University of Alberta

PROACTIVE LAYDOWN YARD MANAGEMENT USING GENETIC ALGORITHM FULLY-INTEGRATED WITH SIMULATION

by

Pejman Alanjari

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science

in

Construction Engineering and Management

Department of Civil and Environmental Engineering

Fall 2013
Edmonton, Alberta

To my parents and sister Leila, Bahman and Niloo

For their love, dedication and encouragement

Throughout my life

I love you guys so much

ABSTRACT

Laydown yard management is comprised of planning and controlling of all necessary efforts to ensure that the correct quantity of materials are available where and when they are needed on construction storage yards. Applying the right material management methodology in construction projects would result in real savings, improved labor productivity and reduced surplus.

An integrated framework which performs dynamic layout optimization of materials arriving at construction yards is presented in this research. Process improvement in the field of material handling is achieved in this study by evaluating two policies in yard laydown management, namely, proactive and reactive material placement policies. Analytical optimization methods are implemented to compare and contrast such placement policies through case studies from the steel fabrication industry where tight consumption schedule, frequent change orders and revisions and late design drawings provide a sensitive environment in which an effective materials handling method could be of great significance.

ACKNOWLEDGEMENTS

Several individuals have had major contributions to my research and my education improvement since I came to the University of Alberta.

First and foremost, I would like to thank Professor Simaan AbouRizk whose wisdom, understanding and appreciation have led my educational career ever since I came here. I do have enormous respect for him who taught me many lessons. I wish to extend my acknowledgements to the members of my examining committee Dr. Yaman Boluk and Dr. Yasser Mohamed who reviewed my thesis and provided very useful comments and feedback.

I would like to express my especial gratitude towards Stephen Hague who was truly the beacon of hope throughout these 18 months. Steve is so concise and right-to-the point—the qualities I like the most in an instructor. I would also like to thank my friend Reza who was always there if I needed anything.

Many thanks to Waiward Steel personnel, Paul Zubic, Jim Kanerva, Mario the yard foreman, Robi DeSanje the shop foreman, Amin Alvanchi and of course kind-hearted, good-tempered Ray Kanerva, who were genuinely helpful even though they were extremely busy. Without their help, I could not have done it!

Finally, I would like to sincerely acknowledge my parents and my sister who have patiently supported me and loved me whole-heartedly. I am proud to have them as my family. I love them so much.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

1.1. OVERVIEW

The Construction Industry Institute (CII) defines planning for materials management as "consideration for receiving, storage, control, and distribution of materials at the construction site" (1988). Such provisions include layout and organization of laydown areas and warehouse facilities and development of storage plans. Proper laydown yard management will bring about improved craft labor productivity due to easy, quick and inexpensive access to the material, minimized material surplus and reduced rework. An optimized yard layout entails efficiency in terms of time and cost for decision makers who seek increased performance in material tracking, availability and accessibility. On large construction yards, equipment units such as overhead cranes, forklifts and carts are deployed to transfer the key material from the laydown areas on the storage yard to the consumption unit. Under a tight schedule, it would be paramount that the right materials are delivered in a timely manner. Moreover, the use of equipment should be minimized to reduce costs as hourly rate of equipment use could be significant. Thus, the significance of proper laydown yard management has prompted experts to seek tools by which they can quantify the best management techniques and to identify the key steps towards finding the optimum layout design for construction yards.

One of the mathematical tools that has been widely used in academia, and very recently in practice, is simulation. Many researchers have strived to define simulation, such as Pristker (1986), who believes computer simulation is the process of devising a mathematical model of an actual world system and experimenting with the model on a computer. However, several assumptions are made once a real-world system is converted into a simulation model. Some of those assumptions oversimplify the real system and do not properly reflect the occurrence of actual events (AbouRizk 1990). Simulation as a tool should serve the construction practice by being comprehensive and yet easy for engineers to use in construction projects for the purpose of quantification. However, several research works are being carried out without considering the engineering assumption in problem solving and oversimplification. On the other hand, some researchers tend to overcomplicate the problems by providing very abstract ideas or compacted mathematical solutions that do not appeal to practicing engineers.

In the present study, efforts have been made to stay within the boundary of practicality by evaluating the current practice in material placement on yards. For process improvement, this research attempts to propose optimized and practical policies which can be easily understood and applied by those without a background in simulation, e.g. yardmen who are responsible for identifying and determining the laydown areas on the yard. Based on this line of thought, an integrated framework which performs dynamic layout optimization of materials arriving at construction yards is presented in this research. The framework uses genetic algorithm (GA) as a suitable heuristic method, fully integrated with simulation, to propose the best possible layout for the incoming materials in terms of their daily consumption. The proposed method provides a continuous interaction between simulation and genetic algorithm for optimization. This interaction is established within the genetic algorithm, which uses a fitness function to rank the different laydown arrangements. Simulation serves as the fitness function within the genetic algorithm and provides continuous feedback for the optimization trend.

Process improvement in the field of material handling is achieved in this study by evaluating two very common policies in laydown yard management, namely, proactive and reactive layout designs. Analytical optimization methods are implemented to compare and contrast placement policies through case studies in the steel fabrication industry where tight consumption schedule, frequent change orders and revisions, late design drawings and failure to meet approval schedules provide a sensitive environment in which an effective materials handling method could be of great significance.

1.2.PROBLEM STATEMENT

Experts in construction are mindful of the fact that the business they practice is full of risks and uncertainties. These risks and uncertainties make construction dynamic and unstable, mostly by creating change orders among construction processes. In particular, when a project suffers from improper planning, those changes can cause disruption of the construction process. For this reason, a detailed change management document is of profound significance to construction projects. Change management acknowledges changes as part of business continuity, new business requirements and continuous improvement. It also reduces impact and disruption by properly evaluating possible effects, planning and controlling the execution of changes. Such measures improve the predictability of capital project cost and schedule by establishing project controls systems to monitor and predict project outcomes. Effective control systems identify deviations from project plans and commitments early enough to eliminate surprises and allow corrective actions. Many companies have established an enterprise resource planning system and business management processes to support business functions, planning and change management. The premise is to track business functions throughout all working disciplines to minimize the adverse effects of change orders. It would be of interest to the managers to discover how changes impact each working process, including material handling.

One of the other major problems that construction managers face is loss of labor productivity as a result of missing materials or inability to locate the material quickly. Late material delivery will result in labor idleness which leads to loss of productivity and demoralization. Labor is an asset to any industry, especially steel fabrication. The decision makers in steel fabrication would rather change the design to use up more material than add up man-hours for detailed cuttings of optimized structural design. This problem is exacerbated in the event of a very tight schedule, or if the workload is significantly high so that late material delivery would impact the project production cycle. Furthermore, incoming material batches and outgoing material consumption dynamically alter the inventory daily, which further complicates the problem.

This research strives to address the aforementioned concerns and problems. The tools, methods and techniques provided in this study will help improve material handling processes at a level most practical to the engineers and managers. In particular, attempts have been made to propose dynamic and optimum placement arrangements for large construction storage yards where materials are stocked and hauled to the consumption units using hauling equipment, in the order they are needed and consumed. It is understood in this work that in order for laydown yard management to be optimized, the most practical working mentalities should be evaluated first and then efforts should be made to proceed with process improvement using practical tools and methods. Results of this study show room for improvements in laydown yard management in terms of time and possibly cost contributing to overall change and materials management processes.

1.3.RESEARCH OBJECTIVES

The research presented in this thesis has the following objectives:

- Identifying a dynamic, optimum storage yard layout for incoming materials by applying the reactive placement policy of yard foreman, where yard personnel have no information in regards to the consumption schedule and instead react to daily incoming batches upon placement on the laydown areas. This policy can be optimized in terms of time and cost of haulage from the laydown areas to the consumption unit.
- Providing a detailed, dynamic and daily record of the yard inventory. By implementing the proposed placement policies, an exact, daily record of the inventory is automatically obtained.
- Identifying a dynamic, optimum storage yard layout for incoming materials by applying a proactive placement policy in which the yard foreman knows what materials are going to be consumed and

strives to place them in the order of consumption time and volume for a certain period of time.

- Presenting a fully integrated framework where simulation and genetic algorithm have continuous interaction and information exchange to propose an optimum solution to a construction problem in which a process can be suitably simulated by using a simulation program.

1.4.RESEARCH METHODOLOGIES

This research initially attempts to identify what is actually practiced by the yard foreman when s/he faces the daily incoming batches to the yard. It is understood that this method might actually be the most common placement philosophy of many construction yards since it has been widely practiced for many years by the company in question. In this case, efforts will be made to help the yard foreman place the materials on the laydown areas in a more sophisticated manner by considering the yard hard constraints and available equipment. Simulation can be of great assistance to serve this purpose, as it can model resource interactions intelligently. Moreover, to propose an optimum or near-optimum solution, all possible placement combinations must be examined, which is impossible due to the great number of laydown areas and variety of material types. As a result, genetic algorithm lends itself to examining cases and discovering the optimum layout through iterations within the algorithm. It should be emphasized that GA is not used separately from simulation. Conversely, a framework has been established in this research where a continuous information exchange is maintained throughout the analysis, in which simulation and GA help find the optimum solution step-by-step up to the final results.

Simphony is used as the simulation program as it is not only a suitable simulation program, but it also has flexible programmable core services that can be easily accessed, developed and customized. It also provides an interactive graphic user interface, where models can be easily created and then run in a computer program which is what is intended in this research. *Simphony* as a simulation tool will serve the objective function of genetic algorithm. GA is selected as the optimization engine due to the nature of the present problem in this research, which is a large and not perfectly smooth and unimodal search space.

Reactive policies are in place in many storage yards as a result of numerous change orders that are part and parcel of construction projects. In the event that changes do not impact the predefined consumption schedule, a clear, predetermined bill of materials exists that can be handed to the yard personnel to help them improve their placement policies. This is the next step of the research: to propose optimized, proactive placement policies on the laydown areas. Simulation, GA and consumption schedule are employed to find the optimum, dynamic layout of the yard materials. Comparisons are also made between these policies to draw useful conclusions.

1.5.THESIS ORGANIZATION

This thesis has been divided into six chapters. Chapter two provides a thorough review of previous studies related to algorithms and applications of computer simulation, genetic algorithms and site and yard layout control and management in the construction field.

In chapter three, the problem at hand is explained in detail, and reactive placement policy is tried, modeled, evaluated and optimized using GA integrated with simulation.

Chapter four discusses the proactive, improved placement policy where consumption schedule exists and the yard foreman knows what materials are used for a period of time in advance. Flowcharts and graphical illustrations of the algorithms and GA-simulation interaction are discussed in detail in the third and fourth chapter and theoretical backgrounds are clearly explained.

Chapter five presents several case studies (real and fictional) to validate the suitability and usefulness of proposed algorithms. A real case study from the steel fabrication industry is used as an example of the application of the present placement policies on a construction yard.

Finally, chapter six concludes the thesis with a summary of the work along with its contributions, limitations and recommendations for future enhancements.

CHAPTER 2: LITERATURE REVIEW

2.1.INTRODUCTION

This chapter presents a summary of the state-of-the-art developments in the following areas:

1. Application of simulation in construction with a focus on site and yard layout optimization problems.
2. Materials management and handling and its role in construction projects.
3. Genetic algorithm and its application in construction and layout modeling.

The first section focuses on a brief introduction to simulation and its application in construction engineering management, with a particular view towards site and yard layout modeling. The second section concentrates on materials handling and management, and the last section covers literature on the application of heuristic methods (in particular, genetic algorithm) in modeling of construction processes.

2.2.HISTORY AND APPLICATION OF SIMULATION IN CONSTRUCTION

Construction simulation has been defined by several researchers as a powerful mathematical-logical tool, based on a real system, that can be utilized by experts for productivity measurements, risk analysis, resource planning, design and analysis of construction methods, and project duration measurements (Shawhney et al. 1998, Pritsker 1986). Simulation has shown to be sophisticated in modeling of a number of situations that other tools fail to model, including examining the interaction between flow activities, determining the idleness of productive resources, and estimating the duration of construction projects, since it provides a fast approach to experimenting with different scenarios without changing the systems themselves (Zhou 2006). Haplin (1977) introduced one of the first generations of simulation programs called CYCLONE (CYCLic Operation Network). This program is a discrete-event simulation algorithm used to analyze and model construction processes and activities, and led to huge acceptance and recognition in academia. CYCLONE takes advantage of some graphical elements to model construction situations by considering the repetitive activities in a construction project.

With the introduction of CYCLONE, a variety of other simulation tools were also offered. Paulson et al. (1987) introduced INSIGHT (INteractive Simulation using Graphics Techniques) which provided an economical approach to collect production time data in the field using field-collected videotapes, and to make powerful simulation analysis and design techniques available on computers at the field-office level (Appleton, B. J.A., 2002). Martinez and Ioannou (1994) presented STROBOSCOPE (State and ResOurce Based Simulation of Construction ProcEsses), which is an open source simulation framework, to model common processes in construction engineering. Stroboscope models consist of a series of programming statements that define a network of interconnected modeling elements which control the simulation. Stroboscope modeling elements have attributes, defined through programming codes, which describe how they behave throughout a simulation. Attributes represent duration, priority of an activity, queue time, and the amount of resource that flows from one element to another. CIPROS (Odeh 1992) is a knowledge-based construction planning simulation system that takes advantage of a hierarchical object-oriented representation for resources and their properties. The integration of process-level and project-level planning by representing activities through process networks is one of the strongest features of this program (Martinez 1998).

CYCLONE and some of the abovementioned programs have successfully modeled construction projects, but modeling complications existing in such simulation tools have made their application limited to academia. Such inherent limitations involve arduous and time-consuming tasks in modeling which makes the modeling considerably unappealing for practical applications

(Hajjar and AbouRizk 1999). To overcome such problems, a user-friendly platform was needed to not only serve the general modeling of construction processes, but also to provide opportunity to develop special-purpose simulation templates within the framework. Simphony (Hajjar and AbouRizk 1999) is such an environment in which practitioners can not only model general construction processes easily, but also allows development of special-purpose templates within the framework to suite the needs of special fields and projects. Developers can use Simphony to implement highly flexible simulation tools that support graphical, hierarchical and integrated modeling very conveniently, while providing a user-friendly graphic interface by which users can perform the modeling with great ease.

Special Purpose Simulation (SPS) templates aim at one particular domain and enable the practitioners to model a project within the domain in a manner where symbolic representations, navigation schemes within the environment, creation of model specifications, and reporting are completed in a native format (AbouRizk 1998). "By making the model environment specific for a given industry many advantages are gained including wider acceptance and use in a practical settings. SPS tools help bring simulation to the desks of construction engineers who have little or no experience with simulation theory" (Hajjar and AbouRizk 1996). Examples of some SPS templates are Ap2Earth (Hajjar and AbouRizk 1996), CSD (Hajjar et al. 1998) and CRUISER (Hajjar and AbouRizk 1998).

2.3. MATERIALS HANDLING AND ITS ROLE IN CONSTRUCTION PROJECTS

In simple terms, materials handling is moving, loading and unloading of materials. In order to achieve such goals safely and economically, equipment and techniques are used in alignment with an overall view at the project production cycle. In fact, in any industry involved in construction and manufacturing, materials management and handling shall be practiced with the utmost care from the point of receipt and storage of raw materials, through production, installation and commissioning. It should be emphasized that materials handling is not inherently a value-adding process to the finished product, but it maintains the continuity of the materials flow in the production life cycle. American Materials Handling Society defines materials handling as “the art and science involving the moving, packaging and storing of substances in any form” (Bolz and Hagemann 1976). Throughout the materials handling process, three important factors should be considered:

- Materials handling shall be operated with the lowest possible cost and time to avoid decrease in craft labor productivity. Timely movement of materials and optimized use of resources for materials haulage are paramount in any handling process.
- Handling operation should take advantage of proper methods and equipment so as not to compromise safety and workflow continuity.
- Space utilization for materials should be optimized, that is, minimum space shall be used for materials storage and handling as space is a resource, especially in congested yards and sites.

Some experts define materials handling as “the art and science of conveying, elevating, positioning, transporting, packaging and storing of materials” (Siddhartha 2007). Materials handling scope of work covers a broad range including (Siddhartha 2007):

- Bulk materials which fall particularly within the scope of mining and construction industry.
- Industrial packaging of semi-finished or finished goods.
- Warehousing from raw materials to finished product stage.

The significance of materials handling is becoming increasingly bolder to practitioners as many enterprises go out of business because of inefficient materials practice. Perhaps one of the most important merits of a proper materials handling system is increased productivity, and thereby, higher profitability (Materials Handling Manuals 2008).

Materials handling is, however, a part of a broader domain of management called *materials management.* This discipline covers a vast scope containing major functions of identifying, acquiring, distributing, and disposing of materials required on a construction project. Applying the right material management methodology in construction projects would result in real savings, improved labor productivity, reduced surplus, and improved cash flow. However, it has always been difficult to convince the industry of the necessity of such techniques for materials, even though material cost constitutes more than half of the total project costs. Moreover, any delay in material delivery and supply would incur major cost and delay in projects in today's competitive market. The need for long-term investment in materials management is recognized by construction companies which aspire to be competitive.

As indicated above, materials management covers a broad range from identification of suppliers to warehousing and material tracking on construction sites. Below, a brief introduction of different sub-disciplines is given to further explain what category the thesis problem falls into, compared to the entire materials management domain (CII report 1988).

2.3.1. Project planning and communication

According to the Construction Industry Institute (CII) report on costs and benefits of materials management (1988), planning and communication are the two most important elements of any effective materials management system, since effective communication between different stake-holders reduces the risks of misalignment and defaults. The latter will result in change in scope or in developments which normally incur costs of rework or change.

In a robust materials management system for construction projects, all responsibilities shall be clearly defined in advance through proper communication. Very often, these responsibilities are defined in a way that the owner and/or engineer assume(s) risks for engineering equipment and major items, while the construction manager and/or contractor assume(s) all responsibility for bulk materials. Such arrangements should be approached with care since they tends to generate a fragmented management system as opposed to an integrated one. This is an excellent time for a proper communication system to notify the contractor about the specific project characteristics or constraints which may change the scope or cost of the materials management effort. Examples of such constraints include restricted site access and lay-down areas, schedule compressions and changes and purchasing approvals.

2.3.2. Material takeoff (MTO)

One of the early stages in material takeoff is coding. Construction companies might have different coding systems for different projects, but they shall maintain a unified coding system throughout different phases of projects. Effective material takeoff concerns a comprehensive material tracking system which goes beyond simple material sheets preparation and talks about application of sophisticated material tracking systems such as RFIDs and GPS. Further descriptions of such systems are provided in the next sections.

2.3.3. Vendor inquiry and evaluation

Several factors are considered in vendor evaluations including cost, delivery, production capacity, geography, owner preference, laws and regulations and previous owner performance. For expedition purposes, experts are often hired to evaluate vendor performance and past experience. Vendors' submittals in past projects can be saved electronically to assist in saving time and cost in search for vendor performance evaluation.

2.3.4. Purchasing

Bills of materials constitute the project material requirements, but purchase orders which result from the bills define the actions that have been taken to satisfy such requirements. Again in this stage, computer systems may be of assistance, providing vendor quotations and purchase orders directly from line-item data stored in the bill of materials file. It is also important that the purchasing functions be fully integrated into the overall materials management system.

2.3.5. Expediting and transportation

Expediting consists of a series of functions which assist material vendors in meeting their contractual agreements. It could also be practiced to provide timely information regarding expected material deliveries to all concerned project personnel. This sub-discipline requires efficient and proactive communication between field personnel and the project expeditor to update changes and amendments quickly.

A material transportation plan should be formulated early in the project and address factors such as rates, routing, inspections and claim resolution. When it comes to large industrial projects, material transportation will be of profound importance, since issues such as permitting, import licenses, port clearances, etc. might arise. Loss or damage to major items might incur huge costs, which should be avoided thorough a comprehensive material transportation plan.

2.3.6. Warehousing

Once materials come to the lay-down yard, it is important to:

- Place the material in the right place to minimize the time and cost of material haulage to the consumption unit in terms of resources available on the yard.
- Maintain an accurate inventory, a detailed record of available materials on the yard, through timely inventory recording.
- Safeguard the key components on the yard and implement careful warehousing techniques.

Sometimes, especially in construction sites, the contractor does not select the lay-down areas. Owners usually dictate the layout of the lay-down areas due to their own limitations. However, within the dictated areas, contractors have freedom to place the materials in the places they desire based on resource availability and minimization of time and cost of material haulage.

Figure 2-1 illustrates a discipline-breakdown of a typical material management system.

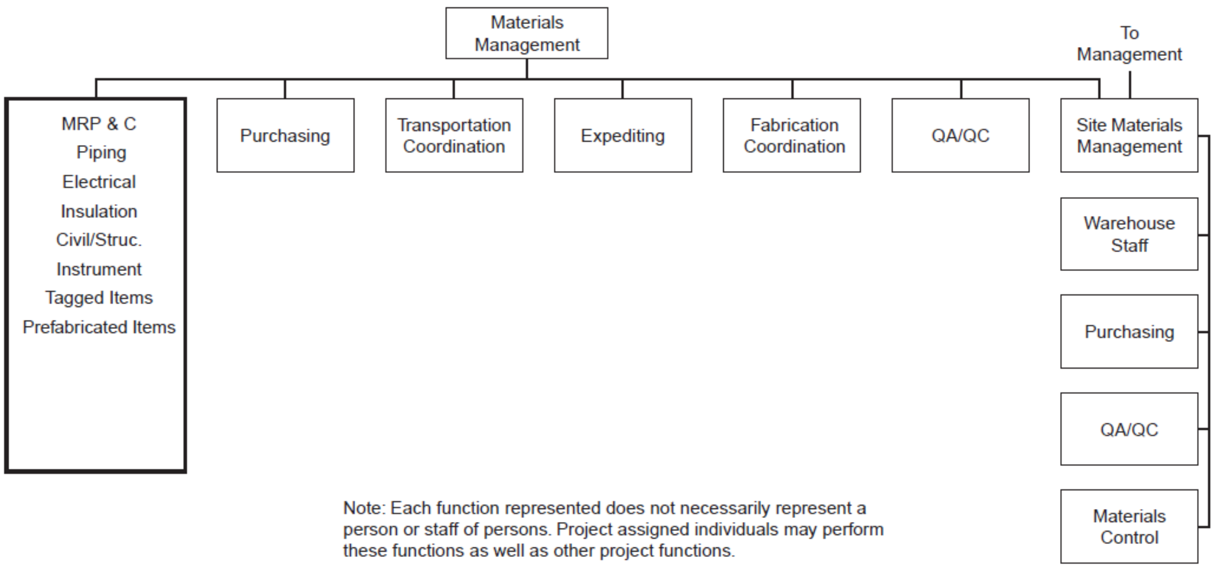


Figure 2-01. Material management discipline breakdown structure (Construction Industry Institute 1999)

It can be inferred from this figure and the aforementioned explanation of different disciplines in materials management that materials handling concerns warehousing and to some extent purchasing and expediting. It can also be said that site materials management is greatly impacted by proper materials handling, but the focus of this study is mostly materials handling within construction laydown yards. Efficient yard operations provide benefits throughout the entire supply chain, while effective yard management plays a critical role in the successful flow of goods in and out of distribution centers, manufacturing plants and warehouse facilities.

Some researchers have tried to formulate materials placement and handling on construction yards, such as Gambardella et al. (1998), who tried to address spatial allocation of containers on terminal yards. They presented a decision support system for the management of an intermodal container terminal. They also investigated the problem of spatial allocation of containers on the terminal yard, the allocation of resources and the scheduling of operations, in order to maximize a performance function based on some economic indicators. Zhang et al. (2003) studied the storage space allocation problem in storage yards of terminals. This problem is related to all the resources in terminal operations, including yard cranes, storage space, and internal trucks. They addressed the problem using mathematical programming approach to optimize the space allocation.

Crainic et al. (1993) investigated space allocation by studying the space and time dependency of events and proposed space optimization method based event handling of the incoming materials (container being the materials) on terminals. Shen and Khoong, C. M. (1995) established a decision support system to solve a large-scale planning problem concerning the multi-period distribution of empty containers for a shipping company. Appleton (2002) developed a special purpose template in which he used priority rating logic to handle process interaction for the tower crane. The significance of his research to this study is that he prioritized the tower crane tasks in the same way as an important resource should—based on the significance of the incoming activity, and modeled it in a special purpose simulation template. The same priority rating could be used for incoming materials to the site based on the significance of the job for placement operations.

Today, materials handling experts on large construction yards take advantage of sophisticated materials tracking on the yard to reduce the cost and time of finding the right materials. Large construction projects naturally have large storage yards. A great amount of time might be consumed to track and find materials within a huge yard, which results in craft labor loss of productivity, delay and confusions and missing key materials. Figure 2-2 shows a large storage yard with massive amount of materials and containers, which makes detection of the right material very arduous.

Figure 2-2. Large, congested construction yard

In industrial construction projects where there are usually small pieces, such as pipe-spools and appurtenances, it is of significance to find a method by which managers and field engineers can efficiently track such materials. One of the very effective tools by which it is possible to track materials accurately is radio-frequency identification (RFID). RFID uses a wireless non-contact system, which takes advantage of radio-frequency electromagnetic fields to transfer data from a tag attached to an object, for the purposes of automatic identification and tracking. RFID tags do not always use electric power and are often powered by the electromagnetic fields used to read them. Others use a local power source and emit radio waves. The tag contains electronically stored information which can be read from up to several meters (yards) away. An advantage of RFID over barcodes is that the tag does not need to be within line of sight of the reader and may be embedded in the tracked object.

Several researchers studied the application of RFID in material tracking in construction projects. For the purpose of brevity, only two of the works on the application of RFID are mentioned here. Wing (2006) studied the potential of RFID tagging technology in construction and management and reviewed a number of applications for improving efficiency. Studies done by Grau et. al (2009) demonstrated the effectiveness of using RFID tags in tracking materials to improve productivity in construction.

2.4. GENETIC ALGORITHMS IN CONSTRUCTION OPTIMIZATION PROBLEMS

Genetic Algorithm (GA) is a search algorithm based on philosophy of natural evolution and biogenetics. Detailed description of the nature of GA will be included in chapter three, but here the discussion is limited to the literature review of the application of this heuristic method in construction engineering management. In this section, discussion on the literature is divided into two separate groups. First, applications of GA in construction management and optimization problems in general are discussed. Then, attention is paid on GA application in site and yard layout design and optimization.

2.4.1. Application in construction optimization problems in general

GA has been successfully applied to numerous areas in construction engineering and management as a very effective heuristic method. One of the early attempts of resource optimization using GA is the work of Ugwa and Tah (1994). Their research was an exploratory work investigating the integration of genetic algorithms (GAs) with organizational databases to solve the combinatorial problem in resource optimization and management. They used two levels of knowledge (declarative and procedural) to address the problems of numerical function, and combinatorial optimization of resources. The research showed that GAs can be effectively integrated into the evolving decision support systems (DSSs) for resource optimization and management. Chan (1996) investigated resource scheduling with the aid of

GA. One year later, Feng and Liu (1997) introduced an algorithm based on the principles of GA for time-cost trade-off optimization in construction. Leu and Chen (2001) developed a similar framework for time-cost trade-off under uncertainty. Hegazy and Wassef (2001) used GA to determine the optimum combination of construction methods, number of crews, and interruptions for each repetitive activity. In 2003, the same researchers offered an approach for resource management and optimization in construction projects using a combination of simulation and GA. However, their research does not provide integration between GA and simulation. Toklu (2002) presented a genetic algorithm applicable to projects with or without resource constraints. In this application, chromosomes are formed by genes consisting of the start days of the activities. This choice necessitated introducing two mathematical operators (datum operator and left compression operator) and emphasizing one genetic operator.

Zheng (2003) presented a multi-objective approach for optimizing two resources at the same time on the basis of GA. Hegazy and Petsold (2003) developed a model for performing dynamic project monitoring and control by means of the overall GA-based optimization of project intermediate schedules. Senouci (2004) studied a genetic algorithm based approach for resource scheduling. He provided a sophisticated model that considered several aspects such as time-cost trade-off, cost minimization, multiple strategies and precedence relationships. In the same year, Hegazy and Elhakeem introduced a distributed scheduling model for resource planning, and cost optimization of large construction projects which involve multiple distributed sites. Kim and Yoon (2004) applied an integrated program containing a neural network engine and GA algorithm to help find the optimal parameters of back-propagation algorithm, enhancing the accuracy of cost estimation.

Zhou (2006) developed a special purpose template for constructing the shaft in tunneling operations. In her work, she developed a GA based site optimization algorithm and incorporated it in a simulation model which used the optimized site-layout as a starting point of simulation and then shaft construction would begin at later stages. She employed hard and soft constraints as target functions (or fitness functions) for optimization purposes. However, her work does not facilitate interaction between GA and simulation and data exchange is carried out only once at the beginning of the simulation. Similar to the work of Kim and Yoon (2004), Feng et al. (2010) utilized GA for optimizing back propagation method to handle low convergence speed. The method they proposed was based on analyzing the basic fundamental that states how to use genetic algorithm to improve the ability of back propagation. Kim (2009) presented an improved elitist GA for resource-constrained scheduling of large projects. His proposed algorithm allocated multiple renewable resources to activities of a single large-sized project to achieve the objective of minimizing the project duration. A permutation-based decoding procedure was developed using the improved parallel schedule generation scheme. Finally, Kim (2010) developed an adaptive hybrid genetic algorithm search simulator (AHGASS) for resource scheduling problems. In this work, he dealt with algorithm performance in regard to algorithm runtime, especially against runtime used in generating optimality. Since the major drawback of using GA is the great length of time required, it is meaningful to investigate the significance in algorithm runtime between AHGASS and optimality. To address this issue, he attempted to investigate the difference in algorithm performance with regard to algorithm runtime.

2.4.2. Application in site and yard layout management

Site and yard layout optimization have differences and similarities. They are similar in the sense that some space is occupied by materials on a storage yard and some space is occupied by working units on a site. The objectives are the same; time and cost trade-offs shall be maintained so that productivity is raised, safety is improved and quality targets are fulfilled. However, on a construction site, there are several working units in which working processes are carried out, and resource consumption and interaction are far more complex than those on a construction yard. For instance, in a bulk material movement project, heavy trucks, as the most important resource, are delayed in a depot unit if there is a queue for unloading. The problem is more pronounced if the trucks in question are to be used in another working unit such as loading source (e.g. in a reclamation project). Thus, it can be concluded that in a site layout problem, the following shall be taken into account:

- Proximity of the working units with respect to one another.
- Resource consumption and interaction within the working unit and between the working processes.

On a construction yard, however, materials handling on the laydown yard concerns a single working unit in which placement optimization may be carried out by using the resources (labor, equipment and materials) consumed solely within the unit.

Despite the fact that many studies have been conducted to plan and optimize site layout, their practicality has always been in question. In fact, it cannot be guaranteed that the layout enhanced from the optimization process is the optimal site layout when facilities are interacting (Zhou et al. 2009). There are many working process factors such as production rate, resource allocation, equipment idleness, and complex relations between activities in construction projects that should be accounted for in site layout planning.

Zouein and Tommelein (1994) introduced a heuristic model that allocates site space to resources associated with an activity schedule so as to prevent spatial conflicts. Their work addressed the dynamic layout planning problem, which involves creating a sequence of layouts that span the duration of project construction. They named their program implementation *MoveSchedule* and introduced it as a unique model builder which addresses the reuse of space by resources whose presence on site depends on a schedule. They stated that most of the efforts in the area of site and yard layout planning have focused on static layouts where heuristic methods are employed to optimize the layout based on the adjacency constraints, and the devised layout remains constant throughout the project life span. However, since the construction project is dynamic in nature, a sequence of layouts to span the duration of the project with regards to activity schedule shall be taken into account. They explained that the sequenced layouts must span a specific time interval and accommodate the resources that are scheduled to be present during this interval. Based on this concept, resources are divided into two groups, dependent and independent, on the basis of timing of their presence on the site. The dependent resource frees the space area it occupies with respect to the activity duration.

A valuable work by Paul and Chanev (1998) discussed the notion of improving simulation with the aid of GA. They stated that most simulationists build their models to solve specific problems, that is, that some parameters are set at the beginning of the modeling to find some desired outputs such as time or cost. Model parameters are selected based on initial guesses or engineering decisions which might not be optimum parameters. Efforts to solve the inverse problems have rarely been made. The primary question they strived to solve is how one can find the optimum parameters to establish the simulation model. Paul and Chanev (1998) took advantage of GA to optimize the input parameter by integrating a programmed-simulation model with GA to improve the simulation model at each iteration. They explained that this integration will consume a considerable computer run-time, which necessitates the design of an advanced algorithm to maintain the feasibility of the technique. The problem they solved was simulation of a steel production line, in which the target was the costs of steel waste. The input parameters they attempted to optimize were the number of torpedoes, cranes, steel furnaces and volume of the torpedoes in tons.

Tommelein (1999) attempted to solve the ‘tool-room’ problem in construction projects, which involved determining the best number of tool rooms and their configuration among different alternatives to meet the demands, using simulation by assessing workers' travel time and waiting time. The developed model can examine various parameters in different scenarios, but cannot optimize the layout.

Azadivar and Wang (2000) presented a facility layout optimization technique which takes into consideration the dynamic characteristics and operational constraints of the system as a whole, and is able to solve the facility layout design problem based on a system's performance measures, such as the cycle time and productivity. These researchers were among the few who applied genetic algorithm for optimization of layout for manufacturing effectiveness, and at the same time, they used simulation as a system performance evaluation tool. They argued that most facility layout solutions focused on minimizing the amount of transportation, and the effect of a given layout design on the production function of a manufacturing system had been just limited to cost of materials handling. They highlighted the significance of other aspects of production such as shorter cycle times in manufacturing of industrial products. Their study exclusively focused on industrial design and manufacturing, underlining the multi-objective optimization approach where time and cost are of mutual significance. The problem they tried to overcome is composed of a manufacturing system consisting of several workstations in which a specific number of parts need to be processed. The parts require processing on different subsets of the same number of workstations and obviously have different processing times. The objective is to find a desired design for the system in which the arrangement of such workstations has been optimized. They used GA as the optimization tool, and simulation to process production rate and cycle times. The main challenge one might face in using GA is encoding (this concept will be described in detail in the next chapter). Azadivar and Wang (2000) took advantage of slicing method for chromosome representation. The system they proposed consisted of a GA package, a simulation package, an automatic simulation model generator, and a graphical user interface. Their results show significant improvement in the field of process layout design improvement.

Tawfik and Fernando (2001) focused on safety and space analysis by specifying hazard zones and identifying the moving path of vehicles and workers. In this study, a simulation model was used as a mathematical tool. In order to plan stock yard layout, a simulation model was presented by Marasini et al. (2001), to evaluate "what-if" layout scenarios. For this analysis, three parameters, namely, product handling cost, throughput time for a lorry, and vehicle waiting time, were the assessed outputs of the model. GA was also integrated with the simulation model to optimize the allocation processes of products to different storage facilities. That is, the GA application in this model was only for optimization of input data, not for optimization of the layout.

Marasini et al. (2001) focused on identifying the appropriate methodology for designing and managing the stockyard layout that ensures efficient storage and dispatch of products, and provided the convenient flow of rotation of products within the yard. They introduced a mixed simulation model employing a heuristic method to evaluate 'what-if' scenarios, and to recommend a suitable methodology for the management of stockyard space for precast concrete products. They focused on concrete stockyard, and stated that concrete products are stocked on the yard intuitively. As a consequence, the industry experiences space congestion for both the storage and retrieval of different concrete products. The main objective of their study was to reduce the throughput time. Their work was divided into three distinctive steps:

- A preliminary layout design considering the hard and soft constraints such as space requirements for offices, plants, storage spaces, roads and aisles.
- Development of a simulation model to study the behavior of the stockyard (business process improvement and problem objectives such as cost and time).
- Optimize the stockyard layout using GA. They used GA to allocate products to different locations, considering the storage spaces are fixed and the products can go anywhere.

