve the DZA problem. It should be pointed out that in the process of assigning line-side buffers to drivers to form working zones, the throughput constraint must be checked (by simulation) under the partial assignment. Infeasible partial assignments will not be considered due to the monotonicity property of the throughput and material handling related cost.

If the first phase obtains a feasible solution, but the feasibility probability is not high enough, we will start the second phase to check whether the solution is statistically feasible. In the second phase, backtracking is introduced. We adopt the Nested Partitions Algorithm idea of exploiting promising sub-regions (i.e., feasible child partial assignments). If some sub-regions are promising they should be backtracked to their super-region (i.e., the parent partial assignment) (Shi & Ólafsson, 2009). We will introduce randomness to help the algorithm to avoid being trapped in a neighborhood of an infeasible zoning assignment while the problem is feasible. The algorithm is described as follows.

Initialization

1) Calculate the workload w_{ij} of line side buffer $b_{ij} \in B$ and define the set of unassigned line-side buffers $\cup = \{b_{(1)}, b_{(2)}, ..., b_{(N)}\}$, which is a list of all line-side buffers sorted in decreasing order of their workloads, where N is the total number of line-side buffers. For the initial stage, we set TP_0, and calculate the lower bound $\overset{\square}{D}_{\min}$ of the minimum number of drivers. If the given number of drivers D< $\overset{\square}{D}_{\min}$, report that no feasible assignment exists for D; otherwise, go to Step 2.

2) Initialize the current partial assignment $A = A_0$, the current minimum material handling related cost $C_{\min} = C_0$, accumulated visited times $n_A = 0$ and driver workloads $\Omega_i(A)$ =0, i=1,2,…,D. Set two small positive values ε , η , the maximum number of the iterations r, a desired probability P^* that observed feasible zoning assignment are statistically feasible, and a sufficiently large n as the maximum number of random transitions. Go to step 3.

Phase One (Assigning buffers to balance the workload of drivers)

3) Assign the first buffer $b_{(j)}$ in U to driver d_i based on the current partial assignment $A = (A_1, A_2, ..., A_D)$, calculate sample mean of the throughput and material handling cost $\overline{TP}^{\pi}(A_i)$, $\overline{C}^{\pi}(A_i)$, under the new child partial assignment $A_i = (A_1, A_2, ..., A_i^{'}, ..., A_D)$ of A through simulation (Chang et al., 2012), where $A_i^{'} = A_i \cup \{b_{(j)}\}$. Accordingly, A will be called as the parent of A_i. If $\overline{TP}^{\pi}(A_i) < TP_0$ for i=1,2,…,D, go to Step 5; otherwise, go to Step 4.

4) Remove the first $b_{(j)}$ from U. Let $i^{'} = \arg\min_{i \in I} \Omega_i(A)$, where I is the set of indices of all observed feasible partial assignment A_i. Calculate $\Omega_{i^{'}}(A_{i^{'}}) + w(j)$ and replace *A* by $A_{i^{'}}$. If U is not empty, go to Step 3, otherwise increase n_A by 1, calculate P_A and go to Step 6 if $P_A \ge P^*$ or go to Step 5 if not.

Phase two (randomly assigning with backtracking to avoid type I, II errors)

5) Evaluate the throughput under each child $A_{i^{'}}$ of the current partial assignment *A* by simulation and then obtain the sample mean $\overline{TP}^{\pi}(A_i)$. If

$\overline{TP}^{\pi}(A_i) < TP_0, \forall i \in \{1,2,...,D\}$, let the backtracking probability $P_b = 1-\varepsilon$, otherwise $P_b = \varepsilon$ .Generate a random number ξ following Uniform (0, 1). If there exists some $i^{'} \in \{1,2,...,D\}$ such that $((i^{'}-1)(1-P_b)/D) < \xi \le (i^{'}(1-P_b)/D)$ and $\overline{TP}^{\pi}(A_{i^{'}}) \ge TP_0$, replace A by $A_{i^{'}}$, otherwise replace A by its partial assignment $A^{'}$. Adjust list U accordingly. If U is empty, increase n_A by 1, calculate P_A, decrease n by 1. If $P_A \ge P^*$ or n=0, go to Step 6; otherwise, go to Step 5.

Output

6) If $\max_{A \in A_0} n_A > 0$, $C^{\pi}(A) < C_{\min}$, denote $C^{\pi}(A)$ as the new $C_{\min}$, if the difference between new $C_{\min}$ and the previous $C_{\min}$ is less than η , or the number of iterations reaches the maximum value r, stop and output the current complete assignment; otherwise, go to step 2 and start our iteration from the beginning .

Remark 4.1: In this algorithm, ε is introduced so that the algorithm can visit all assignments with positive probability. If ε is too small, the algorithm may be trapped in a neighborhood of an infeasible zoning assignment; if it is too large, it will take a long time for the algorithm to visit an assignment in Phase two. Thus $\varepsilon \in ((1/10(D+1)),(1/(D+1))$ would be appropriate.

Remark 4.2: The monotonicity property of the throughput and the MH related cost can help to reduce the searching space and computational efforts in the first phase of the algorithm.

