2002

SIMULATION AND OPTIMIZATION OF A CROSSDOCKING OPERATION IN A JUST-IN-TIME ENVIRONMENT

Karina Hauser
University of Kentucky, karina@thehausers.net

ABSTRACT OF DISSERTATION

Karina Hauser

The Graduate School

University of Kentucky

2002

SIMULATION AND OPTIMIZATION OF A CROSSDOCKING OPERATION IN A JUST-IN-TIME ENVIRONMENT

Abstract of Dissertation

A dissertation submitted in partial fulfillment
of the requirements for the degree of Doctor of Philosophy in the
College of Business and Economics
at the University of Kentucky

By

Karina Hauser

Lexington, Kentucky

Director: Dr. Chen Hua Chung, Gatton Endowed Professor of DSIS

University of Kentucky

Lexington, Kentucky

2002

ABSTRACT OF DISSERTATION

SIMULATION AND OPTIMIZATION OF A CROSSDOCKING OPERATION IN A JUST-IN-TIME ENVIRONMENT

In an ideal Just-in-Time (JIT) production environment, parts should be delivered to the workstations at the exact time they are needed and in the exact quantity required. In reality, for most components/subassemblies this is neither practical nor economical. In this study, the material flow of the crossdocking operation at the Toyota Motor Manufacturing plant in Georgetown, KY (TMMK) is simulated and analyzed.

At the Georgetown plant between 80 and 120 trucks are unloaded every day, with approximately 1300 different parts being handled in the crossdocking area. The crossdocking area consists of 12 lanes, each lane corresponding to one section of the assembly line. Whereas some pallets contain parts designated for only one lane, other parts are delivered in such small quantities that they arrive as mixed pallets. These pallets have to be sorted/crossdocked into the proper lanes before they can be delivered to the workstations at the assembly line. This procedure is both time consuming and costly.

In this study, the present layout of the crossdocking area at Toyota and a layout proposed by Toyota are compared via simulation with three newly designed layouts. The simulation models will test the influence of two different volumes of incoming quantities, the actual volume as it is now and one of 50% reduced volume. The models will also examine the effects of crossdocking on the performance of the system, simulating three different percentage levels of pallets that have to be crossdocked.

The objectives of the initial study are twofold. First, simulations of the current system, based on data provided by Toyota, will give insight into the dynamic behavior and the material flow of the existing arrangement. These simulations will simultaneously serve to validate our modeling techniques. The second objective is to reduce the travel distances in the crossdocking area; this will reduce the workload of the team members and decrease the lead time from unloading of the truck to delivery to the assembly line. In the second phase of the project, the design will be further optimized. Starting with the best layouts from the simulation results, the lanes will be rearranged using a genetic algorithm to allow the lanes with the most crossdocking traffic to be closest together.

The different crossdocking quantities and percentages of crossdocking pallets in the simulations allow a generalization of the study and the development of guidelines for layouts of other types of crossdocking operations. The simulation and optimization can be used as a basis for further studies of material flow in JIT and/or crossdocking environments.

KEYWORDS: Crossdocking, Simulation, Optimization, Genetic Algorithms

Karina Hauser

August 16, 2002

SIMULATION AND OPTIMIZATION OF A CROSSDOCKING OPERATION IN A JUST-IN-TIME ENVIRONMENT

By
Karina Hauser

Dr. Chen Hua Chung
Director of Dissertation

Dr. Michael Tearney
Director of Graduate Studies

August 16, 2002

RULES FOR THE USE OF DISSERTATIONS

DISSERTATION

Karina Hauser

The Graduate School

University of Kentucky

2002

SIMULATION AND OPTIMIZATION OF A CROSSDOCKING OPERATION IN A JUST-IN-TIME ENVIRONMENT

Dissertation

A dissertation submitted in partial fulfillment
of the requirements for the degree of Doctor of Philosophy in the
College of Business and Economics
at the University of Kentucky

By

Karina Hauser

Lexington, Kentucky

Director: Dr. Chen Hua Chung, Gatton Endowed Professor of DSIS

University of Kentucky

Lexington, Kentucky

2002

Acknowledgements

This dissertation, while an individual work, benefited from the insight and direction of several people. First, my dissertation chair, Dr. Chen Chung, examplifies the high quality scholarship to which I aspire. His ability to understand when I needed a gentle push in the right direction and when I needed to work on my own is greatly appriciated. In addition, I wish to thank Dr. Muralidhar, who guided me throught the labyrinth of statistical analyses and cheered me up whenever I felt incompetent by sharing the personal experiences of his dissertation adventure. I also wish to thank the rest of my advisory committee, Dr. Clyde Holsapple, Dr. Al Lederer, Dr. Kozo Saito and the outside reader, Dr. John Yingling, for their time. Their insight guided my thinking and improved the finished product. I would like to thank Toyota not only for financial support but also for allowing me to use their data in my research. Their logistic manager at the plant in Georgetown, Mike Botkin showed immense patience in explaining the processes and data involved in this project. I also would like to thank Dr. George Huang, who, while not directly involved in this thesis, helped me make the decision between a pursuing a Master of Engineering or a Ph.D. in Business. His good advice proved to be invaluable in today's job market. Finally, I would like to thank my husband Thomas. He provided me not only with technical support but also with moral support throughout the challenging phases of this last four years. Without his support, I would not have been able to complete this dissertation process.

Contents

List of Figures

List of Tables

List of Files

DissertationKarinaHauser.pdf

Chapter 1

Introduction

The pressure to produce a wide variety of models has made mixed-model assembly lines an integral part of the Just-in-Time (JIT) production system. On a mixed-model assembly line, several different models of a basic end product are produced at the same time, for example, Camrys with and without moon-roof, with right or left steering. This leads to the problem of balancing and sequencing the different models on the assembly line. One of the goals of sequencing is to keep the usage of every part in the assembly line constant to ensure a smooth production. Many algorithms have been developed to help with the sequencing of mixed-model assembly lines, but little attention has been paid to the challenges that frequent deliveries pose for the support people in the logistics area. The goal to keep inventory low leads to frequent deliveries and the need for innovative storage and transportation solutions. In an ideal situation, the suppliers would deliver the needed parts directly to the workstation at the assembly line in the exact quantity at the exact time and in the sequence needed. In this ideal case, the inventory level at and between all workstations would be zero. In reality, only a few parts are delivered directly in sequence to the assembly line, for example, car seats, thus different intermediate storage solutions have been developed:

Flowracks or floor staging area: Depending on their size, the incoming parts are stored in flowracks or in a designated storage area on the floor. If parts are needed at the assembly line, they are replenished out of the inventory in this area. Either internal kanban cards, or call buttons, a type of electronic kanban, are used as a signal for the internal replenishment; external kanban cards are used for replenishment from the suppliers. A kanban card is a piece of paper/cardboard that has all vital information on it for the parts that are in the box it belongs to, such as part number, part description, quantity, supplier etc. In a JIT system, a kanban card has three main functions: identification tag, job instruction tag and transfer tag [Shingo, 1981].

Internal Sequencing: If parts are needed in a special sequence at the line, they are stored in a sequence area and sequenced before delivery to the line.

Lane storage: Here the incoming parts are sorted by line and then are immediately delivered to the line. This sorting process is called crossdocking. Traditionally, crossdocking is defined as “a logistic technique that eliminates the storage and order picking functions of a warehouse while still allowing it to serve its receiving and shipping functions” [Bartholdi III and Gue, 2001] and it is used in the less-than-truckload freight industry. In this study, the shipping function is replaced by the consumption of the parts at the assembly line.

1.1 Statement of the Problem

The project will be performed in cooperation with Toyota Motor Manufacturing Kentucky (TMMK). Personnel planning in the lane storage area poses a problem for TMMKs internal logistic manager. Team members complain about the unbalanced workload; some team members are unable to handle their workload, whereas others have too little work. In addition, this workload imbalance varies during a typical work day. Team members support each other, but they would prefer a solution equalizing workloads overall and during the whole day. An evenly distributed workload not only establishes a sense of equity among workers but, more importantly, increases the output.

The part requirement schedule and the delivery schedule of the incoming parts are the two factors that directly influence the workload balance in the crossdocking area. Studying the influence of these two factors is beyond the scope of this dissertation. The other factor that influences the workload balance is the workload itself. By reducing the overall workload, the remaining workload is easier to balance; so this study concentrates on minimizing the workload in the crossdocking area.

The logistics manager also would appreciate a tool to better understand the factors leading to this imbalance. For example, how changes in the volume of incoming parts, influence the behavior of the material flow in the logistics area.

1.2 Description of the Lane Storage Area at Toyota

A layout of the lane storage area and adjacent areas is illustrated in Figure 1.1. Trucks get unloaded in 4 pits, which are designed so that the forklifts have access on ground level, eliminating unnecessary up and down movement of the pallets and therefore increasing safety for the team members in that area. The trucks have retractable sides which allow unloading from both sides simultaneously. Two forklift drivers, dedicated to unloading, are able to unload the whole truck within 5 minutes.

Between 20 and 30 trucks per pit are unloaded every day, which totals between 80 and 120 trucks per day. The truck schedule generally remains constant, although some trucks do not come in on a daily basis. Once a month the sequence schedule for the assembly line changes, and the truck schedule changes accordingly. These schedule changes also take into account the mileage per carrier and attempt to equalize it. In addition to the parts that are handled in the lane storage area, the trucks carry parts for other storage areas, such as sequencing parts and flowrack parts.

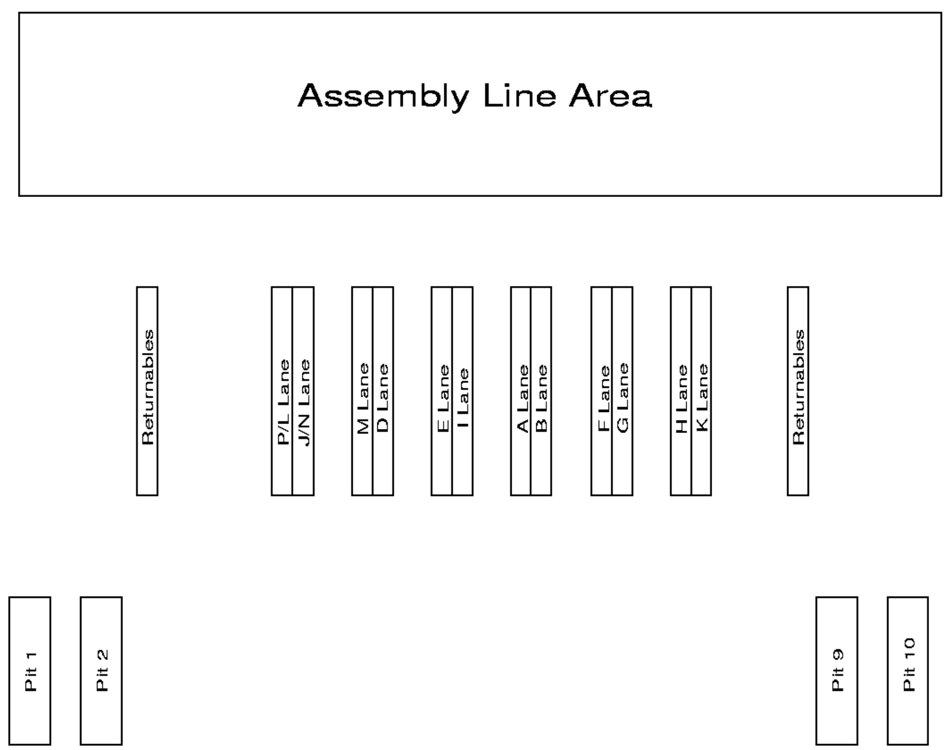

Figure 1.1: Layout of the unloading/lane storage/assembly line area

Each incoming box is accompanied by a kanban card designating the lane and line to which the parts ultimately belong. Approximately 1300 different parts are handled in the lane storage area. A limited number of parts are used at more than one workstation. For these parts with multiple destinations, the kanbans for each destination are printed with the different lineside/lane addresses. Therefore, in this study, parts with multiple uses and destinations can be considered as different parts. The flow of the kanban cards is shown in Figure 1.2.

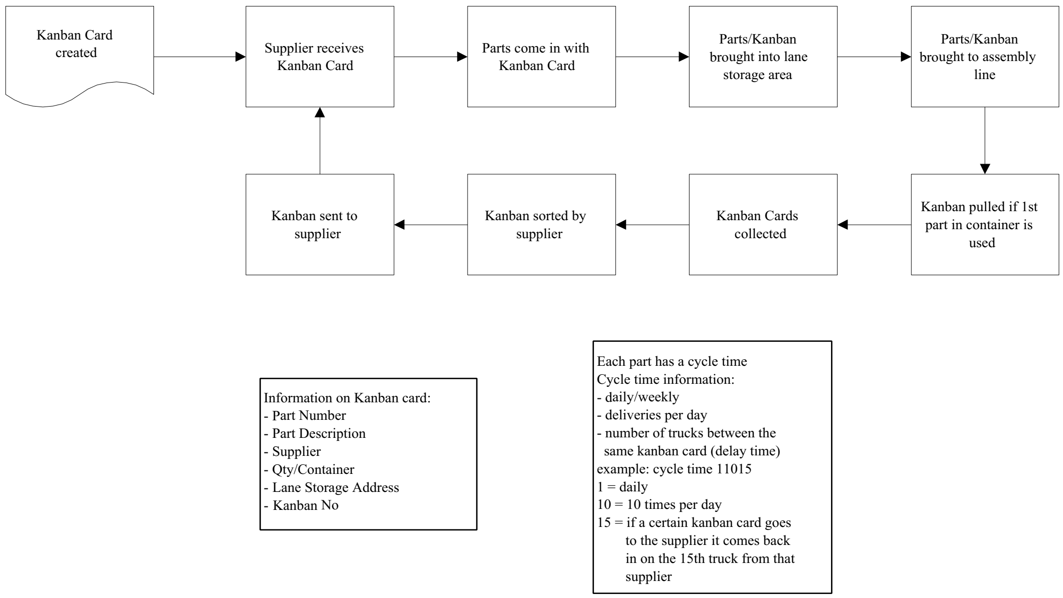

Figure 1.2: Flow kanban cards

The number of team members is currently fixed at 8 working in the delivery area and 3 team members working in the sorting area. Every two hours, team members in the lane storage area rotate between crossdocking and delivery to line. Forklift drivers rotate jobs on a daily basis.

The lane storage area consists of 14 lanes, with two sets of lanes (P/L and J/N, as shown in Figure 1.1) sharing the same physical space; thus for this study they are considered one lane, so overall there are 12 lanes. The lanes and lines have corresponding labels, e.g., all dollies/parts from lane E go to a part of the assembly line that is also labeled E. For the remainder of this study, the lanes are labeled according to their position in the layout, e.g., lane P/L will be labeled lane 1, lane J/N will be labeled lane 2, etc. Each lane is separated into 3 sections, as shown in Figure 1.3.

- Lane Section 1: Unloading area
  5 dollies
  Parts are unloaded from the truck and brought into the unloading area of the designated lane via forklift.

- Lane Section 2: Crossdocking area
  5 dollies
  The crossdocking area is separated from the waiting area by a red line; only electric cars, called tuggers, operate behind the red line, no forklifts are allowed. A team member pulls all full dollies from the loading area into the crossdocking area, removes the packaging material and sorts out parts (crossdocking) that do not belong to that lane. If the lane is close by, the team members bring the boxes there directly; if not, the parts are stored on a dolly that stands between the lanes. When the team member has time, the mixed dolly is unloaded at the proper lanes .

- Lane Section 3: Line delivery area
  5 dollies
  After crossdocking, the team member pulls the dollies into the ready area where they wait until a team member from the delivery team is able to bring them to the assembly line.

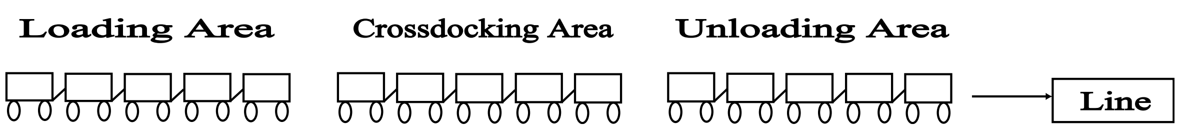

Figure 1.3: Lane layout

At the line, the parts are unloaded into a designated row in a flowrack. If the flowrack is full, the parts go into the overflow area for that workstation.

1.3 Research Goals and Contribution

In 1985, US manufacturers purchased material valued at 60% of total sales revenue [Gunasekaran, 1999]. All this material not only had to be purchased, but also shipped, stored and delivered to the workstations where it was needed. Most of the JIT literature agrees that zero inventory is one of the goals in JIT because inventory is costly. The costs include not only the cost of procurement, storage, insurance, and handling, but also the risk of the inventory becoming obsolete or stolen. High inventory also presents quality issues because a large quantity of defective parts may unknowingly be stored. JIT purchasing considers this issue and attempts to eliminate raw material inventory by using a small, reliable supplier base located close to the buyer's plant to ensure frequent deliveries. Because handling and transportation are viewed as non-value adding elements of a manufacturing operation, they have to be kept to a minimum.

De Haan and Yamamoto [de Haan and Yamamoto, 1999] showed in their case study that zero inventory is, for the moment, more fiction than fact. In a study of inventory methods of eight Japanese companies', seven out of the eight companies inventory methods for raw material, depended on the distance between the supplier and the buyer. Suppliers that are located in close proximity to the buyers' plant deliver daily, whereas the other suppliers have a weekly or even monthly delivery interval. Of the eight surveyed companies, only one, a make-to-order company, found the goal of zero inventory more disruptive to their production process than helpful and had its material delivered on a weekly basis.

The research in this study acknowledges that zero inventory is in reality not possible and that solutions have to be found to handle the incoming material efficiently. The overall goal of this research to identify the factors that can lead to an improvement in the workload of the team members in the crossdocking operation. This will be done through analyzing and optimizing the material flow from the unloading of the material from trucks to the unloading of the material at the workstations where it is used.

The first objective of the simulation is the analysis of the material flow and the identification of all parameters that are involved. After identification of the parameters, the influence of these parameters on the workload of the team members is analyzed. This will lead to a better understanding of the whole system and the identification of potential bottlenecks and problems.

The objective of the optimization is to rearrange the lanes, so that lanes that have the most crossdocking activity are closest together, and that the workload balance among the team members can be further improved. The workload balance is directly influenced by the schedule of part requirements (i.e. production schedule) and the delivery schedule of the incoming material resulting from it. Studying the influence of these parameters is beyond the scope of this dissertation. This work concentrates on minimizing the overall workload for the team members in the crossdocking area. An overall lower workload will simplify the task of workload balancing.

Therefore the overall objective is to reduce the traveling distance of the team members in the crossdocking area. The reduced traveling distance will lead to lower handling cost as well as decreased lead time between unloading of the truck and unloading of the parts at the assembly line. The reduced lead time has two effects: first, it will reduce the workload of the team members, and second, it will reduce the inventory level of raw material.

The remainder of this dissertation is structured as follows. In chapter 2 an overview is given of the related existing literature. Chapter 3 describes the simulation study in detail, followed by the analysis and results in chapter 4. In chapter 5 the optimization approach is discussed. The results of the optimization are reported in chapter 6. Finally, concluding remarks and suggestions for future research are given in chapter 7

Chapter 2

Literature Review

This chapter starts with a review of the existing JIT literature related to the delivery/logistics process of the supply chain management, followed by a brief overview of mixed-model assembly line literature, which covers the front end and the back end of the crossdocking operation. Then the existing crossdocking literature is summarized, and finally, an examination of facility layout studies, especially those using the Quadratic Assignment Problem approach, is made.

2.1 Review of JIT Delivery Literature

In JIT delivery, the materials are provided to the production plant just as they are required for use. JIT delivery is part of the larger concept of JIT purchasing, which includes a small, reliable supplier base close to the buyer's plant and frequent deliveries. Schonberger [Schonberger, 1984] describes a smooth flow of materials between suppliers and buyers as one of the key elements needed to ensure a continuous process from receipt of raw material/components through to the shipment of the finished goods.

Tan [Tan, 2001] developed a framework of supply chain management (SCM) literature. He shows that the current holistic approach of SCM literature evolved from two separate paths: The purchasing and supply perspective of SCM, and the transportation and logistics perspective of SCM. The purchasing and supply perspective mainly covers the issue of the buyer-supplier relationship and integration, whereas the transportation and logistics perspective focuses, as the name suggests, on transportation and logistic issues of the buyer-supplier relationship. Our literature review of JIT delivery concentrates on papers that are concerned with transportation and logistic issues.

Hale [Hale, 1999] points out some of the challenges and opportunities awaiting logistics in the new millennium. With more people shopping via the Internet and home shopping channels, he expects a substantial increase in home deliveries. These small order deliveries present new logistical challenges for all partners, including more non-stop logistic movement, such as:

1. crossdocking
2. consolidation of products from multiple manufacturers by third-party logistics providers in a single delivery
3. increased emphasis on point-of- sale driven, pull inventory replenishment systems
4. increased demand for customized deliveries of multi-tier pallets with electronic pallet content identification
5. advanced electronic data interchange (EDI) capabilities

Real time information flow will be an essential component in the logistic chain. These challenges can only be handled by providing logistics managers with new tools such as: high speed networks, satellites for location of transportation vehicles, easier to use activity-based costing systems, and user friendly modeling, simulation and optimization techniques that support managers in their decision.

Fisher [Fisher, 1997] found that the logistic approach should depend on the type of products. He distinguished two different product types: *functional products*, which are characterized by a predictable demand, a high forecast accuracy, low stockout rate and low product variety, and *innovative products*, which are characterized by an unpredictable demand, low forecast accuracy, high stockout rate and high product variety. To handle functional products, he suggests concentrating on minimizing the physical costs that appear in the supply chain, such as cost of transportation and handling. To handle innovative products, he suggests concentrating on the market mediation costs, which occur when the supply is greater than the demand and force prices to drop, or when demand exceeds supply, resulting in lost sales opportunities and dissatisfied customers.

2.2 Review of Mixed-Model Assembly Line Literature

A mixed-model assembly line is a single line capable of making several different models at the same time. While such lines can quickly respond to changes in market conditions, they also present two challenges. The first challenge is the design and balancing of the assembly line, which includes determination of cycle times and number of workstations. The second challenge is the sequencing of the different models on the assembly line, which can be divided into smoothing and leveling. In smoothing, the goal is to assign each workstation in the assembly line an equal amount of work so that the operation time is the same at all workstations. The goal of leveling is to sequence the models so that all subassemblies and components are withdrawn equally and so that the overall variability is minimized, which at the end leads to a minimized overall inventory. Sequencing mixed-model assembly lines has gotten a lot of attention in the literature.

Leu et al. [Leu et al., 1996] give an excellent illustration of the difficulties faced while sequencing a mixed-model assembly line. They developed a genetic algorithm that improves upon Toyota's Goal Chasing Algorithm and gets results within seconds. The algorithm was tested on 80 problems with the result of improved sequence in 50 of the problems. Using Toyota's variability of part consumption criterion, the algorithm achieved a performance advantage of 2% across all 80 problems. Korkmazel and Meral [Korkmazel and Meral, 2001] first compare the performance of some well-known approaches [Inman and Buffin, 1991] [Miltenburg, 1989][Ding and Cheng, 1993a][Ding and Cheng, 1993b] for solving the leveling problem to the optimal solution obtained by using the shortest path algorithm of Burkard and Derigs [Burkard and Derigs, 1980]. The approaches found to be performing better are extended to incorporate the goal of smoothing the workload. In addition, the conditions under which it is important to take the workload-smoothing goal into consideration are analyzed. They found that high variance in model processing and/or shorter lines makes considering the workload-smoothing goal worthwhile.

Matanachai and Yano [Matanachai and Yano, 2001] propose a new line balancing approach with the emphasis on providing a stable workload on the assembly line while also achieving reasonable workload balance among all workstations. They first compare their heuristic filtered beam search algorithm with a commercial mixed-integer optimizer for a small problem and report improvements of 22% to 41%, depending on the average utilization of the line and the variability of the task processing time. They then used their approach on a set of larger problems and also found substantial improvements in 90% of the problems.

Baykoc and Erol [Baykoc and Erol, 1998] used simulation to study the performance of a multi-item, multi-line, multi-stage JIT system and showed how this system reacted under different factor settings. They tested the effects of four factors, namely, number of kanbans, coefficient of variation of processing times, degree of imbalance, and degree of demand uncertainty, on system performance measures such as total output rate, waiting time on work-in-process (WIP) points, WIP length, and station utilization. For all experiments, output rate and station utilization improves as the number of kanbans increases to two, but no further improvements occur after that. Increasing the number of kanbans also results in an increase in waiting times and WIP length. On the contrary, an increase in the coefficient of variation of processing time or degree of imbalance leads to a decrease in output rate and utilization.

2.3 Review of Crossdocking Literature

The success story of Wal-Mart [Stalk et al., 1992] and its improvement in lead time has brought attention to crossdocking operations. Wal-Mart achieved its goal of providing customers access to quality goods when and where they want them by making the way the company replenished inventory the centerpiece of its competitive strategy. Due to crossdocking, goods cross from one loading dock to another within 48 hours or less. By running 85% of its goods through its warehouse system, Wal-Mart reduced costs of sales by 2% to 3% compared to the industry average.

Gue [Gue, 1999] defines terminal layout as the arrangement of receiving/strip doors and shipping/stack doors, and the assignment of destinations to stack doors. Since the material flow in a crossdocking terminal and the travel distance for workers transporting freight largely depends on the layout of the terminal, the crossdocking literature is mainly concerned with layout studies.

Bartholdi and Gue [Bartholdi III and Gue, 2001] ran a series of computational experiments to determine which shapes of crossdocks have the lowest flow cost and the least traffic congestion. They found that for small to mid-sized crossdocks (up to 150 doors), a rectangle or I-shaped crossdock performed best. For larger docks (150 to 250 doors), the T-shape performed best; for crossdocks that exceed 250 doors, the H-shape performed best.