It should be noted that they did not provide the results of their study in 2001 as they still needed further analysis and study and the work they conducted seemed noticeably complex considering all the aspects in simulation and modeling. However, Marasini and Dawood (2002) continued their efforts, and provided some promising results presenting reduced throughput times once they used GA in collaboration with simulation. They developed a process model for the evaluation of the stockyard layouts for standard precast concrete products. Similar to Azadivar and Wang (2000), they established a framework to optimize the simulation model inputs using GA. As indicated above, the result of their analysis shows significant improvements in throughput time for loading and dispatching of the concrete products using their GA-based allocation.

In the design of service facility layout of a high speed rail station in a renovation project, which was involved in designing the location of new facilities as well as relocating some existing facilities while the other ones were considered fixed facilities, the objective was to reduce walking time of passengers among the facilities (Lee 2012). In this case, simulation and ant colony method were integrated; simulation was implemented to estimate walking time of passengers, and ant colony was used to find optimized layout.

It was revealed that the bottleneck of the simulation application in logistics is input data, special knowledge needed for preparation of reliable and sufficient input data for the simulation model (Koing et al. 2011). Therefore, Koing et al. (2011) focused on data preparation concept in the early planning phase for logistic simulation. They demonstrated that some data, such as material quantities, general activities, and milestones, can be retrieved from other models like Building Information Models (BIM). Some other information, specifically for logistics, such as means of transportation and packing units, can be defined by the user. Their proposed model integrated data models in order to prepare inputs for logistic simulation. Examining different layout alternatives, simulation outputs such as utilization of resources, waiting times, and allocation of storage areas can be analyzed to evaluate the layouts.

Simulation was also applied to plan construction logistics in outfitting processes (Voigtmann and Bargstadt 2009). To choose between two different strategies for storing material, central storage and decentralized storage, two different simulations were run to evaluate the effect of changing different factors on logistic time. In addition to the construction area, simulation has been used in design of industrial plant layouts. It was indicated that simulation tools can provide more information, such as total time in the system in comparison with other techniques that consider only transfer costs between departments in industrial plants (Smutkupt and Wimonkasame 2009). Moreover, simulation can connect the planning stage to operation to reduce costs in production and logistic systems (Wenzel et al. 2010).

Despite the many advantages of simulation for evaluating "what-if" scenarios, the possible application of simulation in selecting the allocated area, the position of construction facilities, the evaluation of different logistic strategies and the complexity of the models with respect to numerous variables have still remained a challenge in simulation. Since in simulation, several experiments along with variable adjustments should be made to find the optimal construction site logistics, identification of relevant factors and elimination of irrelevant ones are of significance. In addition, some dynamic aspects of site layout planning, such as re-location of facilities over time, have not been addressed in the existing simulation models (Voigtmann and Bargstadt 2010).

2.5.CONCLUSION

A thorough overview of the work in the area of simulation, materials handling and management, application of genetic algorithm in construction domain and application of simulation in site and yard layout planning was presented in this chapter. After careful study of the previous works on the topic, the following conclusions can be drawn:

- In the area of materials handling and management, it seems that little study has focused on reduction of throughput time on construction storage yards. Some investigation have been carried out on design of optimum site layout having several workstations or providing some tools to find the optimum placement arrangement of offices, roads as well as laydown areas on concrete stockyards, but they did not provide detailed parametric studies on how different equipment might change the input parameters for the heuristic methods and the simulation engine. Their work has focused on one construction domain (e.g. industrial manufacturing and/or concrete stockyard), and on introducing a tool to solve the optimum layout at hand.
- Some studies have concentrated on the effect of activity schedule, and thereby, incorporated the dynamic nature of the site and/or yard, but they did not compare and contrast the static and optimum dynamic laydown yard management strategies to find the gaps.
- The direct and indirect impact of change orders on materials handling is an area which requires more research. The amount of investigation on this topic indicates the rarity of such studies.
- Interaction and integration of simulation and GA have not been fully studied in past literature. Some researchers (stated previously), used the result of GA and fed it to the simulation or vice versa. Continuous information exchange during the course of analyses and layout design has been scarcely investigated by previous researchers.
- More study is required to capture the effect of material consumption, material size and density, capacity of laydown areas and number of available equipment resources on the reduction of the throughput time.

Based on the aforementioned conclusions, this research aims to fill the gaps the previous investigations have not fully filled. This study strives to improve the process of materials handling on construction stockyards, as well as to provide a tool for optimization. In particular, it attempts to:

1. Identify the challenges the yard foreman usually faces in stocking the materials on the yard and help improve the current practice by means of sophisticated, fully integrated mathematical methods.
2. Propose an improved materials handling process by incorporating the effect of time (dynamism) to material placement practice.
3. Present optimum laydown yard management for several material stocking policies including current placement strategies and proposed, preferred policy in which consumption schedule is known in advance to the personnel on the yard.
4. Compare and contrast the best practices in stocking the materials on the yard to draw conclusions on the superiority, profitability and feasibility of each method.

CHAPTER 3: REACTIVE LAYOUT OPTIMIZATION OF CONSTRUCTION YARDS USING GENETIC ALGORITHM INTEGRATED WITH SIMULATION

3.1.OVERVIEW

Material management is the work process of planning, controlling and executing cross-functional activities to ensure the quality and quantity of engineered and bulk materials are available in a timely and cost effective manner to support the construction execution plan and the facility turnover process. In this area of construction engineering, getting the Right material to the Right place at the Right time and at the Right cost (the 4Rs) are of profound significance. An effective material handling technique should ensure the timely availability of materials, compliant with receipt and installation per the project construction schedule, and contribute to minimizing surplus at project completion.

Material management is a subset of material handling which is a more general area in construction engineering as mentioned in the previous chapters. Material handling itself is in close correlation with other disciplines such planning, estimating, drafting, purchasing, installation and commissioning, etc. Changes, disruption and delay in any of the other disciplines naturally impact material management and handling. In response to such changes, yard management policies, as part of the overall material handling program, react accordingly, and change reciprocally. Based on such interactions, two primary material placement policies in large construction yards can be identified:

- Reactive placement policy: where the receiver (the person who receives the material from the supplier/vendor/mill or any other provider of the material) does not have the arrival schedule for a certain period of time informing him what material arrives at site on the days ahead. The receiver also does not know what material will be consumed and leave the yard in a timely manner (for a certain period of time). S/he only receives daily pick tickets from the consumption unit to feed them right away, and the material arrival list from purchasing telling her/him what material is coming at the same day s/he receives the list. Details of this placement policy are discussed in the next section.

- Proactive placement policy: where the receiver is given a material arrival schedule (as opposed to daily arrival list) informing him about the materials that will arrive at site for a certain period of time. That is, given a 10-day schedule, the receiver knows exactly what material comes to the yard on the fifth day. Moreover, s/he is told in advance what material is going to be used by the consumption unit for the same period of time. In other words, s/he has thorough information (in the form of a schedule) of the incoming and outgoing materials prior to their arrival and release, giving her/him leeway to decide where exactly on the yard s/he can stock the material.

In this chapter, reactive material replacement methodology is discussed first to describe the problem at hand in detail. Then, genetic algorithm (GA) as a powerful heuristic tool is explained, and its different steps are outlined. Then, it is explained how GA can help improve placement strategy as it is suitable to address problems with a large number of possibilities. The chapter proceeds with the role of simulation in the proposed solution and its significance for tackling problems in the construction domain. The continuous interaction and information exchange between simulation and GA is highlighted using descriptive flowcharts and procedures so as to pronounce the effectiveness of the proposed method. After conceptual discussions, computer program implementation as part of the solution strategy is presented. Object oriented programming, as a powerful tool, can be properly utilized in this study as data abstraction, inheritance, class definition for simulation of real objects as well as appropriate encapsulation make programming with such scale much more convenient and self-explanatory. A practical example case study, to describe the suitability of the solution, is presented at the end and results are discussed through parametric studies. At the end, summary and conclusions are given to prepare the discussion which will be laid out in the next chapter.

3.2.REACTIVE MATERIAL PLACEMENT ON CONSTRUCTION YARDS

In the previous section it was discussed that the yard logistic coordinator, who facilitates the most efficient utilization of the laydown and storage areas (the receiver), is faced with a decision as to how s/he should place the materials on the yard. The decision s/he will make could be based on the following:

- Totally random. Wherever there is free space, the material can be placed. This policy is not recommended as there is no order in laydown areas and the yard will face chaos which will ultimately lead to excessive decline in craft labor productivity due to confusion, yard congestion, lost materials, losing accurate record of inventory, etc. Figure 3-1 shows four large, congested laydown yards where any confusion in locating the material could result in considerable losses in time, and consequently, productivity.
- Based on yard segmentation. On most construction yards, provision for the identification of grid-marked storage areas in laydown areas for each item received is carried out to know which material goes where. The laydown yards should have a defined grid location system that can be input into the site material management system. Figure 3-2 shows a typical segmentation of a real construction yard which has been successfully implemented and applied. By using such segmentation, the receiver can track her/is inventory, and use the grid as a map or guideline as to how material placement can be made more efficiently. However, decision-making in this way is solely dependent upon the receiver's personal choice. Generally, the rule of thumb for this decision-making is the availability of free laydown area and proximity to the consumption unit. In most cases, no calculation as to where the different material batches should go is carried out. The receiver arranges the placement based on the grid and her/is own experience in knowing the inventory and an estimate of space availability. This policy is ineffective for the following reasons:

- On large construction yards, it is almost impossible to find the most optimum placement arrangement, especially when the number of incoming batches increases. Imagine an order consists of twenty different materials arrives at a yard that has fifty different grids, fifteen of which have space availability. The receiver is faced with 20^{15} choices to make!
- On large yards, keeping track of space availability is difficult. In some cases, the receiver makes a rough guess that laydown area X has enough space for the batch, but later on s/he realizes that due to safety reasons it is not possible to stack that amount of materials on top of each other. S/he promptly decides to choose the next available laydown area, which may or may not be the best choice.
- If the receiver takes a day off, the next person in charge may not have thorough information of available inventory, nor be familiar with how housekeeping and yard management have been done. Dependency on individuals' personal knowledge is an error-prone practice.

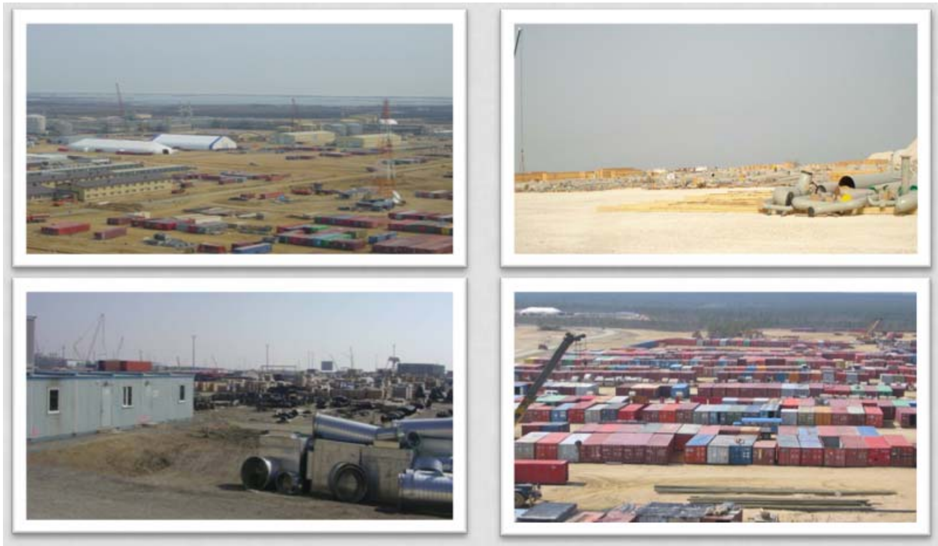


Figure 3-1. Four large, congested laydown yards

- The receiver can be given a daily schedule in advance telling her/im in which grid s/he should stock the material. With a schedule and plan, none of the problems mentioned above can occur. For example, if a batch of material arrives containing twenty different material types to place in twenty different laydown areas, the receiver knows where to place them on the yard grid network, as each material type has a tag with that information. This research proposes such a plan by which the most optimized placement arrangement can be made. The most optimized plan should account for:
    - dynamism of the material flow in and out of the yard,
    - material transfer time/distance from the yard to the consumption schedule,
    - space availability of the laydown areas,
    - especial provisions such as laydown occupancy due to reserved spaces for special jobs,
    - logistics of the yard (yard dimensions, transfer lines to consumption unit, permanent and temporary hauling equipment on the yard)

- and hard and soft yard constraints such as material compatibility constraints (materials of the same type can be stacked in one laydown area).

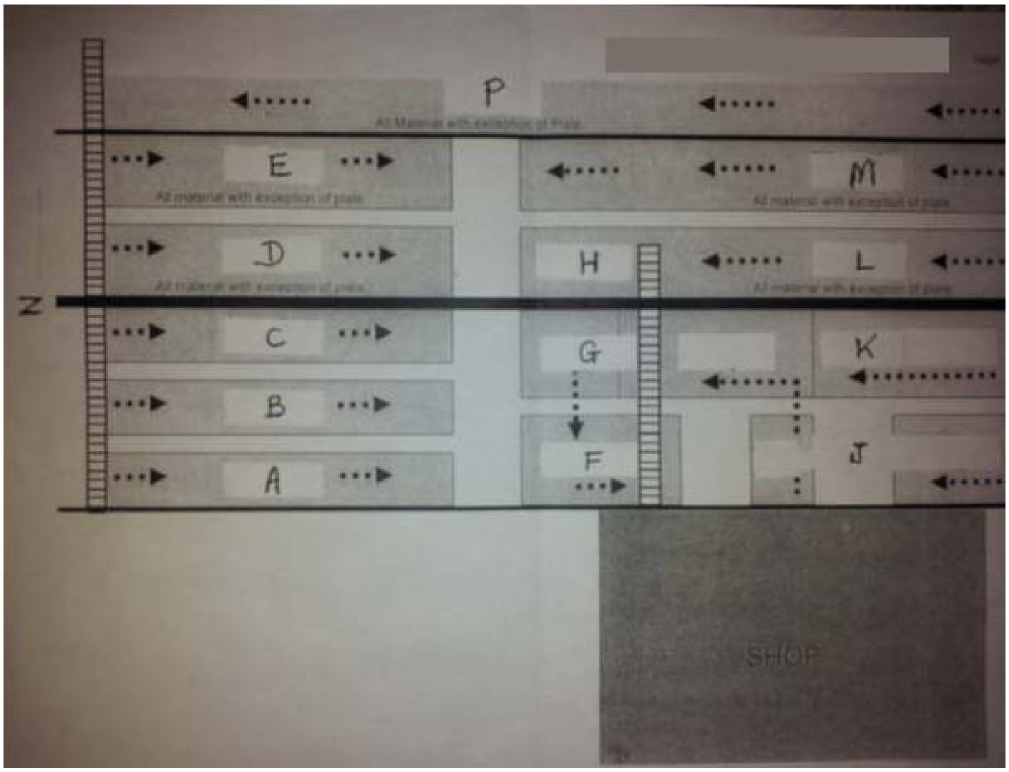


Figure 3-2. A typical, schematic segmentation of a construction yard

An accurate plan having the features outlined above can fall into any of the following categories:

- Single-objective, static, layout optimization of material laydown areas where space availability is the only concern on the yard. The objective is to minimize material travel time to the consumption unit, and no attention is paid to pick tickets/lists by which some material is removed from the yard, and the inventory is updated. In this category of optimization, only a snap-shot of time (i.e. one day) is selected; material arrival for day #n is given to the receiver, and informs him where s/he can place the material.

- Single-objective, dynamic, layout optimization of laydown areas where not only space availability is taken into account, but inventory is updated based on daily consumption. In this case, the assumption would be that the incoming and outgoing schedules are only given for one day, and every day a list of input and output to/from the yard is given to the receiver.

- Single-objective, dynamic, layout optimization of laydown areas where incoming and outgoing materials to/from the yard is known for a longer period of time (say for a month) and decision-making is more holistic. Details of this policy are discussed in detail in the next chapter.

- Multi-objective, dynamic layout optimization of laydown areas where two or more objectives are to be taken into account in optimization such as time and cost.

In a sense, the second category describes a policy which is more practical in nature. The reason that this method is more practical is described herein. Often in construction, the top management strives to avoid unpredictables to minimize risks. Material availability in a supply chain might be the most important aspect of a construction project, since statistically, material costs constitute 50-70% of the total project costs. It would be an ideal case to know exactly what material is needed, consumed and circulated through the project supply chain and life-cycle. Some companies utilize sophisticated computer systems to track material from the moment it is procured, through warehousing, fabrication, transportation, installation and commissioning, as part of the comprehensive material management system. This is an ideal case where material tractability through different phases of a project not only minimizes the risks, but also makes room for other innovative cost and time-saving measures which can be applied to improve productivity and quality. However, this ideal is mostly out of reach due to the unwanted inherent changes and disruptions which need to be accepted as part and parcel of construction project.

To clarify more, consider a steel fabrication company wins a job, and starts by receiving the design drawings from the client. In most cases, a work schedule exists which informs purchasing and the shop manager what to buy and what to consume in advance to fabrication. Figure 3-3 shows a typical fabrication work cycle in which it is known at the end of the second week what material to procure for the project. It is also seen that due to inevitable revisions to the issued drawings, a seven-week leeway has been foreseen in the schedule. The entire back-drafting and final procurement will be done at the end of week twelve at which time fabrication can commence. It is observed that an advanced bill of materials for procurement is ready at the end of week two, and shop issue (consumption schedule) is available at the end of week twelve. In fact, if it was not for the revisions, two weeks wouldn't be spent on drawing review and shop issue and corresponding cut lists would be sent to the shop foreman two weeks earlier. Nevertheless, if everything goes according to the schedule, incoming and outgoing lists of materials can be provided with a reasonable level of detail and be delivered to the shop and yard foreman for enhanced decision making. But still, the existence of revisions introduces two to three weeks delay in finalizing the cut lists in an ideal case. It should be noted that this does not include change orders that might be incurred during the project life.

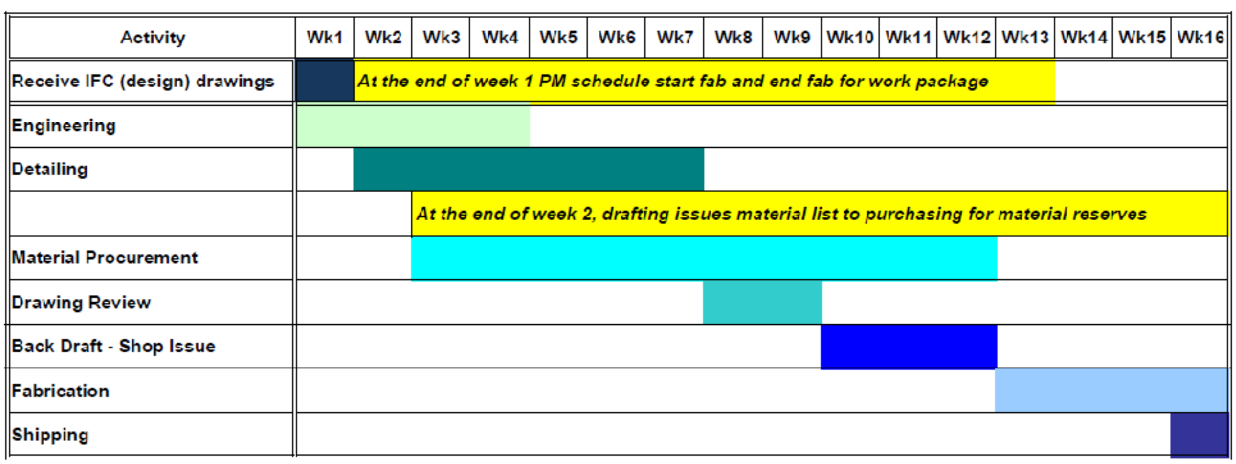


Figure 3-3. Sixteen-week production cycle of a steel fabrication company (Courtesy of Waiward Steel Fabricators)

A procedural representation of the schedule given in Figure 3.3 can be presented in more detail in Figure 3.4 where it is shown how acquiring customer approval has added several activities which are themselves time-consuming. Nonetheless, as also indicated in the schedule, this added time has been anticipated and accounted for in the production life-cycle. The incorporation of customer feedback time into the baseline schedule provides space for proactive material handling and management in which purchase lists and pick lists are known in advance, and leaves room for further implementation of best practices to pursue continual improvement in a construction company. However, a slight change in meeting the milestones totally disrupts the predictions, and incurs unpredictability to the project. Some of these unwanted changes and unpredictabilities are:

- In most cases, design drawings are delivered later than what is anticipated in baseline estimates. This will shift the entire schedule, and as a consequence, the procurement strategy. This is shown by the red rectangle in the flowchart below.
- Revised drawings are returned later than the contractor schedule. In the schedule, two weeks have been foreseen, but depending on different owners and consultants, revised drawings are often not sent back on time.
- Change orders during construction of previous jobs disrupt in-time material handling, procurement and supply. That is, several additional purchase orders are sent to the mills (suppliers/vendors) and arrive at the yard for rush changes and jobs which ultimately interrupt incoming material schedules. The receiver would have to reactively manage the daily additions to the inventory by making room for the new incoming materials and facilitating the swift haulage to the consumption unit (the shop, in the case of steel fabrication).
- Mistakes and errors in drafting, not only on the customer’s side, but also on the contractor’s side would lead to reactive material procurement and supply. Such errors are inevitable in drafting details in construction projects.

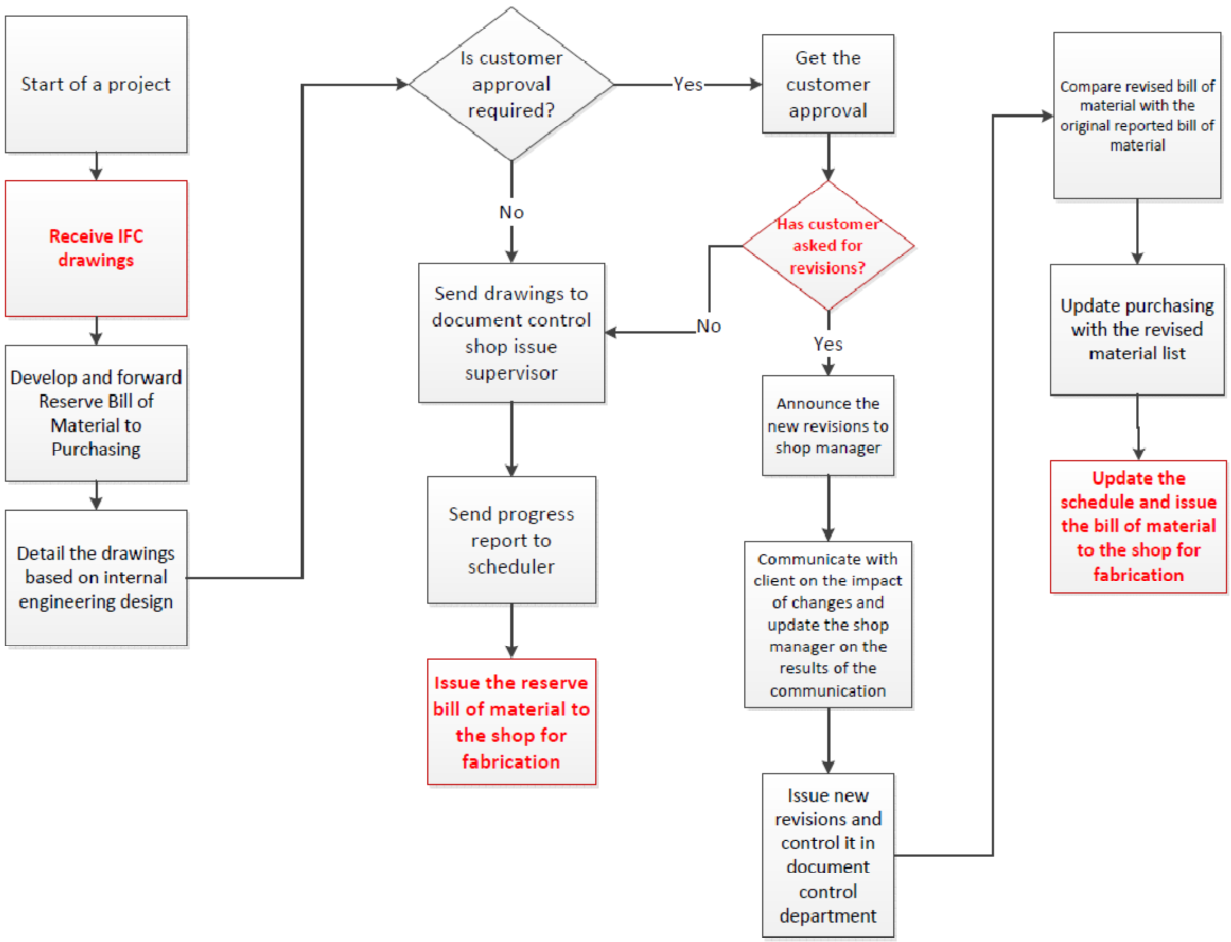


Figure 3-4. Drafting procedure and its interaction with purchasing and consumption of the material

Thus, it is observed that in actuality, the smooth, anticipated material flow to the yard can be easily interrupted, and the problem changes to day-to-day material handling activity on the yard. In fact, this is often experienced in actual construction yards. Now the question is, how we can improve the current placement process and propose an optimized laydown arrangement for a congested yard where material flow is impacted by yard geometry, constraints, hauling equipment and material volume? The receiver faces the following problem: "*How do I lay down the incoming materials that arrive at yard today in the most optimum way?*" Figure 3-5 demonstrates a procedure of placement that a yard foreman or receiver might want to follow to place a material batch on the yard:

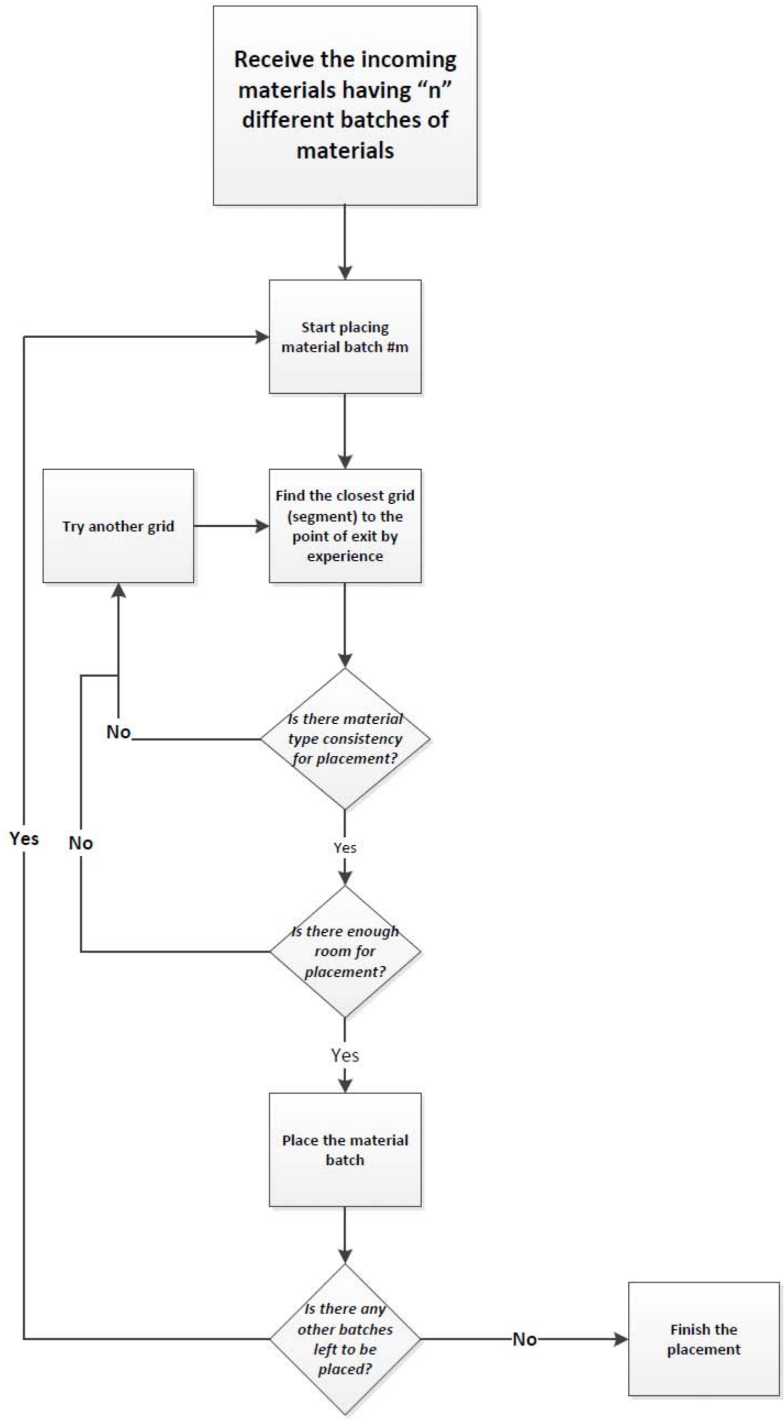


Figure 3-5. A day-to-day, practical material placement policy

Starting from any random day during the working hours, a daily workflow and decision-making process such as the procedure shown above might be followed by the receiver for material placement. Two primary question always exist that need to be answered by the receiver: "*Is there material consistency for* *placement?*" e.g. if s/he wants to place iron angle on any grid, does the same material type (e.g. iron angle) exist on that grid? If not, another grid should be tested. Placing discrepant material would bring chaos and disorder to the yard, and is totally in contradiction with the idea of yard segmentation. The second question involves determination of grid capacity which is the topic of discussion of the next section. In brief, this condition stipulates that the yard segment should not accommodate more than its capacity, not only for obvious physical limitations, but also for safety reasons. It is strictly prohibited by safety provisions to avoid stacking materials more than a certain permitted height.

The material placement policy described in this section is termed as "reactive" placement policy as it reacts to day-to-day incoming material placement problem by finding the best solution to the problem at hand. Although reactive methods in construction engineering are not encouraged, this policy seems to be used in most of the material handling processes in construction companies. As a consequence, this study strives to address this practical problem and proposes practical solutions as it is the need of the current construction industry.

3.3. APPLICATION OF GENETIC ALGORITHM IN PLACEMENT OPTIMIZATION

For material placement in a segmented yard, the receiver must decide which segment would best accommodate the incoming batch, so that resource and time can be used profitably. In a yard with 50 free segments, there will be 50 choices to make to place for a single batch of material. However, the receiver can eliminate most of these choices on the first glance, since in a practical case, her/is criteria for selecting a laydown would first be distance and space availability. It should be noted that these criteria may only be sufficient if a one-day placement problem is to be considered. It will be discussed in the next chapter how other factors need to be accounted for once a wider horizon for laydown selection is considered. Limiting the discussion to a one-day placement problem, the receiver can eliminate most of the available laydown areas by experience, based on the following:

1. S/he needs to identify which segments are either free or have space to accommodate the material. She/he needs to quickly make this decision based on experience and available inventory.
2. Out of these available options, s/he needs to pick one for the first batch based on the proximity of the cell (segment) to the consumption unit. The closer the better. This decision is made again based on experience as s/he does not have an accurate distance measurement tool to determine the closet distance.
3. Decision-making becomes more complicated if there are several material transfer lines for the construction yards. For instance, in a steel fabrication yard, there might be 3 to 4 transfer lines to the fabrication shop, each of which feeds the shop with one particular material type (e.g. iron angle, W-sections, etc.)
4. The decision as to how much material can be stacked in one laydown is again based on experience, and there is rarely a robust procedure to determine the laydown capacity. One rule of thumb, which is formed from safety regulations, is not stacking materials more than a specified height. This decision is subject to receiver preference in most cases.

Considering the aforementioned factors in decision making, the receiver is left with few choices (for instance, 5 choices). For the second batch s/he will have the same factors to impact her/is decision. For simplicity, let's assume that s/he will have the same number of choices to make. Assuming there are 10 batches of materials, the receiver will have 5^{10} choices to make for one day. Even with the few criteria that were discussed above for a one-day placement problem, it is likely that the receiver cannot make the optimum decision in material placement; 5^{10} is almost a countless amount of choices. Needless to say, even computers cannot evaluate 5^{10} choices for a one-day activity.

Facing optimization problems such as the one discussed above where there are countless choices and alternatives to make, heuristic methods can aid the decision making as they are able to evaluate the problem and find the optimum solution through advanced mathematical algorithms. In this study, as discussed in previous chapters, genetic algorithm (GA) is employed to address the problem.

GA was first introduced by Holland in the 60s, and further developed by researchers at the University of Michigan (Goldberg 1989). GA is based on biology, and the fact that natural selection is made to present better populations in consecutive generations. As species evolve, the new attributes are encoded in the chromosomes of individuals. Within this process, evolutionary development such as combination, swap and mutations can occur during breeding. GA then proceeds with survival of the fittest (best) chromosomes over sequential generations. More detailed explanation of GA lingo is given herein.

In GA, a gene is a single encoding of part of the solution space, i.e. either single bits or short blocks of adjacent bits that encode an element of the candidate solution. A chromosome is a string of genes that represents a solution and population is the number of chromosomes available to test. Chromosomes can be bit strings, real numbers, list of rules, program elements or any other data structure. Candidate solutions to the optimization problem play the role of individuals in a population. Some features of GA are:

- Not fast, but covers a large search space.
- Capable of quickly finding promising regions of the search space but may take a relatively long time to reach the optimal solution.
- Good heuristics for combinatorial problems.
- Usually emphasize combining information from good parents (crossover).
- Different GAs use different representations.
- Using mutations and crossovers operators.
- Different selection mechanisms enhance the convergence rate.

Crossover operator recombines the selected parent chromosomes. This operator chooses a random point and swaps the genes before and after that point between the two parent chromosomes in order to create offspring which are two new chromosomes. The basic form of the operator is the random selection of one point within the chromosome before swapping genes. This method is called on-point crossover. There are other crossover methods such as two point crossovers or uniform crossovers, but in this study the simple, one-point crossover method is utilized. Crossover is known as the fundamental step in GA as it provides a method whereby information for differing solutions can be merged to allow the exploration of new areas of the search space.

Mutation operator is designed to avoid falling into local maxima or minima. Using this method, the solution expands the solution space by providing the opportunity to shuffle the population. It is very likely that without mutation, the population would rapidly become uniform under the effect of selection and cross over operators (Coley 1999). To implement this method, some of the genes in a chromosome are randomly changed with a probability equal to the given mutation rate. The GA maintains balance between crossover and mutation operators (Mitchell 1999).

Fitness function is the measure of goodness of the candidate solution. For instance, in a simple mathematical function optimization, the fitness function is the function itself. Depending on the problem to be minimized or maximized, fitness function plays the most important role of presenting the fitness and acceptability of the population. In fact, fitness function evaluates the population and presents information for the GA engine to use for the next step, which is selection. Selection is simply picking the fittest and best member of a population so that the new generation would be a better generation. There are several methods for selection, but in this study Roullete-Wheel selection has been utilized. In this method, better solutions get a higher chance to become parents for next generation solutions. The name Roulette-Wheel stems from the idea that the method assigns each individual a part of the wheel and spins the wheel 'N' times to select 'N' individuals. Roulette-wheel selection is also known as fitness-proportional, which uses a probability distribution in which the selection probability of a given chromosome is directly proportional to its fitness (Reeves 2002). Coley (1999) outlined the steps for implementing this method: first sum up the fitness of all the population numbers, and then choose a random number between 0 and the obtained sum. The next step would be to simply add together the fitness of the population members stopping immediately when the sum is greater than the selected random number. The last individual added is the selected individual and a copy is passed to the next generation. Figure 3-6 illustrates a generic flowchart of genetic algorithm.