4.2.3 A Toy Model Case Study

We illustrate the Sequential Zoning Assignment (SZA) with Backtracking (SAB) algorithm using a simple production system, which consists of two machines, one in-process buffer (excluding the input and output in-process buffers), and five line-side buffers as described in Yao et al. (2010), as shown in Figure 4-3. Parameters of the machines and line-side buffers are displayed in Tables 4-1, 4-2. We set the desired throughput $TP_0 = 0.96TP_{\max}$, where $TP_{\max}$ is the ideal throughput. Failure times and repair times of machines are independently and exponentially distributed, and drivers follow the re-order point delivery policy and re-order time is 15 minutes. In the algorithm, the system throughput and the MH related cost are evaluated by a simulation model (Chang et al., 2012). 10,000 minutes running time with 2,000 minutes warm-up time of the production is simulated and the simulation is run for 20 replications.

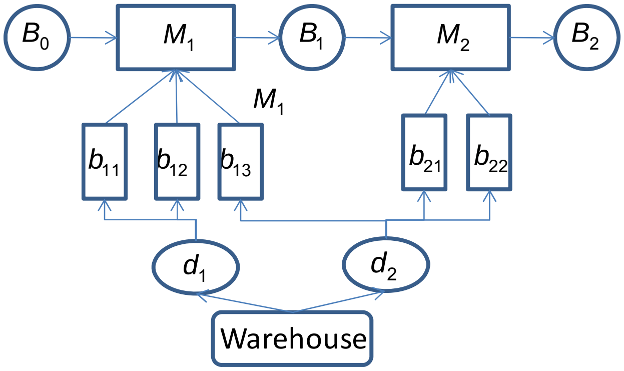

Figure 4-3: The layout of the example line

	MTBF(min)	MTTR(min)	Cycle time(min)
M_1	100	0.5	1
M_2	50	0.4	1.2

Table 4-1: Parameters of the machines

b_{ij}	USG_{ij}	QTY_{ij}	RTT_{ij} (min)	Capacity for b_{ij}
b_{11}	0.75	18	18	100
b_{12}	0.27	7	8.4	100
b_{13}	0.6	20	18	100
b_{21}	0.1	8	10	100
b_{22}	0.17	4	10	100

Table 4-2: Parameters of the line-side buffers

The process of the algorithm is as follows. The workloads of the line-side buffers are calculated based on data shown in Table 4-2. The total workload of the line-side buffers is $\sum_{b_{ij} \in B} w_{ij} = 2.164$ (min), $TP_{\max} = 49.49$ (jobs/hr), Thus, the desired throughput level is $TP_0 = 0.96 TP_{\max} = 47.51$, then we can calculate the lower bound of the minimum number of drivers using (4.6), $D_{\min} = \lceil 1.71 \rceil = 2$. We set $\varepsilon = 0.05, \eta = 0.05, P^* = 0.95, n = 20, r = 15$, and with these initial conditions, we proceed to the first phase of the straightforward search for a feasible zoning assignment. Results of the zoning assignment process in Phase one are summarized in Table 4-3, in which the line-side buffers are listed in decreasing order of their workloads. The process of workload balancing is shown in Table 4-3. In Phase one, the algorithm finds an assignment, shown in Table 4-3, since the feasibility probability of the zoning assignment is $0.972 > P^*$ which means the feasibility probability is high enough, the solution is statistically feasible, the algorithm will not proceed to phase two and stop. Thus, $A = (A_1, A_2)$ where $A_1 = \{b_{11}, b_{12}\}, A_2 = \{b_{13}, b_{21}, b_{22}\}$.

Under this zoning assignment, the estimated system throughput is 47.51 (jobs/hr) with 95% confidence interval (47.32, 47.70) and the driver utilizations are 0.86 and 0.85 respectively. Though SZA is not efficient for large scale problems, we can consider adopting its effectiveness for initial zoning formation.

b_{ij}	w_{ij}	A	TP^{π}
b_{11}	0.75	$A_1 = \{b_{11}\}, A_2 = \varnothing$	49.61
b_{13}	0.54	$A_1 = \{b_{11}\}, A_2 = \{b_{13}\}$	49.61
b_{22}	0.425	$A_1 = \{b_{11}\}, A_2 = \{b_{13}, b_{22}\}$	48.86
b_{12}	0.324	$A_1 = \{b_{11}, b_{12}\}, A_2 = \{b_{13}, b_{22}\}$	48.55
b_{21}	0.125	$A_1 = \{b_{11}, b_{12}\}, A_2 = \{b_{13}, b_{21}, b_{22}\}$	47.51

Table 4-3: The process of drivers' zoning

4.3 Zoning Optimization Based on Particle Swam Optimization (PSO)

4.3.1 Problem Formulation

DZA optimization assigns all line-side buffers into several working zones and assigns each zone to specific drivers, who will be only responsible for their working zones. In previous discussions of initial zoning, we assume a non-overlapping zoning configuration, which means each driver covers only some specific line-side buffers with no overlapping. However, in reality, drivers tend to do team coverage; so both non-overlapping and overlapping zoning should be taken into consideration. Similarly, our goal is to achieve minimum MH related cost utilizing an efficient systematic approach to assign drivers to their working zones while sustain a desired level of throughput. We should note that the cost of cross-training for overlapping is not considered in our research.