In an earlier paper, Bartholdi and Gue [Bartholdi III and Gue, 2000] created several models that guided a local search routine in assigning destination trailers to terminal doors. The goal was to minimize total labor cost, which was defined as the cost of moving freight from incoming trailers to outgoing trailers weighted against the cost of delays due to different types of congestion - in other words, worker travel time and worker waiting time. They found that the improved layouts tend to concentrate activity in the center of the dock. The highest-flow regions on either side in the center are slightly offset so that congestion in the center of the dock is reduced. A typical layout of their model is shown in Figure 2.1 . The improved layout was implemented at a Viking terminal in Stockton and led not only to an improvement in productivity by 11.7 % but also to a noticeable reduction in freight processing time and other unexpected benefits.

Gue [Gue, 1999] investigates the effects of trailer scheduling on the layout of freight terminals. He developed a model of the material flow when a look ahead scheduling strategy is

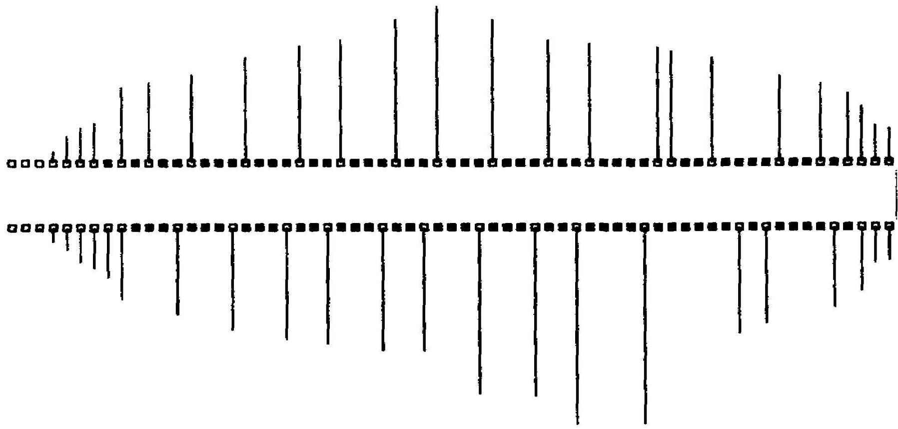

Figure 2.1: A typical layout produced by the models of Bartholdi and Gue [Bartholdi III and Gue, 2000] (Filled squares represent receiving doors and empty squares represent shipping doors. Lines extending from the shipping doors represent the relative flows to those doors.)

used. In a look ahead strategy, to minimize worker travel, incoming trailers are assigned to the door closest the shipping door with the most outgoing freight. Gue first used linear programming to assign trailers to doors and then ran a set of simulations to determine the layout with the lowest expected cost. The look ahead scheduling strategy reduced traveling cost by 15 to 20% compared to a first-come, first-serve policy. The new layout provides further savings of 3 to 30% depending on the mix of freight on incoming trailers.

Tsui and Chang [Tsui and Chang, 1990, Tsui and Chang, 1992] developed a microcomputer based decision support tool for assigning dock doors in freight yards. They used a bilinear algorithm to recognize shipping patterns. Recognizing these patterns leads to an improved assignment of incoming trucks to the receiving doors, minimizing travel distance for the forklift drivers and avoiding congestion.

2.4 Review of Facility Layout Studies

Crossdocking and facility layout studies are closely related. Their common goal is to minimize material handling costs, and they both do so by arranging activities in an optimal way.

The efficiency of a certain layout is typically measured in terms of material handling costs, which increase with the distance between the departments. The two most commonly used measurements for the distance are between I/O points of the department and the centroid-to-centroid method. The two most popular metrics to measure the distance between two points are the rectilinear distance and the Euclidean distance.

Meller and Gau [Meller and Gau, 1996] analyze recent and emerging trends in the facility layout literature from 1986 to 1996. They developed a classification scheme to distinguish three different types of layout studies: Facility layout models and heuristics for block layout, facility layout model extensions, and special cases. Whereas the first two types are concerned with the overall facility layout, the special cases consider the layout of specific areas, for example, flowlines, machine layout and cellular layout design. One emerging trend is the application of genetic algorithms and tabu search to the facility layout problem.

2.4.1 The Facility Layout Problem and the Quadratic Assignment Problem Approach

In the classical facility layout problem, a set of facilities has to be allocated to a set of locations with the objective to minimize cost. Cost is a function of the amount of interdepartmental flow, f_{ij} (the flow from department i to department j); the distance between the departments, d_{ij}; and the unit-cost value, c_{ij} (the cost to move one unit load one distance unit from department i to department j).

$$min \Sigma_i (f_{ij} c_{ij}) d_{ij}$$

The two traditional approaches to solve the problem are the graph-theoretic approach, which assumes that the desirability of locating each pair of facilities adjacent to each other is known, and the quadratic assignment problem approach, which assumes that all departments have equal areas and that all locations are known. In our study, the lanes all have the same size and the locations are known; therefore, the rest of the literature review will concentrate on the quadratic assignment approach.

The quadratic assignment problem was introduced by Koopmans and Beckman [Koopmans and Beckman, 1957] in the late 50's. The quadratic assignment problem belongs to the class of NP-hard (Nondeterministic Polynomial) problems, as shown by Sahni and Gonzalez [Sahni and Gongzalez, 1976], meaning that not even an approximate solution within some constant factor from the optimal solution can be found within polynomial time. Even with the increased computational capabilities, especially the development of parallel computers, over the last several years, only problems with a number of facilities/locations lower than 20 are solvable with exact solution methods, like branch and bound, cutting plane or branch and cut. A number of different heuristic methods which can provide good quality solutions in a reasonable amount of time have been used to solve larger problems. Burkard et al. [Burkard et al., 1998] give a good overview about exact and heuristic methods. Because we choose to use genetic algorithms to find a solution to our problem, the remainder of the literature review will concentrate on papers that use this approach.

Fleurent and Ferland [Fleurent and Ferland, 1994] used a hybrid procedure that combined a genetic algorithm with existing heuristic procedures, namely, local search and tabu search. The genetic hybrid algorithm is used to overcome the problem of stopping at the first local minimum it reaches that is associated with local search procedure. To verify their approach, they used two sets of quadratic assignment problems with large size (n=100) found in earlier literature [Skorin-Kapov, 1990] [Taillard, 1991]. They found that in almost every case, the hybridized local search and tabu search method significantly enhanced the search methods and that they could improve on the already existing best known solutions for most of the larger test problems.

Tate and Smith [Tate and Smith, 1995] showed that their genetic algorithms performed consistently equal to or better than previously known heuristic methods without undue computational overhead. They used character encoding to allow reproduction and mutation functions that work directly on the solution sequence. Mutation took place by selecting two sites at random and reversing the order of all sites within the subsequence bounded by the two selected elements. The reproduction scheme used produced only feasible solutions to minimize computing time. The experimental design consisted of eight different examples defined by Nugent et al. with a range of numbers of facilities from 5 to 30 and a symmetric traffic matrix, meaning that the flow from facility A to B is the same as from B to A, etc. . Multiple runs for each problem were performed with 25%, 50% and 75% of reproduction, meaning % of children created each generation, and 75%, 50% and 25% of probability of mutation during a generation. The best results were obtained using the most stochastic mix of reproduction and mutation, with 25% children and 75% probability of mutation.

Ahuja et al. [Ahuja et al., 1995] suggest a genetic algorithm that incorporates many greedy principles in its design. They created their initial population by using a randomized construction heuristic, developed a new crossover scheme, used a special purpose immigration scheme that promotes diversity, performed periodic local optimization of a sunset of the population, used tournamenting among different populations, and created an overall design that attempts to strike a balance between diversity and a bias toward fitter individuals. The instances in QAPLIB were used as benchmarks for the greedy genetic algorithm which obtained the best known solution for 103 out of the 132 instances, and for the remaining instances(except one) found solutions within 1% of the best known solution.

Huntley and Brown [Huntley and Brown, 1991] developed SAGA, a combined approach of Simulated Annealing and a Genetic Algorithm to solve the quadratic assignment problem.

In their approach, they use a genetic algorithm for finding good initial solutions and then use simulated annealing for a refined local search. They use a crossover operation which splices a portion of the structure of one parent directly into that of the other parent and then resolves conflicts with a simple resolution scheme. One parent is selected at random from among the best structures, the other one is selected completely at random, which increases the greediness of the algorithm. Two test problems are used to evaluate the algorithm; one from Nugent [Nugent et al., 1968] with low flow dominance, and another one from Scriabin and Vergin [Scriabin and Vergin, 1975] with a high flow dominance. Flow dominance is the tendency of items to flow through a bottleneck area; the higher the flow dominance, the harder it is to find good heuristic solutions. Ten runs are made for each problem, and the solutions are compared with solutions found by CRAFT, with the result that SAGA outperformed CRAFT in all twenty trials.

A comparison of the important parameters used in those studies is given in Tables 2.1 and 2.2. There seems to be no predominant set of parameters used in all the studies. Only the coding scheme is the same in all cases; facilities/locations are always represented by real numbers. Because all the studies use different test cases, a direct comparison/evaluation of the parameters is difficult to perform.

2.4.2 Special Layout Cases

Rosa and Feiring [Rosa and Feiring, 1995] simulate a tool room in an aircraft maintenance company with 400 in-out transactions a day. The racks for the tools are arranged in four different layouts, and the traveling distance is measured and compared. In addition, the tool allocation is changed according to the tools request probability. The new tool allocation achieved the biggest improvement, but the rearrangement of the racks also reduced the travel distance by 12%.

Table 2.1: Genetic Algorithm parameters part 1

	Initial population	Selection Method	Crossover Method
Ahuja et al. Greedy GA	construction phase of GRASP	both parents random	path crossover optimized crossover
Version one	construction phase of GRASP	both parents random	path crossover
Version two same as 1 except:	construction phase of GRASP	both parents random	path crossover
Version three same as 2 except:	construction phase of GRASP	both parents random	path crossover
Tate and Smith	randomly generated	bias toward better solutions	genes are randomly chosen from both parents
Version 1 25% children 75% prob.of mutation	randomly generated	bias toward better solutions	genes are randomly chosen from both parents
Version 2 50% children 50% prob.of mutation	randomly generated	bias toward better solutions	genes are randomly chosen from both parents
Version 3 75% children 25% prob.of mutation	randomly generated	bias toward better solutions	genes are randomly chosen from both parents
Fleurent and Ferland	produced by other heuristic methods: local search and tabu search	bias toward better solutions	genes are randomly chosen from both parents
Huntley and Brown	NA	first parent selected at random amon the best structures; second parent selected at random	splicing a portion of the structure of one parent directly into that of the other parent, then resolving conflicts

Table 2.2: Genetic Algorithm parameters part 2

	Mutation/Immigration	**Local Optimization**	**Tournamenting**
Ahuja et al. Greedy GA	immigration of individuals from underexplored search spaces 10%, 20% and variable immigration rate	after 200 trials, first 20% after 400 trials, next 20%	after 100 trials 50% of union of two populations 50% of each population one to one competition
Version one	10% after every 200 trials	20%	none
Version two same as 1 except:	variable, starts with 10%, increased by 2% after every 200 trials	20%	with four teams
Version three same as 2 except:	variable, starts with 10%, increased by 2% after every 200 trials	20%	with eight teams
Tate and Smith	selection of two sites at random and reversing the order of all sites within the subsequence	none	none
Version 1 25% children 75% prob.of mutation	25% children, 75% prob. of mutation	none	none
Version 2 50% children 50% prob.of mutation	50% children, 50% prob. of mutation	none	none
Version 3 75% children 25% prob.of mutation	75% children, 25% prob. of mutation	none	none
Fleurent and Ferland	none, individuals generated by heuristic mehod	none	none
Huntley and Brown	NA	simulated annealing	none

Table 2.3: Genetic Algorithm parameters part 3

	Test cases	Runs	Results
Ahuja et al. Greedy GA	QAPLIB 132 instances	1	obtained best known solutions for 103 problems remaining (except one) within 1% of best known solution algorithm applied only once
Version one	QAPLIB 132 instances	1	used as benchmark algorithm
Version two same as 1 except:	QAPLIB 132 instances	1	better overall performance
Version three same as 2 except:	QAPLIB 132 instances	1	very robust performance
Tate and Smith	8 Nugent 1 Cohoon 1 Steinberg 1 Tate	10	
Version 1 25% children 75% prob.of mutation	9 Nugent 1 Cohoon 1 Steinberg 1 Tate	10	best mix robust with respect to solution quality generally found existing optimum or better, except for one (0/1 flow matrix)
Version 2 50% children 50% prob.of mutation	10 Nugent 1 Cohoon 1 Steinberg 1 Tate	10	not dramatically different from best mix
Version 3 75% children 25% prob.of mutation	11 Nugent 1 Cohoon 1 Steinberg 1 Tate	10	worst
Fleurent and Ferland	larger cases from Chakrapani for initial testing 8 Skorin-Kapov 5 Taillard	5 initial testing 10 after-wards	genetic operators are found to improve the performance of both local search and tabu search improvements on most of the test cases
Huntley and Brown	1 Nugent 1 Sciabin and Vergin	10	comparison with CRAFT better results than CRAFT in all 20 trials

Chapter 3

The Simulation Study

This chapter provides the research questions and details of the simulation model, including the layouts, parameters and performance measures. In addition, the actual data provided by Toyota are analyzed and summarized.

3.1 Definitions

For the remainder of this dissertation, pallets that contain boxes for more than one lane will be abbreviated as CP for Crossdocking Pallets and pallets that contain boxes for only one lane will be abbreviated as NCP for Non Crossdocking Pallets.

3.2 Layouts Simulated

A simulation model is developed to observe the influence of different layouts of the lane storage area on the workload of the team members. A good layout reduces travel distances without creating congestion. In the design of the new layouts, special consideration has been given to minimizing the interference between forklifts and tuggers to insure the safety of the team members. The first two designs (I-shaped and T-shaped) were inspired by the findings of Bartholdi [Bartholdi III and Gue, 2001]who showed that these layouts performed best for crossdocks up to 250 doors. The third new layout was modelled after the U-shaped workstation arrangement found in cellular manufacturing [Miltenburg, 2001]. Since an U-shape would lead to interference while transporting the dollies from the crossdocking area to the line delivery area the shape was adjusted to an open V. Using simulation, the original layout, (shown on page 3 in Figure 1.1) where CP and NCP are unloaded in the same area, and four new layouts are compared. The four new layouts are described in detail below:

1. A three line layout, one for the NCP and two for the CP. The NCP are transported directly from the truck to the dollies in the line delivery area. The CP are pulled between the two lanes in the crossdocking areas, and the parts are distributed from there to the designated dollies. The sorted dollies are then pulled into the line delivery area. The number of dollies in each area depends on the number of pallets that have to be distributed and the ratio of NCP to CP. The first new layout is shown in Figure 3.1.

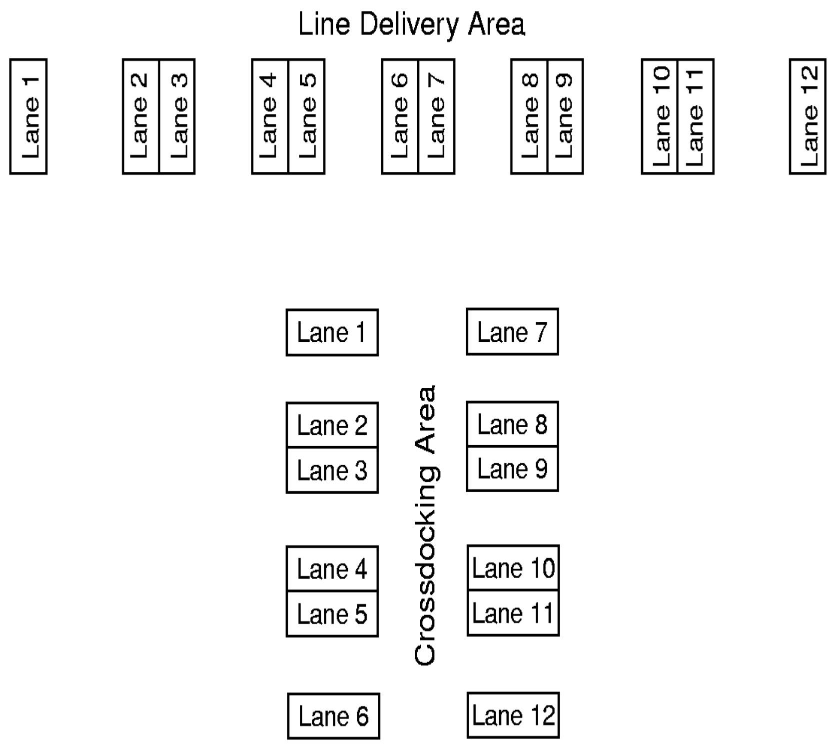

Figure 3.1: New layout 1

2. A layout consisting of two completely separate areas, one for CP and one for NCP. The dollies are pulled from both of these areas directly to the line delivery area. The V-shape design of the line delivery area allows forklifts to work outside the V and tuggers inside the V so interference is minimized. Figure 3.2 presents the second new layout.

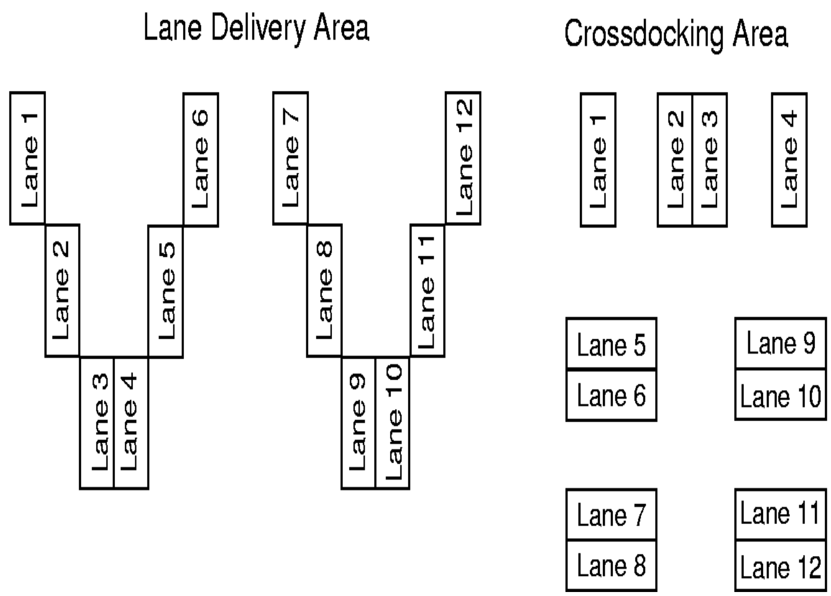

Figure 3.2: New layout 2

3. A layout consisting of two V shaped areas, one for lanes 1-6 and one for lanes 7-12. In the middle of each area, a lane for the CP is created. The third new layout assumes that the supplier will divide the CP into boxes with destination 1-6 and 7-12 to make crossdocking easier. The layout is shown in Figure 3.3.

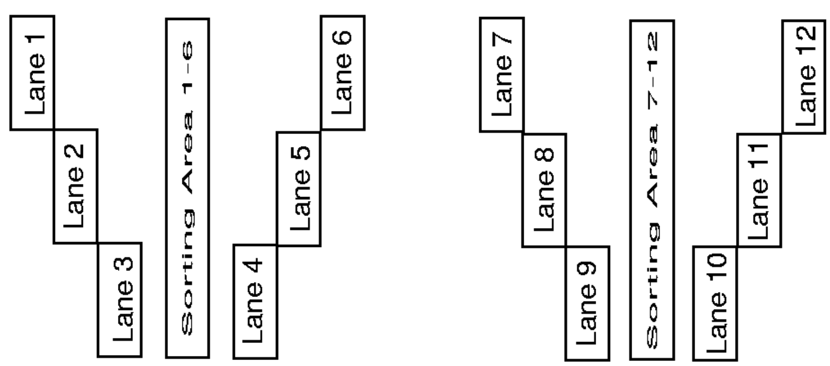

Figure 3.3: New layout 3

4. The fourth new layout was developed by Toyota. The assembly line people suggested reducing the material on their workstations would lead to the reduction of unnecessary movement/walking and therefore workload. Currently, material is stored in up

to three rows of parts in a flowrack, with an overflow area to accept excess material in case all rows are occupied. In a proposed new layout, the material at the line will be reduced to only one row of parts per flowrack and there will be no overflow area. The assembly line team members will request additional material by internal Kanban cards. In the future layout, material with a low volume of containers per truckload and a high quantity of parts per container will be handled in the lane area; material with a high volume of containers per truckload and a low quantity per container will be stored intermittently in either flowracks or a designated floor space. Depending on container size, about 50% of the parts will be stored in flowracks/floor space . The parts will be picked from this area using dollies, which will then wait for the line delivery. The lane storage area will handle the remaining 50% and will be rearranged. Each lane will initially handle the material for two lines, and the final separation will take place during the crossdocking process. After crossdocking, the dollies will go to the same area as the dollies with the parts picked from the flowracks/floorspace, and they will be delivered to the lane together. Toyota's proposed new layout is illustrated in Figure 3.4. The terms future layout and Toyota's proposed new layout are used interchangeably for the remainder of the dissertation.

3.3 Research Questions

The research question that will be answered in the simulation portion of this study are:

- *Research Question 1: Do differences in the percentage of pallets that have to be crossdocked have a significant effect on the workload of the team members?*
- *Research Question 2: Do differences in lane layout organization have a significant effect on the workload of the team members?*
- *Research Question 3: Do differences in the volume of incoming parts have a significant effect on the workload of the team members?*

3.4 Parameters

All simulation runs will test the impact of the layouts, the volume of incoming parts and percentage of pallets to be crossdocked on the workload of the team members:

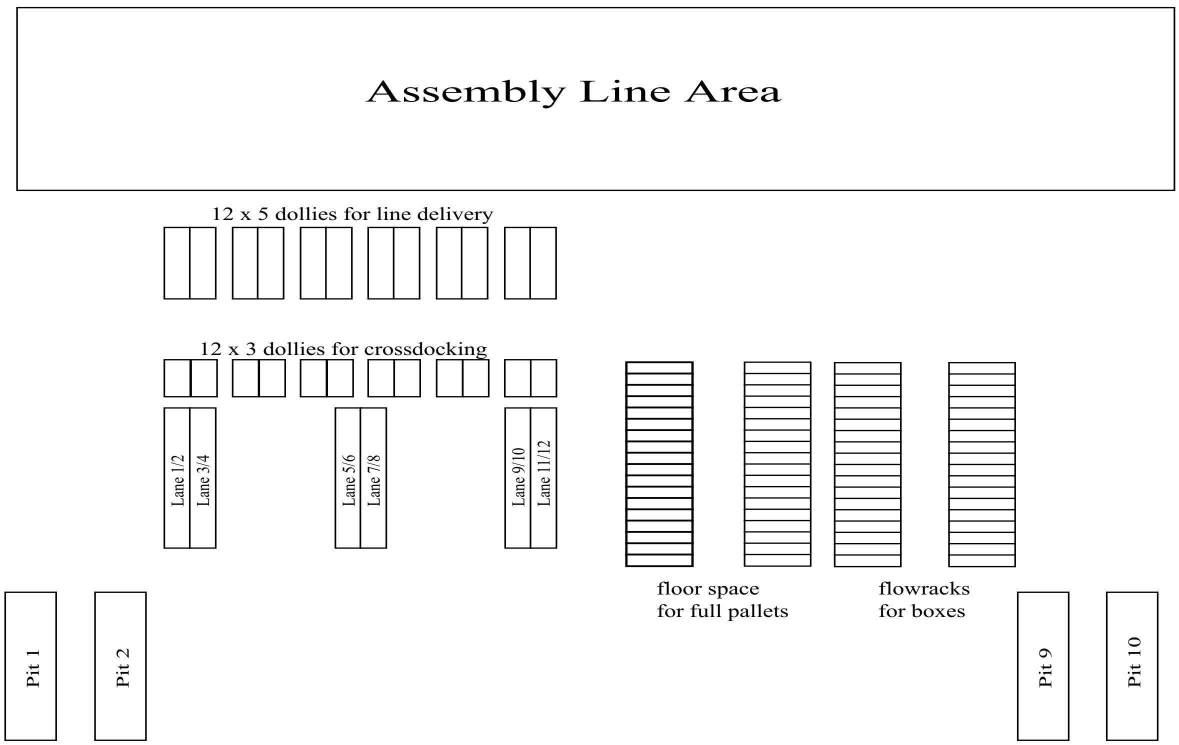

Figure 3.4: New layout proposed by Toyota

1. Percentage of crossdocking pallets
   An analysis of the current and future truck schedule will determine both the current and future average percentage level of pallets to be crossdocked. If there is a practical significant difference (more than 5%) in the two percentage levels both will be used for the simulation runs. In addition, after calculating the current and future percentages, one additional percentage level of crossdocking activity will be determined. This will allow a generalization of the results of the study to a wider variety of JIT companies.

2. Layouts
   The existing layout, the three newly designed layouts and the proposed future layout as described earlier will be simulated and compared.

3. Volume of incoming parts
   The data available at TMMK for the incoming parts is divided into 20 minute time intervals. Each time interval contains information about the number of boxes per supplier in this interval. The information on how many pallets these parts come in is not available, but Toyota requires suppliers to group parts by lane when building the pallets. From that requirement, an algorithm was developed to "arrange" the existing data into pallets. A flowchart of the algorithm is illustrated in Figure 3.5. This algorithm does not give the optimal arrangement of boxes on the pallets, which is itself an NP-complete problem, but it mimics the behavior of the logistic people at the supplier plant, who most likely are not using a sophisticated optimization technique to build the pallets. A distribution function will be fitted to the current and future (50% reduced) volume of incoming pallets and then used for the simulation runs.