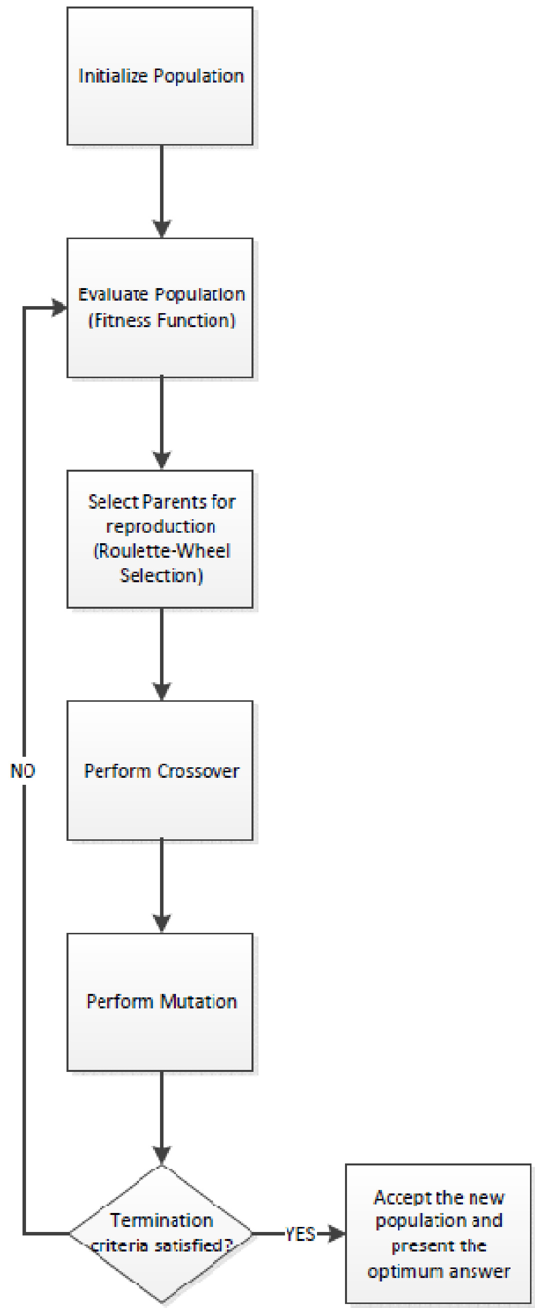


Figure 3-6. A generic flowchart of genetic algorithm

This flowchart will be detailed in later sections as each part is composed of some other components that need to be discussed in more detail.

Some benefits of genetic algorithm are (Al-Tabtabai and Alex, 1998):

- Concept is easy to understand.
- Modular—separate from application (representation); building blocks can be used in hybrid applications.
- Supports multi-objective optimization.
- Good for “noisy” environment.
- Always results in an answer, which becomes better and better with time.
- Can easily run in parallel.
- The fitness function can be changed from iteration to iteration, which allows incorporating new data in the model if it becomes available.

Moreover, Al-Tabtabai and Alex (1998) state that the use of GA in optimization is appropriate when the space to be searched is large, or when it is known not to be perfectly smooth and unimodal.

Some issues with GA are:

- Choosing parameters could be difficult and needs trial and error, including population size, crossover and mutation probabilities.
- Termination criteria.
- Its performance can be too slow but covers a large search space.
- Its success strictly depends on proper and exact definition of the fitness function.

In addition to these issues, incorporation of simulation and definition of chromosome in yard laydown management problem could be two other important problems that should be accounted for in this study.

3.1.ROLE OF THE SIMULATION

In the particular problem of this study, simulation of construction processes and activities is useful as it enables the user to incorporate resource allocation in problem solving. In fact, simulation can easily model the laydown placement operation, and material haulage from laydowns to consumption units no matter how many transfer lines exists. It is also capable of reporting the time of the analysis or determining the distance or cost of the material haulage to the point of exit. Therefore, it could be a perfect candidate for evaluating different placement arrangements which would make the simulation itself a fitness function. The question which arises here is “what exactly needs to be evaluated by the simulation?” The answer to this question would address the problem of chromosome definition for the laydown optimization problem that this study is going to solve. Figure 3-7 shows an imaginary laydown yard with 9 cells, which is hosting incoming materials with four different batches. There is equipment such as forklifts and loaders to transport materials from laydowns to the point of exit (consumption unit). Assuming an arbitrary arrangement of these four batches in yard cells # 2, 7, 6 and 4, a chromosome whose genes represents cell numbers can be formed. Gene #1 has stored the value 2 which is the number of the cell on which material batch #1 has been stacked. Gene #2 stores the value 7 which represents the cell number on which batch #2 has been placed, and so forth.

This figure also shows that each laydown area (cell) accommodates a specific material type, with a specific quantity. It is possible to use simulation to calculate the time/distance/cost of material haulage from the laydowns to the point of exit for all the batches on the yard, and present the sum of the all the times/distances/costs as the output of the simulation. It is concluded that simulation input could be chromosomes containing placement arrangements, and simulation outputs are time/distance/cost of haulage of the material batches to the consumption unit. Figure 3-7 depicts how chromosomes are defined and how simulation can help transport materials from yard cells to point of exit.

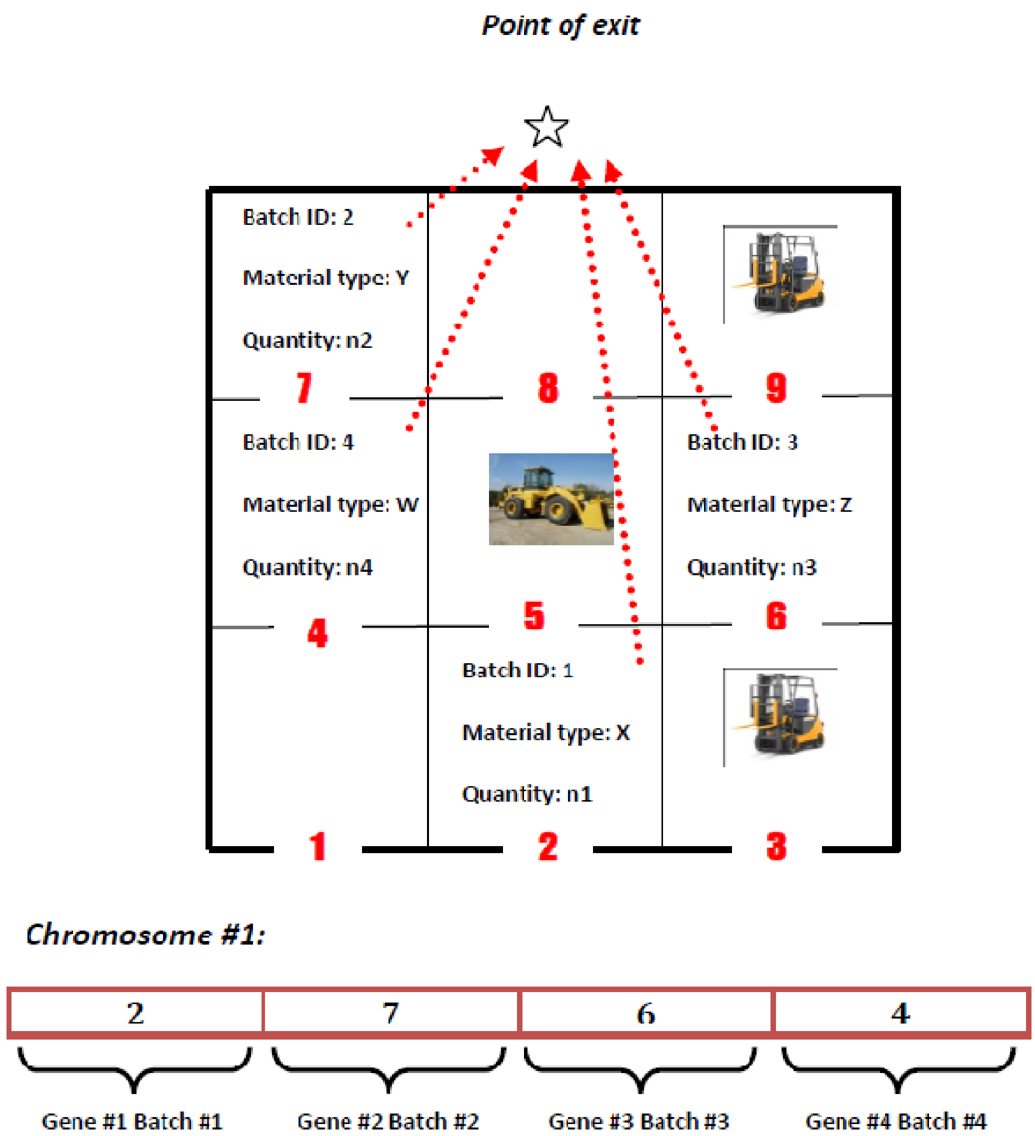


Figure 3-7. Chromosome representation and role of simulation

Figure 3-8 shows how GA and simulation can exchange information continuously throughout the analysis, and Figure 3-9 gives a more detailed version of the flowchart given in Figure 3-6, in which it can be observed how simulation interacts with genetic algorithm.

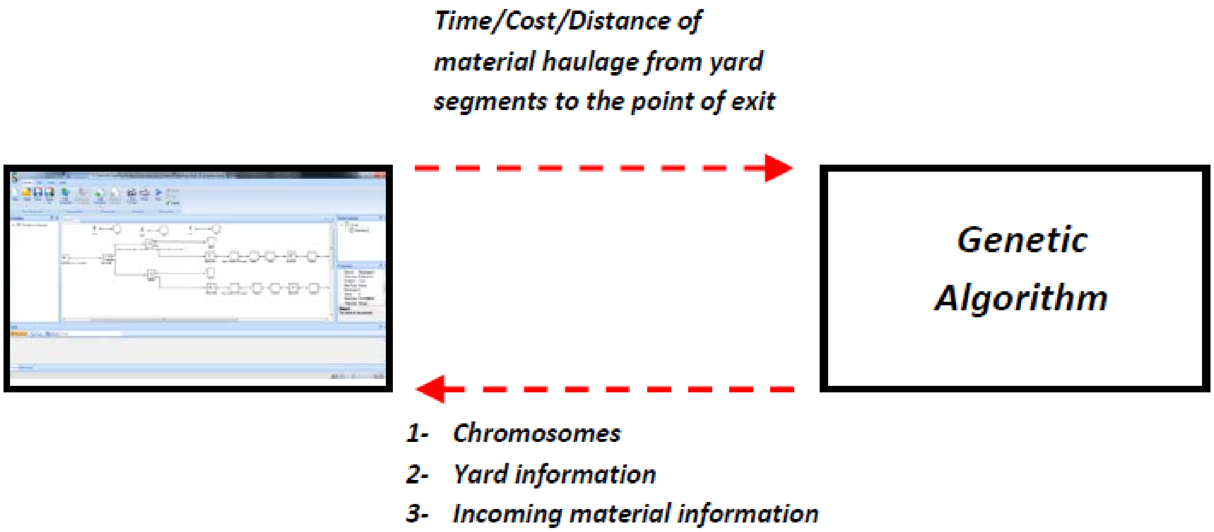


Figure 3-8. Trade-off between simulation and GA

It is shown in Figure 3-8 how continuous information exchange is maintained between simulation and genetic algorithm. In fact, the proposed solution method in this study is not a one-instance integration, but rather it is continuous integration and interaction between two engines (i.e. simulation and genetic algorithm). Genetic algorithm sends chromosomes, yard and incoming material information to simulation, and on the other hand, simulation models the yard and resource conditions and analyzes the material transportation problem, and provides GA with time/cost/distance of material haulage to the point of exit. GA receives this information and uses it as fitness data by which it can evaluate the current population. In other words, simulation in this study plays the role of fitness function in the overall structure of GA.

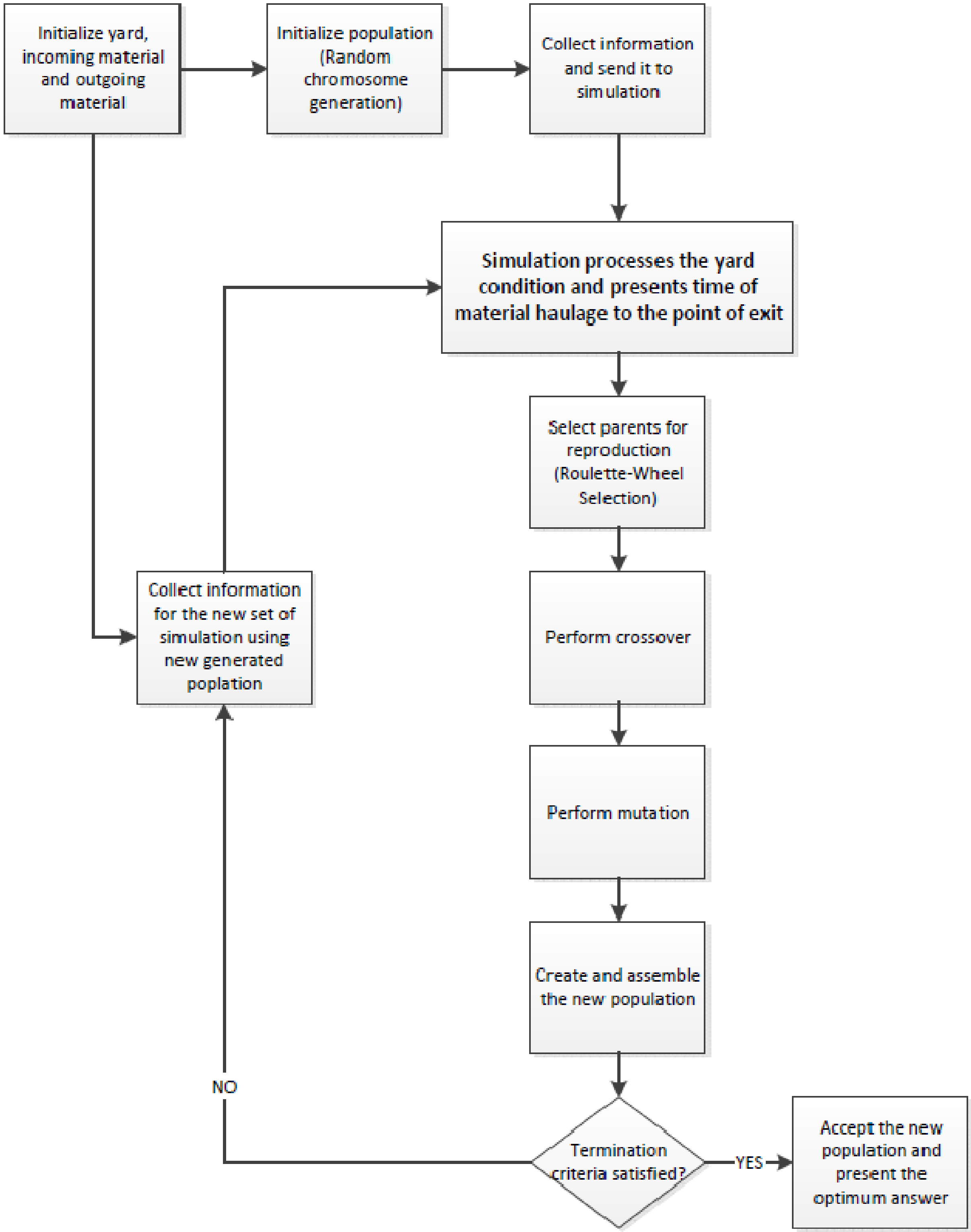


Figure 3-9. Role of simulation in overall genetic algorithm flowchart

Figure 3-9 shows how simulation can help GA evaluate the fitness of a generated population. In this study, the focus is time of material haulage, since cost information cannot be easily acquired, though a separate study for haulage distance determination was also conducted. It should be noted that simulation can effectively process time of material haulage considering resources available for material transportation, whereas distance determination is trivial given the geometry of the yard and simulation may not be necessary for processing haulage distance. In fact, complications such as queue time, waiting time and idleness of equipment (equipment utilization) necessitates and justifies the use of simulation for fitness evaluation of the problem in question. In particular, once the laydown yard is large, containing a multitude of cells and several types of hauling and handling equipment such as forklifts, loaders, gantry and overhead cranes, etc., simulation can readily and sufficiently model the resources, and provides the haulage time and the end of the analysis. Without use of simulation, consideration of the items such as loading/unloading/travel time of equipment, equipment competition over resources (e.g. material and other equipment) and equipment capacity consideration would be very difficult to model.

3.4.OBJECT-ORIENTED PROGRAMMING

Implementation of the procedures explained above requires a robust computer program which is capable of maintaining information flow, data abstraction and realistic simulation of objects. Object-oriented programming (OOD) has been widely used in the past two decades to provide more intuition of real objects for programmers, thus making programming much easier if some particular commonly-accepted rules are observed. OOD combined with .Net framework provides a suitable programming environment in which information flow can be readily maintained between several databases and platforms. The .NET framework is a software framework that runs primarily on Microsoft Windows. It includes a large library and provides language interoperability across several programming languages. As mentioned earlier, the simulation platform used in this study is Simphony.Net which will be integrated with stand-alone object oriented program written to address material handling at hand by using GA.

A starting point for OOD would be to differentiate between several objects of the problem and to develop distinctive classes which characterize the object attributes. It is clear that optimization method has a mathematical basis upon which real objects might be superimposed to address the problem to be solved in real life. Therefore, two separate categories of classes can be distinguished at the highest level, as shown in the class breakdown structure in Figure 3-10. It is seen in this figure that the implementation of GA-related classes can be totally independent of that of material handling-related classes. This will provide more flexibility for the programming as alterations can be made to the material handling classes based on the problem to be studied (i.e. yard, material flow provisions, etc.).

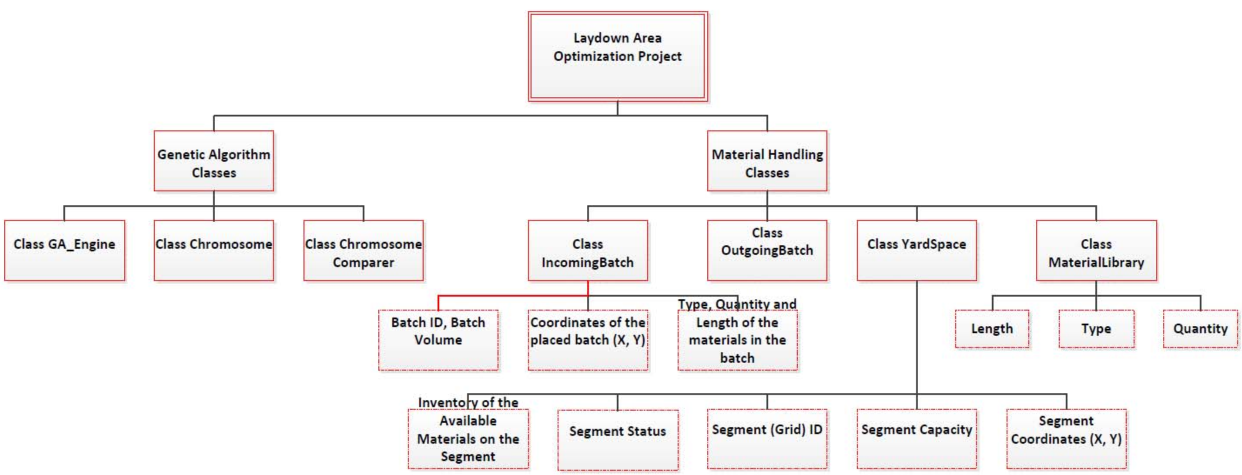


Figure 3-10. Class breakdown structure

Starting from the material handling classes, three totally independent classes, along with one core class can be defined to represent yard activities described in 3.2. In fact, the differentiation shown in Figure 3-10 stems from the totally different processes which occur in a typical construction yard. The first class defined is a class which represents incoming batches to the site. Instantiating this class presents a single batch (e.g. a group of iron angles), having a specific type (e.g. iron angle), quantity and length. An ID will be attached to the batch upon arrival at site, distinguishing it from other batches coming on the same day. Obviously, this batch of material occupies a certain volume and will be placed on a certain cell with a known coordinate, which could be considered a characteristic of the batch. The same attributes (fields and properties) can be defined for the outgoing batch, but it is independent of the incoming material as the consumption unit order might not have anything to do with the receiving materials on the same day.

Another class that can be defined in this context would be the YardCell class which represents each cell of the yard. Instantiating this class to the number of cells in the yard would model the entire yard. The attributes of a yard cell could be cell/segment/grid volume, coordinates, ID and a list of available material (if any) on the segment. Yard ID is readily defined in segmentation but available material inventory on a particular cell would require more attention. If another independent class models the material itself, instances of such a class can be easily stored as private variables/fields/properties of YardCell class. As a consequence, class MaterialLibrary is defined to explicitly model materials having attributes such as quantity, length and type. It should be noted that this program has been written to suit steel fabrication laydown yards. Steel pieces coming to such a yard have the abovementioned attributes. However, it should be mentioned that since the class implementation is based OOD design, and is quite independent of GA implementation, it would be very simple to change the material attributes to suit the need of any special kind of industry in construction.

GA_related classes are composed of three inter-related classes which implement the entire genetic algorithm regardless of what chromosomes would represent. Class GA_Engine triggers the chromosome generation and maintains the information flow between material handling classes and GA_related classes, whereas class Chromosome creates the genes and chromosomes, and ensures their validity and conformity to reality (e.g. yard constraint). This will be further explained later as this conformity check could be considered the core of the program. Class ChromosomeComparer assists class GA_Engine in comparing the fitness of the chromosome and, in particular, in the process of Roulette-Wheel selection where the fittest chromosomes are given the higher probability to be selected for further generations.

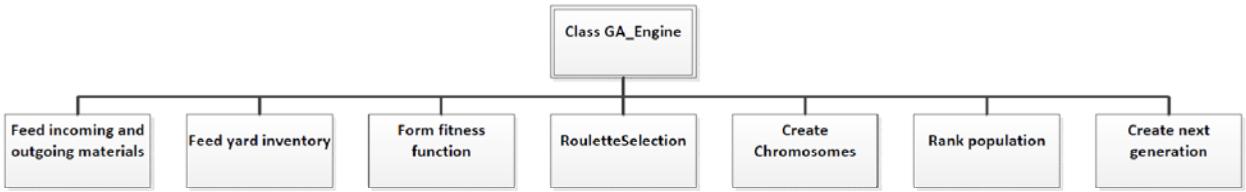


Figure 3-11. Main internal functions of class GA_Engine

Figure 3-11 illustrates the internal functions of class GA_Engine. It is observed in this figure that this class feeds the incoming and outgoing materials as instances of their corresponding classes and generates current population, evaluates their fitness by using fitness function, selects them based on their fitness for further generations through ranking, and manages the termination criteria. It is in this class that simulation is integrated with GA through .Net framework, and interacts with GA, as long as the engine is running. In fact, the function/method FitnessFunction in this class invokes the simulation engine and processes and evaluates the chromosome fitness as many times as required by the GA engine. The nature of this interaction is discussed in 3.6. Code below shows the implementation of fitness function within GA engine:

```
for (int i = 0; i < PopulationSize; i++)
            {
                Chromosome g = ((Chromosome)_thisGeneration[i]);

                g.Fitness = this.FitnessFunction(g.OutGoing,this.YardInventory,i);

                _totalFitness += g.Fitness;

                // Printing the chromosomes and genes
                tw.WriteLine("\nFitness for this chromosome # {0} is: {1}\n",i+1,
(1 / g.Fitness) * (1 / 3600.0));
                for (int j = 0; j < 71; j++)
                    tw.WriteLine("{0}", g.Genes[j]);


                // Printing the outgoing segments
                tw2.WriteLine("\nFitness for this chromosome # {0} is: {1}\n", i +
1, (1 / g.Fitness) * (1 / 3600.0));
                foreach (var d in g.OutGoing)
                    tw2.WriteLine("{0}", d.segmentTag);
            }
```

Figure 3-12 shows class chromosome breakdown structure. Public GA-related functions/methods of this class perform mutation and crossover operators on parents to generate offspring as discussed in the previous section. However, the generated offspring shall comply with yard constraints, that is, as shown in Figure 3-14, any placement arrangement should go through three separate filters representing yard hard constraints. These constraints are as follows:

- Some yard cells are reserved for some special jobs. It is not possible to place any material on these cells.
- Placing material on a cell which contains some material inconsistent with the incoming material is not allowed. For instance, it is not possible to place iron angle on a cell which contains channels. Nonetheless, if the type is consistent and section size differs, the placing is allowed. That is, if placing is to be carried out for iron angle 6x4x3/8 inch, and the yard cell contains two iron angles with two different sizes, the placing is allowed.
- Cell capacity shall be checked. It is not possible to stack material on a cell beyond its capacity. Capacity definition of a cell will be discussed in 3.7.

The generated offspring are randomly created from the parents, but it should be noted that the problem at hand does not span a continuous space. That is, the domain of placement alternatives is discrete depending on the yard constraints discussed above. Thereby, the function PlacementVerifier ensures the compliance of the generated offspring with abovementioned constraints. To that end, two different approaches could be followed:

- To verify whether offspring belong to a range of allowed arrangements. If not, redo the mutation or crossover. This method is quick and effective, but depending on the size of the chromosomes (i.e. number of the genes), none of the combinations might satisfy the hard constraints of the yard. This problem is more pronounced under the following circumstances:

- Single-point crossover, where there are few combinations that can be generated.
- Small number of genes that can be led by small number of incoming batches.
- Very congested yards, where possible alternative arrangements are few.

- To verify whether offspring belong to a range of allowed arrangements. If not, create two random parents again instead and add them to the new generation. This method does not have the limitation presented above, but it does pollute the new generation with possible unfit parents. The direct consequence of this possible contamination would be slower convergence rate.

In assessment of these two alternatives, it was decided to take the second approach, as validity and compliance of the populations and candidate arrangements are of more importance to the present study than rate of convergence.

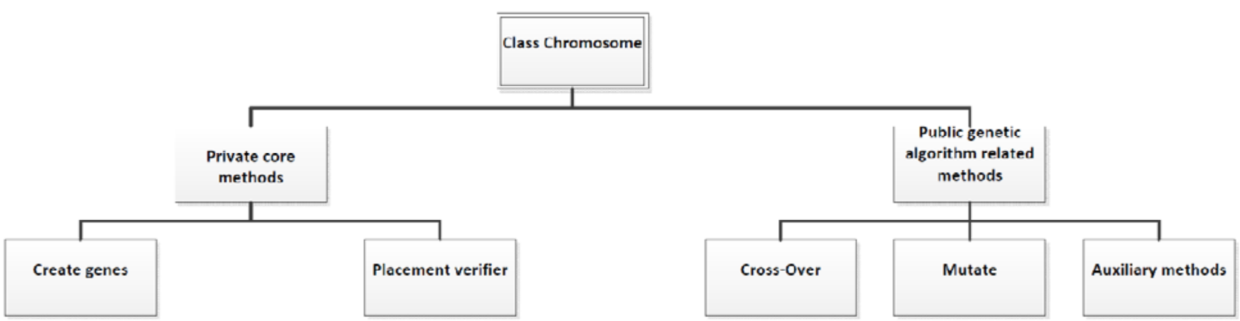


Figure 3-12. Class Chromosome breakdown structure

Figure 3-13 illustrates the overall algorithm of placement which has been implemented in function CreateGenes in the program except for reading information from the database, which is implemented elsewhere. It is seen that a key check is made once placement trial is made. If it is possible to stack the material on the cell, inventory is updated in two stages. The placement material is added to the yard cell inventory first, and then it is checked whether consumption units schedule taking off any material from the cell or not. If yes, the cell inventory is updated once again to provide the accurate inventory for further analyses.

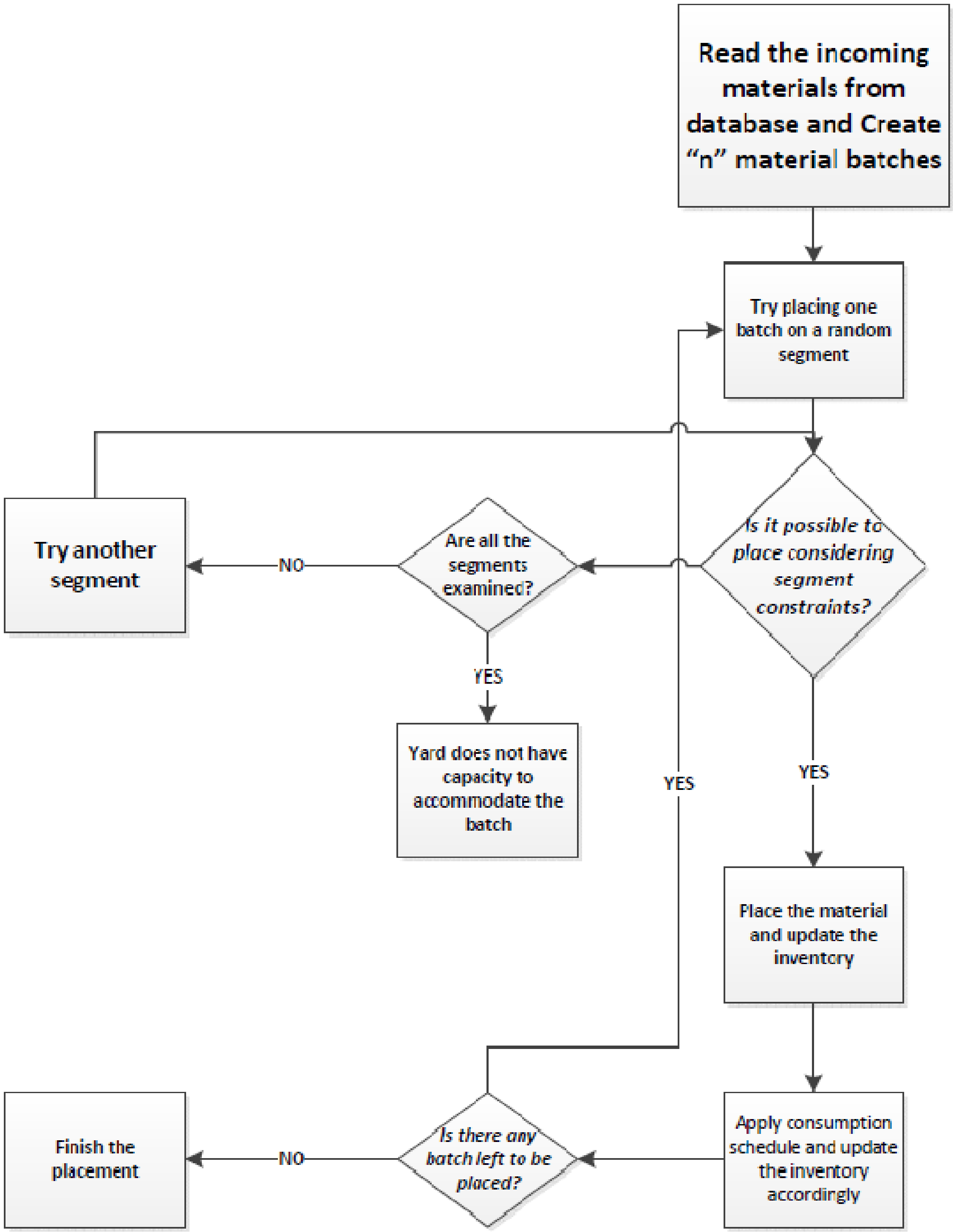


Figure 3-13. Overall algorithm of placement (CreateGenes function)

If there is not any space, however, the program prompts a message to the user that there is no space in the yard for this particular material. This alerts the receiver to plan its placement policy in advance to account for further placement provisions. The program loops through all incoming batches until placement has been fully carried out. The whole operations mentioned above are carried out for all the chromosomes that need to be generated. Assuming 2000 generations with 100 being population size, the algorithm is executed 2000X100 times to produce the optimum solution at the end.

It is seen in Figure 3-13 that during the placement procedure, it is verified whether or not the placement arrangement satisfies the yard constraints. This satisfaction check requires a separate algorithm which is shown in Figure 3-14. It is seen in this figure that three primary checks, that were explained earlier, are done in this algorithm to accept a placement arrangement. If any of these constraints is not satisfied, the program reverts to the first step and rejects the placement. Figure 3-13 and 3-14 combined characterize the operations that are done in CreateGenes and PlacementVerifier functions. The difference between these two functions is that one of them produces the chromosomes based on the checks, while the other solely verifies the compliance of the chromosomes that are sent to the function by Mutate and Crossover functions. By using these two functions, the program ensures that all offspring belong to the domain of possible solutions. It should be noted that inventory update is not done in the program until the final optimum solution has been found. That is, once termination criterion has been satisfied, the inventory update will be done based on incoming and outgoing material.

As for outgoing schedule, a simple algorithm has been implemented in the program that finds the closest material to the point of exit and takes the material off those cells that can supply the consumption unit faster in terms of distance. This is consistent with real-life practice as the receiver solely checks where s/he can take the material s/he needs based on the proximity of the material. Once the closest spots containing the material have been found, this information is sent to the final building block of the program to update the inventory based on consumption. The algorithm written for consumption reads the consumption bill of material and strives to find the closest material to the point of exit for all the listed material. If two cells have the same distance to the exit, the program randomly selects one of the two.

It should be reiterated that the entire discussion given above is totally consistent with the common material handling practices which are actually carried out in steel fabrication yards. It is seen in the next sections how a real-life laydown management problem can be modeled with the aid of the proposed solution described above.

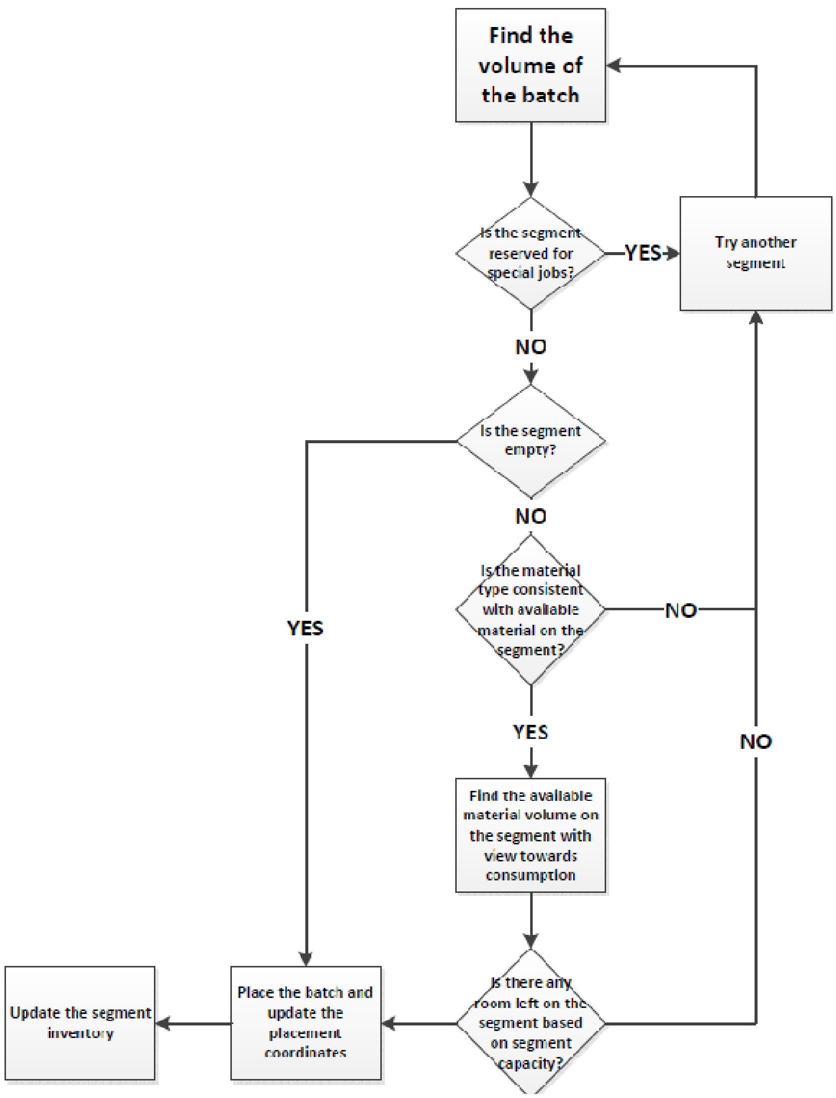


Figure 3-14. An algorithm to satisfy yard hard constraints

3.5.EXAMPLES

After program implementation, a practical example is modeled and solved in this section to show the suitability of the development to optimize material placement arrangement considering yard constraints. It should be noted that during program implementation, it was decided to evaluate one other fitness function besides simulation. This decision was made in light of independence of fitness function from the main GA engine. In fact, the objected-oriented program has been implemented in a way that the fitness function can be independently assigned to any kind of placement policy that the project manager of yard receiver might choose. One possible target/fitness function that might be considered as an optimal placement situation could be the proximity function towards the point of exit. That is, the closer that the incoming batch could be placed to the consumption unit, the better it is. The word closer might mean the closer the receiver might perceive the stocked material to be towards the exit. Often times, this closeness is measured visually by the receiver. Mathematically, however, the distance between placed material and point of exit can be measured by Pythagoras formula. This does not mean that what is thought to be closer distance by the receiver is based on the formula, but rather the formula is a simulation of the receiver perception of distance which is more accurate and based on Euclidian distance between the cell containing the material and the point of exit.