The optimization procedure is described as follows:

Objective function:

Min C^{π} ,

where π is the delivery policy, *C* is the material handling related cost as discussed in section 4.2

Subject to:

1) Current number of drivers
2) Dispatching policy: RP or Integrated
3) Number of drivers covering each zoning area: one or two
4) Delivery order strategy: FCFS or Integrated
5) Maximum number of dolly trains: 2, 3

Starting from the estimated minimum number of drivers as described in 4.2.2.2, an optimal zoning configuration can be achieved by finding out the minimum MH related cost, since under the same production environment, less overall MH related cost indicates higher driver efficiency and lower impact from the MH system. After a certain number of iterations using proper searching rules, we can obtain a converged MH related cost. If the assignment solution meets throughput requirements, the iteration should be terminated, and the current number of drivers and zoning configuration is the assignment solution, otherwise we increase the number of drivers and return to the very beginning of the zoning optimization process.

When delivering one dolly per trip, workload and zoning configuration can be estimated as described in 4.2. However, with multiple dollies per trip, it becomes difficult and time consuming due to complexity. An appropriate and efficient searching process should be utilized.

4.3.2 Alternative Initial Zoning Algorithm (Absenteeism)

Actually, before we investigate the SZA algorithm to form proper initial zones, we simply developed Absenteeism method to serve the same purpose, inspired by routes construction based initialization heuristics (Kennedy & Eberhart, 2001).

This method focuses on scenarios where some drivers do not come to work due to vacation or sickness. Assuming there is no back up drivers, we need to balance the work of other drivers such that the production will continue and tasks are fairly assigned to the drivers.

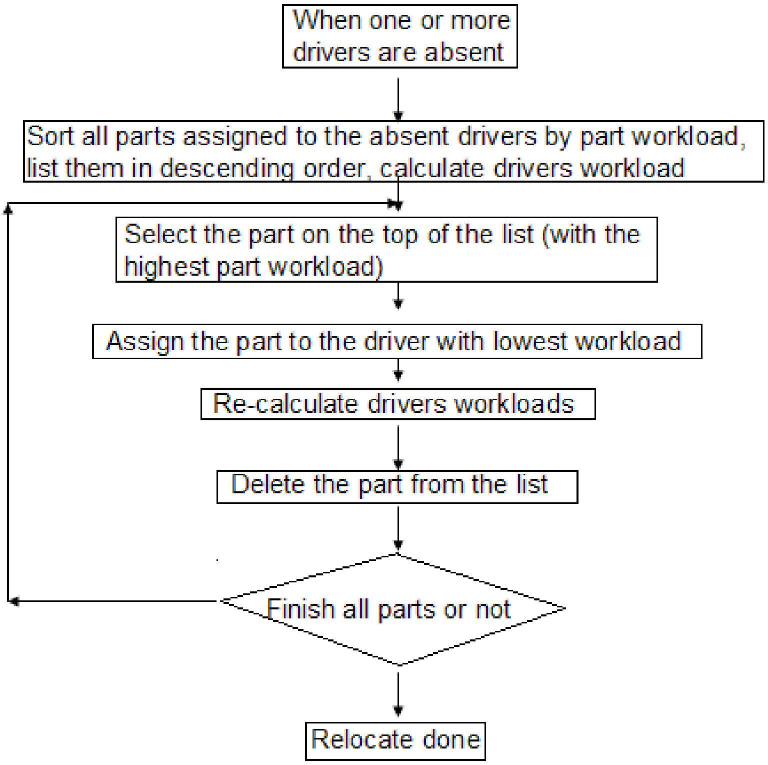
When one or more drivers are absent
Sort all parts assigned to the absent drivers by part workload, list them in descending order, calculate drivers workload
Select the part on the top of the list (with the highest part workload)
Assign the part to the driver with lowest workload
Re-calculate drivers workloads
Delete the part from the list
Finish all parts or not
Relocate done

Figure 4-4: Relocating work of absent drivers

The flow chart in Figure 4-4 is based on balancing the workload. Note that although this algorithm is derived mainly for single dolly train (one dolly per trip), it can be easily extended to multi-dolly case. Surely, by utilizing the system characteristics of DZA, SZA initial zoning method can be expected to outperform the Absenteeism method, but to what extent, we will compare and quantify the difference in the numeric example section.

4.3.3 Meta-Heuristic Searching Algorithm PSO Design

In this section, we will utilize Particle Swarm Optimization (PSO) to obtain converging solution of zoning configuration with minimum MH related cost. The PSO algorithm is an adaptive algorithm: a population of individuals adapts by returning stochastically toward previously successful regions in the search space, and is influenced by the successes of their topological neighbors. It was originally proposed for optimizing hard numerical functions. PSO can be easily implemented and it is computationally inexpensive, since its memory and CPU speed requirements are low. Also, it does not require gradient information of the objective function under consideration and it uses only primitive mathematical operators. Moreover, our PSO optimization process can be easily incorporated with our dispatching and routing approaches, since they share the same objective function to achieve the minimum MH related cost. PSO has been proved to be an efficient method for many optimization problems, such as Design Combinational Logic Circuits, Evolving Artificial Neural Networks, Multiple Object Problems and Travelling Salesman Problem.