A total of 30 simulations (3 levels of CP x 5 layouts x 2 volumes of incoming parts) will be run and analyzed. Table 3.1 provides an overview about all possible combinations of the three parameters.

Table 3.1: Possible combinations of the three simulation parameters

% CD	Quantity	Original Layout	New Layout 1	New Layout 2	New Layout 3	Future Layout
Current	Current					
Future	Current					
New	Current					
Current	Future					
Future	Future					
New	Future					

```
# = number of incoming boxes;
each pallet has on average 12.5 boxes;
** loop1 read all records, write all full pal-
lets into new file, delete the records out of the old file
do until end of file;
start loop1;
    if # > 11.5 and < 12.5 --> 1 pallet;
    if # > 12.5 x = # / 12.5 --
> x pallets, remainder --> new #;
end loop1;
** loop2 “build” pallets
do until supplier changes;
start loop2;
sort all records by quantity;
    if first # > 6.5
        loop 3    ** read first # and find an-
other                   pal-
let so that the sum is closest to 12.5
            add first # and last #;
            if sum < 12.5 --> 1 pallet;
            if sum > 12.5 --> sub 1 last, goto loop3;
        end loop3
    if first # < 6.5
       loop 4
            add first # and next #;
            if sum > 12.5 --> sub next #, first # -->
                     1 pallet;
            if sum < 12.5 --> add next, goto loop 4;
       end loop4;
end loop2;
```

Figure 3.5: Algorithm to build pallets

3.5 Performance Measures

The workload for the truck drivers is defined as the sum of the driving distances between the pits and the lanes. Because the main objective of this study is the optimization of the crossdocking area, the workload of the truck drivers is not considered. The workload for the crossdocking team members is defined as the distance they have to walk to transport the boxes from one lane to another. The main workload of the line delivery people is the unloading of the boxes at the workstations. Because the unloading process is not influenced by the new layouts, it will not be considered as a performance measure.

3.6 The Simulation Model

Experimentation with a real world system is expensive and, in most cases, not practical. In our case, it would mean changing the layout of the lane, observing its performance for a week or month, and risking a shut down of the assembly line should the crossdocking not be done effectively and parts unable to be delivered to the workstation on time. In addition, only the current volume of incoming parts and percentage of CP could be tested. In simulations, on the other hand, testing different scenarios requires only an adjustment of the simulation model, and it is a lot faster since only the actual events are simulated. Therefore, simulation is a much more economical solution.

A discrete event simulation model will be created using ARENA. ARENA uses a graphical user interface (GUI) for SIMAN, a general purpose simulation language providing subroutines for event timing, file handling, and statistical calculations. The GUI speeds up the development of the model and the animation makes it easy for end users, such as the logistics manager, to understand. The simulation model is described in the next section.

3.6.1 Details of the Simulation Model

A general overview of the main program is shown in Figure 3.6. The main program starts with reading the incoming data from a file. Later, the file will be be used to fit a distribution function to the data; this function, in turn, will be used as input for the simulation. For validation of the model, the original data file from Toyota was used. This data contained fixed quantities of incoming parts; therefore, the results, such as quantities per lane, transfer quantities, etc., could be calculated and compared to the simulation output. Each record in the file represents either one pallet, if the parts do not need to be crossdocked, or part of a pallet if the parts have to be crossdocked. The record contains 3 fields: DelayTime, FromLaneToLane and Quantity. The DelayTime field mimics the 20 minute time intervals, with the first record starting with time 0 and each subsequent record in the same time interval containing 1 second. The next time interval starts at 20 minutes minus the number of seconds used for the preceding time interval. The field FromLaneToLane contains the information on the lane to which the parts belong and, if they have to be crossdocked, to which lane the parts should ultimately go. An entry of 0204 would mean the parts first go to lane 02 and from there they are crossdocked to lane 04. The quantity field contains the number of boxes on the pallet.

After reading the file, the next step is to determine in which lane the parts belong and to separate the parts into the lanes. A submodel is used to model the unwrapping process. One challenge in the modeling process was to realistically simulate the behavior of the team members in the crossdocking area. In reality, the team member would determine the lane that has the most pallets in it and unwrap all pallets for that lane, independent of the fact that in the meanwhile, another lane has more waiting pallets. The simulation software, on the other hand would change the allocation of the resource (i.e., the team member) as soon as another lane has more parts waiting to be unwrapped. To solve this dilemma, two counters were needed; the moment a resource starts working on a particular lane, the number of waiting parts in that lane is recorded. Another counter adds the number of parts processed, and the resource does not get released until both numbers match. The submodel used to simulate the unwrapping part is shown in Figure 3.7. To determine which lane has to be processed next, another submodel, shown in Figure 3.8, was created to find the lane with the most parts waiting.

After the pallets are unwrapped, a decision is made as to whether the parts stay in the lane or have to be crossdocked. The crossdocking process takes place in another submodel, shown in Figure 3.9. This submodel uses the same logic to assure the transfer of all parts before the team member switches to another lane. One counter is used to store the sum of parts waiting to be transferred, and another counter is used to sum the transferred parts. If both match, the resource/team member is released and is able to work on the next lane. After all parts in one lane are transferred, the lane with the most parts waiting for transfer is determined, as shown in Figure 3.10, and processed.

Different counters keep track of (1) the number of pallets and quantities of parts coming into each lane; (2) parts being transferred from/to each lane; and (3) parts finally leaving the lane storage area. Histograms show the number of parts waiting to be unwrapped and parts waiting to be transferred. These histograms indicate the number of dollies that should be used. Each dolly can hold one pallet, so the number of parts waiting divided by the number of parts per pallet gives the number of dollies needed.

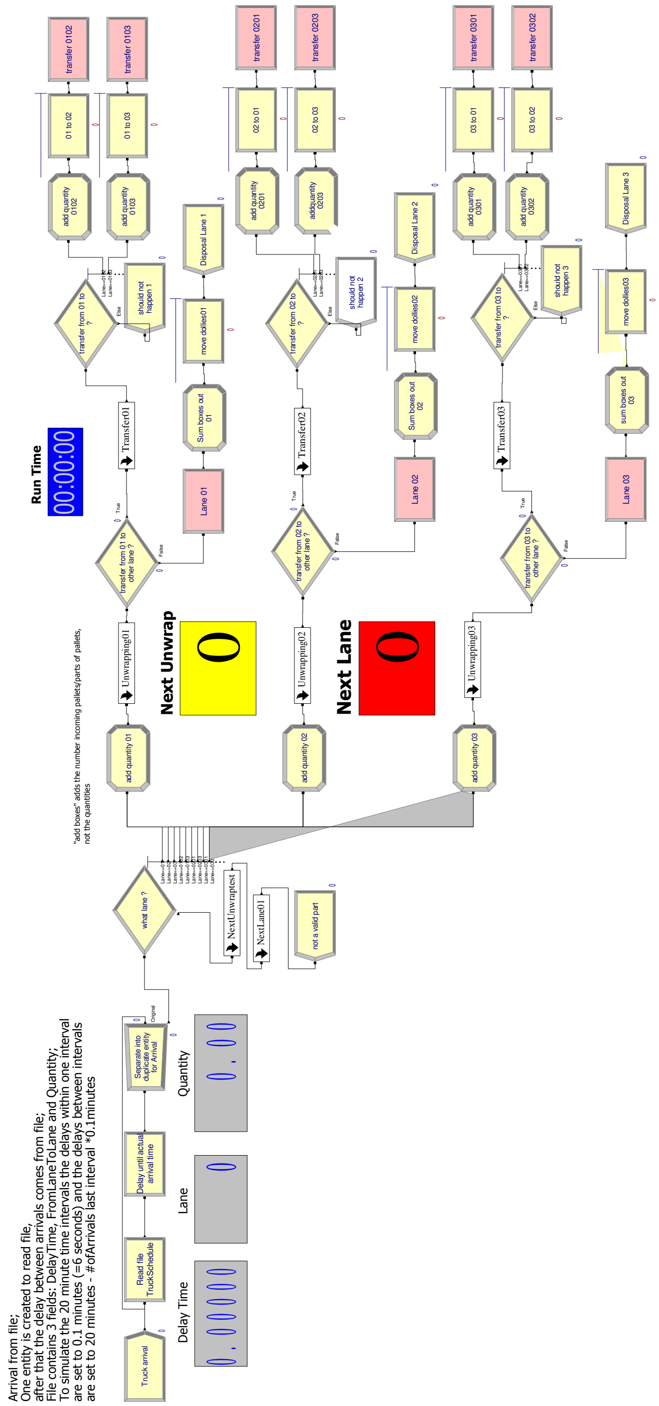

Figure 3.6: Main model

Submodel: Unwrapping

Unwrapping of pallets and moving dollies to line delivery area

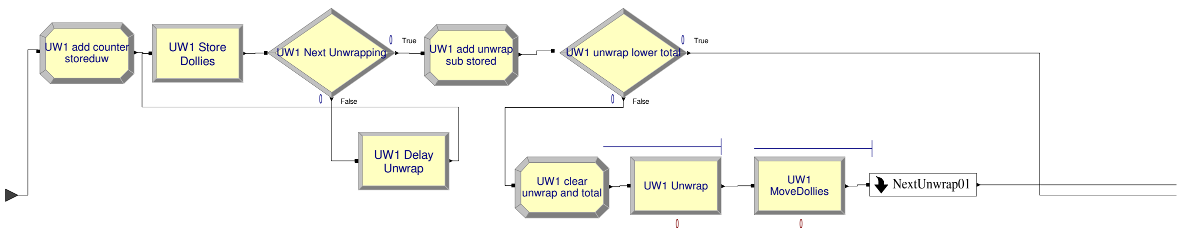

Figure 3.7: Submodel: Unwrapping

Submodel: NextUnwrap

Which lane has to be unwrapped next ?

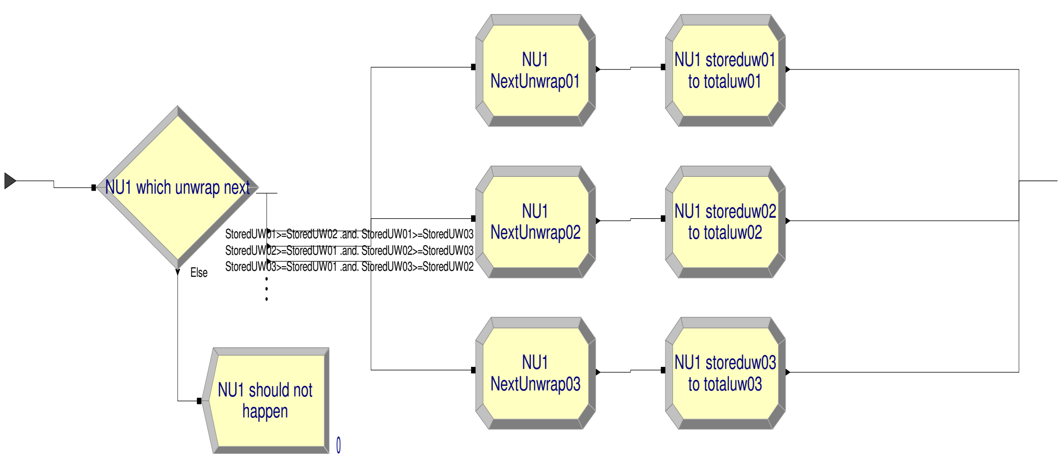

Figure 3.8: Submodel: Next Unwrap

Submodel: Transfer

Which lane should be transferred next?

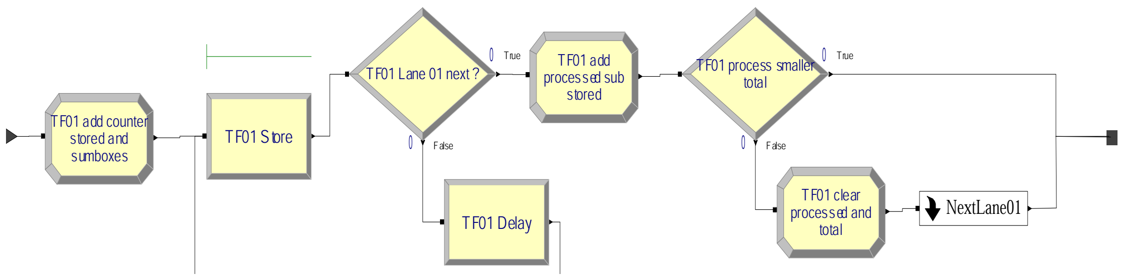

Figure 3.9: Submodel: Transfer

Submodel: Next lane

Which lane has to be transferred next ?

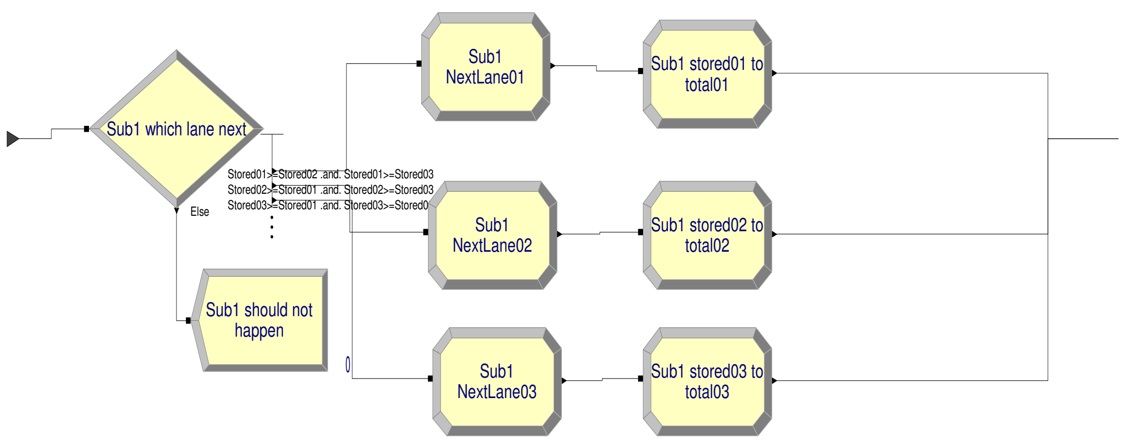

Figure 3.10: Submodel: Next Transfer

Besides the main model, two additional models are used. After 10 minutes of simulated time, the first model sets the lane that has to be unwrapped first. The other model is used to simulate the end of the shift when no more parts are coming in. The two models are shown in Figures 3.11 and 3.12.

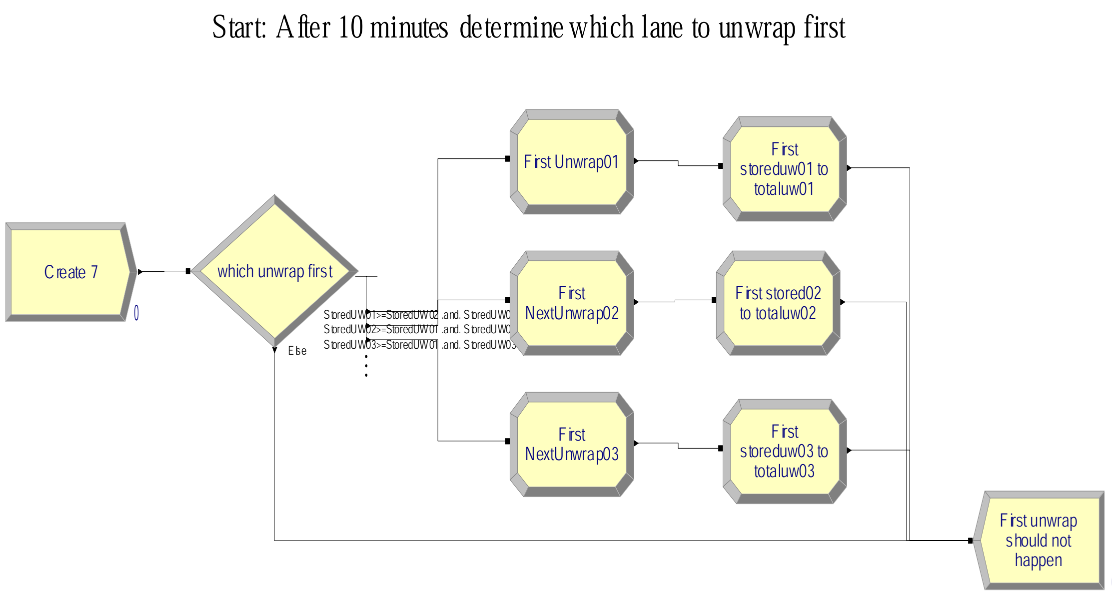

Figure 3.11: Model: Which lane to unwrap first

3.7 Current Toyota Data

The original data, provided by Toyota's logistics department, contained the number of incoming boxes per supplier per line segment (overall 35) divided into 20 minute intervals. Because this study is interested in the number of boxes per lane per time interval, the different line segments and suppliers are added up to the number of containers per lane, as shown in Table 3.2 and Figure 3.13. There is no obvious pattern in the data; three time buckets have no incoming parts at all, whereas two time buckets have over 1260 incoming parts. On average, 508.26 parts come in during a 20 minute time period, with a high variance of 106568.

In addition, the cumulative number of boxes and percentage per lane are shown in Table 3.3. The number of boxes per lane is also graphically illustrated in Figure 3.14. The percentage of incoming volume ranges from 3.72 % for lane 1 up to 11.39 % for lane 4.

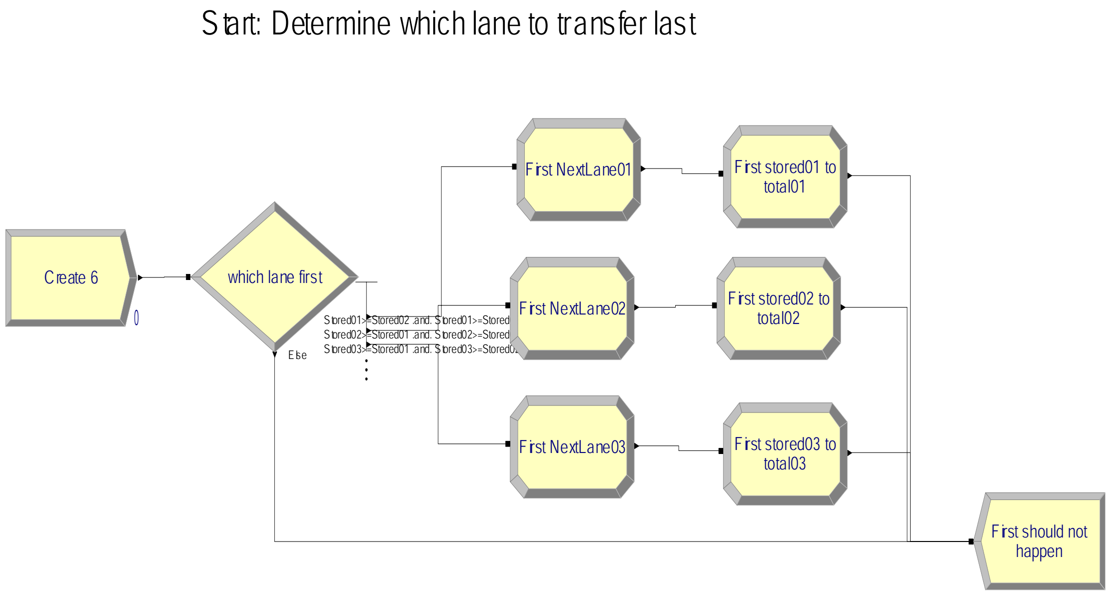

Figure 3.12: Model: End of simulation

Table 3.2: Number of boxes per 20 minute interval for current data

Interval	21	45	3	28	33	9	41	19	17	4	35	5	29	43	12	16
# of Boxes	1266.47	1266.47	1053.87	1045.64	946.91	898.86	867.55	829.65	828.44	722.57	689.54	683.62	683.62	676.88	671.71	671.39
cum	1266.47	2532.94	3586.81	4632.45	5579.36	6478.22	7345.77	8175.42	9003.86	9726.43	10415.97	11099.59	11783.21	12460.09	13131.80	13803.18
%	5.19	5.19	4.32	4.29	3.88	3.68	3.56	3.40	3.40	2.96	2.83	2.80	2.80	2.77	2.75	2.75
cum %	5.19	10.38	14.70	18.99	22.87	26.55	30.11	33.51	36.91	39.87	42.69	45.50	48.30	51.07	53.83	56.58

Interval	37	2	27	11	25	44	40	13	7	31	10	26	34	46	22	23
# of Boxes	645.75	635.98	627.66	627.04	546.28	541.83	527.88	520.44	514.33	514.33	453.82	423.24	418.27	408.80	400.55	358.10
cum	14448.94	15084.92	15712.58	16339.61	16885.89	17427.72	17955.60	18476.04	18990.37	19504.71	19958.52	20381.76	20800.03	21208.83	21609.37	21967.47
%	2.65	2.61	2.57	2.57	2.24	2.22	2.16	2.13	2.11	2.11	1.86	1.73	1.71	1.68	1.64	1.47
cum %	59.23	61.83	64.41	66.98	69.21	71.44	73.60	75.73	77.84	79.95	81.81	83.54	85.26	86.93	88.58	90.04

Interval	36	1	8	15	39	47	30	24	48	32	14	20	38	6	18	42
# of Boxes	346.08	303.84	303.63	277.67	277.67	215.37	125.31	112.94	110.42	90.88	88.36	88.36	88.36	0.00	0.00	0.00
cum	22313.55	22617.39	22921.02	23198.68	23476.35	23691.72	23817.03	23929.97	24040.39	24131.27	24219.63	24307.98	24396.34	24396.34	24396.34	24396.34
%	1.42	1.25	1.24	1.14	1.14	0.88	0.51	0.46	0.45	0.37	0.36	0.36	0.36	0.00	0.00	0.00
cum %	91.46	92.71	93.95	95.09	96.23	97.11	97.63	98.09	98.54	98.91	99.28	99.64	100.00	100.00	100.00	100.00

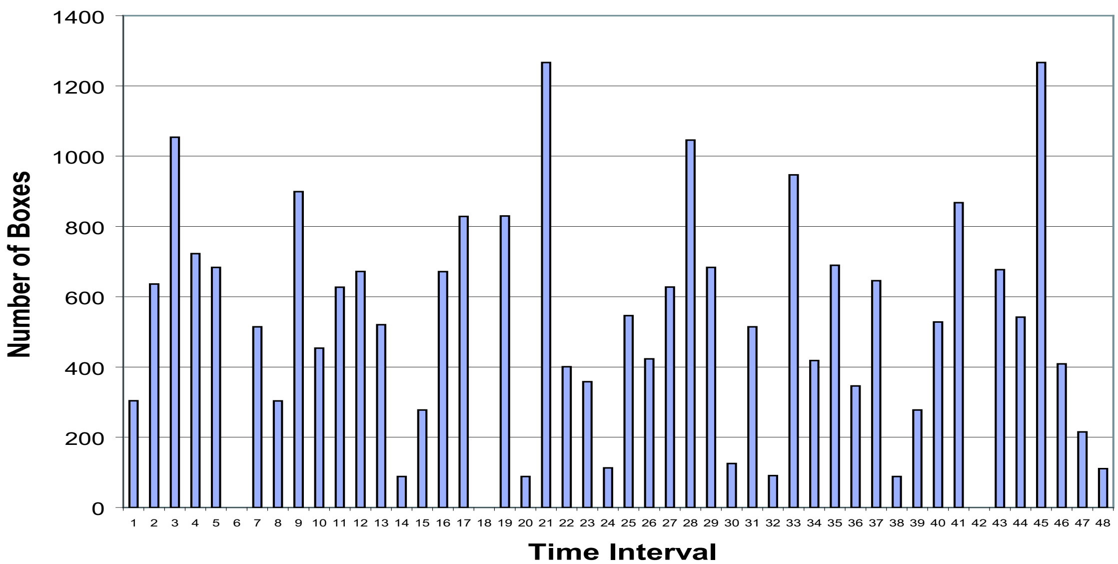

Figure 3.13: Number of boxes per 20 minute interval for current data

Table 3.3: Cumulative data per lane for current data

Lane	# of Boxes	%	cum	cum%
04	2779.77	11.39	2779.77	11.39
05	2639.16	10.82	5418.93	22.21
12	2377.67	9.75	7796.60	31.96
10	2241.34	9.19	10037.94	41.15
03	2128.93	8.73	12166.87	49.87
07	2125.36	8.71	14292.23	58.58
08	2109.15	8.65	16401.38	67.23
11	2041.64	8.37	18443.02	75.60
06	1879.33	7.70	20322.35	83.30
02	1612.53	6.61	21934.88	89.91
09	1552.88	6.37	23487.76	96.28
01	908.59	3.72	24396.35	100.00

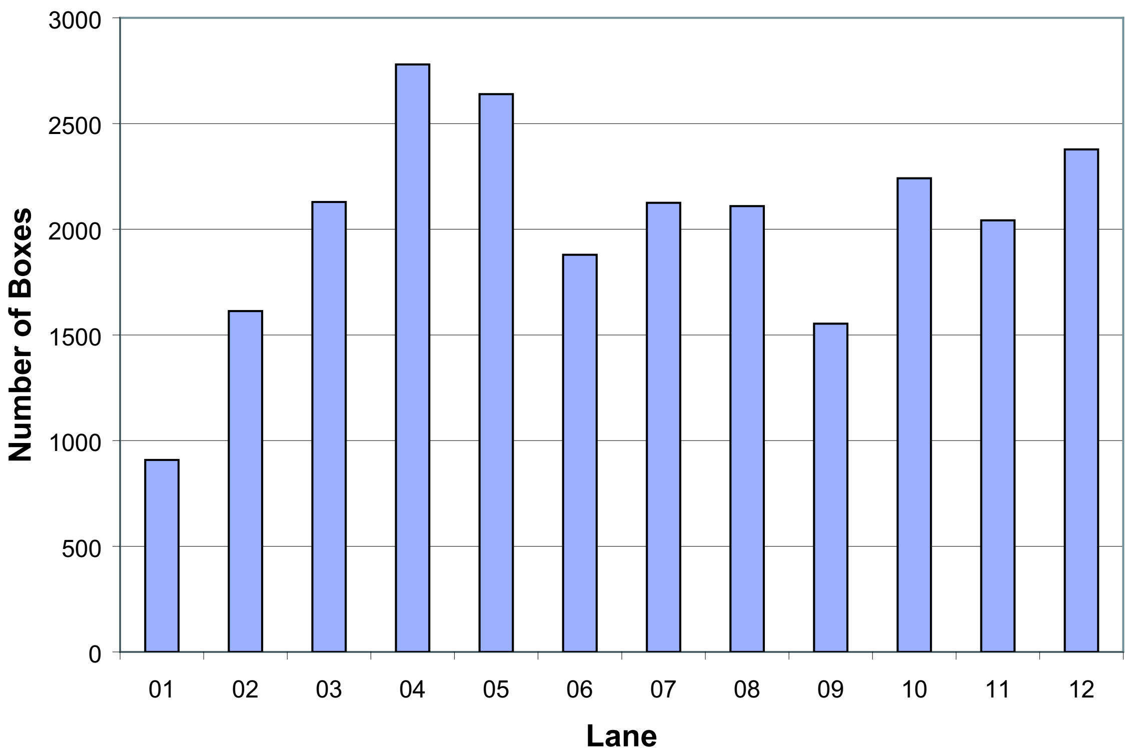

Figure 3.14: Number of boxes per lane for current data

A matrix with all possible lane combination for the crossdocking parts and the number of boxes that flow between them is shown in Table 3.4. From the 24396 parts 1784 or 7.31% have to be crossdocked.