The framework of the program has been established first irrespective of the nature of the fitness function. The fitness function was set to a simple function that identifies the placed material on the yard and calculates the Euclidian distance between the cell containing the material and the consumption unit. Comparing the results of the solution based on a development with the solution based on simulation as fitness function could highlight the significance of using simulation and its interaction with GA as the solution proceeds.

In this section, a sample material handling process in the steel fabrication industry has been modeled using the developed program and simulation. Figure 3-15 illustrates the stock shop yard of the fabrication yard having 20 segments divided by two separate south and north yards. Two overhead cranes span the south and the north yards and haul material from the yard cells to the point where a car and rail system transports the material to the point of exit, as shown in the figure. Crane and car travelling speeds as well as loading and unloading times are given in Table 3-1. The same data that are given in the table has been used in the simulation model of the material haulage that will be explained shortly. It is seen in Figure 3-15 that tow cells have been reserved for special jobs, and no material can be stocked in these laydown areas. It is also seen that the cells are numbered consecutively to facilitate the modeling process. A coordinate system can be assigned to the yard to represent its position with respect to the point of exit. This coordinate system will be used frequently in the program to determine the distances from the cells to the rail-car system and the car to the exit point. The work flow for the material handling is the accommodation of the incoming material by the receiver and the haulage of them to the car, which subsequently carries the steel pieces to the fabrication shop.

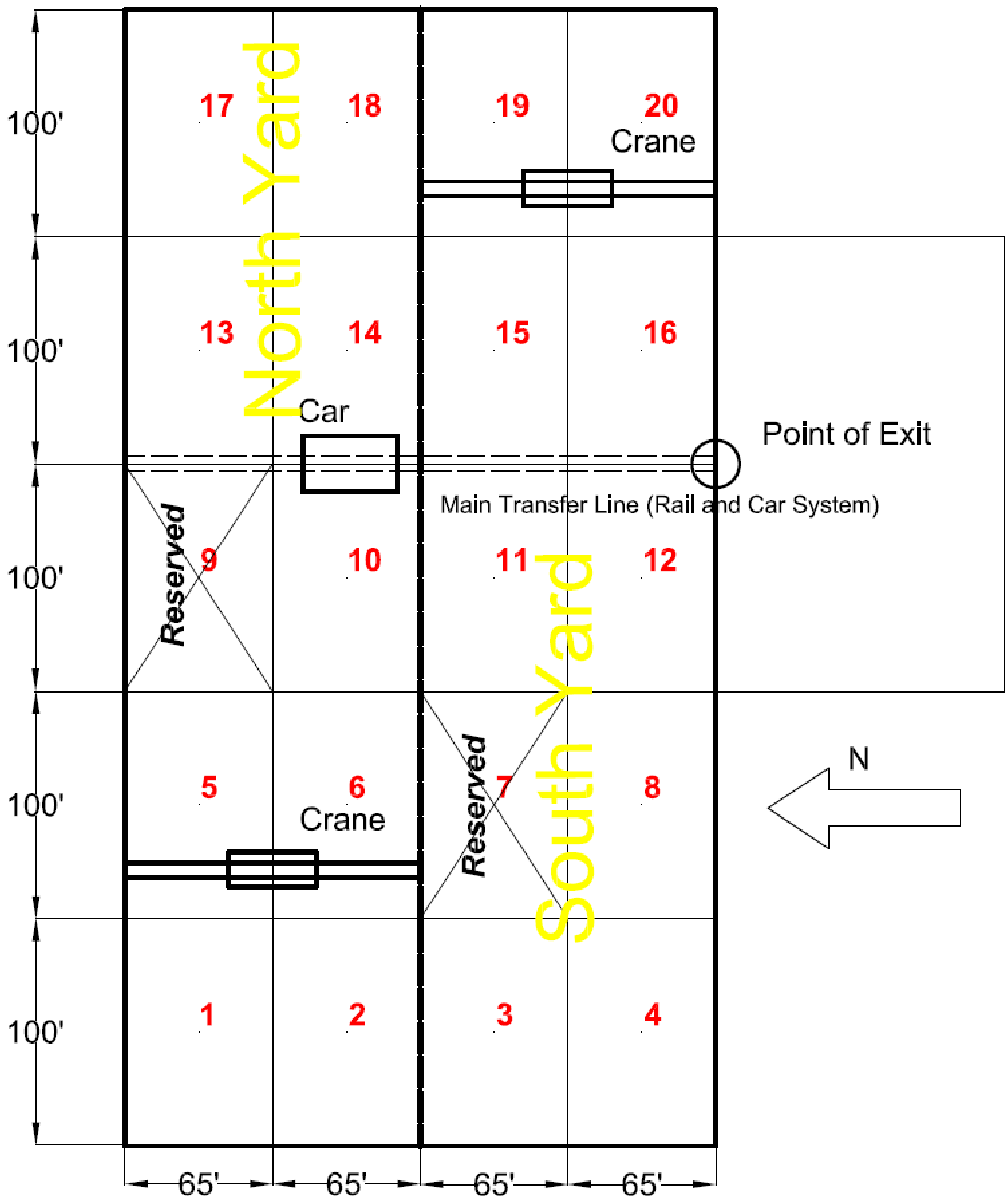


Figure 3-15. A sample construction yard

It should be noted that the existence of one car which serves to overhead cranes poses a challenge to the receiver if s/he wants to utilize the cranes productively. That is, the cranes and the car should work in some form of harmony where cranes do not wait in a queue waiting to be served by the car as it poses safety issues for hoisted load that should not be hanging over the workers on the yard. If the workload is heavy, though, this might be inevitable due to unavailability of the car for the cranes. The car-cranes interaction and its impact along with other material handling issues will be discussed further in the next section.

Table 3-2 gives information on a sample of incoming materials to the yard. The selected incoming materials are taken off an actual purchase order to a steel production facility for the fabrication job. It is understood that some other section types are also ordered in a typical steel fabrication job such as round bars and flat bars, but due to the nature of such steel types, it is decided to account for the four most-commonly used section types, usually circulated on a shop yard.

Table 3-1. Loading and unloading times and traveling speed of the cranes and of the car

Crane Capacity	15 tons
Crane Speed	5 Km/h
Crane Loading Time	20 s
Crane Unloading Time	20 s
Crane Travelling Speed	4 Km/h
Car Travelling Speed	4 Km/h
Car Unloading Time	200 s

Table 3-3 shows a sample daily consumption schedule and bill of materials that are requested by the fabrication shop. It should be noted that shop bill of materials could be totally independent of incoming materials on the same day. In fact, except for a very few cases where rush jobs require the availability of some materials for the rush production, the incoming and outgoing materials are independent from one another.

Table 3-2. Sample incoming materials to the yard

ID	Type	Quantity	Length
1	L6x4x3/8	5	60
2	L6x6x3/8	20	50
3	L8x8x1/8	15	60
4	C10x15.3	200	60
5	C8x13.75	300	40
6	W8x24	50	60
7	W10x30	50	60
8	W14x43	50	35
9	PL3/8	10	8
10	PL1/2	15	8

A sample yard inventory was also created and corresponding steel pieces were placed on the yard. The inventory deliberately suggests a sparsely occupied yard. The reason for creation of such a free yard is to demonstrate the efficiency of the proposed solution to explore a multitude of arrangement options. The improved placement in consecutive runs of the computer program can be exhibited more conveniently with the aid of a less congested yard as material flow throughout the site can be visually shown for instructional purposes.

Table 3-3. Sample outgoing materials to the fabrication shop

ID	Type	Quantity	Length
1	L6x4x3/8	10	60
2	C10x15.3	300	60
3	C8x13.75	450	40
4	W8x24	10	60
5	W10x30	10	60
6	W14x43	10	35
7	PL3/8	10	8
8	PL1/2	15	8
9	PL1	5	8

In Table 3-4 quantity and type of materials are shown. The quantities are selected in such a way that some cells will reach their maximum capacities once excessive placement is imposed upon them. As for the capacity of the cells, an ad-hoc capacity determination has been adopted on the basis of interviews conducted with experienced yard foremen in steel fabrication companies. The rule of thumb is not to stack steel pieces (e.g. iron angles, W-sections, channels, plates) more than 2 meters, as safety regulations would not allow further material stacking, assuming a neatly-arranged stack. To determine the maximum stacking volume, one may need to account for maximums in three perpendicular directions. Two-meter stacking threshold would impose a height limit whereas cell's width and length will constrain the areal placement. There were two approaches to determine the cell capacity:

- Assignment of a universal capacity to similar cells, as the chosen steel shapes roughly occupy a similar volume once they are stacked. This method suffers inaccuracy, but has the benefit of simplicity and reduction of run-time.
- Assignment of capacity based on the section type. An algorithm could be developed to identify the to-be-placed material on the cell and automatically assign the corresponding capacity to the cell. Furthermore, the algorithm should account for the filled segments or the partially filled ones and assign the capacities with respect to the available inventory. For instance, if a cell contains W10x30, and a W section wants to be placed next to it, the current volume should be subtracted from the capacity of the empty cell to present the remaining capacity. The method brings about accuracy, but it furthers implementation complications.

It should be noted that the 2-meter height restriction is by nature an approximate method. Depending on the nature of placement, voids and gaps could be imparted inadvertently into the yard placement, which totally undermines the accurate capacity determination calculations. Thus, it was decided to follow the first capacity assignment approach for this stage of modeling efforts to maintain simplicity.

Table 3-4. Sample yard inventory

ID	Quantity x (Material)	ID	Quantity x (Material)
1	215x(L8x8x1/8)	11	Empty
2	Empty	12	102x(W8x24)+400x(W10x30)+400x(W14x43)
3	Empty	13	350x(C10x15.3)+500x(C8x13.75)+500x(C15x50)
4	170x(W8x24)	14	Empty
5	Empty	15	Empty
6	Empty	16	300x(W8x24)+158x(W10x30)+500x(W14x43)
7	Reserved	17	88x(PL3/8)+30x(PL1)+20x(PL1/2)
8	Empty	18	100x(PL3/8)+20x(PL1)+12x(PL1/2)
9	Reserved	19	33x(PL3/8)+50x(PL1)+55x(PL1/2)
10	Empty	20	Empty

The simulation model works as fitness function as mentioned earlier. Figure 3-16 gives a glance at the entire simulation model that is used to evaluate the population. As shown, three resources have been incorporated into the model, simulating two cranes and one car. Towards the end of the model, two cranes compete for capturing the car resource. The resource attributes (i.e. loading and unloading times, traveling speed) given in 3-1 can be easily changed so as to examine the effect of crane/car travelling speed and loading/unloading times on the performance of the system.

The simulation model has its own way of dealing with cell coordinates, which is totally different from that of the distance evaluator fitness function. While the fitness function is based on determination of Euclidian distance to the exit point, the cell coordinates (where the material lies) are explicitly used in calculation of distance, whereas in simulation, the cell coordinates are used to calculate the parameters such as crane and car travelling times. Having the speed of such equipment, it is possible to determine the orthogonal distances from the cells to the rail system and/or from the car position to the point of exit. By using the speed formula subsequently, travelling times can be obtained.

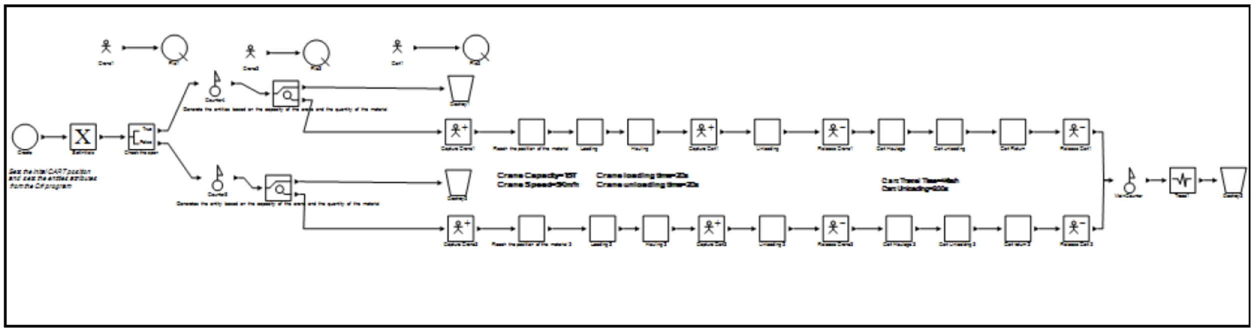


Figure 3-16. A glance at the simulation model of material handling process

Figure 3-17 exhibits a more detailed view of a part of the simulation model where created entities, which represent materials, are sent either to south or north yards on the basis of their locations. A conditional element takes care of the job through its formula evaluator. Moreover, a generate element is utilized to create the required volume of material based on the incoming entities' attribute. One of the coming attributes is volume of the placed material. The other attributes, as mentioned above, are coordinates of the cell which nests the material.

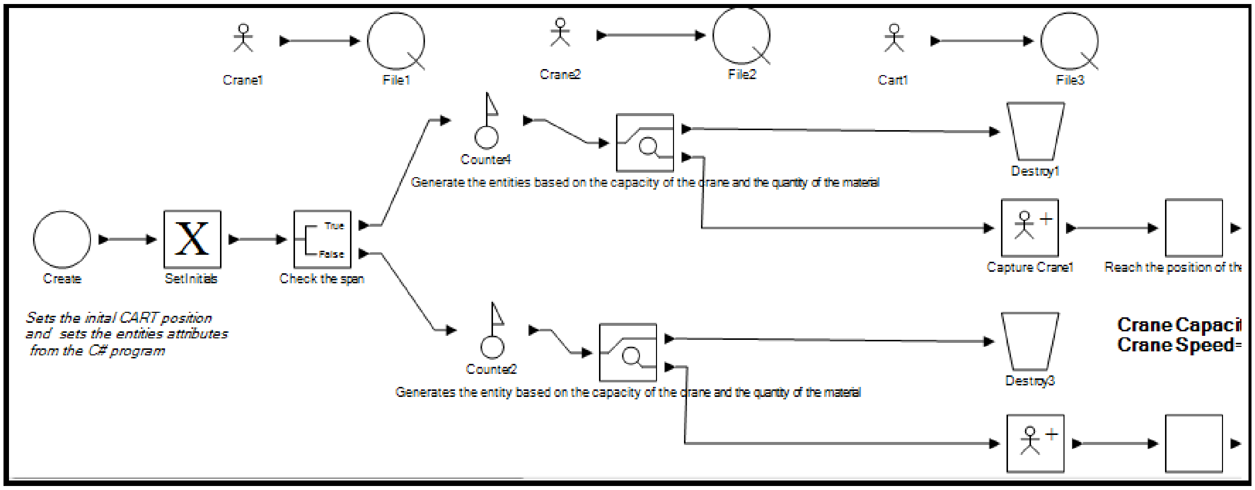


Figure 3-17. Simulation of material division between south and north yard and of crane capacity

The simulation model can be accessed through .Net capabilities and run as many times as required inside the C# program, which is the platform for the material handling solution developed in this study. The model is used as an external source and run to the number of population size multiplied by generation size as all the generated chromosomes shall be evaluated for their fitness.

The current version of program development is written in a console format. After the analysis has been completed, the program presents the best placement arrangement and its corresponding haulage time/distance. However, many other outputs can be requested such as placement arrangement enhancements as GA progresses and their corresponding time/distance to pronounce the competence of the proposed solution in improving the laydown design. One problem that arose during the analysis was the long run-time of the program once the fitness function is set to be simulation. The computer program uses simulation as an external source and runs the simulation file to the number of generations multiplied by the population size. The huge number of times of calling the simulation file inherently incurs a very long run-time (approximately 6 hours for all the generations to be completed).

However, the simulation model in this study has a limitation of being deterministic. That is, the loading/unloading time of the cranes and the carts have not been inserted into the program deterministically. For the simulation model to be stochastic, more field observations needed to be conducted. Moreover, combination of stochastic simulation analysis and optimization using evolutionary methods would further complicate the problem conceptually. Code below shows how simulation is initialized within the C# program within the FitnessFunction:

```
public double FitnessFunction(List<OutgoingBatch> Outgoing, List<YardSpace>
yardCells, int i)
        {

            //////////////////////////////////////////////////
            foreach (var m in Outgoing)
                this.OptimizedLoacations.Add(m.segmentTag);


            var model = new Model();

            using (var stream = new FileStream(@"C:\Users\AA\Desktop\CEM MSc
thesis\Chapter4-Pejman C# program\Model1.sim", FileMode.Open))
            {
                model.Deserialize(stream);
            }

            var scenario = model.Scenarios[0];

            var create = scenario.GetElement<Create>("Create");
            var counter = scenario.GetElement<Counter>("MainCounter");

            var executeElement = scenario.GetElement<Execute>("SetInitials");


            create.Quantity = OptimizedLoacations.Count;
            executeElement.Expression.Function = InitializeEntities;

            model.Simulate();

            ///////////////////////////////////////////////////////
            double time = counter.Time;
            time = 1 / (time + 0.001);
            Console.WriteLine("the transfer time is: {0} hours", (1 / time) * 1 /
3600);
            this.OptimizedLoacations.Clear();
            return time;

        }
```

3.6.DISCUSSIONS AND RESULTS

As indicated earlier, the yard has been deliberately chosen to have several empty spots to demonstrate the suitability of the developed solution to enhance the arrangement as the GA solution progresses. It should be noted that this enhancement is directly affected by the choice of the fitness function within the algorithm. Figure 3-18, illustrates 9 of the best arrangements out of 2000 generations. Table 3-5 gives the GA parameters that were set within the program to obtain the optimum results.

Table 3-5. GA internal parameters

Parameter name	Parameter value
Crossover probability	80%
Mutation rate	5%
Population size	100
Number of generations	2000
Number of genes in a chromosome	10

The 9 arrangements given in Figure 3-18 are randomly selected during the run-time and portrayed in this figure to examine the trend of improvement. The results shown in Figure 3-18 are obtained from the solution in which fitness function was set to be distance evaluator. It is seen how batches tend to approach towards point of exit gradually as the solution progresses. It should be noted that all the generated arrangements satisfy the hard constraints of the laydown areas. The reader is encouraged to verify this with the aid of Table 3-2. Cranes and cars in this series of analysis, have no impact on the results as they have no role in closeness fitness function in which Euclidian fitness function merely evaluates the distance between the stocked materials and exit point. In fact, often times, this is the approach that the material receiver and/or yard foreman applies to estimate the proximity of the placed batches to the consumption unit. Needless to say that the receiver usually does not evaluate the exact Euclidian distance, but the decision s/he makes visually is based on the direct distance to the exit point.

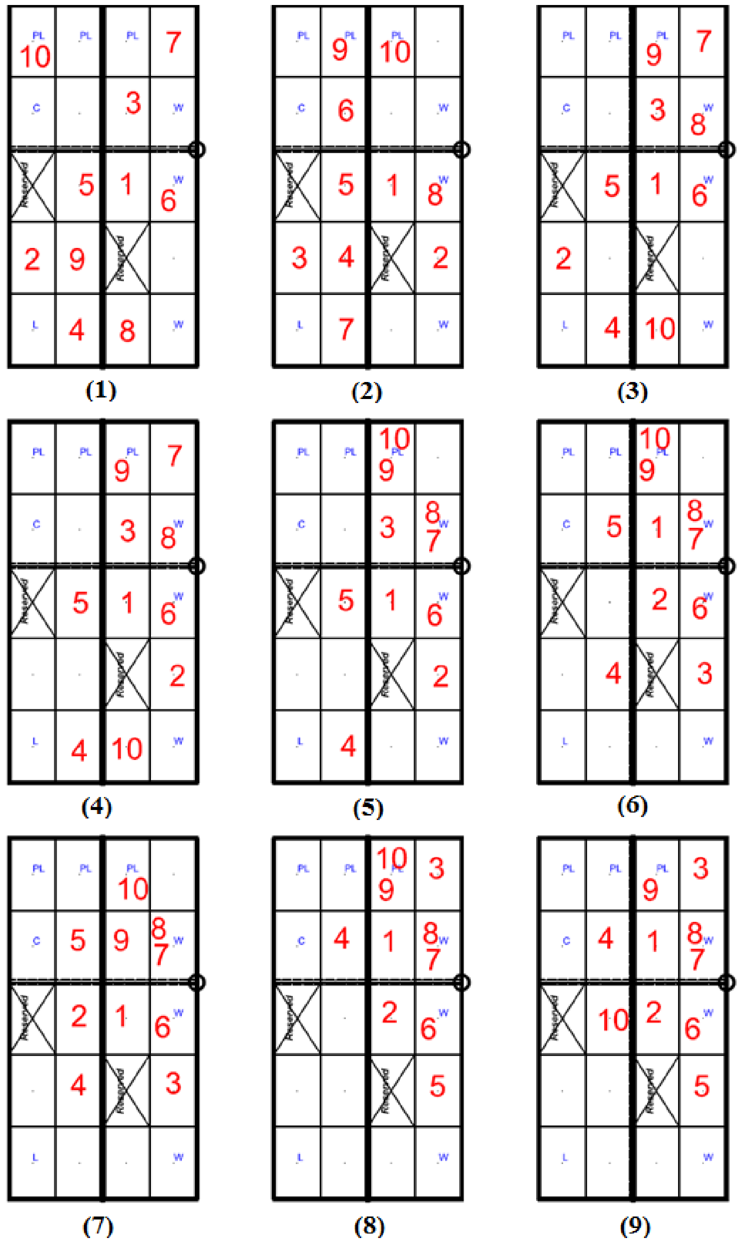


Figure 3-18. Trend of placement arrangement improvement as the GA progresses based on closest distance to the exit point

Figure 3-19 shows how much the summation of all the distance from all the placed batches to the exit point is reduced in several runs. It is observed that the summation of the distances is initially 594.36 meters. As the solution moves forward, the sum of all distances is lowered to 373.43 meters. 221 m improvement proves the competence of the solution in optimizing the results. After 50 generations, there is no significant reduction in distances proving that the arrangement given in generation number 50 could be selected as the optimum arrangement. This explains the termination criterions that have been used in this study. Two alternatives might be chosen for termination criterion:

- Check the last two best arrangements, and subtract their corresponding sum of distances. If this value is less than a small value, the generation process shall come to a halt.
- Run the solution for a certain generation size and track the progress visually. If no significant decrease in sum of distances is observed, the repeated distance value is considered to be the optimum distance, and its corresponding arrangement is the optimum arrangement.

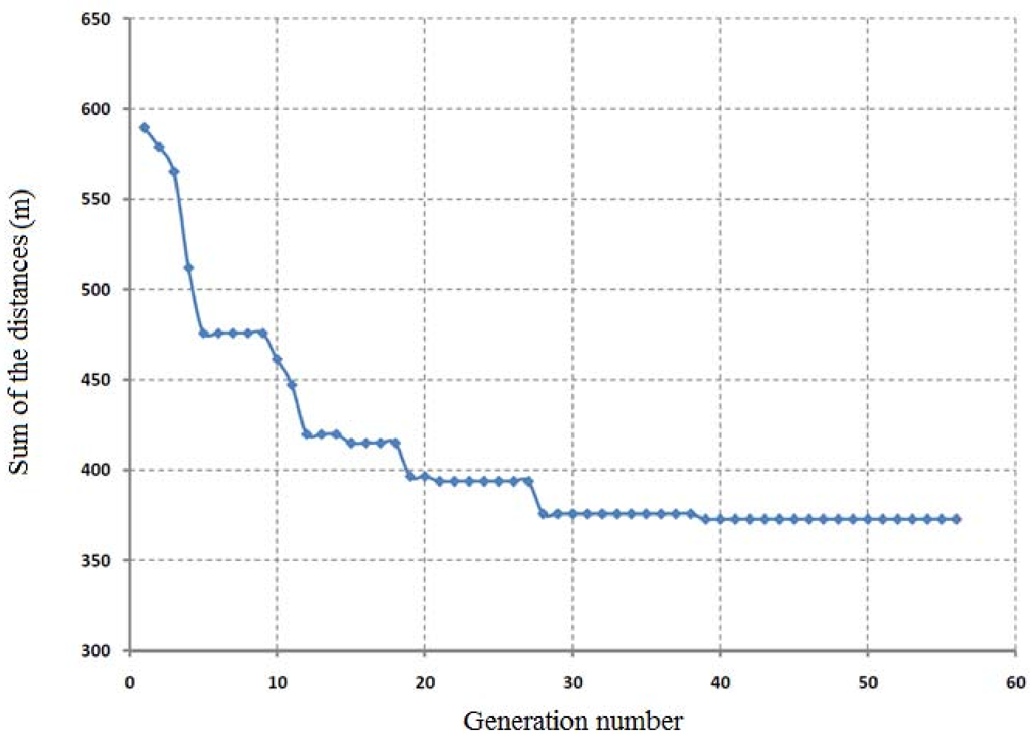


Figure 3-19. Optimization of distance as GA progresses

The first alternative seems more sophisticated in terms of automation of the solution but there is one major complication attached to it. Often times, GA is stuck in some local optima which might seem to be the final answer to the problem at hand. In such cases, mutation in GA will assist the GA to jump to another trend towards which absolute maxima or minima can be achieved. As a consequence, the first alternative might be tricked by such local optima as it only checks the last two solutions, whereas the second alternative suffers from inefficiency but the user observes multiple consecutive runs and selects the best fitness value more robustly. As such, the second approach has been adopted for the current development.

One can't confidently rely on the optimum results unless s/he runs the GA engine several times to ensure the reliability of the GA performance. Figure 3-20 shows 7 series of runs, each of which contains 50 generations. As can be seen, four of these seven runs converge to the same value, but with a different rate of convergence. This is natural as GA strives to find the optimum answer through mutation and crossover operators. One of the solutions cannot find the optimum answer in 50 runs and falls into a local minimum and never escapes. The situation exacerbates in run numbers 5 and 6 which fall within the local minimum which is considerably greater than real minimum. The root causes behind such mistakes could be the following:

- First and foremost, the two private functions "Create Genes" and "Placement Verifier" totally modify the output of crossover and mutation operators. Knowing there are probabilities attached to such operators, it is not guaranteed that new offspring are generated every time that Placement Verifier rejects the output of mutation operator. The program simply does not mutate the parents if mutation is rejected by Placement Verifier. Since mutation is one of the key operators that help the GA engine escape from local optima, such an obstacle in front of the mutation operator might cause the GA engine to fall into a local minimum trap.
- Roulette-wheel selection is not the best selection method. Other enhanced selection method in GA could be utilized.
- Mutation and crossover probabilities should be calibrated. Perhaps some different values could be applied to escape local optima.

It is understood that working on such root causes might help the performance of the proposed solution, but this is left for future efforts of this study.

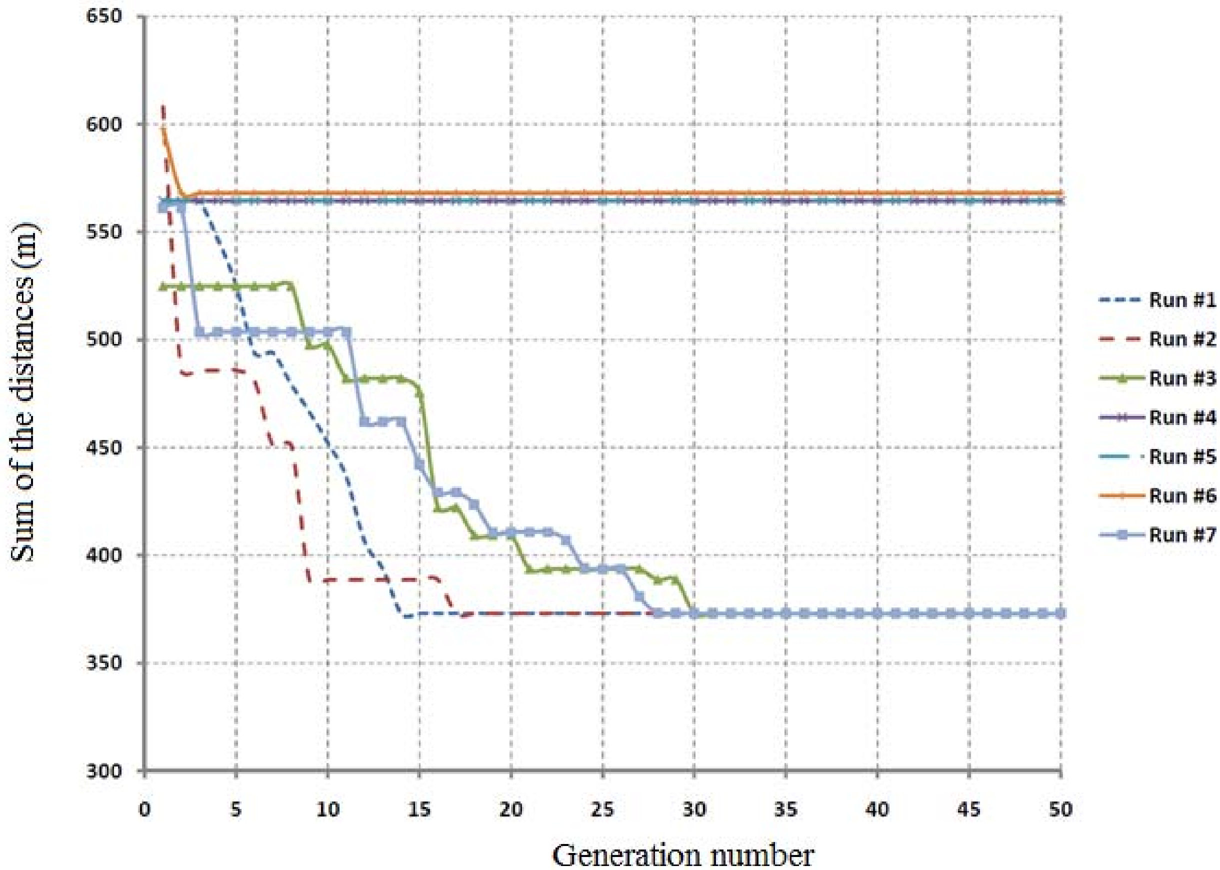


Figure 3-20. Reduction in distance in multiple runs

The same series of post-processing can be carried out once the fitness function is set to use simulation, as discussed before. Material transfer time from cells to the consumption unit is the output of simulation. Similar to Figure 3-18, the trend of improvement in placement can be illustrated in Figure 3-21 in 9 arrangements as the GA engines progress towards better solutions.

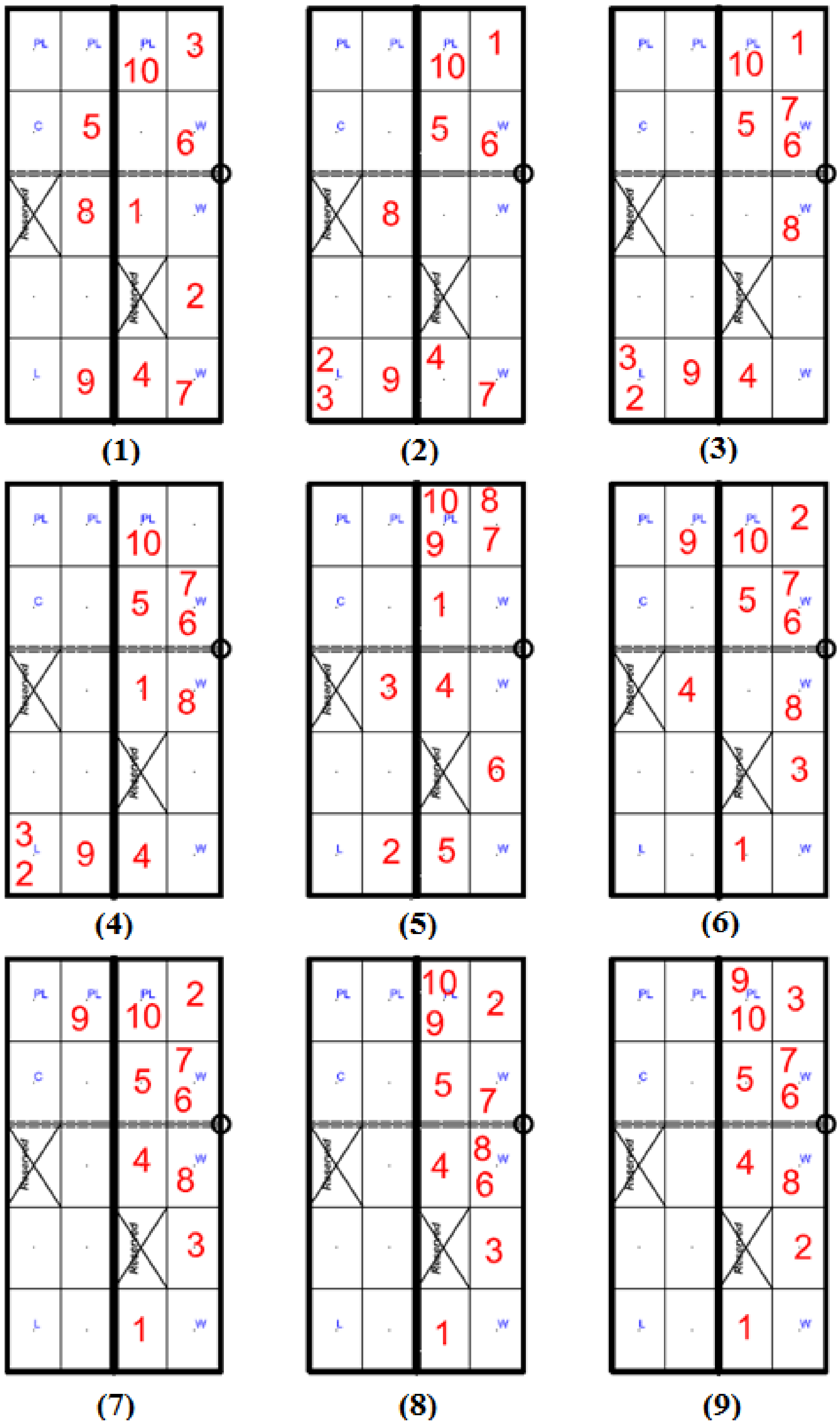


Figure 3-21. Trend of placement arrangement improvement as the GA progresses based on simulation

Figure 3-22 shows the optimization of haulage time as GA engine progresses. 31 minutes reduction in haulage time can be achieved through optimization for the sample material handling problem. This reduction time accounts for 19% of the entire haulage time (the fittest chromosome in initial generations) of the incoming batch on a regular working day. More improvement can be achieved depending on the material handling problem to be addressed. For this particular sample placement problem, the following should be taken into account:

- The presented time by simulation is the summation of the times that the equipment resources take to transfer the material batches from the cells to the exit point. Given the 15 T capacity of the crane and the size of each individual batch given in Table 3-2, the crane can simply haul most of the batches in one instance, which is not what actually occurs in practice. For safety reasons, it is observed that cranes hoist loads significantly lower than their nominal capacities. Safety regulations differ from one material type to the other. For steel pieces, since they come in lengths ranging from almost 10 to 20 meters, it would be dangerous to haul 15 T of steel in one instance, since they might easily swing and deflect under their own weight while suspended by the crane hook. Safety considerations and quantifications are beyond the scope of this study.
- The selected yard is considerably small in comparison with gigantic construction yards. In general, steel fabrication yards are smaller than other types of construction stock yards. As the yards expands in size, travelling time of the equipment also increases. The furthest point to the rail system is less than 100 meters away. Given the 5 Km/h travelling speed of the cranes and the car, the haulage of such material in one instance would take less than 2 minutes.
- There are few equipment resources working on this yard dealing with materials. As the hauling equipment crews increase, the handling time will also be impacted.