In this section, a global version of the PSO algorithm will be developed based on (Ricca & Simeone, 2000). To achieve the global optimum, each particle will move towards its best previous position and towards the best particle in the whole swarm. The global version PSO algorithm can be described as follows:

Let $f : R^m \to R$ be the objective function. Let there be n particles, each with associated positions $A_i \in R^m$ and velocities $v_i \in R^m, i = 1,...,n$. Let $\overline{A}_i$ be the current best position of each particle and let $\hat{g}$ be the global best. Initialize A_i and v_i for all i. One common choice is to take $A_{ij} \in U[a_j, b_j]$ and v_i=0 for all i and j = 1,…,m, where a,b are the limits of the search domain in each dimension $\overline{A_i} \leftarrow A_i$ and $\hat{g} \leftarrow arg\,\min\limits_{A_i} f(A_i)$, i =1,…,n

Basic logic:

```
While not converged:
    For 1<=i<=n
        $A_i \leftarrow A_i + v_i$
        $v_i \leftarrow \omega v_i + c1r1 \circ (\overline{A_i} - A_i) + c2r2 \circ (\hat{g} - A_i)$
        If $f(x_i) < f(\hat{x}_i),\ \overline{A_i} \leftarrow A_i$
        If $f(x_i) < f(\hat{g}), \hat{g} \leftarrow A_i$
```

ω is an inertial constant , good values are usually slightly less than 1. c1 and c2 are constants which indicate how much the particle is directed towards good positions. They represent a "cognitive" and a "social" component, respectively. They will affect how much the particle's personal best and the global best influence its movement. Usually we take, c1, c2 round 1~2. In the beginning of the whole searching process, we will select ω=c1=c2=1 for simplicity. r1, r2 are two random vectors while each component of these two vectors generally is a uniform random number between 0 and 1, detailed tuning processes were discussed in (Ricca & Simeone, 2000).

Initialization

For complicated optimization problem, such as DZA, initialization methods have great effects on the result. Kennedy & Eberhart (2001) discussed routes construction based initialization heuristics for preliminary search. To get a jump start, we need a proper initial zoning, which is essential for convergence effectiveness using PSO (Pan et al., 2008). Both SZA and Absenteeism methods are used to construct initialization configuration.

Solution

We will use PSO to build a proper zoning configuration (Pan et al., 2008) based on our initial zoning. A_i is the configuration of each zone, $\overline{A_i}$ is the local best for A_i according to the lowest MH related cost for driver i specifically responsible for zone A_i. $\hat{g}$ is the global best for all zones to achieve Min C^{π}, where corresponding column of $\hat{g}$ indicates the recorded best configuration of A_i. v_i is the changing factor of zone A_i, indicating which part should be added into A_i, or moved out from A_i. Nearest Neighbor searching algorithm (Kennedy & Eberhart, 2001), is used to move the parts for feasible solutions, based on priori ordering of the parts. All the dispatching and routing procedures are based on our discussion in Chapter 3.

4.4 Numerical Results

The two-stage algorithm will be demonstrated in this case study. Both the Absenteeism methods and Sequential Zoning Assignment (SZA) algorithm are used to form the initial zoning, and then PSO is applied to solve the DZA problem in a real production system with integrated policy and 2 maximum dollies. The structure of the real system is the same as that of the system in (Chang et al., 2012). In the original system, more than 300 line-side buffers are assigned to 16 drivers by trial and error methods based on engineer's experience. The simulation model used to evaluate the system throughput is based on the method in (Chang et al., 2012). 10,000 minutes (about 21 subsequent shifts) with 2,000 minutes warm-up time of the production is simulated and the simulation is run for 20 replications. Under the original zoning assignment in the plant, the ideal throughput that all line-side buffers have sufficient part supply is 30.42(jobs/hr), and $TP_0 = 0.9TP_{\max} = 27.42$.

First, we calculate the lower bound of the minimum number of drivers in the material handling system. The total workload of line-side buffer is 24.72 (min), thus the lower bound of the minimum number of drivers is $D_{\min} = \lceil 27.42 \times 24.72/60 \rceil = \lceil 11.32 \rceil = 12$. In initial zoning, we set $\varepsilon = 0.01, \eta = 0.05, \mathrm{P}^* = 0.99, r = 15$ and $\mathrm{n} = 1{,}000$.

The two stages algorithm is implemented by Visual Studio 2010 and runs on a PC with 2.80 GHz CPU, 4.00 GB RAM. In initial zoning stage, the SZA algorithm provides a zoning assignment of feasibility probability $\mathrm{P_A}$=0.956 and stops in the total of 8 iterations, which implies the SZA algorithm can find a feasible zoning assignment with the minimum number of drivers 12. Then PSO has been implemented, the performance of the system under the SZA+PSO algorithm is shown in Table 4-4. In this case, the developed algorithm provides a feasible zoning assignment for $\mathrm{D} = 12,13,14,15,16$.

Zoning Assignment	D	$\mathrm{TP}^{\pi}(\mathrm{A})$
SZA+PSO	12	27.87 ± 0.06
	13	28.65 ± 0.07
	14	29.33 ± 0.07
	15	29.68 ± 0.05
	16	30.01 ± 0.05
Original zoning	16	30.00 ± 0.05

Table 4-4: Comparison of zoning assignments

Maximum # iteration	Total # of failures in PSO		
	Without Initial Zoning	With Absenteeism	With SZA
100	54	34	11

Table 4-5: Convergence effectiveness comparison

From Table 4-4, we can see that the SZA+PSO algorithm finds a feasible zoning assignment with 12 drivers, which takes about 1.52 hrs. Compared with the zoning assignment implemented in the factory, we can reduce the number of drivers from 16 to 12, and compared with the assignment (Yao et al., 2010), from 13 to 12, while the throughput meets the requirement of throughput $TP_0 = 27.42$. Under the original zoning assignment, the difference among drivers' utilizations could be as large as 44.8%, while it is lower than 8.3% under the zoning assignments provided by the SZA+PSO algorithm. Because of the algorithm's capability of balancing drivers' utilizations, fewer drivers are needed to achieve the required throughput in the production system.