Table 3.4: Flowmatrix for current data

exit	01	02	03	04	05	06	07	08	09	10	11	12
01	785.21		56.40	4.76					7.92			
02	40.01	1502.77	13.20	7.52	31.57	3.13			48.66	45.75	19.79	40.49
03			1877.12	14.30	25.00		12.74					50.44
04		5.30		2547.58			8.40		61.53	43.01	2.48	
05			20.50		2460.22				80.00	16.30	7.86	
06			19.77	47.33		1867.84			30.51	8.52	80.92	1.58
07			4.36		24.80	2.42	2096.50			39.73		
08		16.70	12.22					1861.89				
09	20.54	4.94	15.99	13.53	29.09				1487.58	31.83	9.16	35.07
10	26.99	35.33	42.28	13.26	68.48	5.94	7.72		66.56	2028.25	28.70	2.12
11	26.75	14.87	53.89	89.33					14.59	27.95	1882.44	33.00
12	9.09	32.62	13.20	42.16					2.79		10.29	2214.97

3.8 Toyota Data for the Proposed Changes

The proposed changes lead to a reduction of roughly 50% in the number of boxes that are handled in the crossdocking area, the other 50% are stored into the newly created flowracks. The new quantities per 20 minute bucket are shown in Table 3.5 and Figure 3.15.

Table 3.5: Number of boxes per 20 minute interval for data from Toyota's proposed changes

Interval	3	19	27	43	35	21	45	11	37	16	41	33	17	28	2	10
# of Boxes	522.98	380.72	356.76	355.16	354.59	341.15	341.15	336.48	331.34	321.01	308.10	301.88	298.68	283.38	279.73	277.36
cum	522.98	903.70	1260.46	1615.62	1970.21	2311.36	2652.51	2988.99	3320.33	3641.34	3949.43	4251.31	4549.98	4833.36	5113.09	5390.45
%	4.99	3.63	3.40	3.39	3.38	3.25	3.25	3.21	3.16	3.06	2.94	2.88	2.85	2.70	2.67	2.65
cum %	4.99	8.62	12.02	15.41	18.79	22.05	25.30	28.51	31.67	34.74	37.68	40.56	43.40	46.11	48.78	51.42

Interval	25	13	4	46	34	40	5	29	9	22	7	31	8	23	15	39
# of Boxes	274.70	272.70	263.54	262.56	261.61	257.37	255.08	255.08	254.49	254.31	234.77	234.77	194.88	193.39	173.10	173.10
cum	5665.15	5937.85	6201.39	6463.94	6725.55	6982.92	7238.00	7493.09	7747.58	8001.89	8236.66	8471.43	8666.31	8859.70	9032.80	9205.91
%	2.62	2.60	2.51	2.50	2.50	2.46	2.43	2.43	2.43	2.43	2.24	2.24	1.86	1.84	1.65	1.65
cum %	54.04	56.64	59.16	61.66	64.16	66.61	69.05	71.48	73.91	76.33	78.57	80.81	82.67	84.52	86.17	87.82

Interval	26	44	1	12	47	36	32	14	20	38	30	24	48	6	18	42
# of Boxes	167.21	163.39	162.18	121.96	117.33	99.57	82.36	79.83	79.83	79.83	58.64	33.59	31.06	0.00	0.00	0.00
cum	9373.12	9536.51	9698.69	9820.65	9937.98	10037.55	10119.91	10199.74	10279.57	10359.40	10418.04	10451.63	10482.69	10482.69	10482.69	10482.69
%	1.60	1.56	1.55	1.16	1.12	0.95	0.79	0.76	0.76	0.76	0.56	0.32	0.30	0.00	0.00	0.00
cum %	89.42	90.97	92.52	93.68	94.80	95.75	96.54	97.30	98.06	98.82	99.38	99.70	100.00	100.00	100.00	100.00

The quantities per lane are shown in cumulative form in Table 3.6 . Figure 3.16 pictures the boxes per lane in graphical form, and Figure 3.17 compares the current quantities to

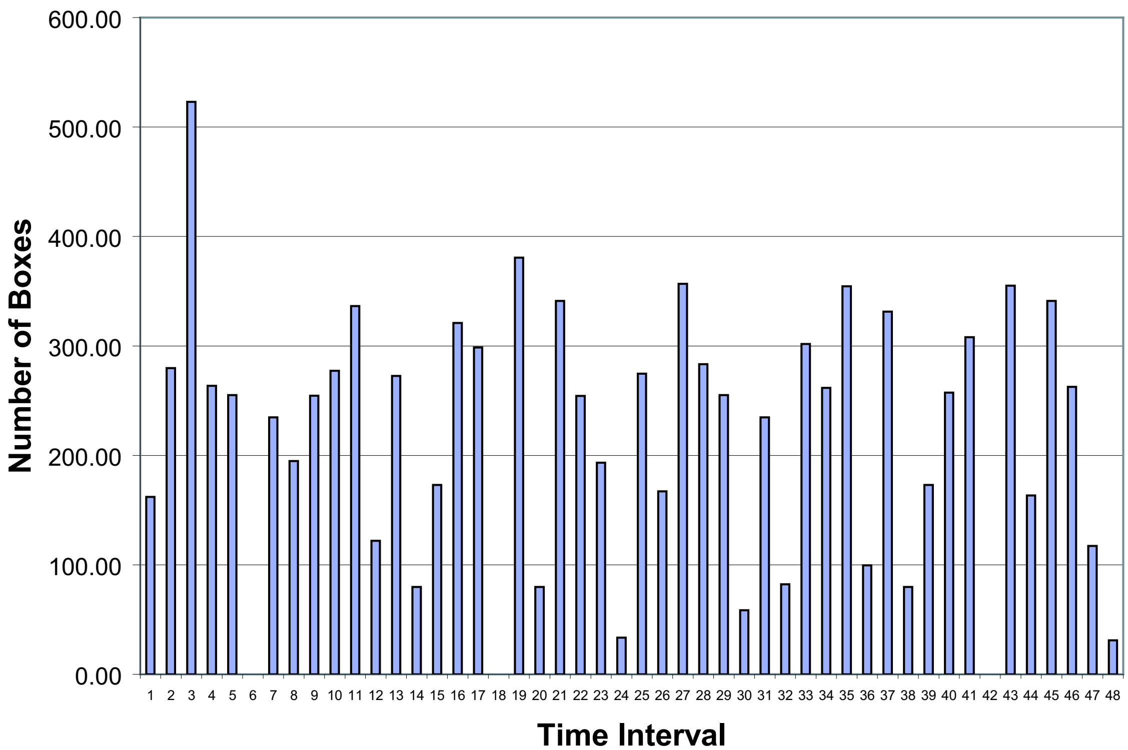

Figure 3.15: Number of boxes per 20 minute interval for data from Toyota's proposed changes

the reduced quantities per lane as of the proposed changes. The reduction is not distributed equally over all lanes, lane 1 has only a 4 % reduction whereas lane 4 has nearly a 75 % reduction.

Table 3.6: Cumulative data boxes per lane for data from Toyota's proposed changes

Lane	# of Boxes	%	cum	cum%
05	1232.62	11.76	1232.62	11.76
11	1181.45	11.27	2414.07	23.03
03	1094.59	10.44	3508.66	33.47
07	1072.37	10.23	4581.03	43.70
08	1059.30	10.11	5640.33	53.81
10	913.01	8.71	6553.34	62.52
01	871.97	8.32	7425.31	70.83
04	703.42	6.71	8128.73	77.54
12	669.89	6.39	8798.62	83.93
06	665.60	6.35	9464.22	90.28
02	627.66	5.99	10091.88	96.27
09	390.81	3.73	10482.69	100.00

The quantities that have to be crossdocked between the lanes are illustrated in Table 3.7. From the 10482 parts 1577 or 15.04% have to be crossdocked.

Table 3.7: Flowmatrix for data from Toyota's proposed changes

to / from	01	02	03	04	05	06	07	08	09	10	11	12
01	684.20	24.14	33.00				5.54		14.87	13.20	27.29	
02		488.68	20.91	16.05			6.09					
03	7.86		879.10				13.20		10.42		8.73	18.18
04		1.13	61.80	603.69		45.63			51.33		12.60	29.64
05				22.30	1160.78		21.60		6.82	52.69		
06	4.16	29.25			20.36	495.25	0.73			6.37	15.18	10.92
07		32.57	50.44		11.13		868.77		27.28		27.20	
08			1.58		30.78	47.33	63.56	989.78		8.52		
09	16.08	9.41	40.49			20.69	17.16		323.17	29.48	35.53	19.79
10	105.14	42.48	7.27	58.59		17.49	9.35		14.03	794.85	1.90	5.30
11	16.50				9.57	38.97	19.80		12.41	7.90	1053.02	21.82
12	38.03			2.79		0.24	46.57					564.24

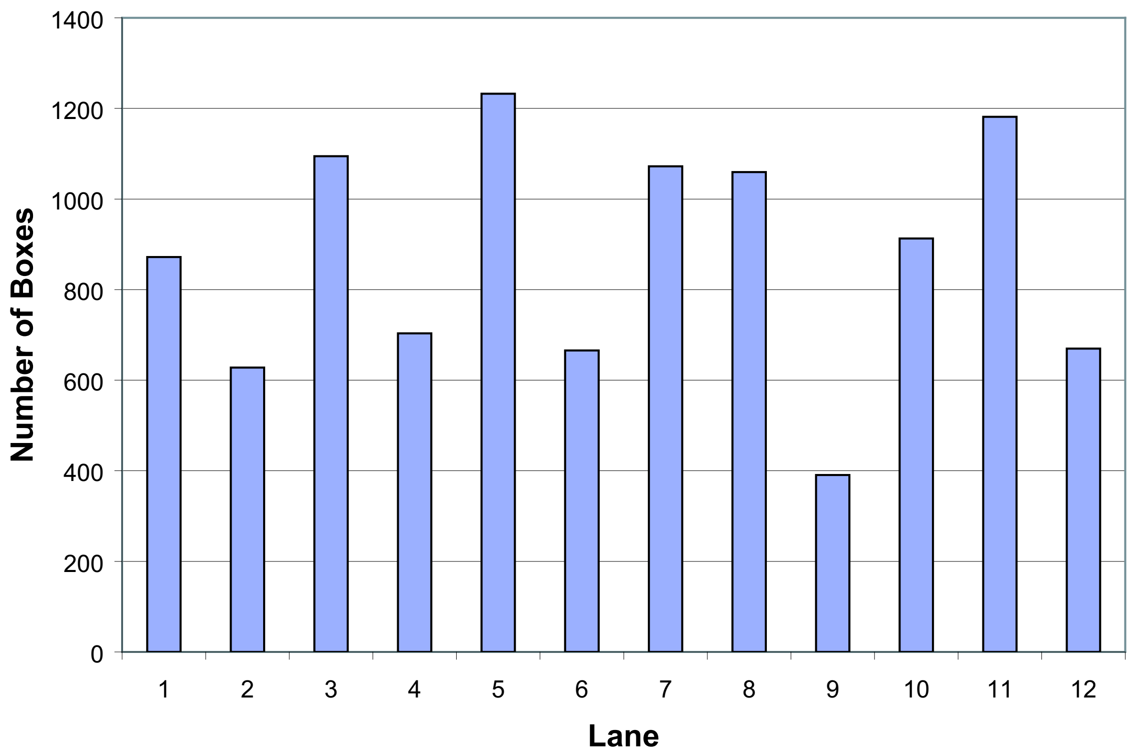

Figure 3.16: Number of boxes per lane for data from Toyota’s proposed changes

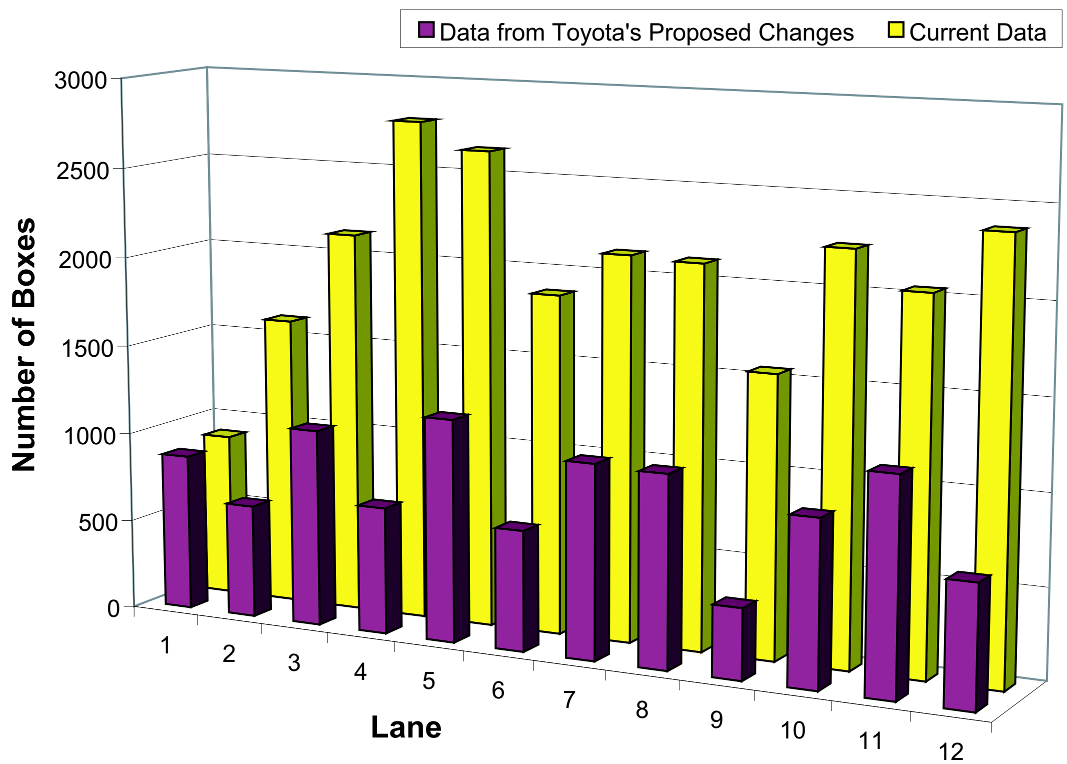

Figure 3.17: Comparison of differences in number of boxes per lane using current data and data from Toyota's proposed changes

3.9 Calculation of the Distances

The distances were measured at the Toyota plant and some minor adjustments made. Obstacles such as steel beams and other structures specific to the Toyota plant were taken out of the distance calculation. The dollies are 4 feet wide and 7 feet long. The lanes are 5 feet wide; the distance between 2 lanes is 14 feet, and the distance between areas is also 14 feet. If there is a row of dollies, the midpoint of the row is used to calculate the travel distance. Because in the new layouts, dollies partially transport the parts, the distance measurement is split into walking distance for the team members and driving distance for the dollies. The following section provides an overview of the measurements of each layout, the walking and driving distances, and an example on how the travel distances are calculated.

3.9.1 Assumptions for all Layouts

- With the exception of the new layout proposed by Toyota, there are always 5 dollies for each line in the crossdocking area.
- Each box is transported separately (worst case scenario).
- Only the travel distances from/to the lane are calculated. Switching lanes is not included in the travel distance (for example, when crossdocking 5 boxes from lane 3 to 7 and after that crossdocking 2 boxes from 1 to 4, the calculation would assume that the team member walks back to lane 3 after delivering the last box to lane 7, and then the next calculation would start with delivering the first box from lane 1 to 4)
- The distance from the unloading area, where the forklift unloads the pallets onto the dollies, to the first unloading point is not included in the calculations of any of the layouts.

3.9.2 Original Layout

The original layout with measurements is shown in Figure 3.18. In this layout it is assumed that the team members walk to transport the boxes from one lane to the other. Table 3.8 shows the walking distances between the lanes for the original layout.

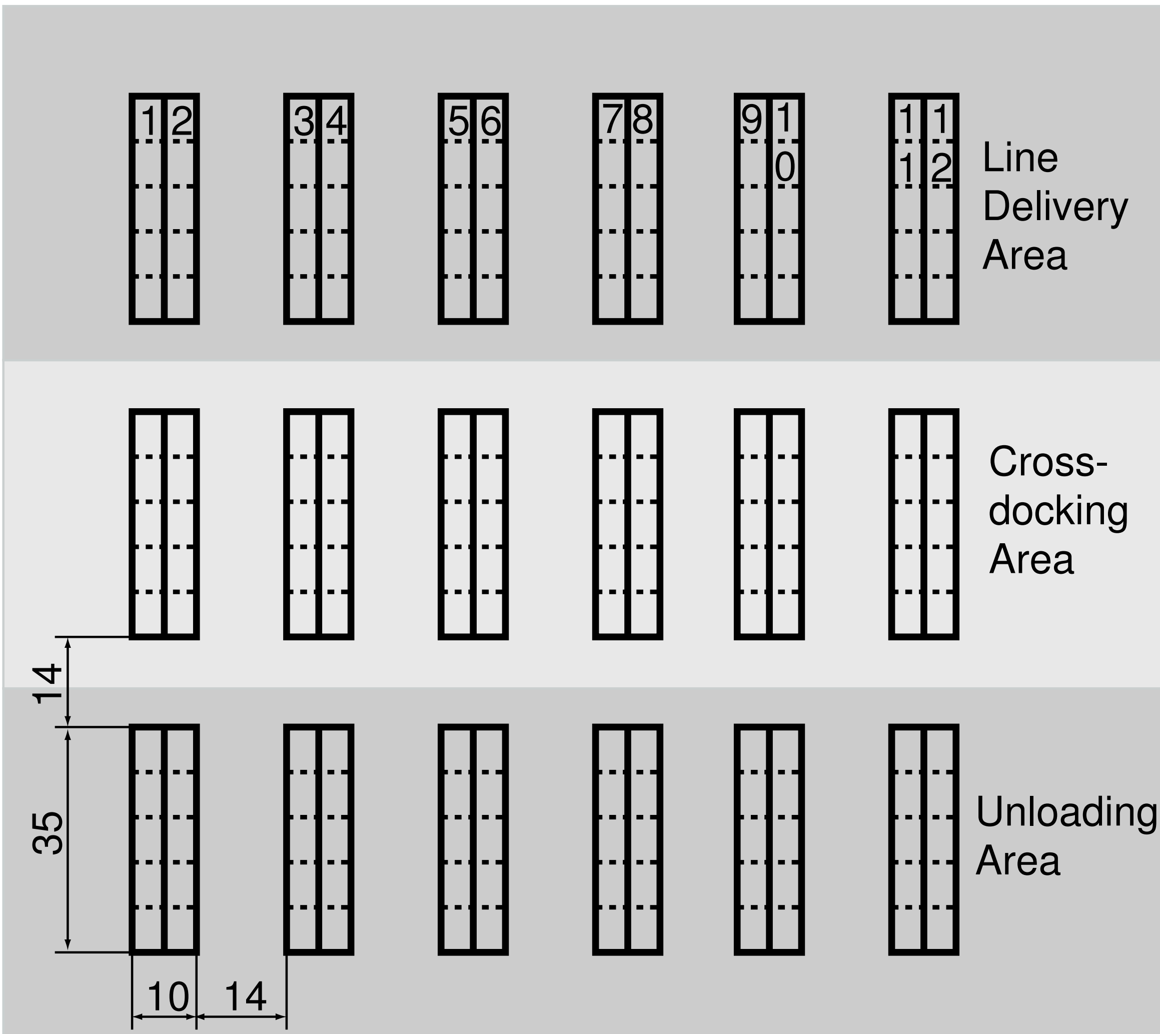

Figure 3.18: Crossdocking area original layout with measurements

Table 3.8: Travel distances original layout

to / from	01	02	03	04	05	06	07	08	09	10	11	12
01		90	118	138	166	186	214	234	262	282	310	330
02	90		28	118	138	166	186	214	234	262	282	310
03	118	28		90	118	138	166	186	214	234	262	282
04	138	118	90		28	118	138	166	186	214	234	262
05	166	138	118	28		90	118	138	166	186	214	234
06	186	166	138	118	90		28	118	138	166	186	214
07	214	186	166	138	118	28		90	118	138	166	186
08	234	214	186	166	138	118	90		28	118	138	166
09	262	234	214	186	166	138	118	28		90	118	138
10	282	262	234	214	186	166	138	118	90		28	118
11	310	282	262	234	214	186	166	138	118	28		90
12	330	310	282	262	234	214	186	166	138	118	90	

Example: 5 boxes have to be crossdocked from lane 3 to 7

- team member walks from midpoint of lane 3 to end of lane: 17.5 feet
- from end of lane to beginning of lane 7: 10 (lane 3 to 4) + 14 (lane 4 to 5) + 10 (lane 5 to 6) + 14 (lane 6 to 7) = 48 feet
- from beginning of lane 7 to midpoint of lane 7= 17.5 feet
- distance one way: 17.5 + 10 + 14 + 10 + 14 + 17.5= 83 feet
- distance for one box: 83 x 2 (walk back to lane 3) = 166 feet
- total distance for five boxes: 5 x 166 feet = 830 feet

3.9.3 New Layout 1

Figure 3.19 shows the measurements of the crossdocking area for the first new layout. In this layout, the pallets that do not need to be crossdocked are transported from the truck directly to the lane delivery area, and team members handle only the crossdocking pallets. The dollies with the crossdocking pallets are driven to the unloading points, and from there a team member carries the boxes to the dollies for the specific lanes. Table 3.9 shows the travel distances of the dollies for all lane combinations. The walking distances for the team members are always 63 feet: 31.5 feet from the unloading point to the dolly and 31.5 feet back.

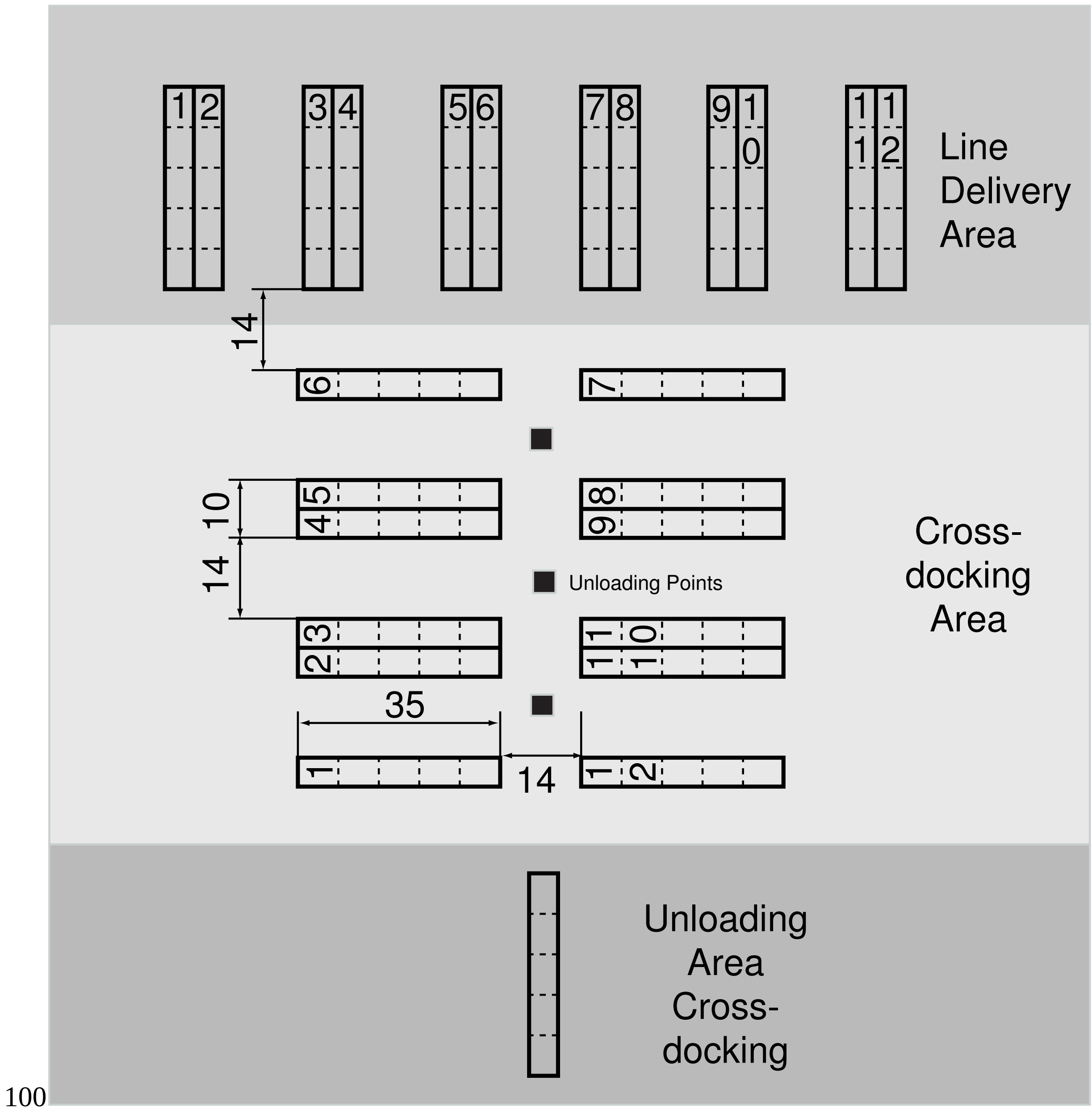

Figure 3.19: Crossdocking area of new layout 1, with measurements

Table 3.9: Travel distances new layout 1

to / from	01	02	03	04	05	06	07	08	09	10	11	12
01			24	24	48	48	48	48	24	24		
02			24	24	48	48	48	48	24	24		
03	24	24			24	24	24	24			24	24
04	24	24			24	24	24	24			24	24
05	48	48	24	24					24	24	48	48
06	48	48	24	24					24	24	48	48
07	48	48	24	24					24	24	48	48
08	48	48	24	24					24	24	48	48
09	24	24			24	24	24	24			24	24
10	24	24			24	24	24	24			24	24
11			24	24	48	48	48	48	24	24		
12			24	24	48	48	48	48	24	24		

Example: 5 boxes have to be crossdocked from lane 3 to 7

- team member drives dollies from unloading point of lane 3 to unloading point of lane 7: 24 feet driving distance
- walking distance for one box: 63 feet
- total distance for five boxes: 5 x 63 feet walking distance + 5 x 24 feet driving distance = 315 feet walking distance + 120 feet driving distance

3.9.4 New Layout 2

The crossdocking area with measurements for the second new layout is shown in Figure 3.20. The travel distances for the dollies are shown in Table 3.10. The walking distance for unloading the dollies is 63 feet for all lane combinations except from all lanes to lanes 5 and 9, where the walking distance is 33.