- The material handling time that has been processed in this study only deals with one day shop supply work. As the number of days increases, the corresponding haulage time also accumulates, and the impact will be more tangible.

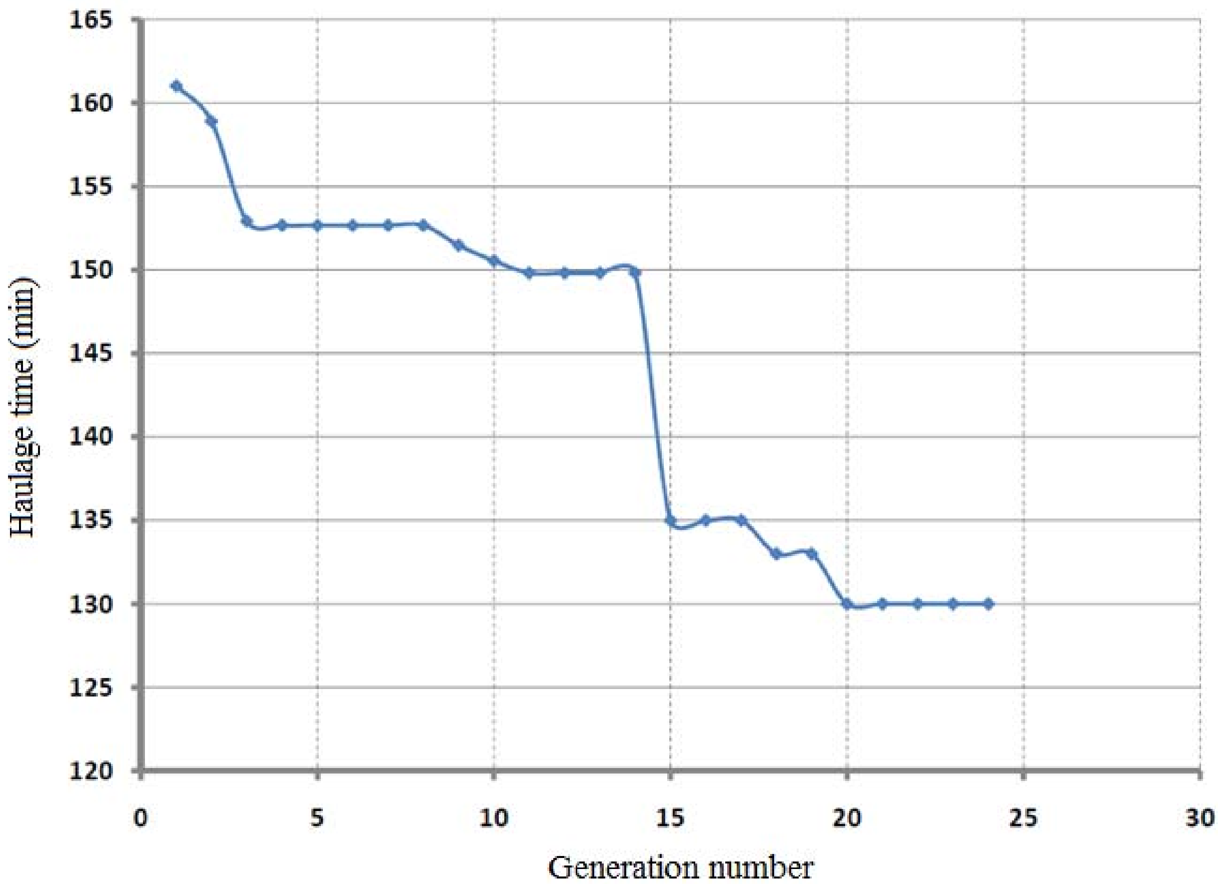


Figure 3-22. Optimization of haulage time as GA progresses

Figure 3-23 illustrates the results of multiple runs of GA that converge to 130 mins at the end. As discussed before, some of the runs fall into local optimum trap. The reason for running the program several times is to identify these local optima and to ensure the robustness of the proposed optimum arrangements. An approximate 20% improvement can be seen in most of the runs comparing the final optimum arrangement and the best arrangement proposed by the first generation.

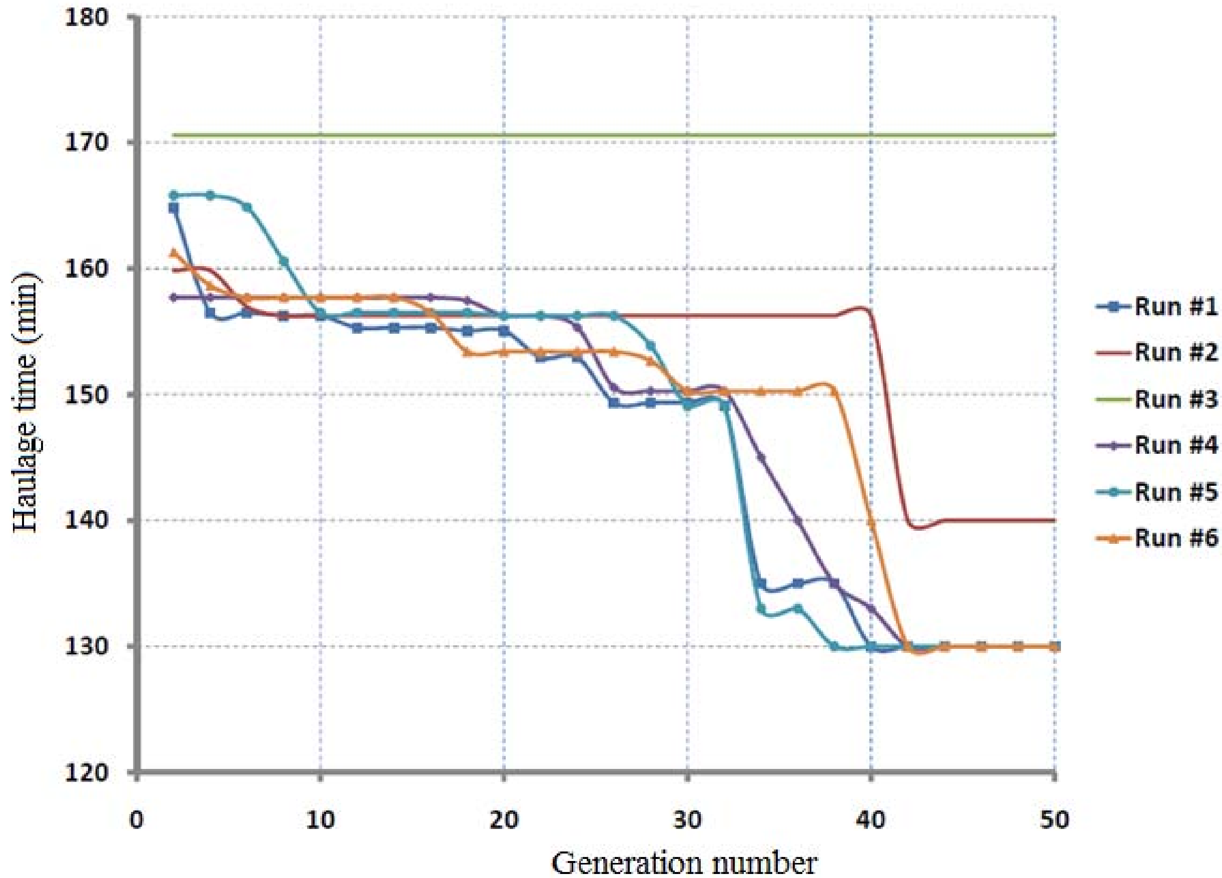


Figure 3-23. Optimization of haulage time as GA progresses (several runs)

By comparing the results that have been given by GA-simulation interaction and GA based on evaluation of distance, the significance of simulation incorporation and its integration with GA can be highlighted. Figure 3-24 exhibits 4 placement arrangements picked from the 9 previously-shown hand-picked arrangements in Figure 3-18. It is seen how material batches move to assemble within a certain radius from the exit point. This figure underlines the impact of fitness function on the performance of GA as well. The reason for such layout of incoming material placement is the simple fact that chromosomes are ranked with respect to their Euclidian distance to exit point. Moving from Figure 3-24 (a) to (d) it is seen that batches with number 10, 2 and 9 move in a way that they stay within a certain radius from the exit point. It should be noted that the length of this radius greatly depends on the yard hard constraints. All the chromosomes shall comply with yard constraints which directly impacts the placement arrangements. In Figure 3-24 (d), all the batches fall within the constant radius that has been shown in all four arrangements. It should be emphasized that this policy (i.e. placement based on distance) does not account for hauling equipment on the yard, nor does it consider the fact that the entire yard has been divided into two distinctive parts. It merely simulates the receiver’s mind and attempts to provide her/him with the optimum placement arrangement.

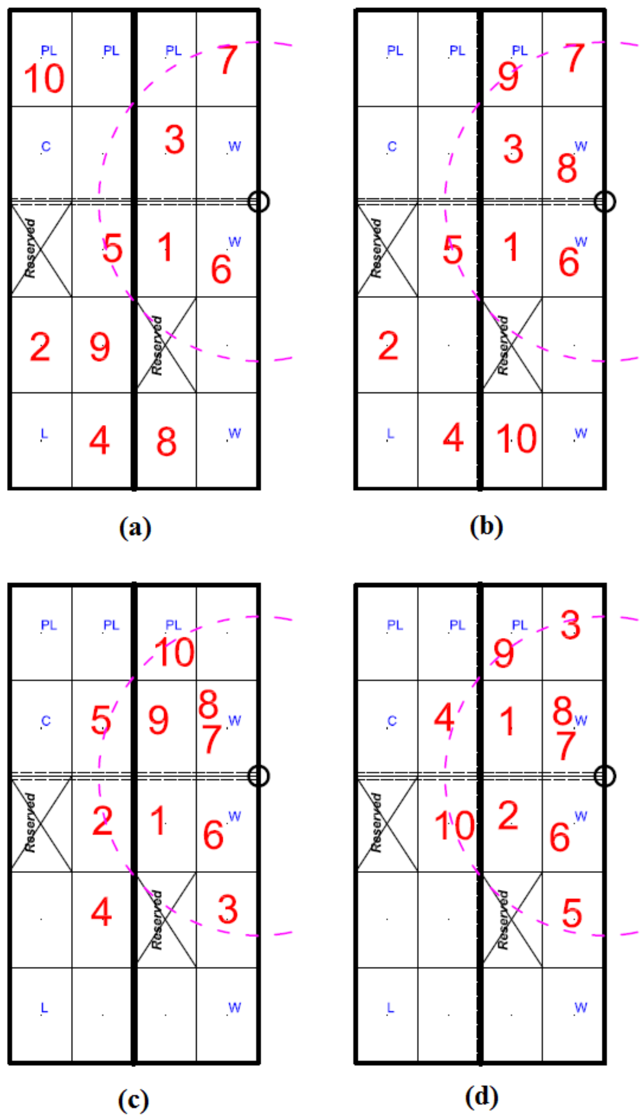
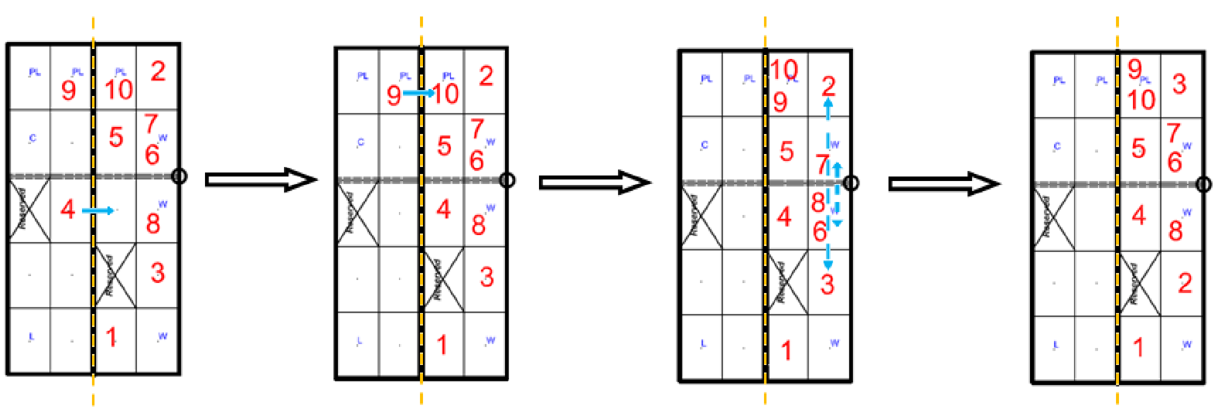


Figure 3-24. Radial congestion of material around the exit point

It was previously shown that in the case of fitness function using distance as the measure of goodness of chromosomes, more than a 30% reduction can be seen in the sum of all distances from placed batches to the exit point, whereas haulage time decrease to 19% from the initial best arrangement to the optimum answer at the end of the analyses. The difference between simulation performance and that of the distance evaluator stems from the nature of simulation in which resource interaction and work processes are modeled accurately. Figure 3-26 can help understand this difference more clearly as it shows four arrangements taken from previously-shown Figure 3-21. It is observed how materials move across the yard in Figures 3-26 (a) and (b) to stay on the south yard so that they can be served by the south overhead crane and minimize the travelling time of the car. In figure (c) also, materials are displaced along the yard to account for different volumes that they have and their impact on the working cycle of the south crane. In (d), and at the end of the analyses, all materials attempt to be stocked in the south yard to be served by one key resource, which is the south crane. This final layout is contrary to the radial placement arrangements given in Figure 3-25, as the fitness function acts completely different.



(a) (b) (c) (d)

Figure 3-25. Movement of material batches on the yard to present the optimum laydown

The final arrangement given in 3-26 (d) suggests the following:

- If all the material batches are nested in the south yard, they are served by the south crane which is itself served by the car with shorter travelling time to the exit point.
- The use of south crane at all times guarantees smoother work process and interaction between the crane and the car as the crane itself does not remain idle and always works since mostly it has to span a wider distance in comparison with the car. The similarity of the travelling speeds of the crane and the car helps prove this fact. In this case, the bottleneck of the simulation would be mostly crane as opposed to the case in which two cranes are served by one car and the bottleneck could be the car in most cases.
- The orders in which the materials are picked are of significance to the overall haulage time. For instance if a chromosome is sent as 11, 8, 20, 3, 14, 16, 4, 10, 2, 19, it is totally up to the receiver to choose which cell s/he shall start first to pick. There is no mathematical background for such selection except for receiver's experience to evaluate the situation. If the materials are disorderly dispersed on both south and north yards, the complication might be greater since the significance of making the right decision would become more conspicuous (refer to Figure 3-21; 1-3). In this particular case, by gathering all the material on the south yard, the GA engine has reduced the risk of making such mistakes.

In order to further highlight the differences between optimization analyses by using distance and simulation as fitness functions, one can simulate the arrangements obtained by distance optimization approach, and compare and contrast the results to discover whether or not the resulting placement arrangements can minimize the haulage time as efficiently as simulation can. Figure 3-27 shows a comparison between four series of analysis results. The first series of the results (as denoted by SARD) represents Simulated Arrangements using Radial Distance as described above. Then the results presented in Figure 3-22 have been redrawn for comparison showing the optimized placement arrangements given by simulation. The third series gives the Simulated Arrangement using Perpendicular Distance calculations. These results were calculated in light of the actual material haulage route within the yard, that is, the cranes traverse along the yard and the car carries the materials afterwards across the yard. It is very likely that the radial distance to the exit point of travel A is less than travel B, but if the batch traverses the yard along and across the yard, the haulage distance of A would be greater than that of B. That is why the fitness function for these series of analyses has been modified properly to generate more realistic distance determinations, which lead to more realistic optimized arrangements. Finally, the fourth series of results represent a further attempt to refine the optimizations given by distance method. Simulated Arrangements given by Weighted Perpendicular Distance determinations (as denoted by SAWPD in the figure) takes into consideration the fact that material haulage time is dependent not only upon the distance to the exit point, but also, they strictly depend on their volumes. That is, transferring a batch with greater volume to the consumption unit would naturally take more time than the batch with lesser volume. A fitness function can be defined to incorporate such volume impact based on Equation (3-1):

$$Total\ cumulative\ target\ distance = \sum_{i=1}^{n} d_i \times V_i \tag{3-1}$$

In this equation d_i is the perpendicular distance of laydown number 'i' to the exit point, and V_i represents the volume of the material batch number 'i'. The fitness that is calculated based on this equation would present an even more realistic estimate of haulage time compared to the last two.

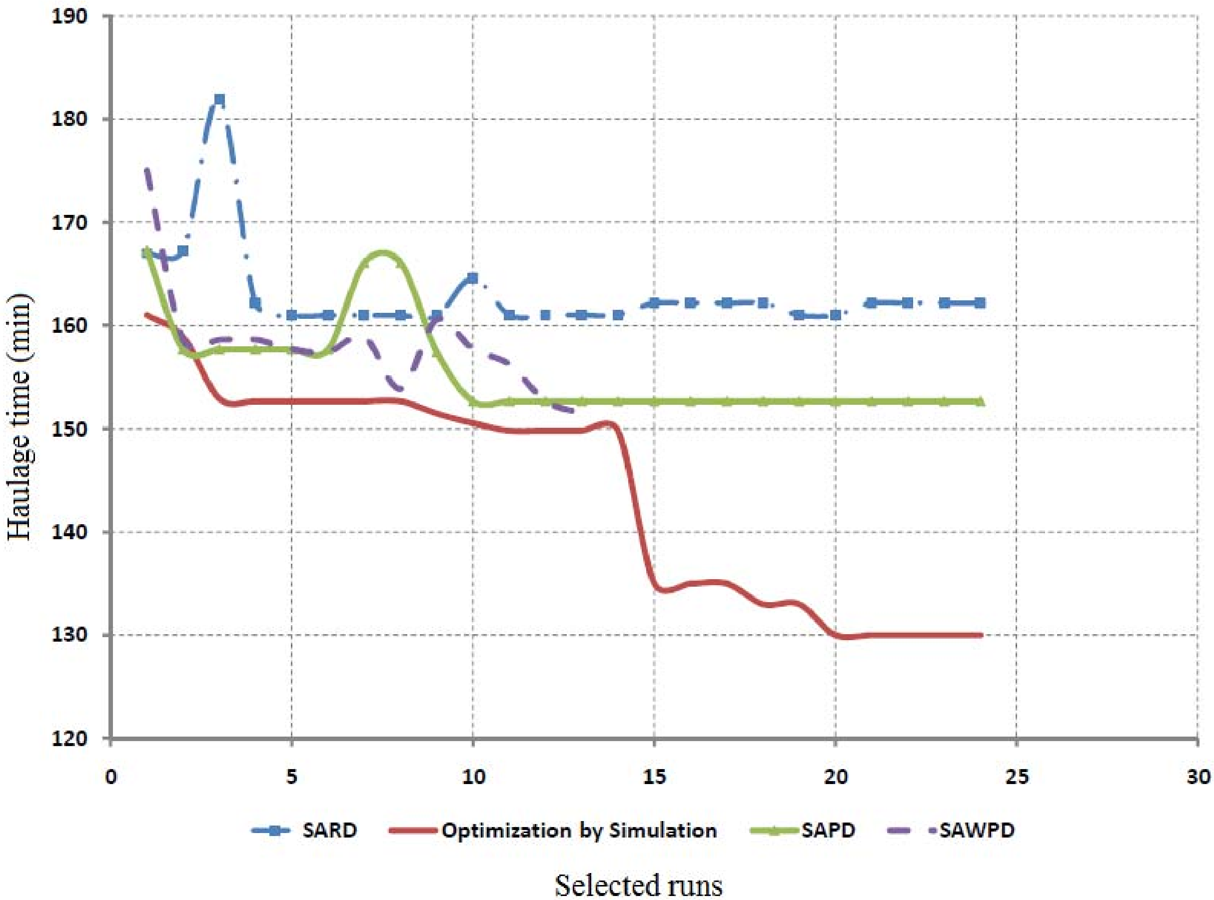


Figure 3-26. Comparison of simulated placement arrangements obtained by different optimization fitness evaluations

As can be observed in Figure 3-27, neither of the distance-generated results can offer optimized haulage time given by simulation. Although improvements can be seen once perpendicular and weighted perpendicular distance fitness functions are used, inconsistencies in optimization trend, fluctuations and excessive overtime compared to the simulation-based results are observed. The following two main reasons can be stated as the root cause of such incompatibilities of the results:

- Distance determination (or weighted distance determination) ignores the capacity of the hauling equipment (cranes and car). The crane could work with its full capacity or a portion of its capacity based on the volume of the material batches. It is very likely that small chunks of materials are hoisted by the crane either due to the original volume of the batch or due to the remaining portion of the materials on a laydown area hauled by the last travel of the crane.

- Distance determination approaches ignore the waiting time of the cranes in south and north yards waiting for the car to serve them. In other words, resource interaction (in particular equipment interaction) is simply disregarded in such analyses whereas simulation can readily incorporate resource interaction through accurate resource modeling.

3.7.SUMMARY AND CONCLUSIONS

In this chapter, a practical material handling problem has been analyzed using the proposed solution. The problem was first investigated in detail and an assessment of current placement policies was offered. It was discussed what ideal policies might exist and which of them might be more practical as there are several complications attached to construction work processes and projects. To address the problem, GA was integrated fully with simulation, and this combination was compared against GA having a distance evaluator fitness function. Results of the analyses in both cases present considerable reduction in haulage distance and time. Optimum arrangement can assist the receiver to make better decisions in placement of the incoming batches considering the yard's hard constraints. The reduction in time and cost can improve craft labor productivity and smooth yard-consumption schedule work interface.

Continuous information flow and interaction between simulation and GA brings about a more realistic modeling of material handling and placement problem and helps present a more accurate optimization. Simulation models work processes and equipment resources which further facilitate the accurate fitness evaluations of proposed placement arrangements on laydown areas within GA. The more complicated the resource interaction is on a laydown yard, the more effective and useful the simulation can be for GA-based optimization problem. The distance evaluator fitness function on the other hand, models what is usually perceived by the receiver as to what the closet laydown would be to place the material. However, simulation might prove that this understanding, which is not fully dependent on the hauling equipment resources and their interactions, might not be always the right placing policy.

CHAPTER 4: PROACTIVE LAYDOWN MANAGEMENT USING GENETIC ALGORITHM INTEGRATED WITH SIMULATION

4.1.OVERVIEW

In the last chapter, a thorough discussion on material placement policies on construction stockyards was presented. It was mentioned how revisions, late-deliveries and changes make the process of scheduling material delivery impossible to program in advance. It was also stated that the direct consequence of such unwanted changes is the fact that the material handling crew should react towards incoming materials, and simply attempt to stock the materials on laydown areas. The tool given in the previous chapter can sufficiently help the decision makers place the materials on the yard cells so as to minimize the travel time from the cells to the consumption unit. The optimization presented in the previous chapter took advantage of simulation to account for the equipment resources available on the yard for materials handling purposes. However, the proposed method does not account for the following:

- The current placement policy does not account for consumption material volume and type. For instance, some material with little volume may be stocked on cell #n which is close to the exit point on day 1, and occupies the cell. Using reactive material placement policy, it is not possible to place other inconsistent material on this cell as long as it is occupied and not consumed. On day 2 a batch arrives at the yard with high volume, based on immediate demand of the consumption unit. Since cell #n, which happens to be close the exit point, is already taken, this batch, whose material type is different from the already-placed material, cannot be stocked on cell #n, and has to be placed elsewhere, probably further from the consumption unit. This placement policy would cause inefficient handling due to high handling costs and time of responding to high consumption demand.

- Proactive material placement policy in which dynamism of the yard material flow has been taken into account. The laydown areas are filled and emptied frequently depending on incoming and outgoing material schedules. Knowing what laydown is going to be when emptied (which day) would help in making a better placement decision.
- Exact inventory and warehousing for a longer period of time (longer than one day). Having an exact inventory depends on knowing which materials come to the yard and which materials leave the yard during a longer period of time. Moreover, it requires a placement schedule which enables the materials management team to know where material batch #n coming on day #m is going to be stocked on the yard. Having this information has many advantages, including minimizing material surplus, higher material handling craft labor productivity, better involvement of material in production cycle, and managing limited space in congested construction sites based on material delivery and consumption schedule. The latter is of profound importance, since in some construction projects, especially on congested construction sites, laydown space is an asset which may not be sufficiently provided to construction crew and contractors. For instance, imagine a construction facility in which several sub-contractors compete to get immediate laydown areas around the massive structure that is being built. Having a proactive material placement policy in place enables the contractors to not only schedule their resources on site more efficiently, but also to argue successfully in case claims, should any be raised for unavailability or sporadic availability of the space.

To further highlight the impact of proactive materials placement, two different cases of materials placement are discussed, as shown in Figures 4-1 and 4-2. In Figure 4-1, two situations have been compared in which in the first one, 20xL8x8x1/8 would be stocked on the laydown space on the far right, and one day after, 5xW14x43 will be placed on two available spaces on the far left. The second situation at the bottom illustrates a swapped situation in which W-sections go the right laydown and iron angles go the left. Based on the reactive material placement policy described in Chapter 3, on day 1, the receiver looks for the closet possible laydown to the exit point and proceeds with the placement. Thereby, placement policy given in Figure 4-1 on the top would be automatically prioritized and implemented. In fact reactive materials placement policy described in the previous chapter is based on the following criteria:

- Proximity to the exit point.
- Work sequence and equipment interactions (e.g. car-crane interaction).
- Hauling equipment capacities and its ratio to the volume of the materials.

Proactive materials management, however, had the schedules available, and made holistic decisions on the basis of consumption demands as well as proximity and equipment interaction criteria. The work suggests that proactive material handling will give freedom to the purchasing manager to procure materials based on demands and place them appropriately on the materials stock yard so that the overall haulage time/cost during the project life time can be minimized. Figure 4-1 at the bottom is based on this placement mentality, in which iron angles are place on either of the far left laydown spaces, even though these spaces are farther to the exit point. The reason for this arrangement is that there would be 4 trips for iron angles and 10 trips for W-sections as of day 2 until day 12. Thus, it would be more reasonable and cost-effective to place iron angles on the left-side laydowns. It is seen in this case that the consumption demand criterion has superseded the proximity preference for the iron angles. It should be noted that in this comparison, consumption of W-sections has started one day after that of the iron angles. On day 2, 10 closer trips for W-sections would take less time than 4 farther trips for iron angles. As such, the proximity criterion still holds, but it is applied in combination with consumption demands.

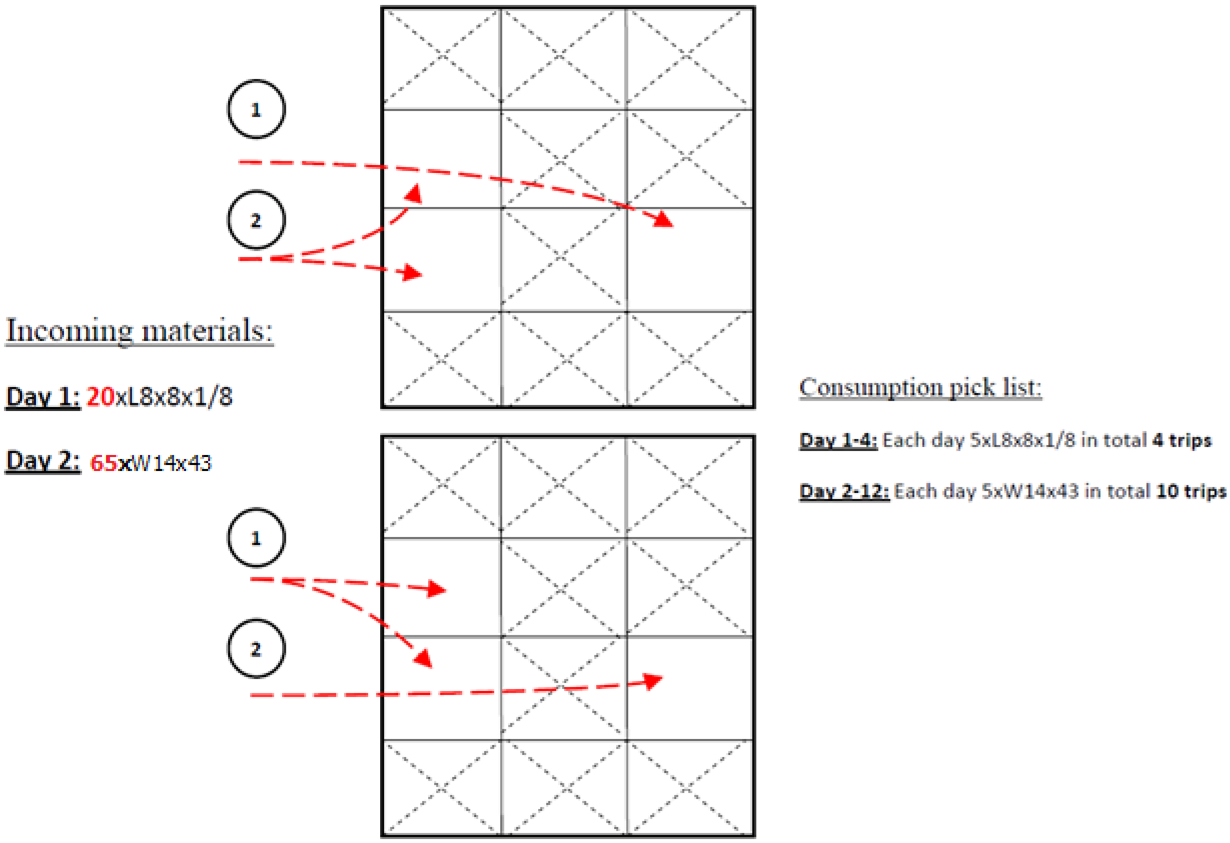


Figure 4-1. An incoming and outgoing materials schedule for 12 days (case 1)

To highlight the impact of proactive materials placement, another situation in which incoming steel plates are added to the previous schedule is studied. Moreover, the consumption of the plates is also appended to the end of the consumption schedule. In this case, plate materials and W-sections are almost concurrently taken off the yard, totaling 25 trips to the consumption unit. Given the fact that iron angles are taken off the yard at the end of day 1, the laydown on the far right will be emptied and ready to accommodate more steel pieces. Now the plates can easily be stocked on the empty spot, and supply the 25-time request of the plate material. This situation is more efficient and cost-effective than the one given in Figure 4-1 at the bottom, where W-sections are placed close to the exit point, since on day 5 when the plates arrive at the yard, they will have no other place to be stocked except for the ones on the left. Although it could be argued that once the demand for W-sections is over, the plates can be placed on their empty spot, the consumption for W-sections adds up to 50 pieces of W-sections, whereas the purchase was done for 65 pieces, leaving 15 pieces on the laydown. The plates cannot be placed on that particular laydown due to the existence of the material consistency constraint (a hard constraint of the yard).

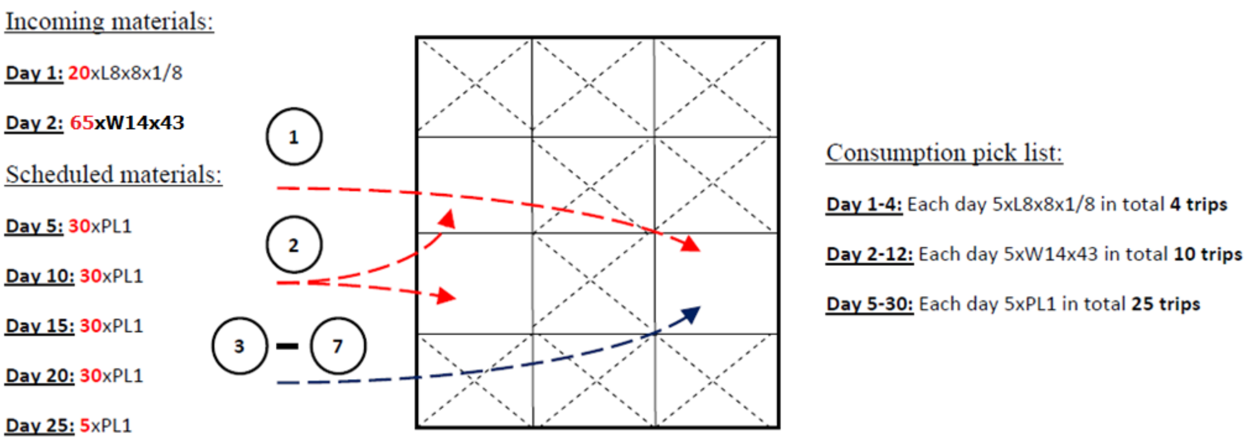


Figure 4-2. An incoming and outgoing material schedule for 30 days (case 2)

For the reasons mentioned above, in this chapter, it is strived to propose a proactive material handling and placement policy in which a placement schedule is presented and material batches are destined to be place on particular cells days before arrival at the yard. The proposed method correctly focuses on the consumption schedule as materials are transported to the consumption unit in practice based on this schedule. To implement this method, the incoming and outgoing material schedules should be known to decision makers in advance, which further requires that material delivery and consumption are not impacted by revisions, late and incorrect deliveries and change orders.

Theory and background of the development is discussed in detail initially to explain the mathematical background of the method. Program implementation is briefly explained afterwards, and a case study is analyzed to verify the suitability of the proposed method in solving material handling problems. Results are presented at the end to evaluate the efficiency of the method to improve material handling and placement processes.

4.2. THEORY AND BACKGROUND

In order to implement a proactive material placement strategy, the time span for material flow to and from the yard shall be expanded to cover a reasonable material flow process. This enables the top management team of construction companies to plan in advance for processes and production cycle in which material circulation plays an essential role. Currently, proactive materials management are practiced successfully in areas such as automotive design in which decisions with regards to different disciplines of material managements are made in advance during bidding or front-end loading. This proactive decision-making minimizes risks and promotes different levels of productivity and process improvements within various processes within an organization. Automotive design has been an inspiration for the construction industry mostly because of its high productivity and quality. Generally, automation is the technique and equipment used to achieve automatic operation or control. Sheer rate of production and customer satisfaction as well as the extremely competitive market has generated an excellent industrial framework for the car industry that can be a role-model to construction, where competitiveness is growing. Automation has mostly necessitated the lean initiatives to lower non-adding values and highlight value-adding activities in production.

One of the lean notions that has been practiced by the car industry is just in time inventory management. This idea is a production and inventory control system in which materials are purchased and units are produced only as needed to meet actual customer demand. Through this process, managers know what materials arrive at storage yards, and what materials are consumed in the front-end planning phase of a manufacturing project. The direct consequence of such accurate planning is the minimization of material storage. It is understood that the construction industry inherently cannot be practiced as “cleanly” as the automotive industry can. However, by promoting an accurate change management program, it might be possible to increase material flow predictability. That is, it would be possible to have incoming material schedule and outgoing material schedule to and from the yard for a broader time-span. This will have the following two direct advantages:

- Exact records of inventory for a specific period of time (preferably project lifetime) which will result in surplus reduction, sophisticated planning, readiness towards changes, minimum unpredictabilities, etc.
- Minimum material haulage time/cost considering the dynamic nature of material flow which will lead to improved craft labor and equipment productivity as well as optimum use of real estate (laydown area) in congested construction sites. It should be emphasized that the discussions and proposed optimization methods in this study are not limited to construction stockyards off-site, but rather, it includes laydown areas which are allocated to different contractors and/or sub-contractors in large construction sites.