From Table 4-5, we can see the effectiveness of SZA as an initial zoning method, for maximum 100 iterations, the failure rate of PSO algorithm to find feasible solutions is significantly reduced with SZA method, compared with Absenteeism method and no initial zoning.

4.5 Conclusions

In this chapter, we first investigated the initial zoning formation. The fixed zoning version of the DZA problem was formulated, and solved by the Sequential Zoning Assignment (SZA) algorithm, which is developed based on two structural characteristics of the problem: the similarity between the DZA problem and the Parallel Machine Scheduling (PMS) problem; the monotonicity property of the throughput in the process of assigning line-side buffers to drivers. Then Particle Swan Optimization (PSO) was implemented based on the initial zoning. The two-stage algorithm was tested, and the results demonstrated that it is effective to solve a practical DZA problem.

CHAPTER 5
CONCLUSIONS AND FUTURE WORK

5.1 Conclusions

The major academic contributions of this research include: 1) Development of an integrated modeling approach with a modified max-plus linear system to accurately estimate the timing information of a production system with coupled general assembly and material handling sub-systems. 2) Development of an integrated dispatching and routing model for the multi-dolly material handling system to achieve minimum MH related cost. And 3) Development of a 2-stage algorithm for drivers' zoning assignment (DZA) with sequential zoning assignment (SZA) and particle swam optimization (PSO).

The main achievements and limitations of this research are summarized as follows:

First, mathematical representations have been derived for a simple two-machine and one-buffer system with finite buffer and blockage & starvation mechanism. This formulation is further extended to multi-machine and multi-buffer systems. We modify the traditional max-plus algorithm by using a "switching" mechanism to deal with concurrency and non-deterministic production interruption. Results showed that our modeling approach can take advantage of max-plus algebra for its quick computational efficiency through matrix manipulation to accurately estimate and predict the timing information. However, some important aspects in a production system, such as machine reliability and product quality may not be reflected easily through our modeling technique. Due to the nature and limitation of max-plus algebra itself, no analytical solutions for system performance such as throughput can be derived directly.

Second, a framework of the multi-dolly material handling system has been established to coordinate and optimize the MH process. Based on dynamic timing information of part replenishment requests derived from the max-plus modeling approach, several dispatching policies have been proposed to improve the performance of the multi-dolly MH system. Numerical results showed that our proposed dispatching policies outperform the existing ones in terms of number of delivery trips, overall trip time and MH related cost. Later we discussed the integration of a vehicle routing problem with time window (VRPTW) based routing strategy with our previously proposed dispatching policies to further reduce the overall material handling related cost. However, due to the tight dispatching schedule, timely delivery performance was essential, while variations in both processing time and transportation need further investigation.

Finally, to investigate the DZA problem and form initial zoning, DZA with fixed zoning was formulated and solved by the sequential zoning assignment, which was developed because of the similarity between the DZA problem and the parallel machine scheduling (PMS) problem. Based on the initial zoning formation, the meta-heuristic algorithm PSO has been successfully implemented to solve large scale DZA problems in real working conditions, which can sufficiently reduce the number of drivers while maintaining a desired level of throughput, compared with existing zoning assignment methods. The 2-stage approach combines the merits of convergence effectiveness and efficiency from SZA and PSO algorithms, respectively.

5.2 Recommendations for Future Work

Recommendations for future work related to this research are as follows:

Current max-plus linear systems should be further extended to fit more generalized discrete event dynamic systems (DEDS), such as a production system with disassembly or rework operations, and to reflect important parameters, such as system reliability and production quality. Besides, in our research, we utilize a re-order point based material handling system, the estimation of the re-order points is still based on a historical average usage rate of certain parts. However, using real time information of line-side buffer inventory, the utilization of parts can be estimated online, which means the re-order time can be updated according to real utilization instead of fixed intervals (i.e., 15 min) before parts depletion. Due to the uncertainties in processing and in material handling delivery, the investigation of the variances of machine processing time and transportation are also interesting research areas.

Dispatching and routing policies have been investigated to minimize the multi-dolly material handling related cost and impact, the application to more generalized MH systems, such as vehicle routing with soft/hard window problem and flight scheduling, would also be of interest. As we mentioned in the literature review, section 1.2.2, for a simpler system, production sequencing is a possible way to improve productivity and material handling performance. Integrated production sequencing with our dispatching and routing policies might be of interests. Integrating a kitting system in a general assembly system with MH is another interesting topic. It can bring great benefits when space reduction and error prevention are essential, however, at the same time, a kitting system may need extra coordination efforts from MH system, and may lack of robustness. Both advantages and disadvantages are worthy of carefully. Even though expedited delivery services are not discussed in our research, it is of great interests for future study due to its ability to avoid starvation. How to identify service priorities, avoid conflicts and setup expedited routing, how the flow of material, sequence of delivery and MH system dynamics will be influenced by expedited services, these questions need to be investigated. Besides, dispatching and routing policies with stochastic production sequencing is another direction for material handling process. Though it cannot be applied directly in real time operations, it is still of interest, considering it might provide insight and prediction for supply-demand trend changes or new system setup preparation.