Example: 5 boxes have to be crossdocked from lane 3 to 7

- team member drives dollies from unloading point of lane 3 to unloading point of lane 7: 55 feet driving distance
- walking distance for one box: 63 feet

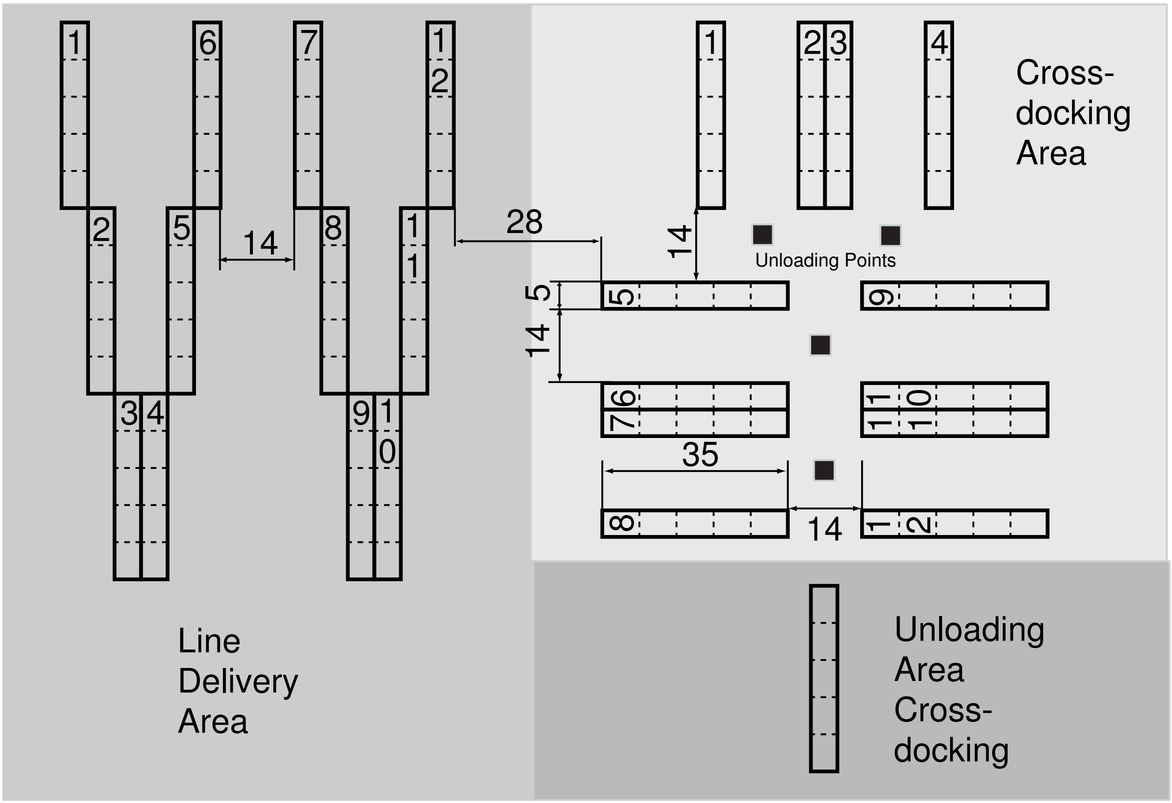

Figure 3.20: Crossdocking area of new layout 2, with measurements

Table 3.10: Travel distances new layout 2

from \ to	01	02	03	04	05	06	07	08	09	10	11	12
01			24	24		31	55	55	24	31	55	55
02			24	24		31	55	55	24	31	55	55
03	24	24			24	31	55	55		31	55	55
04	24	24			24	31	55	55		31	55	55
05			24	24		31	55	55	24	31	55	55
06	31	31	31	31	31		24	24			24	24
07	55	55	55	55	55	24			24	24		
08	55	55	55	55	55	24			24	24		
09	24	24			24		24	24		31	55	55
10	31	31	31	31	31		24	24	31		24	24
11	55	55	55	55	55	24			55	24		
12	55	55	55	55	55	24			55	24		

- total distance for five boxes: 5 x 63 feet walking distance + 5 x 55 feet driving distance = 315 feet walking distance + 275 feet driving distance

3.9.5 New Layout 3

The measurements for the crossdocking area for the third new layout are shown in Figure 3.21.

The driving distances for the dollies are shown in Table 3.11. The walking distances are 19 feet from all lanes to lanes 3/4/9 and 10, 29 feet from all lanes to lanes 2,5,8 and 11, and 39 feet from all lanes to lanes 1/6/7/12, respectively.

Table 3.11: Travel distances new layout 3

from \ to	01	02	03	04	05	06	07	08	09	10	11	12
01		35	70	70	35		241	206	171	171	206	241
02	35		35	35		35	276	241	206	206	241	241
03	70	35			35	70	311	276	241	241	276	311
04	70	35			35	70	311	276	241	241	276	311
05	35		35	35		35	276	241	206	206	241	241
06		35	70	70	35		241	206	171	171	206	241
07	241	206	171	171	206	241		35	70	70	35	
08	276	241	206	206	241	241	35		35	35		35
09	311	276	241	241	276	311	70	35			35	70
10	311	276	241	241	276	311	70	35			35	70
11	276	241	206	206	241	241	35		35	35		35
12	241	206	171	171	206	241		35	70	70	35	

Example: 5 boxes have to be crossdocked from lane 3 to 7

- team member drives dollies from unloading point of lane 3 to unloading point of lane 7: 276 feet driving distance
- walking distance for one box: 39 feet
- total distance for five boxes: 5 * 39 feet walking distance + 5 * 276 feet driving distance = 195 feet walking distance + 1380 feet driving distance

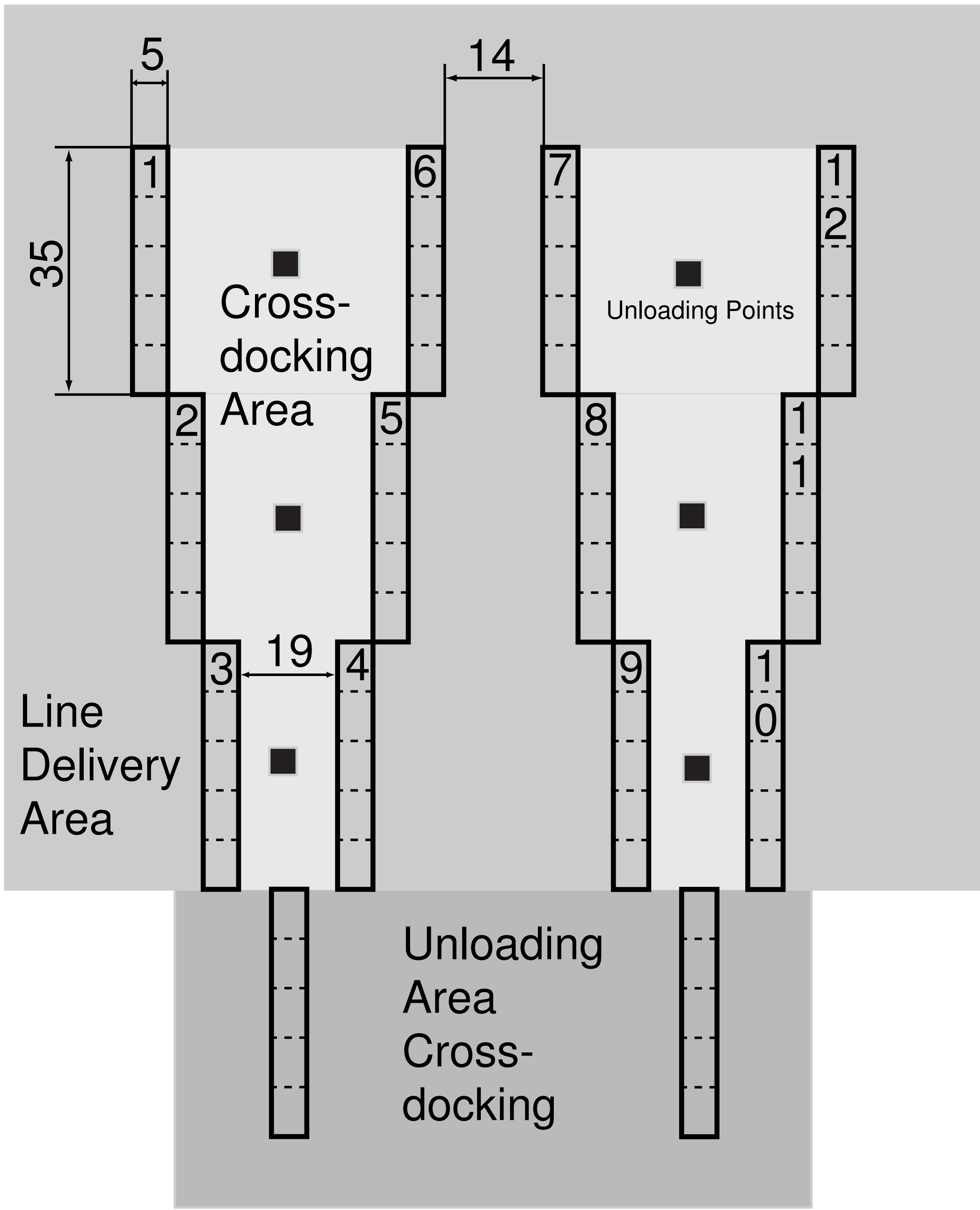

Figure 3.21: Crossdocking area of new layout 3, with measurements

3.9.6 New Layout Proposed by Toyota

The crossdocking area with measurements for the proposed new layout is shown in Figure 3.22.

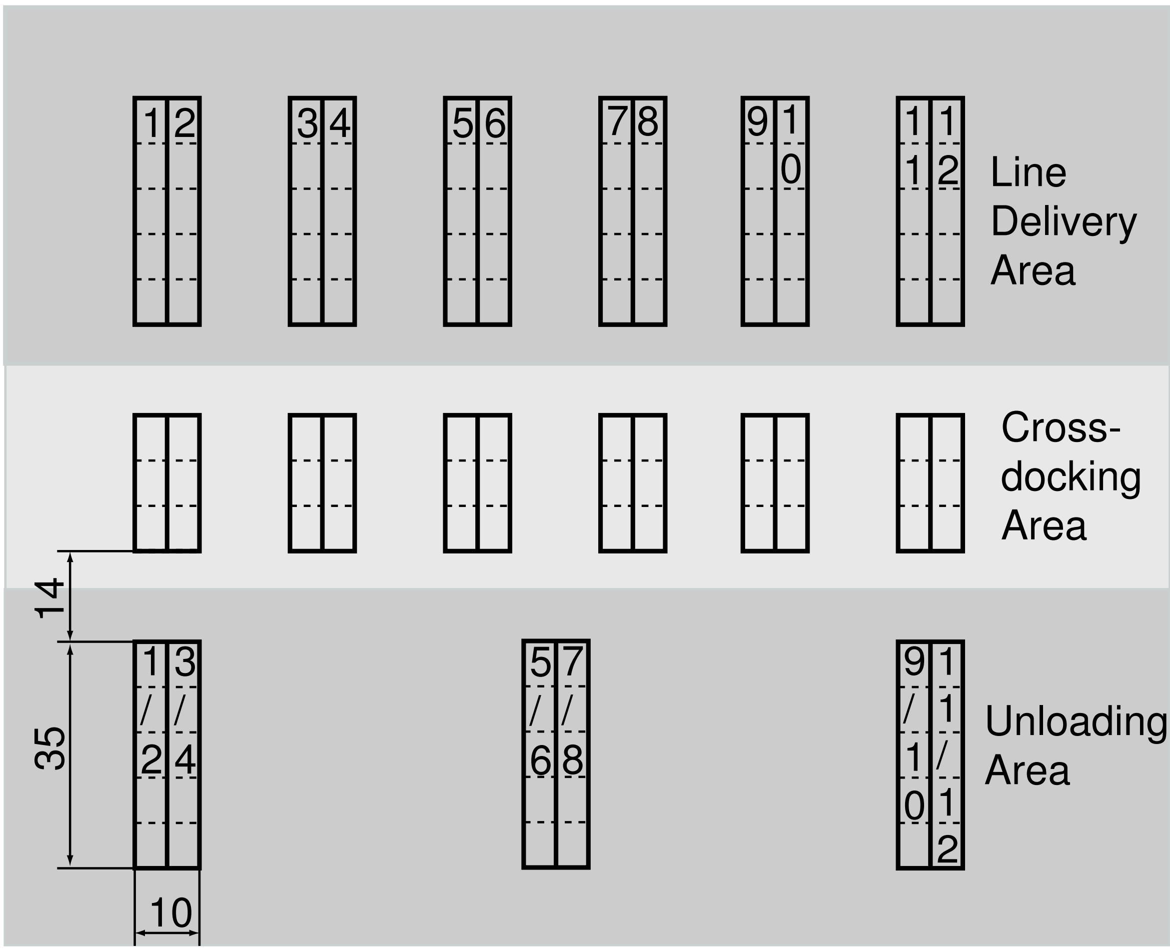

Figure 3.22: Crossdocking area of layout proposed by Toyota with measurements

The walking distances for the dollies are shown in Table 3.12. Since only 3 dollies are used, they are the same as for the original layout minus 24 feet.

Example: 5 boxes have to be crossdocked from lane 3 to 7

- team member walks from midpoint of lane 3 to end of lane: 10.5 feet

Table 3.12: Travel distances for layout proposed by Toyota

to / from	01	02	03	04	05	06	07	08	09	10	11	12
01		62	90	110	138	158	186	206	234	254	282	302
02	62		28	90	110	138	158	186	206	234	254	282
03	90	28		62	90	110	138	158	186	206	234	254
04	110	90	62		28	90	110	138	158	186	206	234
05	138	110	90	28		62	90	110	138	158	186	206
06	158	138	110	90	62		28	90	110	138	158	186
07	186	158	138	110	90	28		62	90	110	138	158
08	206	186	158	138	110	90	62		28	90	110	138
09	234	206	186	158	138	110	90	28		62	90	110
10	254	234	206	186	158	138	110	90	62		28	90
11	282	254	234	206	186	158	138	110	90	28		62
12	302	282	254	234	206	186	158	138	110	90	62	

- from end of lane to beginning of lane 7: 10 (lane 3 to 4) + 14 (lane 4 to 5) + 10 (lane 5 to 6) + 14 (lane 6 to 7) = 48 feet
- from beginning of lane 7 to midpoint of lane 7= 10.5 feet
- distance one way: 10.5 + 10 + 14 + 10 + 14 + 10.5= 69 feet
- distance for one box: 69 x 2 (walk back to lane 3) = 138 feet
- total distance for five boxes: 5 x 138 feet = 690 feet

Chapter 4

Analysis and Results of Simulations

This chapter presents the details of analysis and the results of the simulations. The analysis of the simulation results was split into two sets: the current volume of incoming parts, and the future new volume that would go into the crossdocking area. Overall, 15 simulations were run and analyzed using the current data and 15 using the future data. Since in the three newly designed layouts the parts are partially transported by dollies, the initial measurement of travel distance was divided into two measurements: walking distance and dolly/driving distance. The data provided by Toyota was used to fit distribution functions to the incoming quantities using the Input Analyzer included in the simulation software. The distribution functions were then used to create the incoming quantities in the simulation. The quantities were then distributed into the lanes using the existing percentages of crossdocking activities. The analysis of the data revealed that the current data had a 7.31 % level of crossdocking activity and the future data had a level 15.04 %; thus 25 % was chosen as the third percentage of crossdocking activity. Each replication was run for 16 work hours, reflecting the two 8 hour shifts at Toyota. 1000 replications were run for each simulation.

4.1 Results from the Current Data

The input analyzer found that a Weibull distribution with a scale parameter of 8.2 and a shape parameter of 0.82222 showed the best fit for the current volume of incoming parts.

The percentages of crossdocking activity between the lanes is shown in Table 4.1. Overall for the current data, only 7.31% of all boxes have to be crossdocked, with no predominant lane combination and no lane combination having more than 0.37% of crossdocking activity.

Table 4.1: Crossdocking activity between lanes in percentages for current data

to / from	01	02	03	04	05	06	07	08	09	10	11	12
01	3.22		0.23	0.02					0.03			
02	0.16	6.16	0.05	0.03	0.13	0.01			0.20	0.19	0.08	0.17
03			7.69	0.06	0.10		0.05					0.21
04		0.02		10.44			0.03		0.25	0.18	0.01	
05			0.08		10.08				0.33	0.07	0.03	
06			0.08	0.19		7.66			0.13	0.03	0.33	0.01
07			0.02		0.10	0.01	8.59			0.16		
08		0.07	0.05					7.63				
09	0.08	0.02	0.07	0.06	0.12				6.10	0.13	0.04	0.14
10	0.11	0.14	0.17	0.05	0.28	0.02	0.03		0.27	8.31	0.12	0.01
11	0.11	0.06	0.22	0.37					0.06	0.11	7.72	0.14
12	0.04	0.13	0.05	0.17					0.01		0.04	9.08

Of the possible 132 lane combinations, roughly half (64 NCP vs. 68 CP) have no crossdocking activity at all. For the additional crossdocking levels, the percentages per lane were adjusted to reflect the original distribution; in other words, a lane combination with 0.2% of crossdocking activity at 7.31% will have approximately 0.4% at 15.04%.

4.1.1 Results for Crossdocking Activity Levels

The initial assumption/speculation that the ideal layout depends on the level of crossdocking activity did not hold. For all layouts, the travel distance doubled as the crossdocking level doubled; there seems to be a linear dependency between the two factors. This may be largely due to the fact that the differences in the crossdocking activities between the lanes are not very large, (between 0.01% and 0.37%) and because the percentage distribution for the lanes also increased linearly.

A future research project could study the influence of different percentage distributions on the layouts.

4.1.2 Results for Different Layouts

Because the distance measurements had to be separated into walking distance and driving distance, two different sets of analyses were performed. In the first set, the two measurements are added up to a total distance, making no distinction between walking and dolly distance. This measurement is worse than the actual performance of the system because dollies usually travel faster than people. In the second set of analyses, the assumption was made that dollies travel 3x as fast as the team members can walk.

Results for total distance = walking distance + dolly distance First, an overall analysis of variance (ANOVA) was performed at an alpha level of 0.01 to confirm that the null hypothesis (no difference between the layouts) could be rejected. The result of the ANOVA and some summary data for all four layouts are provided in Table 4.2.

Table 4.2: ANOVA total distance for current data

SUMMARY

Groups	*Count*	*Sum*	*Average*	*Variance*	*Difference*
Original Layout	1000	306892990	306893	1371536008	
New Layout 1	1000	135740538	135741	249775884	171152
New Layout 2	1000	144083423	144083	297435069	162810
New Layout 3	1000	321062858.1	321063	1659724978	-14170
Future Layout	1000	262072580	262073	1037334960	44820

ANOVA

Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	31421307347345	4	7855326836836	8509.16	0.00000	3.32
Within Groups	4611191092880	4995	923161380			
Total	36032498440224	4999				

The analysis shows that there is a statistically significant difference between the average total distance of at least two of the four layouts.

A set of planned, simple comparisons was then performed using the original layout as a control group. Table 4.3 presents the results of the individual t-test. Because the number of planned comparisons is equal to the degrees of freedom between groups, and most sources recommend an adjustment of alpha when the number of comparisons exceeds the degrees of freedom between groups [Sheshkin, 2000], the alpha level was reduced to 0.0025 for the individual t-tests. All tests show a statistically significant difference between the total travel distance of the original layout and the total travel distances of the other layouts. In addition, the minimum required difference for the Scheffé test (designated CD_S) at an alpha level of 0.05 for two means to be significantly different was computed using equation 4.1.

$$CD_S = \sqrt{(k-1(F_{(df_{BG},df_{WG})}}\sqrt{\frac{2MS_{WG}}{n}} \tag{4.1}$$

where the value $F_{(df_{BG},df_{WG})}$ is the tabled critical value that is used for the omnibus F test;

Table 4.3: Individual t-tests for current data

	Original Layout	New Layout 1	New Layout 2	New Layout 3	Future Layout
Mean	306893	135741	144083	321063	262073
Variance	1371536008	249775884	297435069	1659724978	1037334960
Observations	1000	1000	1000	1000	1000
df	1998				
t Stat		134.42	126.02	-8.14	28.88
P-value		<0.0001	<0.0001	<0.0001	<0.0001

k is the number of groups, k=5 in our case;

df_{BG} are the degrees of freedom between groups,

df_{WG} are the degrees of freedom within groups,

and MS_{WG} is the mean square value within groups from the ANOVA Table.

The critical difference for two mean to be significantly different is 4184. Comparing this difference to the differences from Table 4.2 confirms the results of the t-test that statistically all layouts are significantly different regarding the total travel distance.

New layout 1 resulted in the lowest overall travel distance, with a reduction of 55.77%. New layout 2 also shows a significant improvement, with a reduction of 53.05%. Layout 3 performed the worst, with an increase of 4.62% in total travel distance. The use of only three dollies in the future layout accounts for the decrease in travel distance between the original layout and the future layout.

Results for total distance = walking distance + dolly distance/3 The second analysis assumed that dollies drive 3 times faster than team members are able to walk; thus dolly distances were divided by 3.

Table 4.4 furnishes the data for the overall ANOVA.

Since the distances are all lower than they were before individual tests were only necessary for the new layout 3, which resulted in an increase in travel distance for the first test but now also shows an improvement over the original layout. The result of the t-test between the original layout and layout 3 using the faster dolly time is shown in Table 4.5 .

Using the new, more realistic travel distances, layout 2 performed best, with a reduction of 64.52%; layout 1 showed an improvement of 62.43%, and layout 3 now also has a 56.01% reduced overall travel distance.

Table 4.4: ANOVA total distance = walking distance + dolly distance/3 for current data

SUMMARY

Groups	Count	Sum	Average	Variance
Original Layout	1000	306892990	306893	1371536008
New Layout 1	1000	115287671	115288	176886569
New Layout 2	1000	108890834	108891	167291204
New Layout 3	1000	134988125	134988	276760924
Future Layout	1000	262072580	262073	1037334960

ANOVA

Source of Variation	SS	df	MS	F	P-value	F crit
Between Groups	33949723195870	4	8487430798968	14006.54	0.00000	3.32
Within Groups	3026779855899	4995	605961933			
Total	36976503051769	4999				

Table 4.5: T-test results: Original layout vs. new layout 3 for current data

	Original Layout	*New Layout 3*
Mean	306893	134988
Variance	1371536008	276760924
Observations	1000	1000
df	1998	
t Stat		133.90
P-vlaue		<0.0001

To better determine the source of the reduction in total travel distance, the walking distance and dolly distances of the new layouts were compared. The results are given in Tables 4.6 and 4.7.

Table 4.6: Comparison of walking distance for current data

SUMMARY

Groups	*Count*	*Sum*	*Average*	*Variance*
New Layout 1	1000	105061237	105061	146920593
New Layout 2	1000	91294539	91295	118363864
New Layout 3	1000	41950758	41951	25677752

ANOVA

Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	2202421074691	2	1101210537346	11354.16	0.00000	4.61
Within Groups	290671246916	2997	96987403			
Total	2493092321607	2999				

Table 4.7: Comparison of dolly distance for current data

SUMMARY

Groups	*Count*	*Sum*	*Average*	*Variance*
New Layout 1	1000	30679301	30679	19436043
New Layout 2	1000	52788884	52789	48433778
New Layout 3	1000	279112100	279112	1321196567

ANOVA

Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	37809962459997	2	18904981229999	40829.54	0.00000	4.61
Within Groups	1387677321631	2997	463022129			
Total	39197639781628	2999				

A Scheffé test was used to compute the critical difference for two means. The critical difference for the walking distances 2357 and for the dolly distance, 1079. Looking at the average distances from Table 4.6 and 4.7, all distances show a statistically significant difference at an alpha level of 0.05. The third new layout outperformed the other layouts in walking distance by over 50%, but it has a very high dolly distance, largely because the distances are very large when a pallet must be transported from one V to the other (see layout Figure 3.3 page 21). If the suppliers were to pack the containers according to the two areas, the performance of this layout could be improved dramatically.