In general, a yard could have ‘n’ cells/partitions/segments which can accommodate incoming materials. A batch of material being stocked on a cell has a type and quantity/count. On day 1, there could be several placement arrangements considering the yard hard constraint among which there is only one optimum “placement state.” This optimum, however, is unique with regards to optimization fitness/target function. Assuming a time span of 30 days, an optimum “placement state” of the yard could be represented as the matrix below:

$$\begin{bmatrix} (T,C)_1^1 & (T,C)_1^2 & . & (T,C)_1^n \\ (T,C)_2^1 & (T,C)_2^2 & . & . \\ . & . & . & . \\ (T,C)_{30}^1 & (T,C)_{30}^1 & (T,C)_{30}^1 & (T,C)_{30}^1 \end{bmatrix} \quad (4\text{-}1)$$

Each entry in this matrix represents one yard cell which might be occupied or not occupied on any day during the 30-day period. Assuming there are ‘m’ material types, the cost (i.e. time/distance/cost) of hauling the placed materials from the cells to the exit point can be shown in matrix (4-2):

$$\begin{bmatrix} C_1^1 & C_1^2 & . & C_1^n \\ C_2^1 & C_2^2 & . & C_2^n \\ . & . & . & . \\ C_m^1 & C_m^2 & . & C_m^n \end{bmatrix} \tag{4-2}$$

In this matrix, 'C' represents cost, whose subscript shows the day and superscript represents the cell number. Through the process of genetic optimization, this matrix shall be minimized. In general, there might be two costs associated with material flow (Equation 4-3), but in this study the focus is on cost of the removal. GA fitness function concentrates on cost of the removal through its constant evaluation of the proposed chromosomes.

$$Cost(Day_1) = Cost\left(Placement_{Day1}\right) + Cost(Removal_{Day1}) \tag{4-3}$$

Obviously, the total cost of removal/haulage from/to the exit point is the summation of all the costs for all the days for which material schedule is available.

$$Total\ Cost = \sum_{i=1}^{Nmber\ of\ the\ days=n} Cost_i \tag{4-4}$$

It should be emphasized that hauling equipment and crew work are based on the consumption schedule in reality, as opposed to incoming materials. In fact, it is very likely that available yard inventory can respond to consumption unit needs for several days. That is, there might be independency between incoming and outgoing material for a period of time. However, as the project duration increases, yard inventory at project inception will not be able to satisfy the production needs, and ultimately, the incoming and outgoing materials will be dependent on each other.

Figure 4-3 illustrates the overall algorithm of proactive materials placement. It should be noted that this algorithm is generic, and applies to any time period during construction project lifetime. Similar to the last chapter, this chapter takes advantage of steel fabrication jobs as an example of a construction industry. However, the proposed flowchart in Figure 4-3 applies to most construction industries, in which material placement is of significance. It is seen in this flowchart that upon the request of the shop fabrication, a pick-list is created and sent to the yard receiver/foreman. Her/his job is to deploy hauling equipment to transfer the materials to the shop entry point or yard exit point. Normally, the yard foreman searches for the closet availabilities since s/he can carry out the job faster, and prevent shop crew idleness. Based on the pick-list, yard inventory is updated and used as the yard status, which imposes the yard capacity and material consistency constraints for incoming materials (Equation 4-5). Since the discussion so far has been about the first day, input and output for yesterday are zero in Equation (4-5). It should be added that it is assumed that the placement process is carried out after feeding the consumption unit (the shop in this case).

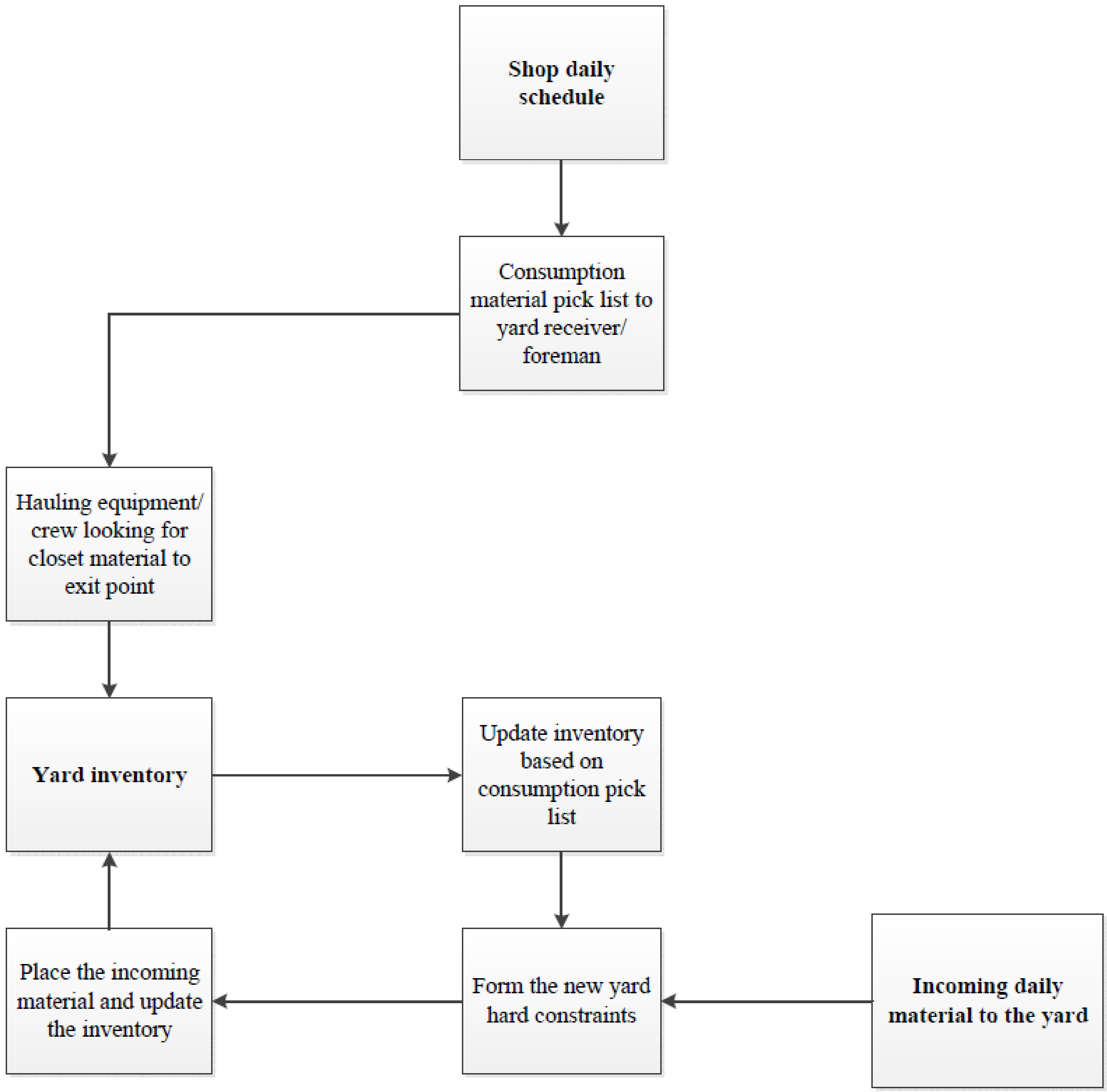


Figure 4-3. Overall algorithm for proactive material placement

Once material batches arrive at the yard, the receiver needs to have the optimum placement arrangements to stock the materials. Assuming there is such an optimum placement, and the placement operation is carried out smoothly, at the end of the first day, the updated yard inventory at the beginning of new day is expressed by equation (4-6).

$$Inventory = (Inventory_{yesterday} - Output_{yesterday}) + Input_{yesterday} \quad (4\text{-}5)$$

This process can be carried out as many days as the incoming and outgoing material schedule exist. As far as GA is concerned, the question to be answered is "which optimum placement arrangement would lead to cheaper/faster shop demand supply?" This question automatically suggests that chromosomes in GA should be made by incoming material batches similar to the previous chapter. Nevertheless, fitness measurements have been performed based on outgoing materials and consumption schedule. As it was mentioned above, this task allocation would not lead to the optimum answer until there is dependency between placements and demands. For a short period of time, it is very likely that there is no relationship between placement layout and what the shop dictates to pick. However, as the time span of the study expands, the inventory would not be able to meet the demands and incoming materials will be used at some point to respond to consumption pick-lists.

Figure 4-2 shows how GA components are related to incoming and outgoing materials. It is seen that outgoing materials play a more important role in proactive material placement optimization than in reactive optimization. It is also seen in this figure how yard hard constraints impact chromosome formation and mutation and crossover operations. As the solution moves from one day to the next, equation (4-5) is used to update the inventory daily, which suggests that hard constraints of the yard change continually. That further entails that chromosomes are generated with regards to varying constraints for each day. Thus, it would be reasonable to indicate that chromosome creation is affected by "dynamic yard constraints," as shown in Figure 4-4.

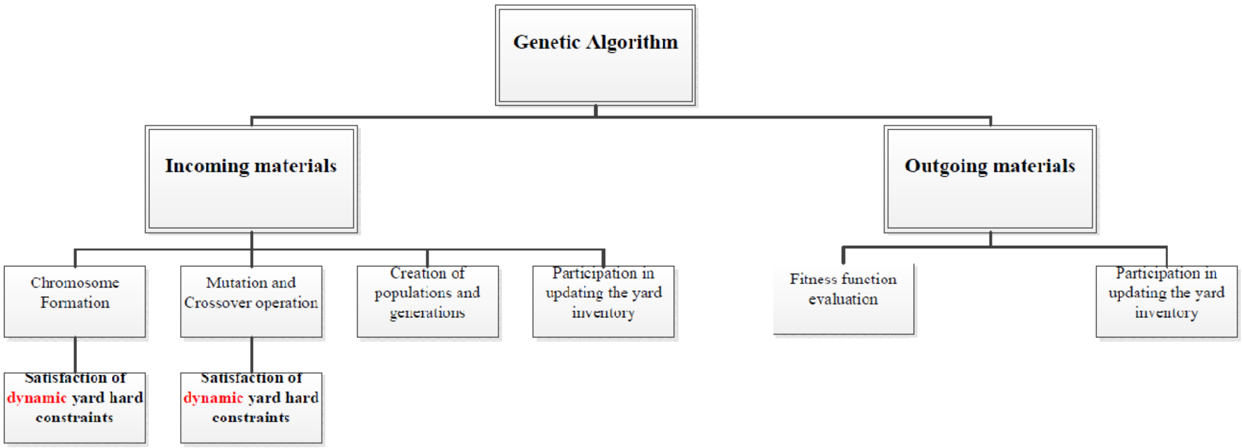


Figure 4-4. GA and its components in relation to incoming and outgoing materials

It is obvious that due to a high number of unknowns, and the extent of solution space, the population size for this solution development should be chosen large enough so that the optimum solution can be reached after reasonable iteration. Later, it will be shown what values are calibrated and selected for GA internal functions through some practical examples.

4.3.PROGRAM IMPLEMENTATION

The computer program written for the proactive material placement is a major extension to the program developed for the reactive material placement. In this section, the focus is only made on the extensions to the previous chapter program implementations. In brief, the extensions made to the reactive placement program are:

- Development of the overall algorithm to include inventory update, daily 'constraint yard' formation, and keeping exact record of inventory for every single day.
- Formation of the chromosomes on the basis of arrangements for any period of time that the proactive materials handling might be performed.

- Fitness function evaluation based on consumption schedule (e.g. pick-list) for any period of time the study is being carried out. This evaluation provides a more realistic simulation of yard material flow as the purpose of placement optimization is to provide placement arrangement in which cost of material haulage to the exit point is minimized. It is understood that incoming materials to the yard are not directly sent to the exit point; often times they are stocked on the yard, but the pick list governs what materials travel to the consumption unit.
- Improved material removal technique based on the closet possible material availability to the exit point.
- Improved validation algorithm for offspring to comply with yard hard constraints. The program iterates until fit offspring that satisfy the yard hard constraints are generated.
- Modification of yard emptying and filling processes in which laydown areas are emptied if material is excessively taken from it, which would provide room for other materials, perhaps with different types, to be stocked on the laydown.

Starting from the first extension, one needs to develop an algorithm to continuously update the yard inventory, and more importantly, provide updated yard hard constraints to the incoming materials to be placed on consecutive days. To that end, it is assumed the yard at the first day of study has inventory 'I_1'. Material pick list is provided to the yard foreman/receiver and s/he attempts to pick the closest materials s/he finds and take them to the exit point to feed the consumption. The remaining inventory forms the hard constraints for the incoming materials every day. Equation (4-6) shows how such constraints are formed every day.

$$I_C^i = I^i - Out^i \tag{4-6}$$

In this equation, 'i' represents the day number and 'I_C' denotes the yard inventory based on which yard constraints are formed for the incoming materials. This inventory is termed as 'constraint yard' herein.

Once the incoming materials satisfy the yard constraints, and are placed on the yard properly, the inventory for the next day can be updated (Equation 4-7).

$$I^i = I_C^{i-1} + Input^{i-1} \tag{4-7}$$

In Equation (4-7) it is seen how 'I^i' in Equation (4-6) is formed. Using these two equations, chromosomes and pick list tags can be updated daily to generate the required information for the GA. Knowing this, Figure 4-3 can be updated to make more sense in terms of GA terminology, as shown in Figure 4-5.

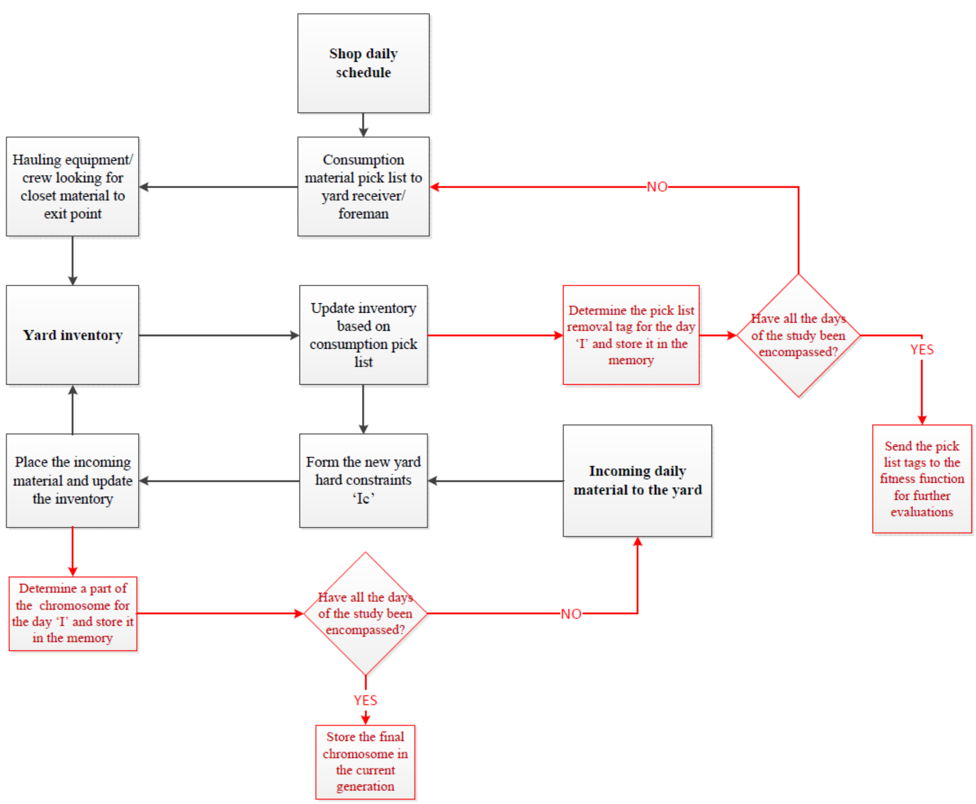


Figure 4-5. Updated overall algorithm for proactive material placement (GA-terms included)

It is seen in Figure 4-5 that the materials are taken off the yard with respect to their proximity to the exit point. An algorithm to find the closet materials is developed in this version of the program to readily find the closet materials on the pick list and update the pick list removal tag, as shown in Figure 4-4. In development of this algorithm, effort is made to put the program in the yard foreman's shoes to discover how s/he determines what material is closer to the exit point. Often times, s/he simply makes the decision based on the following:

- Visual inspection of the yard. This method applies to small, uncongested yards where the foreman can simply find the material visually. In this case, s/he instantly makes an estimate of the proximity and decides where to pick the material. The estimation of distance in a human's mind is often times made on the basis of Euclidian distance.
- Having a sketch of the yard segmentations/cells on a piece of paper along with available inventory. In this case, too, the foreman needs to make quick decisions. Euclidian/radial distance would be the first option that might occur to her/is mind when it comes to determination of the closet availability.

As a consequence, it is decided to use Euclidian/radial distance in the algorithm, which is illustrated in Figure 4-6.

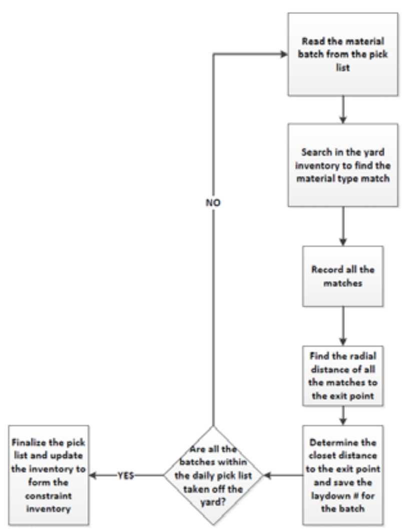


Figure 4-6. An algorithm to form the constraint inventory based on the consumption pick list

In order to store all the updated inventories, and continuously update them with respect to new incoming and outgoing materials, a new class has been developed to store the inventories for all the days. The class has two private fields, namely, day and inventory. The former indicates the day that inventory is stored, and the latter is a list of yard laydown spaces which altogether constitutes the yard inventory. The class instantiates a list of inventory data, which is updated continuously, and forms the inventory for the consecutive days from which materials need to be picked. Moreover, The class instantiates a list that creates a data base of constraint inventories which interact with the updated inventory data base, through the Equations (4-6) and (4-7).

Another extension to the previously-developed program in Chapter 3 is the improvement of placement verifier method/function, which ensures the compliance of offspring to the yard hard constraints. It should be noted that having long chromosomes with a high number of genes would create problems once mutation and crossover operators are to be applied. PlacementVerifier function in the program ensures that offspring generated from the fittest parents satisfy the hard constraints of the yard. However, the probability of creating such a generation is not high once long chromosomes are married together. In this version of the program, iterations were carried out until a fit chromosome was generated. This development has the merit of higher convergence rate but the disadvantage of significantly-prolonging the runtime.

In terms of programming, the engine of the solution was modified to suit the inventory update. Code below shows a small part of the algorithm in which formulae (4-6) and (4-7) are implemented within CreateGene method:

```
List<InventoryDatabase> ThirtyDayInventory = new List<InventoryDatabase>();
            ThirtyDayInventory.Add(new
InventoryDatabase(1,this.ThirtydayInventory(YardInventory,1)));
```

It is seen how Thirty day Inventory is sent to the inventory data base and get updated. Moreover, code below makes sure that all materials for all the thirty days satisfy the yard hard constraints. It creates a loop over the incoming materials on any particular day and suggests random placement as part of the GA chromosome creation strategy, and then makes sure the created chromosomes are in conformance with the yard hard constraints:

```
foreach (var t in IncomingMaterials)
            {

                if (t.date > dayCounter1)
                {
                    UpdatedYardDataBase.Add(new InventoryDatabase(t.date,
ArtificialYard));
                    ArtificialYard.Clear();
                    ConstraintYardDataBase.Add(new InventoryDatabase(t.date,
ThirtydayInventory(UpdatedYardDataBase[t.date - 2].SingleDayYardInventory,
t.date)));
                    //
*****************************************************************

                    dayCounter1 = t.date;

                    // ********************  Clear the empty yard cells
***************
                    foreach (var pp in ConstraintYardDataBase[t.date -
1].SingleDayYardInventory)
                    {
                        double counter5 = 0.0;
                        if (pp.MaterialsOntheYard.Count != 0)
                        {
                            foreach (var a in pp.MaterialsOntheYard)
                            {
                                counter5 = counter5 + a.quantity;

                            }
                            if (counter5 == 0)
                                pp.MaterialsOntheYard.Clear();
                        }
                    }

                }//End of two formula block

                //********** First update the yard based on constraintInventory
******
                // Here I want to make a clone of previously calculated
constraintinventory ****
                if (t.date > dayCounter2)
                {
                    foreach (var pp in ConstraintYardDataBase[t.date -
1].SingleDayYardInventory)
                        ArtificialYard.Add(pp.DeepClone()); //Remember to clear
this temporary yard at the end
                    dayCounter2 = t.date;
                }

                // *************** End of cloning *************************
                counter = 0;
                //Console.WriteLine("Placing material #: {0}\n", m.ID);
                bool placementStatus = false;
                while (!placementStatus)
                {
                    //m.segmentTag = (int)getlocalRandom[counter]; //number of the
segments to be 20
                    t.segmentTag = rnd.Next(1, 21);
                    foreach (var s in ArtificialYard)
                    {
                        if (t.segmentTag == s.ID)
                        {
                            if (s.Status == true)
```

```
                            {
                                if (s.MaterialsOntheYard.Count == 0)
                                {
                                    MaterialLibrary newMaterial = new
MaterialLibrary(t.TypeandQtyandLength.type, t.TypeandQtyandLength.quantity,
t.TypeandQtyandLength.length, t.TypeandQtyandLength.SecProperties.ToArray());
                                    s.MaterialsOntheYard.Add(newMaterial);
                                    placementStatus = true;
                                    //Genes[m.ID - 1] = n.ID;
                                }
                                else if (t.TypeandQtyandLength.type ==
s.MaterialsOntheYard[0].type)
                                {
                                    double tempCap = s.GetSegmentVolume();
                                    if (t.volume <= s.Capacity-tempCap)
                                    {
                                        placementStatus = true;
                                        //Genes[m.ID - 1] = n.ID;

                                        //Inventory update
                                        int flag = 0;
                                        foreach (var p in s.MaterialsOntheYard)
                                        {
                                            if (t.TypeandQtyandLength.ToString()
== p.ToString())

                                            {
                                                p.quantity = p.quantity +

t.TypeandQtyandLength.quantity;

                                                flag++;
                                            }
                                            //if(flag==false)

                                        }
                                        if (flag == 0)
                                        {
                                            s.MaterialsOntheYard.Add(new
MaterialLibrary(t.TypeandQtyandLength.type, t.TypeandQtyandLength.quantity,
t.TypeandQtyandLength.length, t.TypeandQtyandLength.SecProperties));
                                        }


                                    }

                                }
                            }
                        }
                    }
                    counter++;
                    if (counter == 100000) //One hunderd trials
                    {
                        Console.WriteLine("I am afraind there is no space on the
yard!!");

                        Console.WriteLine("the program attempted {0}

times",counter);

                        //return;
                    }
                }

            }//End of material placement for all the days
```

4.4. CASE STUDY

In this section, a case study is selected to exhibit the suitability of the development to perform proactive, optimized material placement on construction yards. The selected case study is a steel fabrication yard of Waiward Steel

Fabricators Ltd. with the dimensions given in the previous chapter. The following changes to the case study presented in the last chapter, have been applied:

- Incoming materials and outgoing materials have been closely monitored for 30 days in October 2012 through discussions with purchasing departments of Waiward Steel Fabricators Ltd. The final proposed incoming materials schedule is inspired by the procurement list that has been made available to the author. The situation was complicated once the shop pick list for the same period of time was to be retrieved. Modifications have been made to the material procurement list as there were some data missing for some days, and it was rather difficult to access the exact information since the company itself did not provide the authors the exact list of consumed materials during a 30-day period. Sporadic information as to what materials were recorded by the yard receiver and shop foreman in Fall 2012 was available to the author, and it was therefore decided to summarize and organize the information for October 2012. The final proposed consumption schedule was inspired by the actual pick list.
- Similar to the previous chapter, focus has been made on 4 main types of materials, each of which has three different section sizes. Enlarging the material diversity database would make the modeling and analyses closer to actual practice, but it would not add further to the scientific value of the research, nor would the lack of it lower the significance and validity of the study.
- Available inventory has been modified properly in the month of October to present more value to the current research. As indicated in the previous chapter, a less-congested yard with some empty spots has been modeled as opposed to current congested yard to provide the readers with more tangible and conclusive results. It should be noted that a more congested yard would have fewer available placement options for the incoming materials, thus shrinking the solution space to fewer possibilities. Furthermore, material flow towards the formation of

most optimum arrangement would not be as conspicuous as a less-congested yard where the movement of materials can be studied more conveniently.

- It is assumed that there is only one transfer line for materials to the shop whereas in reality, there are three to four transfer lines, each of which is designated to transfer some particular material types to the shop. Again, it should be advised that this assumption is made to bring more clarity and intuition to the generated results, and to avoid further complication to the problem, which is unrelated to the main purpose of this study. Addition of transfer lines is a matter of changing the simulation model and fitness function slightly to account for additional exit points.

4.4.1. Incoming and outgoing schedules of materials

As stated before, a 30-day schedule for incoming and outgoing materials has been selected and modified properly. In general, in order to carry out material procurement for a steel fabrication company some provisions should be taken into consideration. The purchasing unit proceeds with procurement in the early stages of the project and in pre-planning phase, as described in the previous chapter. The decision as to what material should be purchased in impacted directly by the demand (i.e. consumption). In proactive materials management, the overall project baseline schedule governs the shop/consumption work schedule in the front-end loading phase of the project. That is, it is understood early in the project what materials are planned to enter the shop, and in what size and volume. Based on such detailed scheduling, the purchasing unit can plan its procurement in early stages of the project. However, the following should be considered in steel procurement (it could be extended to material procurement in general as well):

- Based on the consumption schedule, the volume of the steel pieces going to the shop cannot be greater than a certain value. Similarly, it is unreasonable to load-list a great size of materials into the purchasing system for one particular day. That is, it is not practical to buy a huge volume of materials on one day as the yard receiving operation has a certain capacity. On the other hand, it would be irrational to purchase very small sizes of the material on one day as transportation incurs considerable costs.

- The purchasing should be carried out in a way that shop demands can always be satisfied without interruption, as otherwise, this would cause massive loss in productivity and labor morale. Additionally, the optimum purchasing plan would be one that minimizes the inventory as well. The more a purchasing operation can go towards just-in-time materials management, the more efficiency can be achieved, as discussed previously in brief.
- The purchasing shall be planned in a way that the inventory would not be partially or totally depleted at the end of the project. This is obvious as in practice there are multiple projects that concurrently need to be supplied material by purchasing. Regardless of the number of active projects, the inventory shall not be totally depleted in construction projects as there should always be room for unforeseen events.
- The shop pick demand will be operated by the yard foreman who selects the closet materials, as discussed previously; however, the material procurement unit may not pay attention to the placement as it is the yard receiver's job to proceed to stock the materials on the yard. The primary purpose of this study is in fact to provide either the yard receiver or the purchasing department with the exact placement arrangement that will result in the least expensive supply of the consumption unit.

Figure 4-7 shows a holistic schedule of incoming and outgoing materials for a 30-day period. Inputs and outputs, durations and quantities are clearly specified in this figure.

Material type	I/O	One month duration of material flow on the yard																													
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
L8x8x1/8	Input																														
	Output		10	10	10	10	10	10	10	10	10																				
L6x6x3/8	Input	10	10	10	10	10	10	10	10	10	10																				
	Output		5	5	5	5	5	5	5	5	5																				
L6x4x3/8	Input	10	10	10	10	10	10	10	10	10	10																				
	Output		5	5	5	5	5	5	5	5	5																				
W8x24	Input											35	35	35	35	35	35	35	35	35	35	35	35	35	35	35	35	35	35	35	35
	Output	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30
W10x30	Input																				50	50	50	50	50	50					
	Output	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	30	30	30	30	30	30	30	30	30	30
W14x43	Input									100				100					50					50							
	Output	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
C10x15.3	Input															50					50					50					
	Output											10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
C8x13.75	Input															50					50					50					
	Output											10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
C15x50	Input															50					50					50					
	Output											10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
PL3/8	Input																									5	5	5	5	5	5
	Output	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
PL1	Input																				10					10		10			
	Output	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
PL1/2	Input																				10					10					
	Output	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2

Figure 4-7. Incoming and outgoing schedule of materials in one view

The procurement plan for this schedule has been designed based on the shop consumption. It is seen in this figure that 1) there is constant demand for W-sections and plates throughout the 30 days, 2) iron angles are needed for the first 10 days only and 3) channel sections are needed as of day 10.

Looking at the inventory, it is seen that there is considerable amount of W-sections, plates and channel sections all over the yard, but there is lack of iron angles. Iron angles are only available on laydown number 1 (L8x8x1/8). There are no other types of iron angles, knowing that the shop needs it on the second day. However, it is seen that iron angles are not required as of day 10, thus it would be reasonable to purchase them for the first 10 days to meet the shop's demands and then leave some for future projects. Table 4-1 shows the inventory on day 1 where iron angles are only found on laydown number 1.

Table 4-1. Sample yard inventory

ID	Quantity x (Material)	ID	Quantity x (Material)
1	215x(L8x8x1/8)	11	Empty
2	Empty	12	102x(W8x24)+400x(W10x30)+50x(W14x43)
3	Empty	13	100x(C10x15.3)+100x(C8x13.75)+100x(C15x50)
4	170x(W8x24)	14	Empty
5	Empty	15	Empty
6	Empty	16	300x(W8x24)+158x(W10x30)+50x(W14x43)
7	Reserved	17	88x(PL3/8)+30x(PL1)+20x(PL1/2)
8	Empty	18	10x(PL3/8)+10x(PL1)+10x(PL1/2)
9	Reserved	19	10x(PL3/8)+10x(PL1)+10x(PL1/2)
10	Empty	20	Empty

As of day 10, W-sections need to be purchase due to excessive shop demands. Through discussions with the purchasing department, it was discovered that it is a routine to deliver some W-sections every day due to its frequent use in steel fabrications jobs. On some particular days it might be decided to add more volume of W-sections to stay on the safe side (i.e. days #9, 13, 18, 23). Channel sections are also purchased on certain intervals to supply the shop, and to avoid inventory depletion. As for the plates, it is ensured that the production cycle is not interrupted and the yard is not overloaded.

A better source of information for the purchasing department would be volume bar charts, in which the volume of the material of the entire shop demand, as well as that of the procurement list for 30 days have been compared to evaluate the adequacy of the incoming materials to supply the consumption schedule. Now, based on the company's holistic policy, the top management might decide to decrease the volume of the yard materials at the end of the project or vice versa. The former requires less total volume of inputs compared to that of the outputs, and the latter is the opposite. Figure 4-8 illustrates a comparison chart between volumes of the inputs, that of the outputs, and volume of the available materials on the yard on day 1.

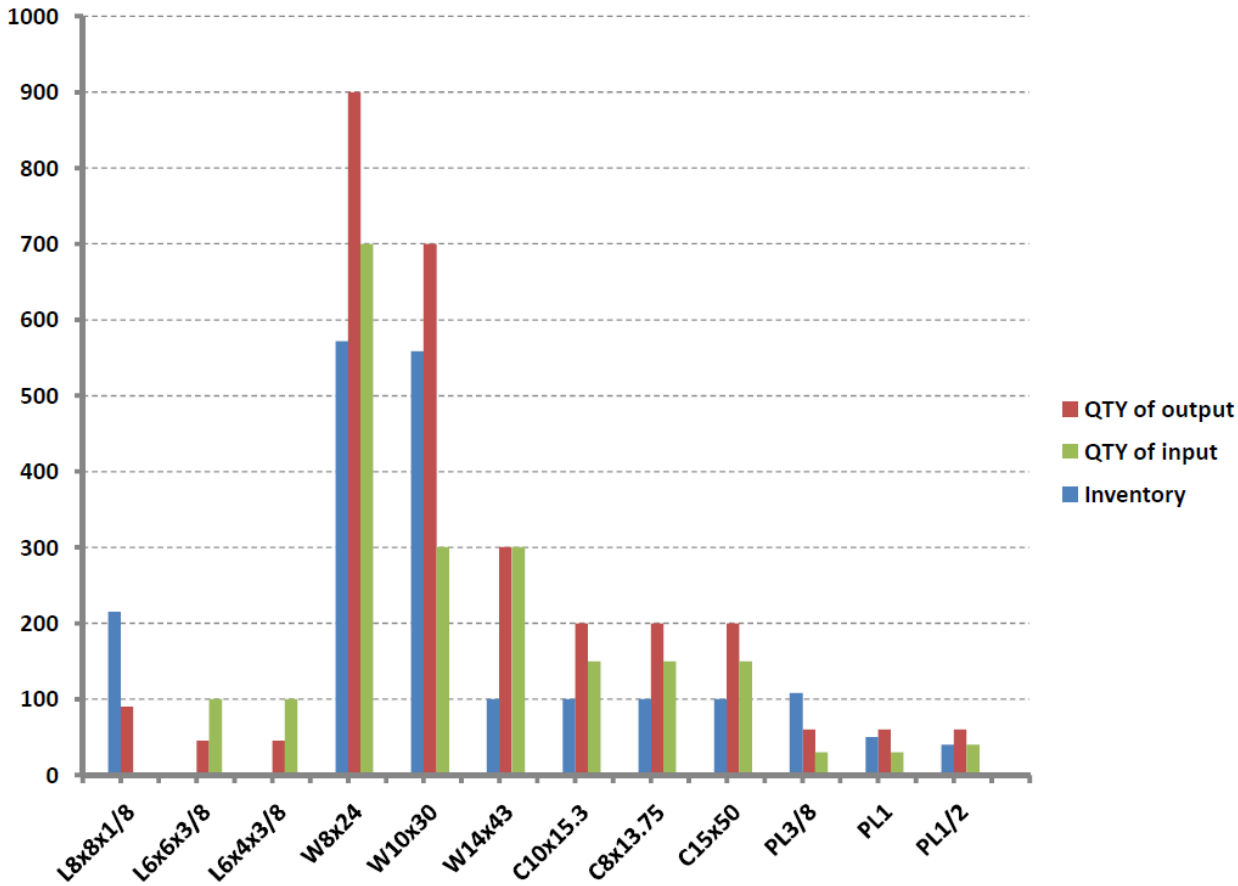


Figure 4-8. Total volume comparison between inputs, outputs and inventory on day 1

In this particular project, the plan was to leave some iron angles on the yard at the end of the thirty-day period, but to lower the volume of the rest of the materials slightly to provide a balance in the inventory. It should be emphasized that planning for the purchasing department takes place as discussed above, and often times, the material flow on the yard and its logistics might not be accounted for meticulously. It is the job of the receiver or material handling manager to decide which laydown spaces would be the best places to stock material on, given this holistic purchasing plan, which is directly impacted by the shop demands.

4.4.2. Analysis and results

The case study discussed in the previous section was modeled within the program and analyses for 2000 generations were performed to obtain the most optimum results. The GA parameters are set the same as the ones presented in Table 3-5 except for the population size that has selected to be 200, twice as much as the population size of the analyses given in the previous chapter. Due to the following reasons, the analysis time was considerably high:

- High length of the chromosomes. Each gene represents one incoming batch, that is, in total there are 71 genes which form one chromosome.
- Population size. As mentioned before, population size is selected to be 200 to reach better accuracy.
- Improved placement verification algorithm. The crossover and mutation operators nest the PlacementVerifier method, which ensures the compliance of the offspring to the yard hard constraints. This entails a great number of iterations over the generated offspring to reach the new, acceptable generation.
- Simulation is run as an external resource. In the current development, simulation is called and applied as an external resource for the C# program which lengthens the runtime. It will also be discussed in the next chapter as a recommendation for future work to take advantage of the open-source nature of Simphony to run the simulation engine within the C# program.

Figure 4-9 shows the reduction in haulage time as the analysis progresses. This time is the total haulage time of the materials from the laydown spaces to the yard exit point or shop entry point. The proposed simulation-GA integrated solution was able to lower the haulage time in excess of 9% of the entire haulage time.