For DZA problems, the system modeling and characteristics have been investigated based on a fixed zoning configuration. Though we may solve the flexible zoning problem using the meta-heuristic method PSO, however, the mathematical modeling and formation of flexible DZA are of great value. The proof of the global convergence and the improvement of convergent velocity of both SZA and PSO are also areas of future interest.

REFERENCE

Alexopoulos, K., Papakostas, N., Mourtzis, D., Gogos, P., and Chryssolouris, G., Quantifying the flexibility of a manufacturing system by applying the transfer function. *International Journal of Computer Integrated Manufacturing,* vol. 20, no.6, pp. 538-547, 2007.

Altiok, T., *Performance Analysis of Manufacturing Systems*. Berlin: Springer, 1997.

Asef-Vaziri, A., and Laporte, G., Loop based facility planning and material handling. *European Journal of Operational Research* 164, pp. 1-11, 2005.

Askin, R. G., and Standridge, C. R., *Modeling and Analysis of Manufacturing Systems*. John Wiley & Sons; International Edition, April, 1993.

Baccelli, F., Cohen, G., Olsder, G., and Quadrat, J., *Synchronization and Linearity*. New York: John Wiley & sons, 1992.

Balakrishnan, V., Boyd, S., and Balemi, S., Branch and bound algorithm for computing the minimum stability degree of parameter-dependent linear systems. *Int. J. of Robust and Nonlinear Control*, 1(4), pp. 295–317, October–December, 1991.

Banks, J., Carson, J. S., Nelson, B. L., and Nicol, D. M., *Discrete-Event System Simulation*. Upper Saddle River, NJ: Prentice Hall, 2005.

Becker, B., and Lastovetsky, A., Max-plus algebra and discrete event simulation on parallel hierarchical heterogeneous platforms. *Lecture Notes in Computer Science Proceedings of HeteroPar' 2010,* Ischia-Naples, Italy, Aug 31 - Sep 3, 2010.

Bertsekas, D. P., Tsitsiklis, J. N., *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, 1997.

Bihan, H., and Dallery, Y., A robust decomposition method for the analysis of production lines with unreliable machines and finite buffers. *Annals of Operations Research*, 93, pp. 265–297, 2000.

Bozer, Y. A., and Yen, C. K., Intelligent dispatching rules for trip-based material handling systems. *Journal of Manufacturing Systems*, vol. 15, pp. 226-239, 1996.

Bozer, Y. A., and Ciemnoczolowski, D. D., Performance evaluation of small-batch container delivery systems used in lean manufacturing- part 1: system stability and distribution of container starts. *International Journal of Production Research*, Vol. 51, No. 2, pp. 555–567, January 2013.

Bozer, Y. A., and Srinivasan, M. M., Tandem configurations for automated guided vehicles systems and the analysis of single-vehicle loops. *IIE Transactions*, 23, 72-82, 1991.

Brimson, J. A., *Handbook of Process Based Accounting, Leveraging Processes to Predict Results*. America Institute of CPAs, 2002.

Burkard, R., Dell'Amico, M., and Martello, S., *Assignment Problems*. Philadelphia, PA: SIAM, 2009.

Buzacott, J., and Shantikumar, J. G., *Stochastic Models of Manufacturing Systems*. Englewood Cliffs, NJ: Prentice Hall, 1993.

Cassandras, C. G., and Lafortune, S., *Introduction to Discrete Event Systems*. Boston, MA, Kluwer Academic, 1999.

Chang, Q., Pan, C., Xiao, G., and Biller, S., Integrated Modeling of Automotive Assembly line with Material Handling. *ASME Transctions*, 2012(Accepted)

Ciemnoczolowski, D. D., and Bozer, Y. A., Performance evaluation of small-batch container delivery systems used in lean manufacturing- part 2: number of Kanban and workstation starvation. *International Journal of Production Research*, Vol. 51, No. 2, pp. 568–581, January 2013.

Coffman, J. E. G., Garey, M. R., and Johnson, D. S., An application of bin-packing to multiprocessor scheduling. *SIAM J. Compute*., vol. 7, no. 1, pp. 1–17, 1978.

Coffman, J. E. G., and Lueker, G. S., *Probabilistic Analysis of Packing and Partitioning Algorithms*. New York: Wiley, 1991.

Cohen, G., Dubois, D., and Quadrat, J.P., A linear-system theoretic view of discrete event processes and its use for performance evaluation in manufacturing. *IEEE Transactions on Automatic Control*, AC-30(3), 1985.

Cohen, G., Dubois, D., Quadrat, J.P., and Viot, M., Algebraic tools for the performance evaluation of DES. *Proceeding of the IEEE, Special Issue on Discrete Event Systems*, 1989.

Colorni, A. M., and Maniezzo, V., Distributed optimization by ant colonies. *Proceedings of ECAL91 — European Conference on Artificial Life*, Paris, France, pp. 134–142, 1991.