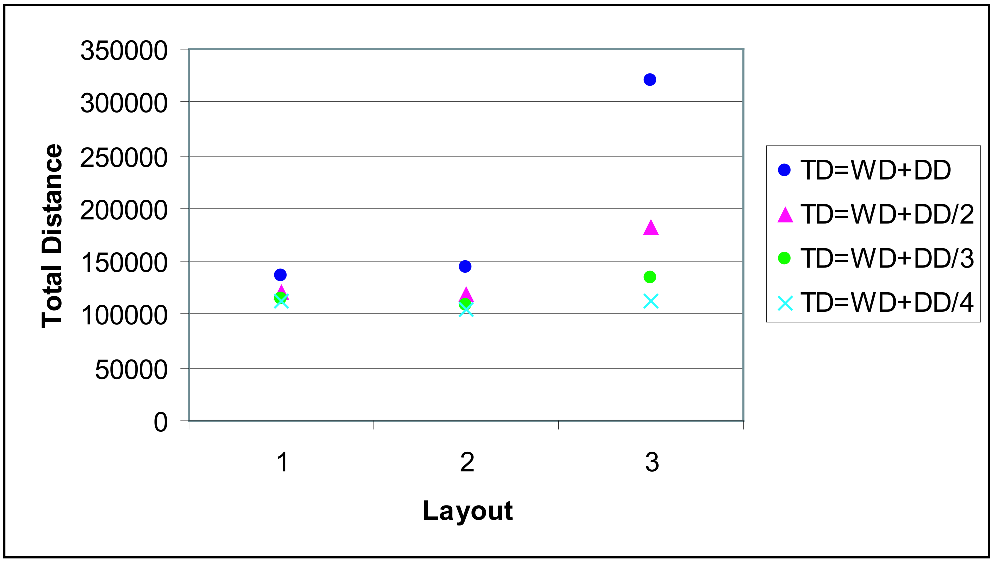

Figure 4.1: Influence of dolly speed on layout performance

Figure 4.1 illustrates how the speed of driving influences the layouts. Four different total distances were calculated for each layout. In the first series, the total distance (TD) equals the walking distance (WD) plus the driving distance (DD); in the second series, the DD is divided by two, which would mean that the dollies drive twice as fast as people walk. In the third series, the DD is divided by three, and in the forth series by four. Whereas the improvements in total speed, shown in Table 4.8, are only between 11.3 % and 16.95% for the first layout, layout 3 already has an improvement of 43.47 % when the speed is doubled and 65.2 % when the speed is four times the speed of walking.

Table 4.8: Improvements of layouts by speed of dollies

TD=WD+DD	TD=WD+DD/2	Impr. %	TD=WD+DD/3	Impr. %	TD=WD+DD/4	Impr. %
135740	120401	11.30	115287	15.07	112731	16.95
144084	117689	18.32	108891	24.43	104492	27.48
321063	181507	43.47	134988	57.96	111729	65.20

The next section contains the results of the analyses for the future data.

4.2 Results from the Data of Toyota's Proposed Changes

The input analyzer found that an exponential distribution with a mean of 5.27 resulted in the best fit of the future volume of parts handled in the crossdocking area.

The percentages per lane for the future data are shown in Table 4.9 Although the incoming

Table 4.9: Crossdocking activity between lanes in percentages for data from Toyota's proposed changes

to / from	01	02	03	04	05	06	07	08	09	10	11	12
01	6.53	0.23	0.31				0.05		0.14	0.13	0.26	
02		4.66	0.20	0.15			0.06					
03	0.07		8.39				0.13		0.10		0.08	0.17
04		0.01	0.59	5.76		0.44			0.49		0.12	0.28
05				0.21	11.07		0.21		0.07	0.50		
06	0.04	0.28			0.19	4.72	0.01			0.06	0.14	0.10
07		0.31	0.48		0.11		8.29		0.26		0.26	
08			0.02		0.29	0.45	0.61	9.44		0.08		
09	0.15	0.09	0.39			0.20	0.16		3.08	0.28	0.34	0.19
10	1.00	0.41	0.07	0.56		0.17	0.09		0.13	7.58	0.02	0.05
11	0.16				0.09	0.37	0.19		0.12	0.08	10.05	0.21
12	0.36			0.03			0.44					5.38

volume was reduced by over 50% (24396 current vs. 10483 parts future), the percentage of parts that need crossdocking doubled (7.31% vs. 15.04%). The number of lane combinations with crossdocking activities stayed the same with 68, but the highest percentage of boxes one lane combinations has gone up from 0.37% to 1%.

The same statistical test as for the current data were performed for the future data, starting with an overall ANOVA to ensure that there is a difference between the total travel distances of the layouts, and then performing individual t-tests, this time using the future layout as the control group.

4.2.1 Results for Crossdocking Activity Levels

Because the analyses of the different crossdocking levels for the current data revealed a linear relationship, no further analysis was done for the future data.

4.2.2 Results for Different Layouts

Just as was done for the current incoming volume of parts, two sets of analyses were performed for the future incoming volume of parts. First, the total distance was calculated by simply adding walking distance and dolly distance; second, assuming a faster travel time for dollies the dolly distance was divided by 3 .

Results for total distance = walking distance + dolly distance From the results of the ANOVA (Table 4.10) it is inferred that there is a statistically significant difference at the 0.01 level between the average total distances for at least two of the four layouts.

Table 4.10: ANOVA total distance for data from Toyota's proposed changes

SUMMARY

Groups	*Count*	*Sum*	*Average*	*Variance*	*Difference*
Future Layout	1000	291640620	291641	577799668	
New Layout 1	1000	118850826	118851	107970952	172790
New Layout 2	1000	125382299	125382	106949508	166258
New Layout 3	1000	363713718	363714	920333710	-72073
Original Layout	1000	345543430	345543	769019977	-53903

ANOVA

Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	56518233454042	4	14129558363510	28463.21	0.00000	3.32
Within Groups	2479591742316	4995	496414763			
Total	58997825196357	4999				

The results of the individual t-tests (Table 4.11) show that statistically all layouts are significantly different from the future layout. Also, the critical difference for the Scheffé test, at 2760, was much lower than the actual differences.

Table 4.11: Individual t-tests for data from Toyota's proposed changes

	Future Layout	***New Layout 1***	***New Layout 2***	***New Layout 3***	***Original Layout***
Mean	291641	118851	125382	363714	345543
Variance	577799668	107970952	106949508	920333710	769019977
Observations	1000	1000	1000	1000	1000
df	1998				
t Stat		208.65	200.92	-58.88	-46.45
P-value		<0.0001	<0.0001	<0.0001	<0.0001

For the future volume of incoming parts, layouts 1 and 2 resulted in an improvement, whereas the total distance for layout 3 and for the original layout are greater than the total distances in the future layout. Since the only difference in the crossdocking area of the future and the original layout is the number of dollies used, this result was expected.

Results for total distance = walking distance + dolly distance Next, the driving distances for the dollies were divided by three to give a more realistic result for the overall travel distance. Table 4.12 gives the results of the overall ANOVA, which shows that there is a statistically significant difference between the average total distance of at least 2 of the 4 layouts. The result of the t-test between Toyota's proposed new layout and new Layout 3 are

Table 4.12: ANOVA total distance = walking distance + dolly distance/3 for data from Toyota's proposed changes

SUMMARY

Groups	*Count*	*Sum*	*Average*	*Variance*
Future Layout	1000	291640620	291641	577799668
New Layout 1	1000	94093139	94093	92346244
New Layout 2	1000	88756866	88757	77837686
New Layout 3	1000	149205078	149205	131882193
Original Layout	1000	345543430	345543	769019977

ANOVA

Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	55563069152369	4	13890767288092	42121.68	0.00000	3.32
Within Groups	1647236882181	4995	329777154			
Total	57210306034549	4999				

given in Table 4.13. A comparison of the walking distances (Table 4.14) and dolly distances

Table 4.13: T-test results: Toyota's proposed new layout vs. new layout 3

	Future Layout	*New Layout 3*
Mean	291641	149205
Variance	577799668	131882193
Observations	1000	1000
df	1998	
t Stat		169.08
P-value		<0.0001

(Table 4.15) give similar results to the comparison of the current data. Layout 3 performs best regarding walking distance and worst in dolly distance.

A comparison of the improvements between the control layout and the new layouts is given in Table 4.16.

Table 4.14: Comparison of walking distance for data from Toyota's proposed changes

SUMMARY

Groups	Count	Sum	Average	Variance
New Layout 1	1000	81714295	81714	88877890
New Layout 2	1000	70444150	70444	71334149
New Layout 3	1000	41950758	41951	25677752

ANOVA

Source of Variation	SS	df	MS	F	P-value	F crit
Between Groups	840009484894	2	420004742447	6778.29	0.00000	4.61
Within Groups	185703901072	2997	61963264			
Total	1025713385966	2999				

Table 4.15: Comparison of dolly distances for data from Toyota's proposed changes

SUMMARY

Groups	Count	Sum	Average	Variance
New Layout 1	1000	37136531	37137	13031998
New Layout 2	1000	54938149	54938	24157124
New Layout 3	1000	321762960	321763	864063955

ANOVA

Source of Variation	SS	df	MS	F	P-value	F crit
Between Groups	50841527151299	2	25420763575649	84618.06	0.00000	4.61
Within Groups	900351823528	2997	300417692			
Total	51741878974827	2999				

Table 4.16: Comparison of improvement percentages control layouts vs. new layouts

Improvements over Current Layout		
	TD=WD+DD	TD=WD+DD/3
New Layout 1	***55.77***	62.43
New Layout 2	53.05	***64.52***
New Layout 3	-4.62	56.01
Future Layout	14.60	14.60

Improvements over Future Layout		
	TD=WD+DD	TD=WD+DD/3
New Layout 1	***59.25***	67.74
New Layout 2	57.01	***69.57***
New Layout 3	-24.71	48.84
Original Layout	-18.48	-18.48

4.3 Conclusions from the Simulation Results

This section gives the findings of the simulation experiments in regard to the three research questions addressed in this study.

- *Research Question 1: Do differences in the percentage of pallets that have to be cross-docked have a significant effect on the workload of the team members?*
  The simulations show that there is a linear relationship between the level of crossdocking and the workload of the team member when the percentage of boxes for each lane combination is kept proportional to increase/decrease in percentage of crossdocking activity. Future studies are needed to determine whether this finding holds when the percentages of boxes for the lane combinations fluctuate.

- *Research Question 2: Do differences in lane layout organization have a significant effect on the workload of the team members?*
  Via simulation, four layouts were compared to a control layout; for the current data, the current layout at Toyota was used as a control; for the data from the proposed changes, the new layout proposed by Toyota was used. Layout 1 performed best when the total distance was calculated as walking distance plus driving distance, with a decrease in overall travel distance of over 55% for the current data and over 59% for the data from the proposed changes in layout. When the driving distance was adjusted to a more realistic speed, layout 2 gave slightly better results than new layout 1, with an improvement of over 64% and over 69%, respectively.
  New layout 3 showed an increase in overall travel distance when the travel speed for the dollies was not adjusted.

- *Research Question 3: Do differences in volume of incoming parts have a significant effect on the workload of the team members?*
  The first instinct would be to think that a reduction in volume will lead to an equal reduction of workload for the team members in the crossdocking area. The analysis of the original Toyota data led to further investigation. A comparison of the current data and the data from the proposed changes revealed that the number of parts that had to be crossdocked was only slightly reduced, from 1784 or 7.31% for the current data to 1577 or 15.04% for the data from the proposed changes, whereas the volume was reduced by more than half, from 24396 to 10482. A direct statistical comparison of the two sets of data was not possible because the reduction was not distributed evenly for all suppliers and parts.
  Eight examples (see Table 4.17) were constructed to illustrate the possible relationship

between the incoming quantities, part mix and crossdocking activity. The pallet size in all examples is 10 boxes per pallet. Parts labeled part A belong in lane A, parts labeled part B belong in lane B, etc. All parts come from the same supplier in the same truck.
In example 1, the original quantity of 80 parts is reduced by 50%, but since each part still fits on exactly 1 pallet, no crossdocking activity is needed.
In example 2, the original quantity of 80 parts is again reduced by 50%, but in this case, 10 parts or 25% have to be crossdocked.
In example 3, the original quantity was 40 parts; here a reduction by 50% will lead to 10 parts, or 50% of crossdocking activity.
In example 4, 10, or 25%, of the original 40 parts had to be crossdocked. In this scenario, a reduction of 50% will eliminate the crossdocking activity since all parts for lane A fit on the first pallet and all parts for lane B fit on the second pallet.
Example 5 shows that the same quantity could result in drastically different crossdocking levels; an incoming quantity of 10 parts could either result in no crossdocking activity at all, when all parts belong to the same lane, or, result in 7 parts or 70% that have to be crossdocked.
Examples 6 to 8 illustrate that an increase in quantity has the same effects as a reduction in quantity; either no change in crossdocking activity (example 6), an increase in crossdocking activity (example 7) or a reduction in crossdocking activity (example 8).
In all these examples, the parts came from one supplier on one truck. The complexity of the problem increases when more suppliers are involved and the incoming quantities are split up into time intervals/trucks, making prediction of the relationship between the incoming parts and the workload of the team members even more complicated. Due to the complexity of the problem, this dissertation is limited to the identification of the factors involved, namely, pallet size, incoming quantity, part mix, number of suppliers, and delivery interval/number of trucks. The analysis of the relationships among these factors and their influence on the workload is left to future research.

Table 4.17: Examples of different Quantities and Crossdocking %

Example 1	Quantity	Part A	Part B	Part C	Part D	Pallets	CD Parts	CD %
	80	20	20	20	20	8	0	0.00
Reduction by 50 %	40	10	10	10	10	4	**0**	**0.00**

Example 2	Quantity	Part A	Part B	Part C	Part D	Pallets	CD Parts	CD %
	80	20	20	20	20	8	0	0.00
Reduction by 50 %	40	15	15	5	5	4	**10**	**25.00**

Example 3	Quantity	Part A	Part B	Part C	Part D	Pallets	CD Parts	CD %
	40	10	10	10	10	4	0	0.00
Reduction by 50 %	20	5	5	5	5	2	**10**	**50.00**

Example 4	Quantity	Part A	Part B	Part C	Part D	Pallets	CD Parts	CD %
	40	15	15	5	5	4	10	25.00
Reduction by 50 %	20	10	10	0	0	2	**0**	**0.00**

Example 5	Quantity	Part A	Part B	Part C	Part D	Pallets	CD Parts	CD %
	10	10	0	0	0	1	0	0.00
same quantity	10	3	3	2	2	1	**7**	**70.00**

Example 6	Quantity	Part A	Part B	Part C	Part D	Pallets	CD Parts	CD %
	5	5	0	0	0	1	0	0.00
Increase by 100%	10	10	0	0	0	0	**0**	**0.00**

Example 7	Quantity	Part A	Part B	Part C	Part D	Pallets	CD Parts	CD %
	5	5	0	0	0	1	0	0.00
Increase by 100%	10	5	2	2	1	1	**5**	**50.00**

Example 8	Quantity	Part A	Part B	Part C	Part D	Pallets	CD Parts	CD %
	10	5	2	2	1	1	5	50.00
Increase by 100%	20	10	10	0	0	2	**0**	**0.00**

Chapter 5

Optimization of Lane Arrangement for Each Layout Type

5.1 Introduction

Having found the best design through simulation, we further seek optimization for the layouts by rearranging the lanes. Lanes that have a high level of crossdocking activity should be close together to minimize unnecessary material movement. A genetic algorithm (GA), a stochastic search technique, will be used for the optimization. In recent years, GAs have become very popular as tools for optimization in the operations management field. They are used for machine layout problems [Cheng et al., 1996, Cheng and Gen, 1998], assembly line balancing [Falkenauer and Delchambre, 2000][Rubinovitz and Levitin, 1995], workload smoothing [Kim et al., 1998] and many other optimizations problems. Using more conventional methods, such as linear programming, these kind of problems often result in exponential growth of computing time and require increased memory resources when the data set gets large. Additionally, genetic algorithms are very flexible, able to include multiple constraints as well as non-linear and non-convex objective functions. Genetic Algorithms are only one of a variety of stochastic search methods; others are simulated annealing and tabu search. A comparison of the different methods applied to optimization of crossdocking layouts is left for future research.

5.2 The Genetic Algorithm Logic

Genetic algorithms mimic the biological evolutionary process of genetic inheritance and survival of the fittest. When working with a GA, five important criteria have to be chosen: the genetic representation, the evaluation or objective function, the genetic operator or genetic operators, the selection criteria and the stopping point.

5.2.1 Genetic Representation

The first step in designing a GA is to create a genetic representation of potential solutions to the problem. The choice of a genetic representation is tightly related to the nature of the problem. In our case, the lines will simply be represented by their number. The position of the number will equal the position of the lane in the layout. For example, a potential solution would be to arrange the lines as follows: 12,10,4,5,3,11,8,9,7,1,2,6, meaning line 12 is the left-most line, line 10 the second left-most line, and so forth. Because there are 12 lines, the number of possible arrangements is 12! or $479{,}001{,}600$ different possibilities. Each of these candidate solutions is called an individual or chromosome, all individuals together are called a population. The single elements (lanes) in the individuals are called genes. In this study, each individual consists of 12 genes. In this first step, the size of the population and the initial population are the two parameters that have to be chosen. The initial population can either be created at random, or, when good solutions are already available, specific individuals can be chosen.

5.2.2 Evaluation Function

In the second step, each individual will be evaluated for fitness. This evaluation is intimately related to the objective of the problem being solved. Since the objective of this study is to reduce the workload of the team members, the evaluation function in this study will be the travel distance of the team member. The shorter the travel distance, the better the candidate solution. The average number of boxes per lane per trailer during one day will be used to determine the travel distance. Since the daily variance in the number of boxes is minimal, it will not be considered.

5.2.3 Selection Criteria

After evaluating each individual, the fittest individuals are selected. The selection method must take into account the population diversity. If the population diversity decreases too fast, the algorithm may converge prematurely. If the population diversity does not decrease fast enough, the algorithm may run longer than necessary. These fittest individuals will be used as parents for the next generation. The important parameters in this step are the selection criterion, the number of individuals that are considered for reproduction and the number of individuals that will be replaced. The number of individuals in the population can either stay the same with each iteration or it can increase or decrease. If redundant solutions are detected, they are removed decreasing the population size.

5.2.4 Genetic Operators

The fourth step is the mutation and/or crossover of the fittest individuals selected in step three. When mutation is used, a portion of the number of individual parent genes will be randomly altered to construct the new individual, called child or offspring. In crossover, the genes of two parents are separated at one or more crossover points and then re-combined, mixing the strings from the parent chromosomes to create two new individuals. In crossover, the important parameters are the crossover method and the number/percentage of the population to which the crossover operator is applied. In mutation, the important parameter is the mutation operator and the number/percentage of individuals that are mutated.

Each iteration of steps two to four produces a new generation.

5.2.5 Stopping Point

The last parameter that has to be selected is a stopping parameter. This could be either a certain number of generations, the level to which the objective function is satisfied or some other criteria.

All the different parameters mentioned above must be carefully chosen to find a balance between the exploitation of good individuals and the exploration of the search space. In addition, exploration of the search space has to be done very carefully to prevent premature convergence to local optima. The multitude of parameters has led to studies on how to determine the proper values of these parameters [Grefenstette, 1986], but finding the proper values still seems to be more an art than a science [Michalewicz, 1994].

In our case, the GA parameters will be determined by preliminary experiments using the current data and original layout as input.

5.3 Example of a Genetic Algorithm

The following section uses an 8 lane layout to provide a simple, detailed example of the different steps a genetic algorithm goes through during one iteration.

5.3.1 Random Creation of a Start Population

One important issue in GAs is the size of the start population. In our example,we start with a population of 10 individuals. The genetic encoding of the first arrangement of lanes is: Individual 1: 2-4-5-7-1-3-8-6. This means that line 2 is the most left lane in the layout, line 4 is the next ..., and lane 6 is the furthest right lane in the layout.
The complete start population is shown in Table 5.1

Table 5.1: Start population for example

Lane position/Gene	1	2	3	4	5	6	7	8
Individual 1	2	4	5	7	1	3	8	6
Individual 2	1	6	3	4	7	5	2	8
Individual 3	5	7	1	3	4	8	6	2
Individual 4	1	7	8	2	4	6	3	5
Individual 5	4	8	7	1	3	2	6	5
Individual 6	7	5	4	2	1	3	8	6
Individual 7	1	4	5	7	2	3	6	8
Individual 8	2	4	6	8	1	3	5	7
Individual 9	5	6	7	1	3	2	4	8
Individual 10	1	2	3	4	6	7	8	5

5.3.2 Evaluation Function

In this study, our objective is to minimize the travel distance for the team members. To test the objective function, we use the arrival of one pallet as an example. The number of boxes and their lane designations are presented in Table 5.2.

The distances between the unloading point and the lanes are shown in Table 5.3.

Table 5.2: Example pallet

Number of Boxes	Lane
2	1
3	2
1	3
5	4
7	5
6	6
3	7
11	8

Table 5.3: Distances between unloading point and the lanes

Lane Position	1	2	3	4	5	6	7	8
Travel distance	4	5	3	2	2	3	5	4
Individual 1	2	4	5	7	1	3	8	6

From these two tables, the travel distance (td) in meters (m) for the individuals can be calculated as follows:

Individual 1: 2-4-5-7-1-3-8-6

$td_1 : 2 * 2m + 3 * 4m + 1 * 3m + 5 * 5m + 7 * 3m + 6 * 4m + 3 * 2m + 11 * 5m = 150m$

$td_2 : 2 * 4m + 3 * 5m + 1 * 3m + 5 * 2m + 7 * 3m + 6 * 5m + 3 * 2m + 11 * 4m = 137m$

$td_3 : 136m$

$td_4 : 129m$

$td_5 : 157m$

$td_6 : 154m$

$td_7 : 143m$

$td_8 : 131m$

$td_9 : 151m$

$td_{10} : 140m$

5.3.3 Selection of the Individuals with the Best Fitness Function

Many different selection algorithms exist, ranging from those selecting none of the individuals and creating a new population every time, so called non-overlapping populations, to steady-state genetic algorithms that use overlapping populations. For our example, we use an overlapping algorithm with an elitist strategy where the two best individuals are chosen as parents for the next generation and 8 new individuals are created from them. The two fittest individuals are individual 4, with a travel distance of 129 meters, and individual 8, with a travel distance of 131 meters.

5.3.4 Reproduction

Assumption: We want our population size to stay the same with each iteration.
From the wide variety of genetic operators, we have chosen a mutation that swaps genes at random. Each parent will be used to create 4 new children.

Parent 1 = 1-7-8-2-4-6-3-5

First, randomly chosen genes 2 and 6 are swapped.

Child 1 = 1-6-8-2-4-7-3-5

Second, randomly chosen genes 4 and 5 are swapped.

Child 2 = 1-7-8-4-2-6-3-5

Third, randomly chosen genes 1 and 6 are swapped.

Child 3 = 6-7-8-2-4-1-3-5

Forth, randomly chosen genes 7 and 8 are swapped.

Child 4 = 1-7-8-2-4-6-5-3
Genes that are swapped are the same for both parents

Parent 2 = 2-4-6-8-1-3-5-7

Child 5 = 2-3-6-8-1-4-5-7

Child 6 = 2-4-6-1-8-3-5-7

Child 7 = 3-4-6-8-1-2-5-7

Child 8 = 2-4-6-8-1-3-7-5

5.3.5 Evaluation Function for the New Generation

The iteration starts again at point 5.2.2, with each new generation being evaluated until the stopping point is not reached.

5.4 Research Question

The research question that will be answered in this part of the dissertation is:

- *Research Question 4: Does the arrangement of the lanes have a significant effect on the workload of the team members?*

5.5 GAlib

In this study, an existing software package, GAlib, is used for the optimization of the lane arrangements. GAlib is a genetic algorithm software package developed by Matthew Wall at the Massachusetts Institute of Technology [Wall, 1995]. It contains a set of C++ tools to solve optimization problems with the help of GAs. Besides including documentation and examples of how to implement GAs, GAlib includes a wide variety of representation types, genetic operators, selection methods and stopping criteria. The source code is freely available; therefore, should the existing parameters not be sufficient, modifications to all these parameters can be made. Because the objective function depends largely on the problem, it must be written by the user. In addition, the BLITZ++ [Veldhuizen, 1998] library was used to simplify the array handling. As discussed earlier, selecting parameters in a GA is a difficult process and few guidelines have been developed. GAlib makes it possible to try a variety of genetic operators so that a nearly optimal solution can be found.

5.6 Experiments for Choosing the GA Parameters

As mentioned earlier, the performance of a GA is largely influenced by the following parameters: representation of the individuals, crossover method, mutation method, population size and number of generations. For this study, the representation of the individuals did not pose a problem: the line are represented by their position in the layout. Two crossover methods were compared: The edge recombination crossover (ERX) and the partial match crossover (PMX), both described in detail in sections 5.6.1 and 5.6.2. The population size (pSize) and the number of generation (nGen) was varied between 100 and 1000. Two levels (10% and 50%) of the percentages of individuals were chosen for mutation (pMut, and three levels (10%, 50% and 100%) of the percentage of individuals selected for crossover (pRep) were tested. For each parameter combination, 100 runs were carried out using random seed points (i.e. randomly created individuals in the initial population). The results are presented in

Table 5.4. In addition to the GA parameters, Table 72 also reports the number of runs that found the best result and the CPU time used by the program.