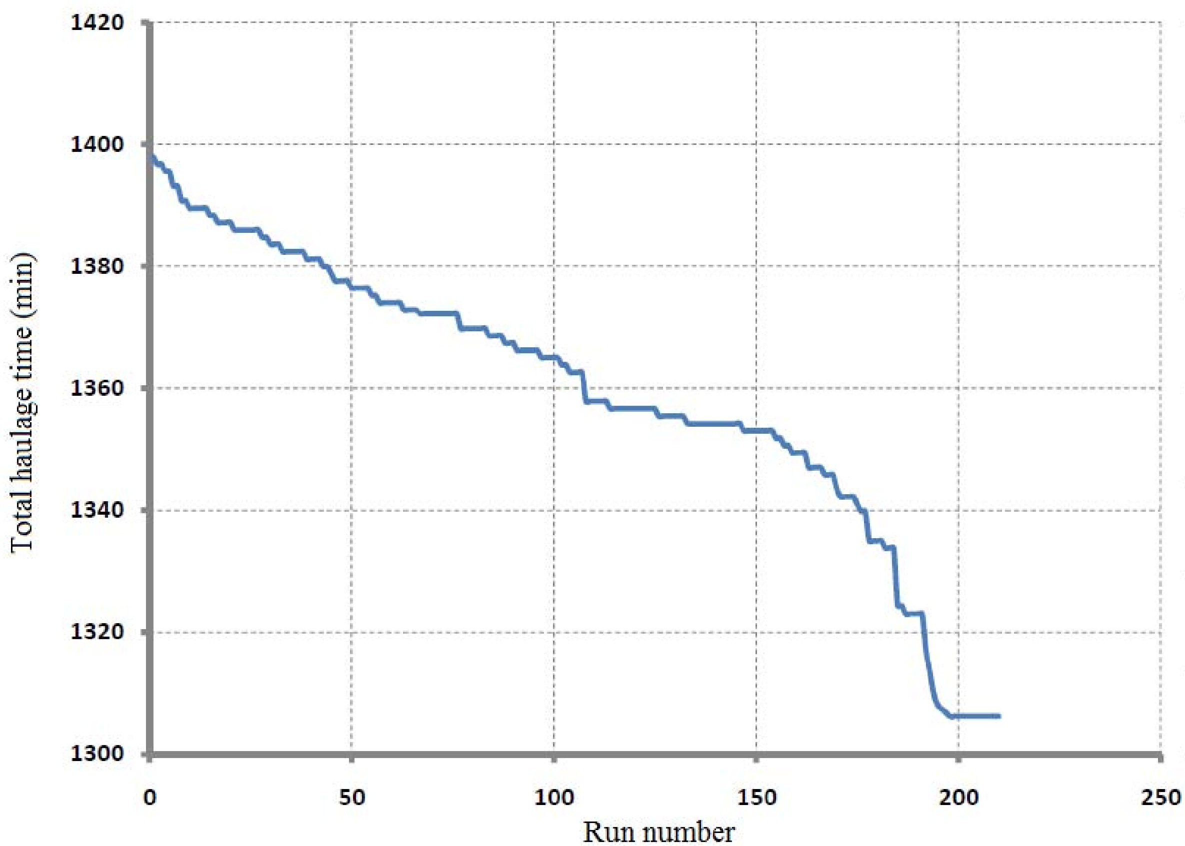


Figure 4-9. Optimization of the haulage time of 271 material batches to the yard exit point

The termination criterion was chosen to be the same as the one used in the previous chapter. In this study, it was observed that after 205 generations, there were 10 similar results, suggesting a local optimum being achieved. The runtime for these 215 generations was 7 hours, which is very long, due to the reason explained above. It is understood that more generations were needed to reach an absolute optima but the runtime would be have been considerably longer. Nevertheless, the local optimum achieved in this analysis will be shown to present very efficient placement arrangement. Table 4-2 shows some selected placement arrangements which demonstrate the ability of the solutions to optimize the arrangement for the incoming materials. It should be noted that the fitness values shown above the arrangements are the corresponding shop consumption supplies that are directly impacted by the incoming materials. That is, performing the placement arrangements shown in the table would result in minimum haulage time of materials from laydown spaces to the shop on the basis of shop pick lists for 30 days. In Appendix 1 the corresponding removal arrangements for these incoming schedules are shown. There are 271 outgoing batches, which results in a table having 271 rows, as given in Appendix 1.

A closer study of the optimum arrangement shown on the column on the far right (the best proposed arrangement with the least haulage time) would highlight the significance of proactive materials handling on the yard. In the previous chapter, it was discussed that due to the arrangement of the hauling equipment on the yard, the yard has been divided into south and north yards, in which south and north overhead cranes work correspondingly. It was stated that the optimum arrangement proposed by the simulation is the situation in which all the materials can be stocked on the south yard. The reasons for that are first the proximity of the south yard to the exit point, and second the car-crane interaction, which would add delay if the car were to serve two cranes as opposed to serving only the south overhead crane. In the reactive solution given in the previous chapter, there was no provision for the materials coming on the later days, and there were only two criteria for optimizations to be carried out as discussed above. However, once the planner knows about the incoming and outgoing materials for a longer period of time (e.g. duration of a fabrication project), s/he will be able to manage the placement proactively, and take other criteria into consideration as discussed in 4.1.

Figure 4-10 graphically illustrates how the proposed solution has provided the planner with the optimized arrangement. Material flow for only two days is shown for brevity. Starting from day 1, materials are removed from the yard based on the first day pick list. As discussed earlier, this process is performed on the basis of closet possible cells to the exit point, as shown in Figure 4-10. Then it comes to the first day incoming materials which are iron angles. They are placed on laydowns 3 and 8. These laydowns are on the south yard. They are suitable places for the south overhead cranes to be served.

Table 4-2. Seven placement arrangements proposed by the GA-simulation engine

		Chromosome fitness (material batch haulage time to the exit point- Time unit is hour)						
		23.27813685	23.0790768	22.9795468	22.45199083	22.2529308	21.894623	21.79509283
Day No.	**Batch No.**	**Yard laydown number (where the material batches are to be placed)**						
1	**1**	2	6	14	11	1	5	8
1	**2**	10	3	6	14	10	2	3
2	**3**	10	8	3	11	14	20	14
2	**4**	14	8	10	20	11	8	5
3	**5**	1	2	6	5	20	14	20
3	**6**	10	20	15	14	8	5	15
4	**7**	5	15	11	6	11	14	20
4	**8**	20	11	8	14	2	2	8
5	**9**	20	6	2	10	19	20	1
5	**10**	10	19	3	8	19	10	5
6	**11**	3	11	2	8	19	6	5
6	**12**	10	2	1	2	2	10	6
7	**13**	14	20	20	15	10	15	6
7	**14**	6	15	6	5	20	1	1
8	**15**	5	1	1	14	14	10	20
8	**16**	6	15	19	1	19	11	14
9	**17**	6	6	8	15	8	1	14
9	**18**	19	8	1	20	6	5	10
9	**19**	4	12	10	3	4	12	2
10	**20**	6	15	14	5	3	18	18
10	**21**	19	20	20	18	3	8	5
10	**22**	20	5	10	12	4	16	16
11	**23**	15	18	5	3	16	12	15
12	**24**	11	18	16	11	11	4	16
13	**25**	14	10	5	11	12	15	12
13	**26**	4	4	15	16	4	12	4
14	**27**	12	4	18	4	16	3	4
15	**28**	13	14	13	13	15	20	13
15	**29**	13	13	11	19	13	20	8
15	**30**	18	14	13	13	15	19	19
15	**31**	12	5	10	3	16	12	4
16	**32**	8	10	10	4	4	16	12
17	**33**	8	12	10	11	12	16	15
18	**34**	11	16	4	4	11	12	15
18	**35**	16	10	18	16	18	16	3
19	**36**	20	16	10	16	5	3	20
20	**37**	18	13	19	19	15	19	19
20	**38**	13	14	13	19	20	20	8
20	**39**	16	18	11	4	12	15	2
20	**40**	13	11	13	13	20	11	19
20	**41**	17	17	17	17	17	17	11
20	**42**	17	17	17	17	17	17	11
20	**43**	8	16	4	12	11	3	3
21	**44**	8	5	11	4	4	3	2
21	**45**	15	10	11	12	11	12	4
22	**46**	12	10	10	11	18	15	20
22	**47**	4	10	12	3	5	3	12
23	**48**	12	16	12	4	11	3	2
23	**49**	12	18	16	11	11	15	20
23	**50**	4	5	11	16	4	15	16
24	**51**	11	10	18	16	11	15	4
24	**52**	4	18	15	16	4	12	3
25	**53**	4	5	11	12	5	4	3
25	**54**	18	13	19	13	15	11	19
25	**55**	13	13	19	19	13	11	8
25	**56**	13	13	13	19	13	20	8
25	**57**	17	17	17	17	17	17	17
25	**58**	17	14	17	17	20	19	11
25	**59**	17	14	17	17	20	17	11
25	**60**	20	18	15	3	12	12	4
26	**61**	17	14	17	17	20	17	17
26	**62**	15	12	5	12	12	4	4
27	**63**	17	11	17	17	17	19	11
27	**64**	17	11	17	17	17	19	17
27	**65**	14	4	10	11	5	12	4
28	**66**	17	11	17	17	17	19	11
28	**67**	20	18	11	12	18	4	3
29	**68**	4	10	4	16	16	3	2
29	**69**	17	17	17	17	17	17	11
30	**70**	8	5	10	4	11	3	8
30	**71**	17	17	17	20	20	17	**11**

On day 2, the shop needs two types of iron angles, namely, L6x6x3/8 and L6x4x3/8, which have been stocked on the yard the day before, thereby the shop access them easily in little time. There are other materials on the list that are fed to the yard based on their proximity, as show in Figure 4-10 at the right bottom.

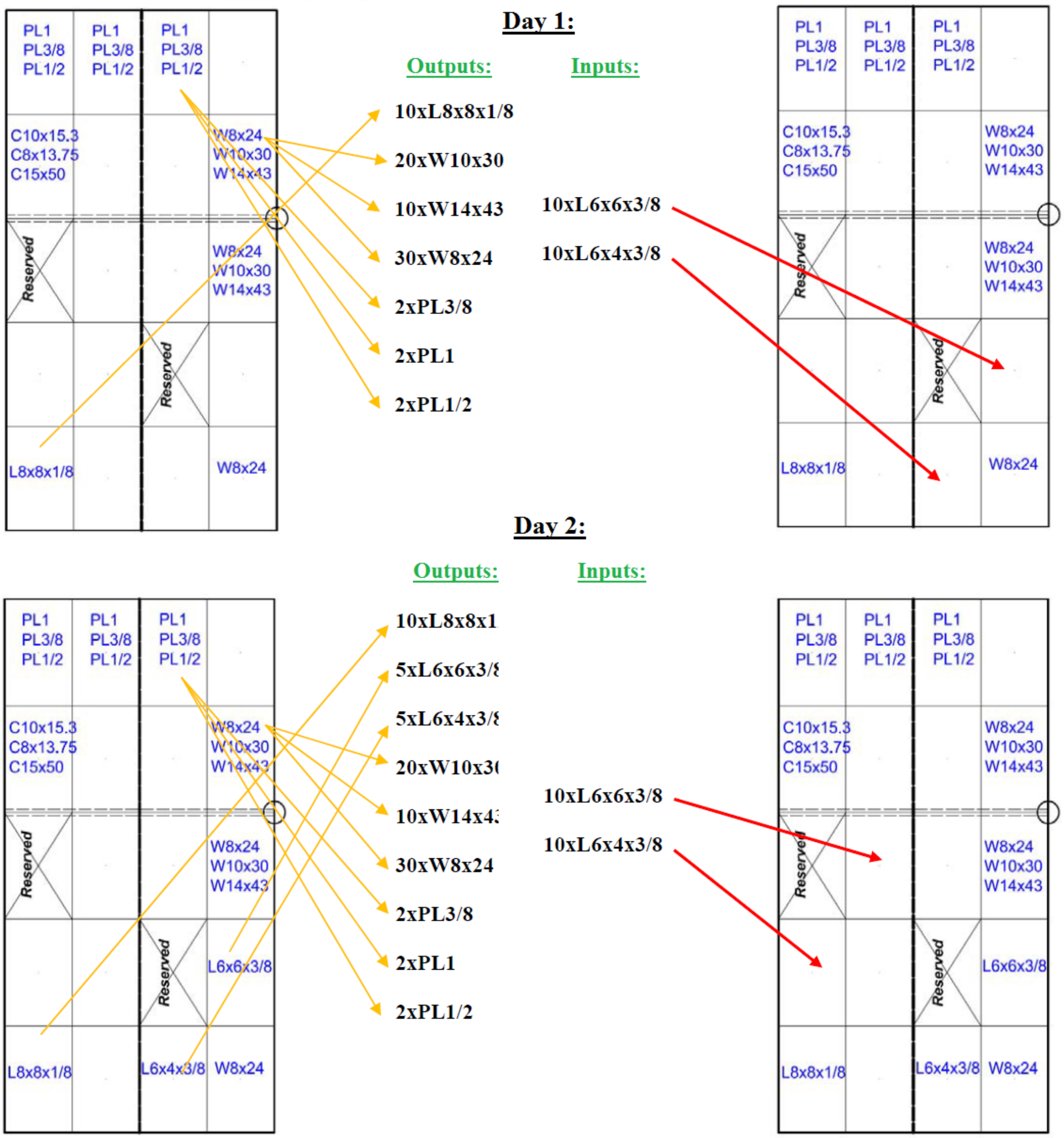


Figure 4-10. Two-day, optimum material flow on the yard

On the same day, two more batches of iron angles arrive at the yard waiting to be placed. Surprisingly, the program suggests placing them on the north yard on laydown 5 and 14. One might inquire why the program does not suggest placing the iron angles on the south yard, preferably on the same spots or closer to the exit point. Further search through the placement arrangement for all the thirty days reveals that iron angles are variably placed on laydowns 1, 3, 5, 6, 8, 14, 10, 15, 18 and 20. Of these proposed placements, laydowns 3, 8, 15 and 20 are located on the south yards and the rest are on the north yard. The placement for iron angles continues until the 10th day where there is no procurement of iron angles afterwards, due to sufficiency of the shop supply.

Table 4-3. Proposed placements for all the L6x6x3/8 and L6x4x3/8 types of iron angles

Day #	Batch #	Material type	Placement tag
1	1	10xL6x6x3/8	8
1	2	10xL6x4x3/8	3
2	3	10xL6x6x3/8	14
2	4	10xL6x4x3/8	5
3	5	10xL6x6x3/8	20
3	6	10xL6x4x3/8	15
4	7	10xL6x6x3/8	20
4	8	10xL6x4x3/8	8
5	9	10xL6x6x3/8	1
5	10	10xL6x4x3/8	5
6	11	10xL6x6x3/8	5
6	12	10xL6x4x3/8	6
7	13	10xL6x6x3/8	6
7	14	10xL6x4x3/8	1
8	15	10xL6x6x3/8	20
8	16	10xL6x4x3/8	14
9	17	10xL6x6x3/8	14
9	18	10xL6x4x3/8	10
10	20	10xL6x6x3/8	18
10	21	10xL6x4x3/8	5
Total L6x6x3/8 placement on laydown# 20:			30
Total L6x4x3/8 placement on laydown# 15:			10
Total L6x6x3/8 placement on laydown# 8:			10
Total L6x4x3/8 placement on laydown# 8:			10
Total L6x4x3/8 placement on laydown# 3:			10

Table 4-3 highlights the proposed south laydowns and summarizes the quantities of the stocked iron angles on these spots. The sums of quantities for the iron angles stocked on south laydowns are as shown at the bottom of the table. Table 4-4 on the other hands searches for the same iron angle types in the output plan proposed again by the program on the basis of closet possible laydowns to the exit point. Adding all the quantities on the same south laydown spaces (i.e. 3, 8, 20 and 15) reveals that the same amount of materials are removed from the yard by the shop leaving the previously occupied south laydowns totally empty for the W-sections, channels and plates.

Table 4-4. Proposed removal plan for all the L6x6x3/8 and L6x4x3/8 types of iron angles

Day #	Batch #	Material type	Removal tag
2	9	5xL6x6x3/8	8
2	10	5xL6x4x3/8	3
3	18	5xL6x6x3/8	8
3	19	5xL6x4x3/8	3
4	27	5xL6x6x3/8	20
4	28	5xL6x4x3/8	15
5	36	5xL6x6x3/8	20
5	37	5xL6x4x3/8	15
6	45	5xL6x6x3/8	20
6	46	5xL6x4x3/8	8
7	54	5xL6x6x3/8	20
7	55	5xL6x4x3/8	8
8	63	5xL6x6x3/8	14
8	64	5xL6x4x3/8	6
9	72	5xL6x6x3/8	20
9	73	5xL6x4x3/8	14
10	81	5xL6x6x3/8	20
10	82	5xL6x4x3/8	14
Total L6x6x3/8 take off from laydown# 20:			30
Total L6x4x3/8 take off from laydown# 15:			10
Total L6x6x3/8 take off from laydown# 8:			10
Total L6x4x3/8 take off from laydown# 8:			10
Total L6x4x3/8 take off from laydown# 3:			10

The rationale behind this is that the program discovers that a great amount of W-sections and channels are coming to the yard from the 10th day forward. As a consequence, it tries to place the iron angle based on the following principles:

- The south laydowns shall be emptied after the 10th day so that W-sections and channels, which have higher flow volumes to the yard, are placed close to the exit point. If a higher amount of materials was placed on the south laydowns, there would be iron angle left over on the south yard, preventing the channels and W-section from being placed close to the yard.
- Overall 200 pieces of L6x6x3/8 and L6x4x3/8 come to the yard and 90 pieces are to be consumed. 70 pieces of 90 pieces are taken from south laydowns and only 20 pieces are taken from the north laydown, which shows the suitability of the proposed placement for iron angles in terms of satisfying proximity criterion.
- Iron angles are not going to be used after day 10, thus it would be reasonable to stock the ones which are to be placed on the north yard as far as possible so that there would be room for other materials which may congest the yard in later days. For instance, laydown #18, which is located on the north yard, and is considerably far from the exit point, contains plates. The program waits for the day that plates are taken off from laydown # 18, and quickly places the iron angle on the 10th day on the farthest possible place.

Given the placement principles above, one can follow the placement trend for other materials and conclude similar optimum placement strategies that the solution can bring about. Integrating simulation with GA has further empowered the solution to incorporate equipment interactions, which is precisely what brings about sophistication to the placement policy.

To further highlight the significance of simulation, one can delve a little deeper in simulation capabilities in modeling the resource interaction on the yard. It would be noteworthy to determine the waiting time of the cranes for the car. Simulation can provide the total waiting time of the cranes for the car as the car is the key resource during the material haulage operation serving the two cranes. The amount of waiting time for the car may reveal how important resource interaction could be in the entire optimization process. Table 4-5 shows total simulation time of 7 previously-shown arrangements in Table 3-5. It is observed that the amount of waiting time could go in excess of 1% of the entire haulage time. This figure would be more tangible if compared with the 9% total haulage time reduction that the proposed solution has provided after 200 generations. It is also seen in this figure that as the optimization progresses from arrangement #1 to #7, the waiting time decreases significantly, which further underlines the role of simulation in optimization.

Table 4-5. Crane utilization and waiting time for the car

Arrangement #	Haulage time		North Crane		South Crane	Total(mins)	Percentage
	Hour	Minute	Waiting Time for the car (min)	Utilization	Waiting Time for the car(min)		
1	23.2781	1396.686	4.29	63%	10.3	14.59	1.04%
2	23.079	1384.74	4.41	59%	9.4	13.81	1.00%
3	22.979	1378.74	4.3	57%	8.9	13.2	0.96%
4	22.4718	1348.308	4.17	49%	7.4	11.57	0.86%
5	22.2529	1335.174	4.16	45%	6.86	11.02	0.83%
6	21.8946	1313.676	4.09	39%	6.05	10.14	0.77%
7	21.7951	1307.706	4.11	37%	5.77	9.88	0.76%

Additionally, simulation can readily present the utilization time of the cranes, which reveals how equipment resources are used during the placement operation. Table 4-5 shows utilization percentage for the north crane, proving that the optimum arrangement is the one that strives to stock the materials on the south yard as much as possible. The utilization data shows the optimization is more resource-oriented than distance-oriented, as the utilization of the north crane decreases as the fitter solutions are introduced. Smooth interaction of the south crane and the car, without having to wait for the north crane to be served, would be a good leading roadmap for optimization process. This can be further proven by separating the waiting time of the south crane from that of the north crane. It is seen that as the utilization of the north crane decreases, the waiting time of the south crane also decreases, which would ultimately lead to a more optimized arrangement. However, it is understood that once the yard is more congested, the north crane would have to be utilized to serve the materials on the north yard. The discussions given above would merely highlight the significance of the simulation in constantly evaluating the proposed arrangements (chromosomes). The merits that can be brought upon the GA optimization problems in construction cannot be simply ignored in favor of the more simplified fitness evaluation methods such as distance evaluations and/or weighted fitness calculations.

4.5. SUMMARY AND CONCLUSIONS

The concept of proactive material placement on construction stock yards was discussed and developed in this chapter. It was explained how more realistic, more efficient material placement would be if incoming and outgoing material schedules existed. This would further facilitate proactive procurement of the materials based on the consumption schedule. Furthermore, the existence of such smart, unchangeable purchasing plan would ensure continuous and consistent flow of materials to the stock yard. Having such consistency, the yard material receiver can plan the placement process for a longer period of time, which leads to a sophisticated placement policy, minimizing time and costs of material handling and warehousing.

A powerful placement optimization solution was developed to perform proactive placement on construction stock yards. The proposed solution strategy improves material handling process by directly integrating incoming and outgoing schedules of materials into the solution engine. Then, it maintains a continuous flow of information between simulation and genetic algorithm to present the most realistic, optimized placement arrangements based on the yard hard constraints. The results are detailed material placement and removal plans for material handling and consumption units of any construction company dealing with massive material flow.

Results of the analyses show clear merits of proactive material placement over the reactive strategy described in the previous chapter. Although it is understood that reactive techniques are practiced more frequently in construction stock yards due to unforeseen events, the advantages of proactive material handling would encourage decision makers to improve other pertinent processes to approach the ideals of proactive methods so as to save as much time and money as possible. Furthermore, it is seen in this chapter how simulation can model the complicated resource interactions to further approach the actual material handling processes, which contain several influential factors such as hauling equipment idleness, waiting time and capacities, material volumes and laydown locations and proximities to the exit point of the yard. Genetic algorithm cannot simply account for all these factors by mere application of weighted target functions or equivalent cost computation methods. Fitness evaluations of chromosomes are best carried out with the aid of construction simulation, which is capable of creating resource-constraint models. However, it should be mentioned that the continuous flow of information between genetic algorithm and simulation requires a sophisticated programming technique so that runtime of the solution would be still appealing to the users.

CHAPTER 5: CONCLUSIONS AND RECOMMENDATIONS

5.1.RESEARCH SUMMARY

Proactive and reactive materials placement and handling were discussed in this study. It was explained that the difference between the two methods lies in how the material handling management decides to place the material on the construction stock yard. If there is no specific plan as to how materials arrive at the yard, the foreman is faced with the problem of finding the best laydown areas to host the material batches. In order to do so, some criteria need to be taken into consideration such as proximity to the exit point to reduce the haulage time/costs, yard hard constraints including volume and consistency constraints, equipment interaction and working process, etc. With these considerations in place, the foreman would have to quickly decide what the optimized laydown areas would be out of several options/combinations. The explained placement situation could be an everyday working experience of the yard foreman/receiver to deal with. It is understood that the reactive placement policy is the direct result of not having reliable schedules for materials that arrive at the yard, and more importantly, not having the reliable and detailed consumption schedule for the duration of the project for which supply chain needs to be maintained and managed. The reason for the lack of such schedules in most construction jobs could be change orders, late issuance of owner/customer drawings, non-conformances, management policies, etc.

The proposed solution enables her/im to quickly and conveniently determine how to react with the placement decisions as the developed program takes all the aforementioned considerations into account with the aid of built-in constraint checking functions within the genetic algorithm (GA) as an optimization tool customized to suit material handling problems. Furthermore, simulation is fully integrated with GA to evaluate the fitness of the generated placement arrangements (chromosomes in GA terminology), and to maintain continuous flow of information throughout the analyses. The use of simulation as fitness evaluator could be considered an ideal case in construction optimization problems as it models the actual, resource-constraint work process which provides the actual fitness evaluation to the GA engine. GA then proceeds with gradually improving the placement layout through crossover and mutation operations.

No matter how frequently reactive policies might apply to construction yards, the policy cannot account for dynamism of material flow in and out of the yard during a construction project. Congested laydown areas are emptied and occupied frequently from day-to-day as the project progresses. A perfect laydown arrangement for one day period might be very uninteresting for the day after since the yard inventory and yard laydown spaces are not temporary resources in construction projects, and are utilized frequently during the project lifetime. An optimum laydown management is one that accounts for variation in time as the project progresses towards completion. Proactive materials placement technique addresses the aforementioned concern by taking the dynamic nature of material flow on the yard into account. However, the implementation of proactive design of placement arrangements is stipulated by the existence of incoming and outgoing material schedules informing the decision makers what the inputs and outputs to the yard are. Schedule of materials is required for the duration for which the proactive plan is to be devised. Then it would be possible to precisely determine what laydowns the material batches are to be placed on. The designated laydown areas ensure the optimum material flow on the yard, minimizing haulage time and costs, while maximizing craft labor productivity.

A sophisticated optimization computer program was developed in this study which is capable of the following:

- Modeling the yard hard constraints including consistency and volume.
- Optimization of the placement based on consumption.
- Modeling the material removal process from the yard as close as possible to actual practice.

- Integrating the incoming and outgoing schedules of materials with the optimization engine to account for the dynamism of the yard material flow.
- Improved, built-in placement verification to maintain the validity of the generated placement schemes.
- Incorporation of simulation into the optimization engine to evaluate the fitness of the generated chromosomes.

By using the developed solution in this study, each material batch would have a placement tag in advance to arriving at the yard, facilitating the material placement process for the yard foreman, and improving the material handling process for the materials management team.

5.2. RESEARCH CONTRIBUTIONS

The present study has the following contributions to the related research area.

- Change orders and late issuance of owner drawings in the steel fabrication industry may result in variability in material consumption and procurement plan. It can be concluded that materials management as a whole is directly and indirectly impacted by such non-conformances. This study was the first in its kind that was able to address one direct result of these impacts in the area of materials handling by clearly differentiating between reactive and proactive materials placement policies. The research was initiated by offering the reactive optimization which is practical and easy-to-use by the yard receiver, and continued to present comprehensive proactive materials placement policy in which dynamic material flow on the yard is explicitly modeled. Then conclusive comparisons are made to highlight the significance of having consistent, reliable and unchangeable materials input and output schedules.

- Integration of simulation and GA in the area of yard laydown management has been carried out for the first time in this study. Continuous information exchange between simulation as the heart of optimization engine, and genetic algorithm was maintained throughout the solution in this development. The application of this sophisticated combination on the area of materials handling, for steel fabrication in particular, has been implemented for the first time. The merits of using simulation were clearly discussed by comparing the results with some commonly-used fitness evaluations such as closest distance and closest weighted distance. It was shown that the resource-constraint nature of the construction processes necessitates the application of simulation if one is to obtain the most realistic results.
- Materials incoming and outgoing schedules were incorporated into the GA directly to facilitate the automatic data input into the optimization engine. These schedules are updated automatically at the end to contain extra information regarding the placement and removal of the materials on and from the yard.
- GA internal engine contains crossover and mutation operators which create offspring for the new generations. Such offspring shall comply with yard hard constraints so that new generations could be considered valid solutions. In this study, placement verifications are performed for each and every newly-generated offspring to ensure their conformance to the yard constraints. Furthermore, iteration algorithm was internally developed to ensure that offspring are generated in a one-to-one ratio with the old population.

5.3. LIMITATIONS AND DIRECTIONS FOR FUTURE RESEARCH

This study opens up a wide area of research and improvement for future work among which the following can be mentioned.

- Starting from the current development, it is paramount in evolutionary heuristic methods to reduce the runtime as much as possible. Unfortunately, the runtime of the proposed proactive solution might be uninteresting to users. The primary reasons for the slow run times and the corresponding solutions can be presented as follows:
    - Simulation is used as an external resource within C# program. This prolongs the runtime significantly as it takes time for the .Net framework to read through the external model and run it. It is strongly recommended to take advantage of open-source codes of Simphony and integrate the simulation codes within the GA engine to reduce the runtime.
    - Proactive optimization code entails the generation of long chromosomes containing the placement information of all the material batches for the duration of study. Once crossover and mutation operations are to be applied on the chromosomes, they scarcely meet the criteria of valid, yard constraint-compliance chromosomes. As discussed earlier, iteration and verification techniques are applied to the offspring to ensure their validity. This process adds to the run time significantly as thousands of new chromosomes are generated during a typical evolutionary heuristic method such as genetic algorithm. As a remedy, internal, sophisticated algorithms can be written to assist the process of offspring generation. This algorithm might apply yard 'soft constraints' such as local checking of the volumes and proximities with respect to other batches, checking consumption sequence locally, or weight attachment to some batches that are considered as 'frequently-used'. Another internal algorithm that might be applied is verification and improvement at the gene level (as opposed to chromosome level) to locally help the genes to satisfy the yard constraints.

- More advanced selection methods could be applied to avoid falling into local optima during generation enhancement. In this study, roulette selection was used, but it might not be the best selection technique. The proper choice of selection method along with mutation operator can avoid falling into local optima.
- Often times, GA is used in combination with other artificial intelligence methods such as neural network, ant colony, etc. to help GA with faster and more efficient convergence. In this study, it was attempted initially to use the weighted-distance fitness evaluator to help the simulation-GA engine acquire some ideas of the enhanced population to save runtime, but the distance of weighted-distance evaluators were proven to introduce some anomalies and inconsistencies among the consecutive generations. More efficient methods can be sought.
- This study could have been more complete if more steel sections (more material types in general) were also modeled. Moreover, different transfer lines, stochastic observations of the equipment loading and unloading times, exact modeling of inventory, exact list of input and output materials for the duration of study, etc. could be modeled to further bring this study to the actual material handling practice in industry.
- The proposed solution in this research incorporates the schedule of materials into the optimization and updates the placement tag for each incoming batch. It helps the material handling management in decision making, and provides them with useful information to supply the consumption unit efficiently and conveniently. However, it does not facilitate the purchasing department as they make their decisions based on the consumption schedule as to what materials to procure and when to procure them. An extended version of this program would not only improve the handling process, but could also optimize the material procurement plan incorporating the company's overall policies for the project, shop demand, yard inventory at the end of the project, logistics,

etc. The program can simply optimize the incoming material schedule while updating the placement tag. This proposed development should not be cumbersome as it only needs to add some more soft and hard constraints to the existing hard constraints. The problem with such optimization would be the choice of heuristic. A more advanced optimization algorithm might be more interesting as evolutionary methods inherently require long runtimes.

- The developed optimization solution is single-objective (i.e. time as the only objective for simulation and GA). Multi-objective, proactive laydown management could be the next step in development. Objectives such as cost and safety could be incorporated into the problem and addressed in a more comprehensive, multi-objective solution strategy.
- The simulation model in this study has a limitation of being deterministic. That is, the loading/unloading time of the cranes and the carts have not been inserted into the program deterministically. For the simulation model to be stochastic, more field observations needed to be conducted. Moreover, combination of stochastic simulation analysis and optimization using evolutionary methods would further complicate the problem conceptually.
- The current development addresses the yard laydown management and process improvement. Within the yard, there is no working process among the stocked batches. In site-layout management and optimization, though, working units such as staff trailer, erection yard, stockpile, laydown areas, fabrication unit and many other working units depending on the industry type interact with one another. Resources are shared among the units and processed accordingly to perform the job. Site-layout optimization is a more resource-intensive problem that requires the use of more sophisticated simulation models. Multi- or single-objective site-layout optimization problems could be the next

step of this development in which simulation and heuristic method cooperate closely to provide the most optimum solution and layout for a construction site.

- The current development could be more enhanced to create a material handling special purpose template. The template would be a user-friendly medium in which the users could conveniently generate the yard, equipment and transfer lines, while reading the input and output schedule of materials directly from some scheduling software package such as P6 or MS project. All the aforementioned future developments could be incorporated in such a special purpose template so that the template can be easily used by managers and decision makers in the industry.

REFERENCES

AbouRizk, S. M. (1990). "Input modeling for construction simulation." Ph.D. dissertation, Purdue University, West Lafayette, IN.

AbouRizk, S. M. (1998). "Simulation and modeling." *Proc., 5th Annual Canadian Construction Research Forum*, Edmonton, AB, 55-68.

Al-Tabatai, H., and Alex, A.P. (1998). "An evolutionary approach to the capital budgeting of construction projects." *Cost Engineering*, AACE, 40(10), 28-34.

Appleton, B. J. A. (2002). "Special purpose simulation for tower crane construction operations management." M.Sc. thesis, University of Alberta, Edmonton, AB.

Azadivar, F., and Wang, W. (2000). "Facility layout optimization using simulation and genetic algorithms." *International Journal of Production Research*, 38(17), 4369-4383.

Bolz, H. A. and Hagemann, G. E. (ed.). (1976). *Materials Handling Handbook*, Ronald Press.

Chan, W. T., Chua, D. K. H., and Govindan, K. (1996). "Construction resource scheduling with genetic algorithms." *Journal of Construction Engineering and Management*, ASCE, 122(2), 125-132.

Chen, P., Fu, Z., Lim, A., and Rodrigues, B. (2004). "Port yard storage optimization." *Trans Robotics Automat*, IEEE, 1, 26-37.

Coley, D. A. (1999). *An Introduction to Genetic Algorithms for Scientists and Engineers*, World Scientific Publishing Co., Singapore.

Construction Industry Institute. (1988). *Costs and Benefits of Materials Management Systems*.

Construction Industry Institute (1999). "Procurement and materials management: a guide to effective project execution." *Implementation Resource*, 7-3.

Crainic, T. G., Gendreau, M., and Dejax, P. (1993). "Dynamic and stochastic models for the allocation of empty containers." *Oper Res* 41, 102–126.

Feng W., Zhu W., and Zhou Y. (2010). "The application of genetic algorithm and neural network in construction cost estimate." *Proc., 3rd International Symposium on Electronic Commerce and Security Workshops*, Guangzhou, P.R., China, 151-155.

Feng, C. W., Liu, L., and Burns, S. A. (1997). "Using genetic algorithms to solve construction time-cost trade-cost trade-off problems." *Journal of Construction Engineering and Management*, ASCE, 11(3), 184-189.

Gambardella, L., Rizzoli A., and Zaffalon M. (1998). "Simulation and planning of an intermodal container terminal," *Simulation*, 71, 107–116.

Goldberg, D. R. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, Massachusetts.

Grau, D., Caldas, C. H., Haas, C. T., Goodrum, P. M., and Geong, J. (2009). "Assessing the impact of materials tracking technologies on construction craft productivity." *Automation in Construction*, 18, 903 – 911.

Hajjar, D., and AbouRizk S. M. (1999). "Simphony: an environment for building special purpose construction simulation tools." *Proc. Winter Simulation Conference*, Phonenix, AZ., 998-1006.

Hajjar, D. and AbouRizk, S. M. (1998). "Modeling and analysis of aggregate production operations." *Journal of Construction Engineering and Management*, ASCE, 124(5), 390-401.

Hajjar D., and AbouRizk, S. M. (1996). "Building a special purpose simulation tool for earth moving operations." *Proc. Winter Simulation Conference*, San Diego, CA, 1313-1320.

Hajjar D., AbouRizk, S. M. and Zu, J. (1998). "Construction site dewatering analysis using a special purpose simulation-based framework." *Canadian Journal of Civil Engineering*, 25, 819-828.

Halpin D.W. (1977). "CYCLONE-method for modeling job site processes." *Journal of Construction Division*, ASCE, 103(3), 489-499.