Cunninghame-Green, R.A., Process synchronization in a steelworks – a problem of feasibility. *Proceedings of the 2nd International Conference on Operational Research* (Aix-en-Provence, France, 1960), J. Banbury and J. Maitland, Eds. London: English Universities Press, pp. 323–328,1961.

Cunninghame-Green, R. A., Minimax algebra, ser. *Lecture Notes in Economics and Mathematical Systems*. Berlin, Germany: Springer Verlag, vol. 166, 1979.

Dallery, Y., and Gershwin, S. B., Manufacturing flow line systems: a review of models and analytical results. *Queueing Systems*, 12, 3-94, 1992.

Das, S. R., Adaptive protocols for parallel discrete event simulation. *The Journal of the Operational Research Society*, vol. 51(4), pp 385-394, 2000.

Driels, M., and Klegka, J. S., Analysis of alternative rework strategies for assembly manufacturing systems. *IEEE Transactions on Components, Hybrids and Manufacturing Technology*, Vol. 14, No. 3, pp 637-644, 1991.

Dueck, G., and Scheuer, T., Threshold accepting: A general purpose optimization algorithm appearing superior to simulated annealing, *Journal of Computational Physics*, 90, 161-175, 1990.

Egbelu, P. J., Pull versus push strategy for automated guided vehicle load movement in a batch manufacturing system. *Journal of Manufacturing System,* 16(3): 271 – 80, 1987.

Fujimoto, R. M., Parallel and distributed simulation systems. *Proceedings of the 2001 Winter Simulation Conference*, pp. 147-157, 2001.

Garey, M. R. and Johnson, D. S., *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco, CA: W.H. Freeman, 1979.

Gaubert, S., Performance evaluation of (max, +) automata, *IEEE Transactions on Automatic Control*, 40(12), 1995.

Gershwin, S. B., *Manufacturing Systems Engineering*. Englewood Cliffs, NJ: PTR Prentice Hall, 1999.

Giffler,B., *Mathematical Solution of Production Planning and Scheduling Problems*, IBM ASDD, Tech. Rep., 1960.

Glover, F., Tabu search - part I. *ORSA Journal on Computing*, 1, 190-206, 1989.

Gondran, M., and Minoux, M., Graphs and algorithms. *Wiley-Interscience Series in Discrete Mathematics, Discrete Mathematics Series*: 1-484, 1984.

Gorilli, P., Manzi ,C., Pennisi, A., Ricca, F., Simeone, B., Evaluation and optimization of electoral systems, *SIAM Monographs on Discrete Mathematics and Applications, SIAM, Society for Industrial and Applied Mathematics*, Philadelphia,1999.

Govil, M. C., and Fu, M. C., Queueing theory in manufacturing: A survey. *Journal of Manufacturing Systems*, vol. 18, pp. 214-240, 1999.

Govind, N., Roeder, T. M., and Schruben, L. W., A simulation-based closed queueing network approximation of semiconductor automated material handling systems. *IEEE Transactions on Semiconductor Manufacturing*, vol. 24, pp. 5-13, 2011.

Graham, R. L., Bounds for certain multiprocessing anomalies. *The Bell System Technical Journal*, vol. 45, no. 9, pp. 1563–1581, 1966.

Harding, J. A., Popplewell, K., Simulation: an application of factory design process methodology. *The Journal of the Operational Research Society*, vol. 51(4),pp. 440-448, 2000.

Hirao, S., and Tamaki, M., Optimal dispatching control of an AGV in a JIT production system. *Production Planning & Control*, 13(8). 746-753, 2002.

Hodgson, T. J., and Wang, D., Optimal hybrid push/pull control strategies for a parallel multistage system: part I. *International Journal of Prod. Res.,* 29(6):1279 – 87, 1991.

Huang, C. C., and Kusiak, A., Manufacturing control with a push-pull approach. *International Journal of Prod. Res.,* 36(1):251 – 75, 1998.

Inman, R. R., and Bulfin, R. L., JIT sequencing mixed model assembly lines. *Management Science*, 37(7), 901-904, 1991.

Inman, R. R., Bhaskaran, S., Blumenfeld, D. E., In-plant material buffer sizes for pull system and level-material –shipping environments in the automotive industry. *International Journal of Production Research*, 35(5), 1213-1228, 1997.

Loannou, G., and Minis, I., A review of current research in manufacturing shop design integration. *Journal of Intelligent Manufacturing* 9, 1, 57 – 72, 1998.

Loannou, G., An integrated model and a decomposition-based approach for concurrent layout and material handling system design. *Computers and Industrial Engineering*, forthcoming, 2007.

Johnson, D., Demers, S. A., Ullman, J. D., Grahanm, R. L., Worst-case performance bounds for simple one-dimensional packing algorithms. *SIAM J. Comput*., vol. 3, no. 4, pp. 299–325, 1974.

Johnson, M. E., and Brandeau, M. L., Stochastic modeling for automated material handling system design and control. *Transportation Science*, vol. 30, pp. 330-350, 1996.

Kennedy, J., and Eberhart, R, C., *Swarm Intelligence*, Morgan Kaufman Publishers, 2001.

Kirkpatrick, S., Gelatt C. D. Jr and Vecchi, M. P., Optimization by simulated annealing. *Science*, 220, 671-680, 1983.