The partial match crossover method with a population size of 1000, number of generations, 50% mutation rate and 50% crossover rate found the best solution in 96 of the 100 run with a CPU time of only 19.23 minutes. Even if this parameter combination is not the overall best combination, it provides a good balance between processing time and the number of runs that found the best solution. This parameter combination was chosen for the optimization of the layouts. The partial match crossover methods showed both better performance, in most of the cases, and also smaller processing times, but the results also demonstrate that there is no visible relationship between the parameters.

5.6.1 The Edge Recombination Crossover

In the edge recombination crossover (ERX) a table is created which contains each gene and the gene it is connected to; the head and tail gene are connected to each other.

For example:

Parent 1: 1-5-8-4-2-7-3-6

Parent 2: 2-4-7-1-6-8-3-5

would result in the connection table shown in step 1 of Table 5.5.

The first gene of the child is selected by choosing the gene with the smallest number of connected genes/edges. In this example, gene 1 is connected to only two other genes; thus it will be chosen as the first gene of the new offspring. After that, the selected gene is removed from the table, resulting in the new connection table as presented in step 2 of Table 5.5. This process is repeated until all genes are selected, steps 3 to 8 in Table 5.5. The child resulting from this procedure would be:

Child: 1-6-4-7-2-5-8-3

5.6.2 The Partial Match Crossover

The partial match crossover was first described by Goldberg [Goldberg, 1989]. It creates children by transforming genes from one parent and preserving the order and position of as many genes as possible from the other parent.

For example:

Table 5.4: Results of GA parameter selection experiments

pSize	nGen	pMut	pRep	Crossover	# best	sec	min	hours
100	100	10	10	ERX	0	31	0.52	
100	100	10	50	ERX	49	129	2.15	
100	100	10	100	ERX	2	245	4.08	
100	100	50	10	ERX	2	252	4.20	
100	100	50	50	ERX	2	131	2.18	
100	100	50	100	ERX	3	254	4.23	
100	100	10	10	ERX	0	31	0.52	
100	1000	10	50	ERX	52	1209	20.15	
100	1000	10	100	ERX	5	2402	40.03	
100	1000	50	10	ERX	7	2423	40.38	
100	1000	50	50	ERX	8	1228	20.47	
100	1000	50	100	ERX	4	2421	40.35	
1000	100	10	10	ERX	4	546	9.10	
1000	100	10	50	ERX	15	1484	24.73	
1000	100	10	100	ERX	16	2673	44.55	
1000	100	50	10	ERX	71	2769	46.15	
1000	100	50	50	ERX	13	1568	26.13	
1000	100	50	100	ERX	76	2778	46.30	
1000	1000	10	10	ERX	19	3635	60.58	
1000	1000	10	50	ERX	19	12808	213.47	3.56
1000	1000	10	100	ERX	24	25138	418.97	6.98
1000	1000	50	10	ERX	86	25623	427.05	7.12
1000	1000	50	50	ERX	32	11536	192.27	3.20
1000	1000	50	100	ERX	81	25839	430.65	7.18
100	100	10	10	PMX	2	24		
100	100	10	50	PMX	14	99		
100	100	10	100	PMX	15	194		
100	100	50	10	PMX	0	26		
100	100	50	50	PMX	21	103		
100	100	50	100	PMX	71	202		
100	1000	10	10	PMX	13	206		
100	1000	10	50	PMX	18	958	15.97	
100	1000	10	100	PMX	13	1925	32.08	
100	1000	50	10	PMX	13	215	3.58	
100	1000	50	50	PMX	32	1004	16.73	
100	1000	50	100	PMX	75	1990	33.17	
1000	100	10	10	PMX	54	388	6.47	
1000	100	10	50	PMX	89	1349	22.48	
1000	100	10	100	PMX	89	2123	35.38	
1000	100	50	10	PMX	5	450	7.50	
1000	***100***	***50***	***50***	***PMX***	***96***	***1154***	***19.23***	
1000	100	50	100	PMX	92	2122	35.37	
1000	1000	10	10	PMX	35	1470	24.50	
1000	1000	10	50	PMX	85	9955	165.92	2.77
1000	1000	10	100	PMX	92	19824	330.40	5.51
1000	1000	50	10	PMX	92	2601	43.35	
1000	1000	50	50	PMX	94	10379	172.98	2.88
1000	1000	50	100	PMX	97	10701	178.35	2.97

Table 5.5: Connection Table and selection of genes to create offspring

Step 1: Create connection table

Gene	Connected to			
1	6	7		
2	4	7	5	
3	6	7	5	8
4	8	2	7	
5	1	8	3	2
6	3	1	8	
7	2	3	4	1
8	5	4	6	3

Step 2: Selection of gene 1

Gene	Connected to			
2	4	7	5	
3	6	7	5	8
4	8	2	7	
5	8	3	2	
6	3	8		
7	2	3	4	
8	5	4	6	3

Step 3: Selection of gene 6

Gene	Connected to			
2	4	7	5	
3	7	5	8	
4	8	2	7	
5	8	3	2	
7	2	3	4	
8	5	4	3	

Child: 1,6,

Step 4: Selection of gene 4

Gene	Connected to			
2	7	5		
3	7	5	8	
5	8	3	2	
7	2	3		
8	5	3		

Child: 1,6,4

Step 5: Selection of gene 7

Gene	Connected to			
2	5			
3	5	8		
5	8	3	2	
8	5	3		

Child: 1,6,4,7

Step 6: Selection of gene 2

Gene	Connected to			
3	5	8		
5	8	3		
8	5	3		

Child: 1,6,4,7,2

Step 7: Selection of gene 5

Gene	Connected to			
3	8			
8	3			

Child: 1,6,4,7,2,5,8

Step 8: Selection of gene 8

Gene	Connected to			
3	8			

Child: 1,6,4,7,2,5,8,3

Parent 1: 1-5-**8-4-2**-7-3-6

Parent 2: 2-4-**7-1-6**-8-3-5

First a matching section is defined (bold), and from that an interchange mapping table is developed:

$\{8 \Leftrightarrow 7, 4 \Leftrightarrow 1, 2 \Leftrightarrow 6\}$,

which is then used to create the offspring. Each gene of the first parent is compared with the elements in the interchange mapping table. If the table has an entry for the gene, it is exchanged with the corresponding element. If there is more than one corresponding entry in the table, one entry is chosen at random. If the gene has no entry in the table, it is simply copied over to the child.

Parent 1: 1-5-8-4-2-7-3-6

Child 1 : 4-5-7-1-6-8-3-2

The procedure is then repeated for the second parent

Parent 2: 2-4-7-1-6-8-3-5

Child 2 : 6-1-8-4-2-7-3-5

5.7 The Optimized Lane Arrangements

Using the parameter combination found earlier, the GA was run for all quantities/layout combinations. See Appendix 7.3 for a printout of the program showing the objective function. The original Toyota data were used for the incoming quantities. The distances were calculated as total distance equals driving distance plus walking distance. Table 5.6 presents he best layouts found by the GA. In most cases, more than one layout gave the lowest distance; only the layout used in the simulation runs is shown.

The simulation models were changed to use the optimized arrangement of lanes for the distance calculation, and all 8 simulations were run again. The results are presented in Chapter 6.

5.8 Validation of the GA

As mentioned earlier, the number of possible line combinations is 12! or $479{,}001{,}600$ different possibilities. To validate the results of the GA, an exhaustive search algorithm was

Table 5.6: Optimized layouts

Quantity	Layout	Optimized Arrangement of Lanes											
Current	Original Layout	7	0	2	11	1	4	8	9	3	10	5	6
Current	New Layout 1	4	12	9	2	3	1	7	8	5	10	11	6
Current	New Layout 2	1	8	5	7	3	10	12	4	9	2	11	6
Current	New Layout 3	6	4	5	9	10	11	8	12	1	3	2	7
Proposed Changes	Toyota's proposed layout	2	1	10	4	3	9	12	7	11	6	8	5
Proposed Changes	New Layout 1	12	1	4	7	11	5	8	6	3	9	2	10
Proposed Changes	New Layout 2	7	8	2	6	5	1	12	9	11	10	3	4
Proposed Changes	New Layout 3	4	1	2	3	10	9	5	7	11	6	12	8

applied to one parameter combination of incoming quantities and layouts, namely, the current data and the original layout. Because the run time of such a program would be too lengthy for normal workstations, the problem was divided into smaller subproblems and distributed onto several computers. Each machine independently computes the solution for its subproblems and the minimum from all of the solutions will be the overall minimum distance. This exhaustive search utilized 132 out of 968 processors of the Platinum cluster supercomputer at the National Center for Supercomputing Applications (NCSA) at the University of Illinois. The line layout problem is an ideal candidate for parallel processing because all subsearches are independent of each other. The first processor computes all the possible lane combinations starting with lane 1 in position 1 and lane 2 in position 2. The second processor computes all the possibilities starting with lane 1 in position 1 and lane 3 in position 2, ... and the 132nd processor computes all the possibilities starting with lane 12 in position 1 and lane 11 in position 2. The program was written in C++ and the parallel communication was implemented using the parallel programming library MPI [ref]. A listing of the program can be found in Appendix 7.3.

The CPU time was 55:23 minutes on each 1 GHz Intel Pentium III which means it would have taken 132x55 minutes or over 5 days to run the program serial on one computer. Such an exhaustive search is not feasible in a production environment, where time and cost are an issue, and requires, other, faster solutions, such as a GA.

The exhaustive search found that layout 7-0-2-11-1-4-8-9-3-1-5-6 resulted in the lowest travel distance, which is the same lane combination found by the GA.

Chapter 6

Results and Analysis of Optimized Lane Arrangements

The following chapter gives the results and analysis of the simulations performed for the optimized lane arrangement. As in the analysis of the original simulations, it is split into current data and the data from the proposed changes.

6.1 Results from the Current Data

First an overview over the overall improvements is given. Table 6.1 reveals that all layouts can be further improved by rearranging the lanes. For the current layout and the new layout 3, the improvement would be substantial at nearly 34%.

Table 6.1: Overview improvements for current data

Current Data			
	Travel Distance		
	before GA	after GA	Improvement
Original Layout	306893	202751	33.93%
New Layout 1	135741	131485	3.13%
New Layout 2	144083	123679	14.16%
New Layout 3	321063	212194	33.91%

To better understand the improvements, the data were analyzed using the two different measurements, driving distance and walking distance.

In the original layout (Figure 3.18, page 41) all the parts are carried and no driving takes place so only one t-test was performed, resulting in an overall improvement of nearly 34%. In the new layout 1 (Figure 3.19, page 43), the walking distance is the same regardless of the arrangement of the lanes. The improvement in the driving distance is close to 14%, but since the driving distance accounts only for 1/3 of the overall distance the overall improvement is a much smaller 3.13%. The second new layout (Figure 3.20, page 45) shows a total improvement of 14.16% after rearranging the lanes, mainly because of the high improvement in the driving distances. New layout 3 (Figure 3.21, page 47) also shows also a substantial improvement of nearly 34%. Nearly all of the improvement is gained in the driving distance, which is not surprising considering the large distance between the two areas. As mentioned earlier, the performance of this layout is largely influenced by the amount of material that flows between the two areas, and a separation of material into the two areas by the supplier would be worth investigating. The results of the t-test are summarized in Table 6.2. The mean distances before and after the GA are statistically all significantly different at an alpha level of 0.01 with P-values <0.0001 for all cases except for the walking distance in new layout 3, which has a P-value of 0.35604.

6.2 Results from the Data of Toyota's Proposed Changes

The improvements due to the rearrangement of the lanes for the data from the proposed changes are similar to those for the current data. All distances can be reduced. An overview for the improvements is given in Table 78. The greatest reduction in overall travel distance is gained for the layout proposed by Toyota and the new layout 3.

A more detailed analysis reveals that the improvements are mainly in the driving distance, as observed earlier in the current data. The results of the t-tests are summarized in Table 79.Statistically all distances are significantly different except the walking distance for the new layout 2. The alpha level used in this analysis was 0.01.

6.3 Conclusions from the Optimization Results

The research question the optimization part of this dissertation is trying to answer is:

- *Research Question 4: Does the arrangement of the lanes have a significant effect on the workload of the team members?*

Table 6.2: Results analysis for current data

Current Data								
	before GA		after GA		Mean	%		
	Mean	Variance	Mean	Variance	Difference	Difference	T-Stat	P-value
Original Layout Total Distance	306893	37034	202751	27293	104142	33.93	71.59	<0.0001
New Layout 1 Total Distance	135741	15804	131485	15546	4255	3.13	21.62	<0.0001
New Layout 1 Driving Distance	30679	4409	26424	4395	4255	13.87	6.07	<0.0001
New Layout 2 Total Distance	144083	17246	123679	15016	20405	14.16	28.22	<0.0001
New Layout 2 Driving Distance	52789	6959	34315	5498	18473	35.00	65.87	<0.0001
New Layout 2 Walking Distance	91295	10880	89363	10693	1931	2.12	4.00	<0.0001
New Layout 3 Total Distance	321063	40740	212194	31639	108869	33.91	66.74	<0.0001
New Layout 3 Driving Distance	279112	36348	170326	28009	108786	38.98	74.97	<0.0001
New Layout 3 Walking Distance	41951	5067	41868	4981	83	0.20	0.37	0.35604

Table 6.3: Overview improvements for data from Toyota's proposed changes

Data from Proposed Changes			
	Travel Distance		
	before GA	after GA	Improvement
Toyota's Proposed Layout	291641	216057	25.92%
New Layout 1	162783	159228	2.18%
New Layout 2	172705	162052	6.17%
New Layout 3	380836	261713	31.28%

Table 6.4: Results analysis for data from Toyota’s proposed changes

Data from Toyota's Proposed Changes								
	before GA		after GA		Mean	%		
	Mean	Variance	Mean	Variance	Difference	Difference	T-Stat	P-value
Toyota's Proposed Layout Total Dist.	291641	24037	216057	17671	75584	25.92	80.12	<0.0001
New Layout 1 Total Distance	162783	11977	159228	11657	3555	2.18	6.73	<0.0001
New Layout 1 Driving Distance	37137	3610	33582	3351	3555	9.57	22.82	<0.0001
New Layout 2 Total Distance	172705	12965	162052	12127	10652	6.17	18.97	<0.0001
New Layout 2 Driving Distance	54938	4915	43973	4326	10965	19.96	52.96	<0.0001
New Layout 2 Walking Distance	117767	8673	118079	8714			-0.80	0.21102
New Layout 3 Total Distance	380836	33180	261713	23713	119123	31.28	92.37	<0.0001
New Layout 3 Driving Distance	321763	29395	208303	20753	113460	35.26	99.71	<0.0001
New Layout 3 Walking Distance	59073	4453	53410	4043	5663	9.59	29.78	0.35604

Overall, the arrangement of the lanes has a significant effect on the workload of the team members. The computational results demonstrate that GAs comprise a promising approach for crossdocking layout problems. The eight comparisons made demonstrate that the travel distances for all 8 layouts could be reduced by using a GA to find an optimal, or nearly optimal layout. The decreases in travel distance were between 2.18% and 33.93%, which would reduce the workload of the team members in the crossdocking area by over 1/3.

Chapter 7

Conclusion

7.1 Introduction

Crossdocking has been studied before only in the context of the airline industry and distribution industry. This is the first research, known to the author, that studies crossdocking in the manufacturing industry.

This dissertation is concerned with the workload of the team members in a crossdocking operation in a JIT environment. The layout and data of an existing crossdocking operation of a particular JIT company, namely Toyota, was chosen to test the influence of different factors on the workload:

- percentage of incoming pallets that have to be crossdocked
- layout of the crossdocking area
- volume of incoming quantities

Workload was measured as total travel distance; in layouts where parts are carried and also traveled by dolly, the measurement was split into walking distance and driving distance. Simulation studies were used to test the three factors.

As a further step, the arrangement of lanes in the layout was optimized. Lanes that have a high level of crossdocking activity should be close together. Since an exhaustive search would be too costly, a Genetic Algorithm was used to optimize the lane arrangement.

7.2 Conclusions about research questions

For the first research question we found a linear relationship between the percentage of pallets that have to be crossdocked and the workload of the team member. Keeping all the other factors constant and adjusting he percentage of quantities each lane combination receives, the workload doubled when the percentages of crossdocking pallets doubled.

For the volume of incoming quantities and the layout aspect of the study, the analysis of the simulation results was split into two sets: the first set of analyses used the current data, observed at Toyota, the second set of analyses used data resulting from a proposed change suggested by Toyota. For the current data the current layout was used as a control group for the comparison with three newly designed layouts. For the analysis of data from the proposed changes, the proposed changed layout was used as the control group. In both cases the control group performed worse than two of the new layouts. The reduction of workload was substantial, over 50% for the best layout. Whereas the best shape in the study of Bartholdi and Gue [Bartholdi III and Gue, 2001] was related to the number of receiving/outgoing doors, the performance of the layouts test in this dissertation depended on how the travel distance was calculated. Two different measurements were used to calculate the total travel distance. First, the total distance was calculated as walking distance plus driving distance, which is the worst case scenario. In this case, the first newly designed layout performed best. Second, the driving distance was divided by three, assuming that dollies travel three times faster than people walk. In this case, the second newly designed layout performed best.

But workload is not the only parameter that has to be taken into account when designing a new layout. Other things, like overall workload, floorspace and safety of the team members also have to be considered.

The third research question was more challenging than originally thought. An increase in volume of incoming parts does not necessarily lead to an increase in the workload of the team members. The same is true for a decrease in volume of incoming parts, as shown in the examples of Table 4.17. This research had to be limited to the identification of the factors involved: pallet size, number of incoming parts, number of suppliers, part mix and delivery interval/number of trucks.

The last research question was concerned with the arrangement of lanes in the layouts. Because calculating all the possibilities would require either a large number of computers or a very long time (see section 5.8 for more detail), a GA was used for the optimization of the arrangement of the lane. Wheres a GA does not necessarily give the overall best solution, it will find a good solution in a very short time. All the layouts could be improved by rearranging the lanes. The results of the GA were very encouraging; it found the optimal layout in 95 of 100 test runs and it did it in only seconds. This would make it a good tool for practitioners since it provides a good relationship between the goodness of the solution and computational cost/time.

Overall, this study shows that a reduction of the workload of the team members in the cross-docking area is possible, which would allow companies not only to reduce handling cost but also to achieve a decrease in lead time between unloading of the truck and unloading of the parts at the assembly line.

7.3 Limitations and Future Research

Because this is the first study of crossdocking in a JIT environment, assumptions had to be made to define the proper scope of the research questions. Increasing the complexity of the research questions opens a wide range of research possibilities.

One of the research questions in this study found that there is a linear relationship between the percentage level of crossdocking activity and the workload of the team members when the percentage of quantities each lane combination (from lane/to lane) receives is kept the same. An extension of this study could investigate the influence of changes in the percentage of quantities each lane combination receives. This factor could not only influence the workload of the team members but also the choice of the best layout design or lane arrangement.

Further, in this dissertation the design of the different crossdocking layout is restricted to 12 lanes. Bartholdi and Gue [Bartholdi III and Gue, 2001] have shown in their study about the shape of a crossdock in the distribution industry that the best layout varies with the size of the crossdocking operation. A similar study with different numbers of lanes would be worth exploring.

The relationship between the volume of incoming parts and the workload of the team members is another area that needs further investigation. Not only should the direct relationship of the factors identified in this study on the workload be examined in more detail, but also their relationship to each other.

Another possible research area would be to extend the existing studies of the sequencing and balancing of mixed model assembly lines to include as one of its objectives a minimized and/or balanced workload on the crossdocking area.

The results found in this study could also be applied to other crossdocking operations, for example, at large warehouses such as Walmart.

One limitation of this study is that there is no direct data exchange between the simulation and the Genetic Algorithm. The GA uses the original data from Toyota as input whereas the simulation study used a distribution data developed from the original data as input. Thus there is a slight chance that the best layout found by the GA is not the best layout for the data from the simulation but since all layouts showed improvements the additional work of creating an interface between the two application might not be necessary.

Another limitation of the study is that the travel distance is the only workload that is considered for the team members in the crossdocking area. The work resulting from unwrapping the pallets, and picking up and setting down the boxes is not considered. Additionally, the workloads for the forklift drivers and the team member that unload the boxes at the assembly line are not included in the study. The inclusion of all of this factors in further studies would give a better insight into the overall work in a crossdocking operation.

Appendix A
Parallel Exhaustive Search Program

```
#include <iostream>
#include <string>
#include <sstream>
#include <fstream>
#include <iomanip>

#include "blitz/array.h"
#include "mpi.h"

using namespace std;

// reading from argv[1] the distance between lanes
// reading from argv[2] the boxes between lanes
// writing to argv[3]best.txt argv[3]worst.txt the solution
int main(int argc, char** argv)
{

  const int NLANES = 12;

  // MPI start
  int ierr = MPI_Init(&argc, &argv); if (ierr != MPI_SUCCESS) exit(1);
  int myRank, totalSize;
  ierr = MPI_Comm_rank(MPI_COMM_WORLD, &myRank);
  if (ierr != MPI_SUCCESS) exit(3);
  ierr = MPI_Comm_size(MPI_COMM_WORLD, &totalSize);
  if (ierr != MPI_SUCCESS) exit(4);
  //
  if (argc != 4) {
    cerr << argv[0] << "dist.txt boxes.txt outfile\n";
    exit(5);
  }
```

```
blitz::Array<int, 2> distLanes(NLANES,NLANES);
blitz::Array<double, 2> nboxesLanes(NLANES,NLANES);
//
// read the distances between the lanes from file
ifstream inDist(argv[1]);
if(!inDist)  {
  cerr << "could not read data file" << argv[1] << "\n";
  exit(1);
}
cout << "reading distance file " << argv[1] << endl;
int ii = 0;
blitz::Array<int, 1> d(NLANES);
while (inDist >>
       d(0) >> d(1) >> d(2) >> d(3) >> d(4) >> d(5) >>
       d(6) >> d(7) >> d(8) >> d(9) >> d(10)>> d(11)) {
  for (int j=0; j < NLANES; ++j) {
    distLanes(ii,j) = d(j);
  }
  ++ii;
}
if (myRank == 0) {
  cout << "distLanes array" << endl;
  cout << distLanes << endl;
}

//
// read the number of boxes between each lane from file
ifstream inBoxes(argv[2]);
if(!inBoxes)  {
  cerr << "could not read data file" << argv[2] << "\n";
  exit(1);
}
cout << "reading nboxes file " << argv[2] << endl;
ii = 0;
blitz::Array<double, 1> b(NLANES);
while (inBoxes >>
       b(0) >> b(1) >> b(2) >> b(3) >> b(4) >> b(5) >>
       b(6) >> b(7) >> b(8) >> b(9) >> b(10)>> b(11)) {
  for (int j=0; j < NLANES; ++j) {
    nboxesLanes(ii,j) = b(j);
  }
  ++ii;
}
if (myRank == 0) {
  cout << "nboxesLanes array" << endl;
  cout << nboxesLanes << endl;
```

```
  }
  // now computer what each of the processors has to do
  blitz::Array<int, 1> iwork(NLANES*NLANES-NLANES);
  blitz::Array<int, 1> jwork(NLANES*NLANES-NLANES);
  int pn = 0;
  for (int i=0; i < NLANES; ++i)
    for (int j=0; j < NLANES; ++j) {
      if (i!=j) {
        iwork(pn) = i;
        jwork(pn) = j;
        ++pn;
      }
    }

  // now do the work
    blitz::Array<int, 1> cl(NLANES);            // current lay-
out
    blitz::Array<int, 1> bl(NLANES);            // best layout
    double bd;                  // best distance
    double cd;                  // current distance
    blitz::Array<double, 2> dl(NLANES,NLANES);  // distance ma-
trix for current layout
    int distance;
    double boxes;
    bd = 1.e30;

    int i = iwork(myRank);
    int j = jwork(myRank);
    cout << myRank << ": " << i << ",  " << j << endl;
    for(int k=0; k<NLANES; k++)
     for(int l=0; l<NLANES; l++)
      for(int m=0; m<NLANES; m++)
       for(int n=0; n<NLANES; n++)
        for(int o=0; o<NLANES; o++)
         for(int p=0; p<NLANES; p++)
          for(int q=0; q<NLANES; q++)
           for(int r=0; r<NLANES; r++)
            for(int s=0; s<NLANES; s++)
             for(int t=0; t<NLANES; t++)
              {
               if (i!=t && i!=s && i!=r && i!=q && i!=p && i!= o
&& i!= n && i!=m && i!=l && i!=k && i!=j &&
                   j!=t && j!=s && j!=r && j!=q && j!=p && j!= o
&& j!= n && j!=m && j!=l && j!=k &&
                   k!=t && k!=s && k!=r && k!=q && k!=p && k!= o
&& k!= n && k!=m && k!=l &&
```

```
                    l!=t && l!=s && l!=r && l!=q && l!=p && l!= o
&& l!= n && l!=m &&
                    m!=t && m!=s && m!=r && m!=q && m!=p && m!= o
&& m!= n &&
                    n!=t && n!=s && n!=r && n!=q && n!=p && n!= o &&
                    o!=t && o!=s && o!=r && o!=q && o!=p &&
                    p!=t && p!=s && p!=r && p!=q &&
                    q!=t && q!=s && q!=r &&
                    r!=t && r!=s &&
                    s!=t)
                     {
                      cl(0 )=i;
                      cl(1 )=j;
                      cl(2 )=k;
                      cl(3 )=l;
                      cl(4 )=m;
                      cl(5 )=n;
                      cl(6 )=o;
                      cl(7 )=p;
                      cl(8 )=q;
                      cl(9 )=r;
                      cl(10)=s;
                      cl(11)=t;
                      // calculate distance
                      for(int ii=0; ii < NLANES; ii++) {
                        int icl = cl(ii);

                          for(int jj=0; jj < NLANES; jj++) {
                            int jcl = cl(jj);

                            distance=distLanes(ii, jj);
                            boxes=nboxesLanes(icl,jcl);
                            dl(ii, jj)=distance*boxes;
                          }
                      }

                     cd = 0.;
                     for(int ii=0; ii < NLANES; ii++)
                      for (int jj=0; jj < NLANES; jj++) {
                              cd += dl(ii, jj);
                              }
                      if (cd < bd) {
                      bd = cd;
                      bl(0 ) = cl(0 );
                      bl(1 ) = cl(1 );
                      bl(2 ) = cl(2 );
```

```
                    bl(3 ) = cl(3 );
                    bl(4 ) = cl(4 );
                    bl(5 ) = cl(5 );
                    bl(6 ) = cl(6 );
                    bl(7 ) = cl(7 );
                    bl(8 ) = cl(8 );
                    bl(9 ) = cl(9 );
                    bl(10) = cl(10);
                    bl(11) = cl(11);
                    }
                  }
             }
    // output to screen
    cout << "Best Layout for processor " << myRank << endl;
    cout << bl << endl;
    cout << "Distance: " << bd << endl;

    // output to file
    ostringstream myos;
    myos << argv[3] << myRank << ".txt";
    string fileName(myos.str());
    ofstream outFile(fileName.c_str());
    outFile << "Best Layout for processor " << myRank << endl;
    outFile << bl << endl;
    outFile << "Distance: " << bd << endl;
    ierr = MPI_Finalize();
    if (ierr != MPI_SUCCESS) exit(7);
  exit(0);
}
```