Krivulin, N. K., A max-algebra approach to modeling and simulation of tandem queueing systems. *Mathematical and Computer Modeling*, 1995.

Langevin, A. M., Desrochers, J., Desrosiers, S., Gelinas, F., A two-commodity flow formulation for the traveling salesman and makespan problems with time windows. *Networks* 23 631 – 640, 1993.

Law, A. M., Kelton, W. D, *Simulation Modeling and Analysis*. New York: McGraw-Hill, 2000.

Li, J., Blumenfeld, D. E., and Alden, J. M., Comparisons of two-machine line models in throughput analysis. *International Journal of Production Research*, vol. 44, pp. 1375-1398, 2006.

Magnanti, T., Combinatorial optimization and vehicle fleet planning : perspectives and prospects. *Networks*, 11, 179-214, 1981.

Maxwell, W. L., and Muckstadt, J. A., Design of automated guided vehicle systems. *IIE Transactions* 14, 2, 114 – 124, 1982.

Mokotoff, E., Parallel machine scheduling problems: A survey. *Asia Pacific J. Oper. Res.,* vol. 18, no. 2, pp. 193–242, 2001.

Monden, Y., *Toyota Production System*. Norcross, GA, USA, Institute of Industrial Engineering, 1983

Monden, Y., *Toyota Production System*, (3nd ed) Norcross, GA, USA, Institute of Industrial Engineering, 1998

Morse, C., *Stochastic Equipment Replacement with Budget Constraints*. PhD thesis, University of Michigan, Ann Arbor, MI, 1997.

Okamura, K., and Yamashina, H., A heuristic algorithm for the assembly line model-mix sequencing problem to minimize the risk of stopping the conveyor. *International Journal of Production Research*17: 3, pp. 233-247, 1979

Pan, C., Xiao, G., Chang, Q., Ni, J., Optimization of workforce zoning for dolly material Handling. *Proceedings of the 2008 Industrial Engineering Research Conference, 2008*.

Pentico, D. W., Assignment problems: A golden anniversary survey. *Eur. J. Oper. Res*., vol. 176, no 2, pp. 774–793, 2007.

Pinedo, M., *Scheduling: Theory, Algorithms, and Systems* (2nd Edition), Prentice Hall, 2002.

Qiu, L., and Hsu, W., An algorithm for concurrent routing of AGVs in a mesh. *The 7th Australasian Conference on Parallel and Real-Time Systems*, University of New South Wales, Sydney, Australia, 29-30 November, pp. 202-214, 2000.

Qiu, L., and Hsu, W., Scheduling and routing algorithms for AGVs: A survey. *International Journal of Production Research*, 40 (3), 745-760, 2002.

Ricca, F., and Simeone, B., Local search algorithms for political districting. *Technical Report, Dipartimento di Statistica, Probabilità e Statistiche Applicate*, Università La Sapienza , Seria A - Ricerche, n. 11,2000.

Schruben, D., and Schruben, L. W., Graphical simulation modeling using SIGMA. *Custom Simulation*, 2000.

Shi, L., and Ólafsson, S., *Nested Partitions Method, Theory and Applications*. New York: Springer, 2009.

Sinriech, D., and Tanchoco, J. M. A., An economic model for determining AGV fleet size. *International Journal of Production Res*earch 30, 6, 1255 – 1268, 1992.

Solomon, M. M., and Desrosiers, J., Time window constrained outing and scheduling problems. *Transportation Science*, 22(1), 1-13, 1988.

Sumicharast, R., and Clayton, E. R., Evaluation sequences for paced, mixed model assembly line with JIT component fabrication. *International Journal of Production Research*, 34-11, 1996.

Tanchoco, J. M. A., Egbelu, P. J., and Taghaboni, F., Determination of the total number of vehicles in an AGV-based material transport system. *Material Flow* 4, 1, 33 – 51, 1987.

Tompkins, J. A., White, J. A., Bozer, Y. A., Frazelle, E. H., Tanchoco, J. M. A., Trevino, J., *Facilities Planning* (2nd), John Wiley & Sons, New York, 1996.

Vis, I., Survey of research in the design and control of automated guided vehicle systems, *European Journal of Operational Research*, vol. 170, 677-709, 2006.

Wysk, R. A., Egbelu, P. J., Zhou, C., and Ghosh, B.K., Use of spreadsheet analysis for evaluating AGV systems. *Material Flow* 4, 1, 53 – 64, 1987.

Yan, C., Zhao, Q., Huang, N., Xiao, G., and Li, J., Formulation and a simulation based algorithm for assignment problem in systems of general assembly line. *IEEE Transactions on Automation Science and Engineering*, vol. 7, pp. 902-920, 2010.

Yao, D. D., *Stochastic Modeling and Analysis of Manufacturing Systems*, Springer-Verlag, New York, NY, 1994.

Yim, D. S., and Linn, R. J., Push and pull rules for dispatching automated guided vehicles in a flexible manufacturing system. *International Journal of Prod Res*, 31(1): 43 – 57, 1993.

Zhao, Y., Yan, C., Zhao, Q., Huang, N., Li, J., and Xiao, G., Efficient simulation method for general assembly systems with material handling based on aggregated event-scheduling. *IEEE Transactions on Automation Science and Engineering*, vol. 7, no 4, pp. 762-775, 2010.