Appendix B
Evaluation function of GA

```
#include <math.h>
#include <ga/GASStateGA.h>
#include <ga/GAListGenome.h>
#include <ga/garandom.h>

#include <iostream>
#include <fstream>
#include <iomanip>
#include "blitz/array.h"     // array library

using namespace std;

#define lanes 12
#define Dist_FILE "distances.txt"
#define Quantities_FILE "quantities.txt"

float DISTANCE[lanes][lanes];
double x[lanes],y[lanes];
int nlanes = 0;

blitz::Array<int, 2> distLanes(lanes,lanes);
blitz::Array<double, 2> nboxesLanes(lanes,lanes);
blitz::Array<double, 2> dl(lanes,lanes);
blitz::Array<int, 1> cl(lanes);
blitz::Array<int, 1> layout(lanes);

// You can use either edge recombination crossover or par-
tial match crossover.
// Which one you select makes a HUGE difference in the perfor-
mance of the
// genetic algorithm.  Only one of the two follow-
ing lines should be commented.
```

```
    #define XOVER PMXover       // (Partial Match Crossover)
//     #define XOVER ERXover         // (Edge Recombina-
tion Crossover)


float Objective(GAGenome&);
int   Mutator(GAGenome&, float);
void  Initializer(GAGenome&);
float Comparator(const GAGenome&, const GAGenome&);
int   ERX-
over(const GAGenome&, const GAGenome&, GAGenome*, GAGenome*);
int   PMX-
over(const GAGenome&, const GAGenome&, GAGenome*, GAGenome*);
void  ERXOneChild(const GAGenome&, const GAGenome&, GAGenome*);


int
main(int argc, char** argv) {
  //  cout << "Lane Optimization Program.\n" << endl;

// See if we've been given a seed to use (for testing pur-
poses).  When you
// specify a random seed, the evolution will be ex-
actly the same each time
// you use that seed number.

  unsigned int seed=0;
  for(int ii=1; ii<argc; ii++) {
    if(strcmp(argv[ii++],"seed") == 0) {
      seed = atoi(argv[ii]);
    }
  }

  // read file with distances between lanes

  ifstream inDist(Dist_FILE);
  if(!inDist) {
    cerr << "could not read data file " << Dist_FILE << "\n";
    exit(1);
  }

  int ii=0;
  blitz::Array<int, 1> d(lanes);

  while(ii!=12) {
    inDist >> d(0)>> d(1) >> d(2) >> d(3) >> d(4) >> d(5) >>
```

```
         d(6) >> d(7) >> d(8) >> d(9) >> d(10)>> d(11) ;
    for (int j=0; j < lanes; ++j) {
      distLanes(ii,j) = d(j);
    }
    ii++;

 }


 // read the number of boxes between each lane from file
  ifstream inBoxes(Quantities_FILE);
  if(!inBoxes)  {
    cerr << "could not read data file" << Quanti-
ties_FILE << "\n";
    exit(1);
  }
  cout << "reading nboxes file " << Quantities_FILE << endl;
  ii = 0;
  blitz::Array<double, 1> b(lanes);

  while (ii!=12) {
    in-
Boxes >> b(0) >> b(1) >> b(2) >> b(3) >> b(4) >> b(5) >>
      b(6) >> b(7) >> b(8) >> b(9) >> b(10)>> b(11) ;
    for (int j=0; j < lanes; ++j) {
      nboxesLanes(ii,j) = b(j);
    }
    ++ii;
  }

  GAListGenome<int> genome(Objective);
  genome.initializer(::Initializer);
  genome.mutator(::Mutator);
  genome.comparator(::Comparator);
  genome.crossover(XOVER);

  GASteadyStateGA ga(genome);
  ga.minimize();
  ga.pReplacement(0.5);
  ga.populationSize(1000);
  ga.nGenerations(100);
  ga.pMutation(0.5);
  ga.pCrossover(1.0);
  ga.selectScores(GAStatistics::AllScores);
  ga.parameters(argc, argv);
//  cout << "initializing..."; cout.flush();
```

```
  ga.initialize(seed);
//  cout << "evolving..."; cout.flush();
  while(!ga.done()) {
     ga.step();
//     if(ga.generation() % 10 == 0) {
//       cout << ga.generation() << " "; cout.flush();
//  }
  }

  genome = ga.statistics().bestIndividual();
  cout << "shortest distance " << genome.score() <<" " ;
  cout << "layout"  << genome << "\n";
//   cout << ga.statistics() << "\n";


  return 0;
}


// Here are the genome opera-
tors that we want to use for this problem.

float
Objective(GAGenome& g) {
  GAListGenome<int> & genome = (GAListGenome<int> &)g;
  float dist = 0;

 // calculate distance

  int distance;
  double boxes;

  GAListIter<int>  titer(genome);
  if(titer.head()){
    for(int i=0; i<lanes; i++) {
      cl(i) = *titer.current();
    *titer.next();
    }
  }

  float testdist=0.00;

  for(int ii=0; ii < lanes; ii++) {
    int icl = cl(ii);

    for(int jj=0; jj < lanes; jj++) {
```

```
      int jcl = cl(jj);

      distance=distLanes(ii, jj);
      boxes=nboxesLanes(icl,jcl);
      dl(ii, jj)=distance*boxes;
      testdist = dl(ii, jj);
    }
  }

   dist  = 0.;
  for(int ii=0; ii < lanes; ii++)
    for (int jj=0; jj < lanes; jj++) {
      testdist = dl(ii, jj);
      dist += dl(ii, jj);
    }

  return dist;
}

void
Initializer(GAGenome& g) {
  GAListGenome<int> &child=(GAListGenome<int> &)g;
  while(child.head()) child.destroy(); // destroy any pre-
existing list

  int i,currentlane;
  static int visit[lanes];

  memset(visit, 0, lanes*sizeof(int));
  currentlane=GARandomInt(0,lanes-1);
  visit[currentlane]=1;
  child.insert(currentlane,GAListBASE::HEAD); // the head node

  for( i=1; i<lanes; i++) {
    do {
      currentlane=GARandomInt(0,lanes-1);
    } while (visit[currentlane]);
    visit[currentlane]=1;
    child.insert(currentlane);
  }             // each subsequent node
}

int
Mutator(GAGenome& g, float pmut) {
  GAListGenome<int> &child=(GAListGenome<int> &)g;
  register int n, i;
```

```
  if ((GARandomFloat() >= pmut) || (pmut <= 0)) return 0;

  n = child.size();

  float f1=GARandomFloat();
  if (f1<0.5) {
    int r1=GARandomInt(0,n-1);
    int r2=GARandomInt(0,n-1);

  child.swap(r1,r2); // swap only one time
  }
  else {
    int r3= GARandomInt(1,((int)(n/2-1)));
    int nNodes = r3;       // displace nNodes
    int r4= GARandomInt(0,n-1);

    child.warp(r4);                   // with or without

    GAList<int> TmpList;                              // inver-
sion
    for(i=0;i<nNodes;i++) {
      int *iptr = child.remove();
      TmpList.insert(*iptr,GAListBASE::AFTER);
      delete iptr;
      child.next();
   }

    int invert;
    int r5 = GARandomInt(0,n-nNodes);
    child.warp(r5);
    float f5 = GARandomFloat();
    invert =(f5<0.5) ? 0 : 1;
    if (invert) TmpList.tail(); else TmpList.head();

    for(i=0;i<nNodes;i++) {

      int *iptr = TmpList.remove();
      child.insert(*iptr,GAListBASE::AFTER);

      delete iptr;
      if (invert) ; else TmpList.next();

    }
  }
  child.head();         // set iterator to root node
```

```
  return (1);
}

int
ERXover(const GAGenome& g1, const GAGenome& g2, GAGenome* c1,
GAGenome* c2) {
  int nc=0;
  if(c1) { ERXOneChild(g1,g2,c1); nc+=1; }
  if(c2) { ERXOneChild(g1,g2,c2); nc+=1; }
  return nc;
}

void
ERXOneChild(const GAGenome& g1, const GAGenome& g2,
GAGenome* c1) {
  GAListGenome<int> &mate1=(GAListGenome<int> &)g1;
  GAListGenome<int> &mate2=(GAListGenome<int> &)g2;
  GAListGenome<int> &sis=(GAListGenome<int> &)*c1;

  int i,j,k,t1,t2,currentlane;

  static char CM[lanes][lanes],visit[lanes];
  memset(CM, 0, lanes*lanes*sizeof(char));
  memset(visit, 0, lanes*sizeof(char));

 while (sis.head()) sis.destroy();

  // create connection matrix
  mate1.head();
  for(j=0; j<lanes; j++) {
    t1 = *mate1.current(); t2 = *mate1.next();
    CM[t1][t2]=1; CM[t2][t1]=1;
  }
  mate2.head();
  for(j=0; j<lanes; j++) {
    t1 = *mate2.current(); t2 = *mate2.next();
    CM[t1][t2]=1; CM[t2][t1]=1;
  }

  // select 1st currentlane randomly
  int r6=GARandomInt(0,lanes-1);
  currentlane=r6 ;
  visit[currentlane]=1; mem-
set(CM[currentlane], 0, lanes*sizeof(char));
  sis.insert(currentlane); // the head node
```

```
  GAList<int> PossFollowList;
  GAList<int> FollowersList[5];
  while (PossFollowList.head()) PossFollowList.destroy();
  for(k=0; k<5; k++) {
    while (FollowersList[k].head()) FollowersList[k].destroy();
  }

  // select the following currentlane with the mini-
mal no of next folling currentlanes
  int nPoss,nFollow;
  for(i=1; i<lanes; i++) {
    nPoss = 0;
    for(j=0; j<lanes; j++) {          // no of poss. follow-
ing currentlanes
      if (CM[j][currentlane]) {
        nPoss += 1;
        PossFollowList.insert(j);}
    }
    // nPoss = 0;
    if (nPoss == 0) {
      do {currentlane=GARandomInt(0,lanes-
1);} while (visit[currentlane]); // no follower
      visit[currentlane]=1; mem-
set(CM[currentlane], 0, lanes*sizeof(char));
      sis.insert(currentlane);
    }
    else {
      PossFollowList.head();
      for(j=0; j<nPoss; j++) {
        nFollow = 0;
        currentlane = (*PossFollowList.current());
        for(k=0; k<lanes; k++) {
          if (CM[k][currentlane]) nFollow++;
        }
        FollowersList[nFollow].insert(currentlane);
        PossFollowList.next();
      }
      k=0;
      while (FollowersList[k].size() == 0) k++;
            //  Follower-
sList[k].warp(GARandomInt(0,FollowersList[k].size()));
      currentlane = (*FollowersList[k].current());
      visit[currentlane]=1; mem-
set(CM[currentlane], 0, lanes*sizeof(char));
      sis.insert(currentlane);
```

```
 titer(sis);
      if(titer.head()){
        for(int i=0; i<lanes; i++) {
          layout3[i] = *titer.current();
          *titer.next();
        }
      }
    }
    while (PossFollowList.head()) PossFollowList.destroy();
    for(k=0; k<5; k++) {
      while (FollowersList[k].head()) Follower-
sList[k].destroy();
    }
  }
  sis.head();         // set iterator to head of list


      titer(sis);
      if(titer.head()){
        for(int i=0; i<lanes; i++) {
          layout3[i] = *titer.current();
          *titer.next();
         }
      }
}

int
PMXover(const GAGenome& g1, const GAGenome& g2, GAGenome* c1,
GAGenome* c2) {
  GAListGenome<int> &mom=(GAListGenome<int> &)g1;
  GAListGenome<int> &dad=(GAListGenome<int> &)g2;

  int a = GARandomInt(0, mom.size());
  int b = GARandomInt(0, dad.size());
  int h;
  if (b<a) { h=a; a=b; b=h; }

  int* index;
  int i,j,nc=0;

  if(c1) {
    GAListGenome<int> &sis=(GAListGenome<int> &)*c1;
    sis.GAList<int>::copy(mom);
    GAListIter<int> diter(dad);
    index = diter.warp(a);
    for(i=a; i<b; i++, index=diter.next()){
```

```
      if(*sis.head() == *index){
        sis.swap(i,0);
      }
      else{
        for(j=1; (j<sis.size()) && (*sis.next() != *in-
dex); j++);
        sis.swap(i,j);  // no op if j>size
      }
    }
    sis.head();         // set iterator to head of list
    nc += 1;
  }
  if(c2) {
    GAListGenome<int> &sis=(GAListGenome<int> &)*c2;
    sis.GAList<int>::copy(mom);
    GAListIter<int> diter(dad);
    index = diter.warp(a);
    for(i=a; i<b; i++, index=diter.next()){
      if(*sis.head() == *index){
        sis.swap(i,0);
      }
      else{
        for(j=1; (j<sis.size()) && (*sis.next() != *in-
dex); j++);
        sis.swap(i,j);  // no op if j>size
      }
    }
    sis.head();         // set iterator to head of list
    nc += 1;
  }

  return nc;
}

float
Comparator(const GAGenome& g1, const GAGenome& g2) {
  GAListGenome<int> &a = (GAListGenome<int> &)g1;
  GAListGenome<int> &b = (GAListGenome<int> &)g2;

  int i,j,t1,t2;
  float dist=lanes;

  static char CM1[lanes][lanes],CM2[lanes][lanes];
  memset(CM1, 0, lanes*lanes*sizeof(char));
  memset(CM2, 0, lanes*lanes*sizeof(char));
```

```
  // create connection matrix CM1
  a.head();
  for(i=0; i<lanes; i++) {
    t1 = *a.current(); t2 = *a.next();
    CM1[t1][t2]=1; CM1[t2][t1]=1;
  }
  // create connection matrix CM2
  b.head();
  for(i=0; i<lanes; i++) {
    t1 = *b.current(); t2 = *b.next();
    CM2[t1][t2]=1; CM2[t2][t1]=1;
  }
  //calc distance = how many edges are different
  for (i=0; i<lanes; i++) {
    for (j=i; j<lanes; j++) {
      if (CM1[i][j]&CM2[i][j]) dist--;
    }
  }
  return (dist);
}




//   Here we over-
ride the _write method for the List class.  This lets us see
// exactly what we want (the de-
fault _write method dumps out pointers to the
// data rather than the data contents).
//   This routine prints out the contents of each ele-
ment of the lis // sepa-
rated by a space.  It does not put a new-
line at the end of the list.
//   Notice that you can override ANY function of a tem-
plate class.  This is
// called "specialization" in C++ and it lets you tai-
lor the behaviour of a
// template class to better fit the type.
int
GAListGenome<int>::write(ostream & os) const
{
  int *cur, *head;
  GAListIter<int> tmpiter(*this);
  if((head=tmpiter.head()) != 0) {
    os << *head << " ";
    for(cur=tmpiter.next(); cur && cur != head; cur=tmpiter.next())
```

```
      os << *cur << " ";
  }

  return os.fail() ? 1 : 0;
}


#ifdef NO_AUTO_INST
#include <ga/GAList.C>
#include <ga/GAListGenome.C>
#if defined(__GNUG__)
template class GAList<int>;
template class GAListGenome<int>;
#else
GAList<int>;
GAListGenome<int>;
#endif
#endif
```

Bibliography

[Ahuja et al., 1995] Ahuja, R., Orlin, J., and Tiwari, A. (1995). A greedy genetic algorithm for the quadratic assignment problem. Technical Report working paper no 3836-95, MIT, Sloan School of Management.

[Bartholdi III and Gue, 2001] Bartholdi III, J. and Gue, K. (2001). The best shape for a crossdock. Technical report, The Logistics Institute.

[Bartholdi III and Gue, 2000] Bartholdi III, J. and Gue, K. R. (2000). Reducing labor costs in an LTL crossdocking terminal. *Operations Research*, 48(6):823–832.

[Baykoc and Erol, 1998] Baykoc, O. and Erol, S. (1998). Simulation modelling and analysis of a JIT production system. *International Journal of Production Economics*, 55:203–212.

[Burkard et al., 1998] Burkard, R., Cela, E., Pardalos, P., and Pitsoulis, L. (1998). The quadratic assignment problem. Technical Report Bericht Nr. 126, Karl-Franzens-Universitaet Graz and Technische Universitaet Graz.

[Burkard and Derigs, 1980] Burkard, R. and Derigs, U., editors (1980). *Assignment and matching problems: Solution methods with FORTRAN-Programs*, chapter 1, pages 6–11. Springer, Berlin.

[Cheng and Gen, 1998] Cheng, R. and Gen, M. (1998). Loop layout design problem in flexible manufacturing systems using genetic algorithms. *Computers in Industrial Engineering*, 34(1):5–61.

[Cheng et al., 1996] Cheng, R., Gen, M., and Tosawa, T. (1996). Genetic algorithms for designing loop layout manufacturing systems. *Computers in Industrial Engineering*, 31(3/4):587–591.

[de Haan and Yamamoto, 1999] de Haan, J. and Yamamoto, M. (1999). Zero inventory management: Facts or fiction? Lessons from japan. *International Journal of Production Economics*, 59:65–75.

[Ding and Cheng, 1993a] Ding, F. and Cheng, L. (1993a). An effective mixed-model assembly line sequencing heuristic for just-in-time production systems. *Journal of Operations Management*, 11:45–50.

[Ding and Cheng, 1993b] Ding, F. and Cheng, L. (1993b). A simple sequencing algorithm for mixed-model assembly lines in just-in-time production systems. *Operations Research Letters*, 13:27–36.

[Falkenauer and Delchambre, 2000] Falkenauer, E. and Delchambre, A. (2000). A genetic algorithm for bin packing and line balancing. www.

[Fisher, 1997] Fisher, M. (1997). What is the right supply chain for your product. *Harvard Business Review*, 75(2):105–116.

[Fleurent and Ferland, 1994] Fleurent, C. and Ferland, J. (1994). Genetic hybrids for the quadratic assignment problem. In *Quadratic assignment and related problems: DIMACS Workshop, May 20 1993*, DIMACS Series in Discrete mathematics and theoretical computer science v16, pages 173–187. Providence, R.I. American Mathematical Society c1994.

[Goldberg, 1989] Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*, chapter 5. Addison-Wesley Publishing Company, Inc.

[Grefenstette, 1986] Grefenstette, J. (1986). Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man and Cybernetics*, 16(1):122–128.

[Gue, 1999] Gue, K. (1999). The effects of trailer scheduling on the layout of freight terminals. *Transportation Science*, 33(4):419–428.

[Gunasekaran, 1999] Gunasekaran, A. (1999). Just-in-time purchasing: An investigation for research and applications. *International Journal of Production Economics*, 59:77–84.

[Hale, 1999] Hale, B. (1999). Logistic perspectives for the new millennium. *Journal of Busisness Logistics*, 20(1):5–9.

[Huntley and Brown, 1991] Huntley, C. and Brown, D. E. (1991). A parallel heuristic for quadratic assignment problems. *Computers in Operations Research*, 18(3):275–289.

[Inman and Buffin, 1991] Inman, R. and Buffin, R. (1991). Notes on sequencing JIT mixed-model assembly lines. *Management Science*, 37(7):901–904.

[Kim et al., 1998] Kim, Y. J., Kim, Y., and Cho, Y. (1998). A heuristic-based genetic algorithm for workload smoothing in assembly lines. *Computers in Operations Research*, 25(2):99–111.

[Koopmans and Beckman, 1957] Koopmans, T. and Beckman, M. (1957). Assingment problems and the location of economic activities. *Econometrica*, 25:53–76.

[Korkmazel and Meral, 2001] Korkmazel, T. and Meral, S. (2001). Bicriteria sequencing methods for the mixed-model assembly line in just-in-time production systems. *European Journal of Operational Research*, 131:188–207.

[Leu et al., 1996] Leu, Y., Matheson, L. A., and Rees, L. (1996). Sequencing mixed-model assembly lines with genetic algorithms. *Computers in Industrial Engineering*, 30(4):1027–1036.

[Matanachai and Yano, 2001] Matanachai, S. and Yano, C. (2001). Balancing mixed-model assembly lines to reduce work overload. *IIE Transactions*, 33:29–42.

[Meller and Gau, 1996] Meller, R. and Gau, K. (1996). The facility layout problem: Recent and emerging trends and perspecives. *Journal of Manufacturing Systems*, 15(5):351–366.

[Michalewicz, 1994] Michalewicz, Z. (1994). *Genetic algorithms and data structures*. Springer-Verlag, Berlin, 2nd edition.

[Miltenburg, 1989] Miltenburg, J. (1989). Level schedules for mixed-model assembly lines in just-in-time production systems. *Management Science*, 35(3):192–207.

[Miltenburg, 2001] Miltenburg, J. (2001). U-shpaed production lines: A review of theory and practice. *International Journal of Production Economics*, 70:201–214.

[Nugent et al., 1968] Nugent, C., Vollmann, T., and Ruml, J. (1968). An experimental comparison of techniques for the assignment of facilities to loactions. *Operations Research*, 16:150–173.

[Rosa and Feiring, 1995] Rosa, L. M.-L. and Feiring, B. R. (1995). Layout problem for an aircraft maintenance company tool room. *International Journal of Production Economics*, 40:219–230.

[Rubinovitz and Levitin, 1995] Rubinovitz, J. and Levitin, G. (1995). Genetic algorithm for assembly line balancing. *International Journal of Production Economics*, 41:343–354.

[Sahni and Gongzalez, 1976] Sahni, S. and Gongzalez, T. (1976). P-complete approximation problems. *Journal of the Association of Computing Machinery*, 23:555–565.

[Schonberger, 1984] Schonberger, R. (1984). *Japanes manufacturing techniques: Nine hidden lessons in simplicity*. The Free Press, New York.

[Scriabin and Vergin, 1975] Scriabin, M. and Vergin, R. (1975). Comparison of computer algorithms and visual based methods for plant layout. *Management Science*, 22:172–181.

[Sheshkin, 2000] Sheshkin, D. (2000). *Handbook of Parametric and Nonparametric statistical procedures*, chapter Test 21, page 528. Chapman & Hall.CRC, 2nd edition.

[Shingo, 1981] Shingo, S. (1981). *A study of the toyota production system*. Productivity Press.

[Skorin-Kapov, 1990] Skorin-Kapov, J. (1990). Tabu search applied to the quadratic assignment problem. *ORSA Journal of Computing*, 2:33–45.

[Stalk et al., 1992] Stalk, G., Evans, P., and L.E.Shulman (1992). Competing on capabilities: The new rules of corporate strategy. *Harvard Business Review*, pages 57–69.

[Taillard, 1991] Taillard, E. (1991). Robust taboo search for the quadratic assignment problem. *Parallel Computing*, 17:443–455.

[Tan, 2001] Tan, K. C. (2001). A framework of supply chain management literature. *European Journal of Purchasing and supply management*, 7:39–48.

[Tate and Smith, 1995] Tate, D. and Smith, A. (1995). A genetic approach to the quadratic assignment problem. *Computers and Operations Research*, 22:73–83.

[Tsui and Chang, 1990] Tsui, L. and Chang, C. (1990). A microcomputer bases decision support tool for assigning dock doors in freight yards. *Computers and Industrial Engineering*, 19(1-4):309–312.

[Tsui and Chang, 1992] Tsui, L. and Chang, C. (1992). An optimal solution to a dock door assignment problem. *Computers and Industrial Engineering*, 23(1-4):283–286.

[Veldhuizen, 1998] Veldhuizen, T. L. (1998). Arrays in blitz++. In *Proceedings of the 2nd International Scientific Computing in Object-Oriented Parallel Environments (ISCOPE'98)*, Lecture Notes in Computer Science. Springer-Verlag.

[Wall, 1995] Wall, M. (1995). Galib. http://lancet.mit.edu/ga.

Vita

Date of Birth: May, 8th 1965

Place of Birth: Cologne, Germany

Education:

Master of Business Administration, 1993

Verwaltungs- und Wirtschaftsakademie in Munich, Germany

Honors:

Toyota Fellowship, 1998-2002

Industrial Experience:

Programmer and Consultant at Brain North America, 11/96-10/99

Programmer at Brain Munich, Germany, 07/96-10/96

Head of IT Department at SPEA Software AG, Starnberg, Germany 08/93-06/96

Consultant at Brain Munich, Germany, 04/90-07/93

Programmer at Leicher, Kirchheim, Germany 10/87-03/90

Programmer at Scipio AG, Bremen, Germany 07/86-09/87

Commercial clerk at Atlanta GmbH, Munich, Germany 08/83-06/86