Han, Y., Lee, L. H., Chew, E. P., and Tan, K. C. T. (2008). "A yard storage strategy for minimizing traffic congestion in a marine container transshipment hub." *OR Spectrum* 30, 697–720, DOI 10.1007/s00291-008-0127-6.

Hegazy, T., and Elbeltagi, E. (1999). "EvoSite: evolution-based model for site layout planning." *Journal of Computing in Civil Engineering*, 13(3), 198-206.

Hegazy, T., Elhakeem, A., and Elbeltagi E. (2004). "Distributed scheduling model for infrastructure networks." *Journal of Construction Engineering and Management*, ASCE, 130(2), 160-167.

Hegazy, T., and Petzold, K. (2003). “Genetic optimization for dynamic project control.” *Journal of Construction Engineering and Management*, ASCE, 129(4), 396-404.

Holland, J.H. (1975). *Adaptation in Natural and Artificial systems*, The University of Michigan Press, Ann Arbor, MI.

Kemme, N. (2011). “Effects of storage block layout and automated yard crane systems on the performance of seaport container terminals.” *OR Spectrum*, 34, 563–591, DOI 10.1007/s00291-011-0242-7.

Kim J. (2009). “Improved genetic algorithm for resource-constrained scheduling of large projects.” *Canadian Journal of Civil Engineering*, 36(6), 1016-1027, DOI 10.1139/L09-049.

Kim J. (2010). “Integrated genetic algorithm and its applications for construction resource optimization.” *Proc. Winter Simulation Conference*, 121-125.

Kim, G. H., Yoon, J. E. An, S. H., Cho, H. H. and Kang, K. I. (2004). “Neural network model incorporating a genetic algorithm in estimating construction costs.” *Building and Environment*, 39 (11), 1333-1340.

Kim, B., Koo, J., Park, B. (2009). “A raw material storage yard allocation problem for a large-scale steelworks.” *Int J Adv Manuf Technol* 41, 880–884, DOI 10.1007/s00170-008-1538-x.

Koing, M., Habenicht, I., and Speickermann, S. (2011). On-site logistics simulation in early planning phases. *Proc. 11th International Conference on Construction application of Virtual Reality*, Weimar, Germany.

Lee, H. Y. (2012). “Integration simulation and ant colony optimization to improve the service facility layout in a station.” *Journal of Computing in Civil Engineering*, 26(2), 259-269.

Lee, H. L., Chew, P. E., Tan, K. C., and Han, Y. (2006). “An optimization model for storage yard management in transshipment hubs.” *OR Spectrum* 28, 539–561, DOI 10.1007/s00291-006-0045-4.

Leu, S. S., Chen, A. T., and Yang C. H. (2001). “A GA-based fuzzy optimal model for construction time-cost trade-off.” *International Journal of Project Management*, 19(1), 47-58.

Manuals Materials Handling, (2008). *A Health and Safety Guideline for Your Workplace,* Industrial Accident Prevention Association.

Marisini, R., Dawood, D., and Hobbs B. (2001). “Stockyard layout planning in precast concrete products industry: a case study and proposed framework.” *Construction Management and Economics*, 19(4), 365-377.

Martinez, J. C. and Ioannou, P. G. (1994). “General purpose simulation with stroboscope.” *Proc. Winter Simulation Conference,* Orlando, FL, 1159-1166.

Martinez, J. C. (1998). “General-purpose systems for effective construction simulation.” *Journal of Construction Engineering and Management*, ASCE, 125(4), 265-276.

Mitchell, M. (1997). *An Introduction to Genetic Algorithm,* The MIT Press, Cambridge, London.

Odeh, A. M. (1992). “CIPROS: knowledge-based construction integrated project and process planning simulation system.” Ph.D. dissertation, University of Michigan, Ann Arbor, MI.

Paulson, B. C. Jr., Chan W. T., and Koo, C. C. (1987). “Construction operation simulation by microcomputer.” *Journal of Construction Engineering and Management*, ASCE, 113(2), 302-314.

Peurifoy R., and Oberlender G. (2002). *Estimating Construction Costs*, McGraw-Hill.

Pritsker, A. A. B. (1986). *Introduction to Simulation and SLAM II*, 3rd Ed., Halsted Press, John Wiley & Sons, Inc., New York, NY.

Ray, J. P., Chanev T. (1998). “Simulation optimization using a genetic algorithm.” *Simulation Practice and Theory*, 601-611.

Reeves, C. R., and Rowe, J. E. (2002). *Genetic Algorithms - Principles and Perspectives: A Guide to GA Theory*, Kluwer Academic Publishers, New York.

Sawhney, A., AbouRizk, S. M., and Haplin, D. W. (1998). “Construction project simulation using CYCLONE.” *Canadian Journal of Civil Engineering*, 25, 16-25.

Senouci, A. B., and Eldin, N. N. (2004). “Use of genetic algorithm in resource scheduling of construction projects.” *Journal of Construction Engineering and Management*, ASCE, 130(6), 869-877.

Shen, W. S., and Khoong, C. M. (1995). “A DSS for empty container distribution planning.” *Decis Support Syst* 15, 75–82

Siddhartha, R., (2007). *Introduction to Materials Handling,* New Age International Publication, Kolkata, W.B.

Smutkupt, U., and Wimonkasame, S. (2009). “Plant layout design with simulation.” *Proc. International MultiConference of Engineering and Computer Scientists*, Hong Kong.

Tawfik, H., and Fernando, T. (2001). “A simulation environment for construction site planning.” *Proc. 5th International Conference of Information Visualization*, London, UK.

Toklu, Y. (2002). “Application of genetic algorithms to construction scheduling with or without resource constraints.” *Canadian Journal of Civil Engineering*, 29(3), 421-429, DOI 10.1139/l02-034.

Tommelein, I. D. (1999). Travel-time simulation to locate and staff temporary facilities under changing construction demand. *Proc. Winter Simulation Conference*, Phoenix, AZ, 978-984.

Tommelein, I.D., (1994), “Materials handling and site layout control.” *Automation and Robotics in Construction XI*, 297-304.

Ugwa O. O. and Tah J. H. M. (1994). “Development and application of a hybrid genetic algorithm for resource optimization and management.” *Engineering, Construction and Architectural Management*, 9(4), 304 – 317.

Voigtmann, J., and Bargstadt, H. (2009). “Simulation of construction logistics in outfitting processes.” In, *eWork and eBusiness in architecture, Engineering and Construction*, A. Zarli & R. Scherer, eds., London, UK: Taylor& Francis Group, 195-203.

Waiward Steel Fabricators Ltd. www.waiward.com. (April, 2013).

Wenzel, S., Boyaci, P., and Jessen, U. (2010). “Simulation in production and logistics: trends, solutions and applications.” *Advanced Manufacturing and Sustainable Logistics Lecture Notes in Business Information Processing*, 46(1), 73-84.

Wing, R., (2006). “RFID applications in construction and facilities management.” *Journal of IT in Facility Management*, 11, 711-721.

Zhang, C., Liu, J., Wan, Y. W., Murty, K. G., and Linn, R. J. (2003). “Storage space allocation in container terminals.” *Transp Res B* 37, 883–903.

Zheng, D. X. M., Thomas, N. S., and Kumaraswamy M. M. (2003). “GA-based multiobjective technique for multi-resource leveling.” *Proc. Construction Research Congress*, ASCE.

Zhou, F., (2006). “An integrated framework for tunnel shaft construction and site layout optimization.” M.Sc. thesis, University of Alberta, Edmonton, AB.

Zhou, F., AbouRizk, S. M., and Al-Battaineh, H. (2009). “Optimisation of construction site layout using a hybrid simulation-based system.” *Simulation Modelling Practice and Theory*, 17, 348-363.

Zouein, P. P., and Tommelein, I. D. (1994). “Automating dynamic layout construction.” *Automation and Robotics in Construction XI*, 409-416.

Appendix 1: Removal arrangements for case study in chapter 4

There are 271 outgoing batches which results in a table having 271 rows as shown below:

		Chromosome fitness (material batch haulage time to the exit point)						
		23.27814	23.07908	22.97955	22.45199	22.25293	21.89462	21.79509
Day No.	Batch No.	Yard laydown number (where the materials are to be taken from)						
1	1	1	1	1	1	1	1	1
1	2	16	16	16	16	16	16	16
1	3	16	16	16	16	16	16	16
1	4	16	16	16	16	16	16	16
1	5	19	19	19	19	19	19	19
1	6	19	19	19	19	19	19	19
1	7	19	19	19	19	19	19	19
2	8	1	1	1	1	1	1	1
2	9	2	6	14	11	1	5	8
2	10	10	3	6	14	10	2	3
2	11	16	16	16	16	16	16	16
2	12	16	16	16	16	16	16	16
2	13	16	16	16	16	16	16	16
2	14	19	19	19	19	19	19	19
2	15	19	19	19	19	19	19	19
2	16	19	19	19	19	19	19	19
3	17	1	1	1	1	1	1	1
3	18	10	8	14	11	14	20	8
3	19	14	8	10	20	11	8	3
3	20	16	16	16	16	16	16	16
3	21	16	16	16	16	16	16	16
3	22	16	16	16	16	16	16	16
3	23	19	19	19	19	19	19	19
3	24	19	19	19	19	19	19	19
3	25	19	19	19	19	19	19	19
4	26	1	1	1	1	1	1	1
4	27	10	8	6	11	20	20	20
4	28	14	20	15	20	11	8	15
4	29	16	16	16	16	16	16	16
4	30	16	16	16	16	16	16	16
4	31	16	16	16	16	16	16	16
4	32	19	19	19	19	19	19	19
4	33	19	19	19	19	19	19	19
4	34	19	19	19	19	19	19	19
5	35	1	1	1	1	1	1	1
5	36	5	15	11	11	11	14	20
5	37	20	11	15	14	8	5	15
5	38	16	16	16	16	16	16	16
5	39	16	16	16	16	16	16	16
5	40	16	16	16	16	16	16	16
5	41	19	19	19	19	19	19	19
5	42	19	19	19	19	19	19	19
5	43	19	19	19	19	19	19	19
6	44	1	1	1	1	1	1	1
6	45	20	15	11	10	11	20	20
6	46	20	11	8	8	8	10	8
6	47	16	16	16	16	16	16	16
6	48	12	12	12	12	12	12	12
6	49	16	16	16	16	16	16	16
6	50	18	18	18	18	18	18	18
6	51	18	18	18	18	18	18	18
6	52	18	18	18	18	18	18	18
7	53	1	1	1	1	1	1	1
7	54	20	11	6	8	20	20	20
7	55	10	20	8	8	10	10	8
7	56	16	16	16	16	16	16	16
7	57	12	12	12	12	12	12	12
7	58	16	16	16	16	16	16	16
7	59	18	18	18	18	18	18	18
7	60	18	18	18	18	18	18	18
7	61	18	18	18	18	18	18	18
8	62	1	1	1	1	1	1	1
8	63	14	11	20	15	14	15	14
8	64	10	15	10	14	20	10	6
8	65	12	12	12	12	12	12	12
8	66	12	12	12	12	12	12	12

		Chromosome fitness (material batch haulage time to the exit point)						
		23.27814	23.07908	22.97955	22.45199	22.25293	21.89462	21.79509
Day No.	Batch No.	Yard laydown number (where the materials are to be taken from)						
8	67	16	16	16	16	16	16	16
8	68	18	18	18	18	18	18	18
8	69	18	18	18	18	18	18	18
8	70	18	18	18	18	18	18	18
9	71	1	1	1	1	1	1	1
9	72	14	20	20	15	14	15	20
9	73	10	15	19	14	20	11	14
9	74	12	12	12	12	12	12	12
9	75	12	12	12	12	12	12	12
9	76	16	16	16	16	16	16	16
9	77	18	18	18	18	18	18	18
9	78	18	18	18	18	18	18	18
9	79	18	18	18	18	18	18	18
10	80	1	1	1	1	1	1	1
10	81	6	20	8	15	8	14	20
10	82	10	15	19	20	19	11	14
10	83	12	12	12	12	12	12	12
10	84	12	12	12	12	12	12	12
10	85	16	16	16	16	16	16	16
10	86	18	18	18	18	18	18	18
10	87	18	18	18	18	18	18	18
10	88	18	18	18	18	18	18	18
10	89	13	13	13	13	13	13	13
10	90	13	13	13	13	13	13	13
10	91	13	13	13	13	13	13	13
11	92	12	12	12	12	12	12	12
11	93	4	12	10	3	4	12	2
11	94	12	12	12	12	12	16	16
11	95	17	17	17	17	17	17	17
11	96	17	17	17	17	17	17	17
11	97	17	17	17	17	17	17	17
11	98	13	13	13	13	13	13	13
11	99	13	13	13	13	13	13	13
11	100	13	13	13	13	13	13	13
12	101	12	12	12	12	12	12	12
12	102	4	12	10	3	4	12	2
12	103	12	12	12	12	16	12	12
12	104	17	17	17	17	17	17	17
12	105	17	17	17	17	17	17	17
12	106	17	17	17	17	17	17	17
12	107	13	13	13	13	13	13	13
12	108	13	13	13	13	13	13	13
12	109	13	13	13	13	13	13	13
13	110	12	12	12	12	12	12	12
13	111	4	12	10	3	4	12	2
13	112	12	12	16	12	12	12	16
13	113	17	17	17	17	17	17	17
13	114	17	17	17	17	17	17	17
13	115	17	17	17	17	17	17	17
13	116	13	13	13	13	13	13	13
13	117	13	13	13	13	13	13	13
13	118	13	13	13	13	13	13	13
14	119	12	12	12	12	12	12	12
14	120	14	12	10	11	12	12	12
14	121	15	18	12	16	12	12	12
14	122	17	17	17	17	17	17	17
14	123	17	17	17	17	17	17	17
14	124	17	17	17	17	17	17	17
14	125	13	13	13	13	13	13	13
14	126	13	13	13	13	13	13	13
14	127	13	13	13	13	13	13	13
15	128	12	12	12	12	12	12	12
15	129	14	12	10	11	12	12	12
15	130	12	18	15	12	16	12	12
15	131	17	17	17	17	17	17	17
15	132	17	17	17	17	17	17	17
15	133	17	17	17	17	17	17	17
15	134	13	13	13	13	13	13	13
15	135	13	13	13	13	13	13	13

		Chromosome fitness (material batch haulage time to the exit point)						
		23.27814	23.07908	22.97955	22.45199	22.25293	21.89462	21.79509
Day No.	Batch No.	Yard laydown number (where the materials are to be taken from)						
15	**136**	13	13	13	13	13	13	13
16	**137**	12	12	12	12	12	12	12
16	**138**	14	12	10	11	12	12	12
16	**139**	12	4	10	11	16	12	15
16	**140**	17	17	17	17	17	17	17
16	**141**	17	17	17	17	17	17	17
16	**142**	17	17	17	17	17	17	17
16	**143**	13	14	13	13	15	20	13
16	**144**	13	13	11	19	13	20	8
16	**145**	18	14	13	13	15	19	19
17	**146**	12	12	12	12	12	12	12
17	**147**	14	12	10	11	12	12	12
17	**148**	11	10	10	4	11	16	12
17	**149**	17	17	17	17	17	17	17
17	**150**	17	17	17	17	17	17	17
17	**151**	17	17	17	17	17	17	17
17	**152**	13	14	13	13	15	20	13
17	**153**	13	13	11	19	13	20	8
17	**154**	18	14	13	13	15	19	19
18	**155**	12	12	12	12	12	12	12
18	**156**	14	12	10	11	12	12	12
18	**157**	20	12	10	11	12	16	15
18	**158**	17	17	17	17	17	17	17
18	**159**	17	17	17	17	17	17	17
18	**160**	17	17	17	17	17	17	17
18	**161**	13	14	13	13	15	20	13
18	**162**	13	13	11	19	13	20	8
18	**163**	18	14	13	13	15	19	19
19	**164**	12	12	12	12	12	12	12
19	**165**	11	16	10	11	12	12	12
19	**166**	16	10	10	16	18	16	4
19	**167**	17	17	17	17	17	17	17
19	**168**	17	17	17	17	17	17	17
19	**169**	17	17	17	17	17	17	17
19	**170**	13	14	13	13	15	20	13
19	**171**	13	13	11	19	13	20	8
19	**172**	18	14	13	13	15	19	19
20	**173**	12	12	12	12	12	12	12
20	**174**	11	16	10	11	12	12	12
20	**175**	20	16	10	16	4	12	20
20	**176**	17	17	17	17	17	17	17
20	**177**	17	17	17	17	17	17	17
20	**178**	17	17	17	17	17	17	17
20	**179**	13	14	13	13	15	20	13
20	**180**	13	13	11	19	13	20	8
20	**181**	18	14	13	13	15	19	19
21	**182**	12	16	12	12	12	12	12
21	**183**	11	16	4	11	12	12	12
21	**184**	16	18	11	4	12	15	4
21	**185**	17	17	17	17	17	17	17
21	**186**	17	17	17	17	17	17	11
21	**187**	17	17	17	17	17	17	11
21	**188**	18	13	19	19	15	19	19
21	**189**	13	14	13	19	20	20	8
21	**190**	13	11	13	13	20	11	19
22	**191**	12	16	12	12	12	12	12
22	**192**	11	16	4	11	12	12	12
22	**193**	15	10	11	12	11	12	4
22	**194**	17	17	17	17	17	17	17
22	**195**	17	17	17	17	17	17	11
22	**196**	17	17	17	17	17	17	11
22	**197**	18	13	19	19	15	19	19
22	**198**	13	14	13	19	20	20	8
22	**199**	13	11	13	13	20	11	19
23	**200**	12	12	12	12	12	12	12
23	**201**	11	16	4	11	12	12	12
23	**202**	8	10	12	4	4	12	12
23	**203**	17	17	17	17	17	17	17
23	**204**	17	17	17	17	17	17	11

		Chromosome fitness (material batch haulage time to the exit point)						
		23.27814	23.07908	22.97955	22.45199	22.25293	21.89462	21.79509
Day No.	**Batch No.**	Yard laydown number (where the materials are to be taken from)						
23	**205**	17	17	17	17	17	17	11
23	**206**	18	13	19	19	15	19	19
23	**207**	13	14	13	19	20	20	8
23	**208**	13	11	13	13	20	11	19
24	**209**	12	16	12	12	12	12	12
24	**210**	12	12	16	11	11	12	15
24	**211**	8	4	11	16	4	4	16
24	**212**	17	17	17	17	17	17	17
24	**213**	17	17	17	17	17	17	11
24	**214**	17	17	17	17	17	17	11
24	**215**	18	13	19	19	15	19	19
24	**216**	13	14	13	19	20	20	8
24	**217**	13	11	13	13	20	11	19
25	**218**	12	12	12	16	11	15	20
25	**219**	12	12	16	11	11	12	15
25	**220**	4	18	15	16	4	12	4
25	**221**	17	17	17	17	17	17	17
25	**222**	17	17	17	17	17	17	11
25	**223**	17	17	17	17	17	17	11
25	**224**	18	13	19	19	15	19	19
25	**225**	13	14	13	19	20	20	8
25	**226**	13	11	13	13	20	11	19
26	**227**	12	12	12	16	11	15	4
26	**228**	12	10	16	11	11	15	15
26	**229**	20	18	15	4	12	12	4
26	**230**	17	14	17	17	20	17	11
26	**231**	17	17	17	17	17	17	17
26	**232**	17	14	17	17	20	19	11
26	**233**	18	13	19	13	15	11	19
26	**234**	13	13	19	19	13	11	8
26	**235**	13	13	13	19	13	20	8
27	**236**	12	12	11	12	11	15	3
27	**237**	12	10	16	11	11	15	15
27	**238**	15	12	18	12	12	4	4
27	**239**	17	14	17	17	20	17	11
27	**240**	17	17	17	17	17	17	17
27	**241**	17	14	17	17	20	19	11
27	**242**	18	13	19	13	15	11	19
27	**243**	13	13	19	19	13	11	8
27	**244**	13	13	13	19	13	20	8
28	**245**	12	10	11	12	11	4	3
28	**246**	12	10	16	11	11	15	15
28	**247**	14	4	10	11	12	12	4
28	**248**	17	11	17	17	20	19	17
28	**249**	17	11	17	17	17	19	11
28	**250**	17	14	17	17	20	19	11
28	**251**	18	13	19	13	15	11	19
28	**252**	13	13	19	19	13	11	8
28	**253**	13	13	13	19	13	20	8
29	**254**	11	10	11	12	11	3	3
29	**255**	14	10	4	4	11	15	20
29	**256**	20	18	11	12	18	4	4
29	**257**	17	11	17	17	20	19	11
29	**258**	17	11	17	17	17	19	11
29	**259**	17	14	17	17	20	19	11
29	**260**	18	13	19	13	15	11	19
29	**261**	13	13	19	19	13	11	8
29	**262**	13	13	13	19	13	20	8
30	**263**	8	10	10	12	18	3	2
30	**264**	14	10	4	4	11	15	20
30	**265**	4	10	10	16	16	4	4
30	**266**	17	11	17	17	20	19	11
30	**267**	17	11	17	17	17	19	11
30	**268**	17	14	17	17	20	19	11
30	**269**	18	13	19	13	15	11	19
30	**270**	13	13	19	19	13	11	8
30	**271**	13	13	13	13	13	13	13

Appendix 2: C# program- (Main Functions)

Class Material Library:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace BasicGA
{
    public class MaterialLibrary
    {
        #region Constructors
        /// <summary>
        /// This constructor is used to simply create new instances of materials
and their corresponding quantity only for any class
        /// </summary>
        /// <param name="TypeOfMaterial">Represents the type of the material. It
could be W, C, L and PL</param>
        /// <param name="qty">Quantity of the material</param>
        /// <param name="length">length of the material in foot</param>
        /// <param name="sectionProperties">Represents section properties. User
could add up to three sizes</param>
        public MaterialLibrary(string TypeOfMaterial, int qty, double length,
params string[] sectionProperties)
        {
            this.type = TypeOfMaterial;
            this.quantity = qty;
            this.length = length;
            for (int i = 0; i < sectionProperties.Length; i++)
                this.SecProperties.Add(sectionProperties[i]);

            this.matArea.Add("W8x24", 7.08);
            this.matArea.Add("W10x30", 8.84);
            this.matArea.Add("W14x43", 12.6);
            this.matArea.Add("L6x4x3/8", 3.61);
            this.matArea.Add("L6x6x3/8", 4.38);
            this.matArea.Add("L8x8x1/8", 16.8);
            this.matArea.Add("C10x15.3", 4.48);
            this.matArea.Add("C8x13.75", 4.04);
            this.matArea.Add("C15x50", 14.7);
            this.matArea.Add("PL3/8", 90);
            this.matArea.Add("PL1", 480);
            this.matArea.Add("PL1/2", 240);

        }

        public MaterialLibrary(string TypeOfMaterial, int qty, double length,
List<string> sectionProperties)
        {
            this.type = TypeOfMaterial;
            this.quantity = qty;
            this.length = length;
            for (int i = 0; i < sectionProperties.Count; i++)
                this.SecProperties.Add(sectionProperties[i]);

            this.matArea.Add("W8x24", 7.08);
            this.matArea.Add("W10x30", 8.84);
            this.matArea.Add("W14x43", 12.6);
            this.matArea.Add("L6x4x3/8", 3.61);
            this.matArea.Add("L6x6x3/8", 4.38);
            this.matArea.Add("L8x8x1/8", 16.8);
            this.matArea.Add("C10x15.3", 4.48);
            this.matArea.Add("C8x13.75", 4.04);
            this.matArea.Add("C15x50", 14.7);
            this.matArea.Add("PL3/8", 90);
            this.matArea.Add("PL1", 480);
            this.matArea.Add("PL1/2", 240);
```

```
        }


        public MaterialLibrary()
        {
            this.type = "";
            this.quantity = 0;
            this.length = 0.0;
            this.SecProperties.Clear();
            //this.SecProperties.Add("");
            //this.SecProperties.Add("");
            //this.SecProperties.Add("");
        }
        #endregion

        #region  fields
        //private string a1, a2, a3; //3 parameters defining the section type
        public List<string> SecProperties = new List<string>();
        public string type;// type of the material, could be W, C, L, PL
        public int quantity;//number of the material
        public double length;
        #endregion

        #region dictionaries
        Dictionary<string, double> matArea = new Dictionary<string,
double>();//Contains the area of the standard sections

        public double matVolume { get; set; }
        #endregion

        #region public methods

        public MaterialLibrary Clone()
        {
            return (MaterialLibrary)this.MemberwiseClone();
        }


        public double GetVolume(string Type, double length)
        {
            matVolume = (matArea[Type] * length / (12 * 12)) * this.quantity;

            return matVolume;
        }

        public override string ToString()
        {
            switch (this.type)
            {
                case "W":
                    return string.Format("W{0}x{1}", this.SecProperties[0],
this.SecProperties[1]);
                case "L":
                    return string.Format("L{0}x{1}x{2}", this.SecProperties[0],
this.SecProperties[1], this.SecProperties[2]);
                case "C":
                    return string.Format("C{0}x{1}", this.SecProperties[0],
this.SecProperties[1]);
                case "PL":
                    return string.Format("PL{0}", this.SecProperties[0]);
            }
            return base.ToString();
        }

        public void PrintoutMaterial()
        {

            if (this.quantity == 0)
            {
                Console.WriteLine("this segment does not contain any materials");
```

```
            }
            else
            {

                switch (this.SecProperties.Count)
                {
                    case (1):

                        Console.WriteLine("Type: {0}x{1}, Quantity: {2}, Length:
{3}", this.type, this.SecProperties[0], this.quantity, this.length);
                        break;
                    case (2):
                        Console.WriteLine("Type: {0}x{1}x{2}, Quantity: {3},
Length: {4}", this.type, this.SecProperties[0], this.SecProperties[1],
this.quantity, this.length);
                        break;
                    case (3):
                        Console.WriteLine("Type: {0}x{1}x{2}x{3}, Quantity: {4},
Length: {5}", this.type, this.SecProperties[0], this.SecProperties[1],
this.SecProperties[2], this.quantity, this.length);

                        break;
                }


            }

        }
        #endregion
    }
}
```

Class Chromosome:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Simphony;
using System.IO;

namespace BasicGA
{
    public class Chromosome
    {

        static Random _random=new Random();

        #region public properties

        public static double MutationRate { get; set; }
        public double Fitness { get; set; }
        public int Length { get; set; }
        public int[] Genes { get; private set; }
        public List<OutgoingBatch> OutGoing = new List<OutgoingBatch>();

        #endregion


        #region Constructor
        public Chromosome(int length, List<IncomingBatch> incoming,
List<OutgoingBatch> outgoing)
        {
            Length = length;
            Genes = new int[length];
            //CreateGenes();
        }

        public Chromosome(int length, bool createGenes, List<IncomingBatch>
incoming, List<OutgoingBatch> outgoing)
```

```
        {

            foreach (var m in outgoing)
                this.OutGoing.Add(m.DeepClone());
            Length = length;
            Genes = new int[length];
            if (createGenes)
                CreateGenes( incoming,outgoing);
        }
        public Chromosome(int length)
        {

            Length = length;
            Genes = new int[length];

        }


        public Chromosome(ref int[] genes, List<IncomingBatch> incoming,
List<OutgoingBatch> outgoing)
        {
            Length = genes.GetLength(0);
            Genes = new int[Length];
            for (int i = 0; i < Length; i++)
                Genes[i] = genes[i];
        }
        #endregion

        #region private methods

        private List<YardSpace> ThirtydayInventory(List<YardSpace>
UpdatedInventory,int dateTakeOff, bool placement)
        {
            //***making a copy of the yard inventory for the first day***
            List<YardSpace> ArtificialYard = new List<YardSpace>();
            foreach (var a in UpdatedInventory)
                ArtificialYard.Add(a.DeepClone());


            Dictionary<int, double> DistanceFinder = new Dictionary<int,
double>();
            double XexitPoint = 260.0;
            double YexitPoint = 300.0;

            /// key=which material
            /// value=from which cell
            Dictionary<int, int> FromWhereTakeOff = new Dictionary<int, int>();

            int date = dateTakeOff;
            foreach (var m in this.OutGoing)
            {
                if (m.date == date)
                {
                    foreach (var n in ArtificialYard)
                    {
                        foreach (var o in n.MaterialsOntheYard)
                        {
                            if (m.TypeandQtyandLength.ToString() == o.ToString())
                            {
                                if (m.TypeandQtyandLength.quantity <= o.quantity)
                                    DistanceFinder.Add(n.ID,
this.GetDistance(XexitPoint, n.X, YexitPoint, n.Y));
                            }
                        }
                    }

                    var minPair = DistanceFinder.Aggregate((p1, p2) => (p1.Value <
p2.Value) ? p1 : p2);

                    FromWhereTakeOff.Add(m.ID, minPair.Key);
                    //m.segmentTag = FromWhereTakeOff[m.ID];
```

```
                    //*******************************************************
                    //         Go and take the material off the yard
                    //*******************************************************


                    foreach (var p in ArtificialYard[FromWhereTakeOff[m.ID] - 
1].MaterialsOntheYard)
                    {
                        if (m.TypeandQtyandLength.ToString() == p.ToString())
                        {
                            p.quantity = p.quantity - 
m.TypeandQtyandLength.quantity;
                            //p.quantity = p.quantity - 1;
                            //tw1.WriteLine("*******");
                        }
                    }
                }
                else if (m.date > date)
                {
                    FromWhereTakeOff.Clear();
                    return ArtificialYard;
                }

                DistanceFinder.Clear();
                //tw.WriteLine("Material no {0} on the day {1} will be taken from 
cell # {2}", m.ID, m.date, FromWhereTakeOff[m.ID]);
            }
             return ArtificialYard;
        }
        private List<YardSpace> ThirtydayInventory(List<YardSpace> 
UpdatedInventory, int dateTakeOff)
        {


            //***making a copy of the yard inventory for the first day***
            List<YardSpace> ArtificialYard = new List<YardSpace>();
            foreach (var a in UpdatedInventory)
                ArtificialYard.Add(a.DeepClone());


            Dictionary<int, double> DistanceFinder = new Dictionary<int, 
double>();
            double XexitPoint = 260.0;
            double YexitPoint = 300.0;

            /// key=which material
            /// value=from which cell
            Dictionary<int, int> FromWhereTakeOff = new Dictionary<int, int>();

            int date = dateTakeOff;
            foreach (var m in this.OutGoing)
            {
                if(m.date==date)
                {
                    foreach (var n in ArtificialYard)
                    {
                        foreach (var o in n.MaterialsOntheYard)
                        {
                            if (m.TypeandQtyandLength.ToString() == o.ToString())
                            {
                                if(m.TypeandQtyandLength.quantity<=o.quantity)
                                    DistanceFinder.Add(n.ID, 
this.GetDistance(XexitPoint, n.X, YexitPoint, n.Y));
                            }
                        }
                    }

                    var minPair = DistanceFinder.Aggregate((p1, p2) => (p1.Value < 
p2.Value) ? p1 : p2);

                    FromWhereTakeOff.Add(m.ID, minPair.Key);
```

```
                    m.segmentTag = FromWhereTakeOff[m.ID];
                    //*****************************************************
                    //         Go and take the material off the yard
                    //*****************************************************


                    foreach (var p in ArtificialYard[FromWhereTakeOff[m.ID] -
1].MaterialsOntheYard)
                    {
                        if (m.TypeandQtyandLength.ToString() == p.ToString())
                        {
                            p.quantity = p.quantity -
m.TypeandQtyandLength.quantity;
                            //p.quantity = p.quantity - 1;
                            //tw1.WriteLine("*******");
                        }
                    }
                }
                else if (m.date > date)
                {

                    return ArtificialYard;
                }

                DistanceFinder.Clear();
                //tw.WriteLine("Material no {0} on the day {1} will be taken from
cell # {2}", m.ID, m.date, FromWhereTakeOff[m.ID]);

            }


            return ArtificialYard;
        }

        private void CreateGenes(List<IncomingBatch> IncomingMaterials,
List<OutgoingBatch> OutgoingMaterials)
        {

            List<YardSpace> YardInventory = new List<YardSpace>(); ;

            #region Initializing the yard

            //YardSpace A12 = new YardSpace(227.5, 250, 12, 3360.0, true, new
MaterialLibrary("W", 102, 60.0, "8", "24"), new MaterialLibrary("W", 400, 60,
"10", "30"), new MaterialLibrary("W", 400, 35, "14", "43"));
            YardSpace A1 = new YardSpace(32.5, 50, 1, 3360.0, true, new
MaterialLibrary("L", 215, 40.0, "8", "8", "1/8"));
            YardSpace A2 = new YardSpace(97.5, 50, 2, 3360.0, true, new
MaterialLibrary());
            YardSpace A3 = new YardSpace(162.5, 50, 3, 3360.0, true, new
MaterialLibrary());
            YardSpace A4 = new YardSpace(227.5, 50, 4, 3360.0, true, new
MaterialLibrary("W", 170, 60.0, "8", "24"));
            YardSpace A5 = new YardSpace(32.5, 150, 5, 3360.0, true, new
MaterialLibrary());
            YardSpace A6 = new YardSpace(97.5, 150, 6, 3360.0, true, new
MaterialLibrary());
            YardSpace A7 = new YardSpace(162.5, 150, 7, 3360, false);
            YardSpace A8 = new YardSpace(227.5, 150, 8, 3360.0, true, new
MaterialLibrary());
            YardSpace A9 = new YardSpace(32.5, 250, 9, 3360.0, false);
            YardSpace A10 = new YardSpace(97.5, 250, 10, 3360.0, true, new
MaterialLibrary());
            YardSpace A11 = new YardSpace(162.5, 250, 11, 3360.0, true, new
MaterialLibrary());
            YardSpace A12 = new YardSpace(227.5, 250, 12, 3360.0, true, new
MaterialLibrary("W", 102, 60.0, "8", "24"), new MaterialLibrary("W", 400, 60,
"10", "30"), new MaterialLibrary("W", 50, 35, "14", "43"));
            YardSpace A13 = new YardSpace(32.5, 350, 13, 3360.0, true, new
MaterialLibrary("C", 100, 60.0, "10", "15.3"), new MaterialLibrary("C", 100, 40,
"8", "13.75"), new MaterialLibrary("C", 100, 50, "15", "50"));
```

```
            YardSpace A14 = new YardSpace(97.5, 350, 14, 3360.0, true, new
MaterialLibrary());
            YardSpace A15 = new YardSpace(162.5, 350, 15, 3360.0, true, new
MaterialLibrary());
            YardSpace A16 = new YardSpace(227.5, 350, 16, 3360.0, true, new
MaterialLibrary("W", 300, 60.0, "8", "24"), new MaterialLibrary("W", 158, 60,
"10", "30"), new MaterialLibrary("W", 50, 35, "14", "43"));
            YardSpace A17 = new YardSpace(32.5, 450, 17, 3360.0, true, new
MaterialLibrary("PL", 88, 8.0, "3/8"), new MaterialLibrary("PL", 30, 8.0, "1"),
new MaterialLibrary("PL", 20, 8.0, "1/2"));
            YardSpace A18 = new YardSpace(97.5, 450, 18, 3360.0, true, new
MaterialLibrary("PL", 10, 8.0, "3/8"), new MaterialLibrary("PL", 10, 8.0, "1"),
new MaterialLibrary("PL", 10, 8.0, "1/2"));
            YardSpace A19 = new YardSpace(162.5, 450, 19, 3360.0, true, new
MaterialLibrary("PL", 10, 8.0, "3/8"), new MaterialLibrary("PL", 10, 8.0, "1"),
new MaterialLibrary("PL", 10, 8.0, "1/2"));
            YardSpace A20 = new YardSpace(227.5, 450, 20, 3360.0, true, new
MaterialLibrary());
            YardInventory.Add(A1);
            YardInventory.Add(A2);
            YardInventory.Add(A3);
            YardInventory.Add(A4);
            YardInventory.Add(A5);
            YardInventory.Add(A6);
            YardInventory.Add(A7);
            YardInventory.Add(A8);
            YardInventory.Add(A9);
            YardInventory.Add(A10);
            YardInventory.Add(A11);
            YardInventory.Add(A12);
            YardInventory.Add(A13);
            YardInventory.Add(A14);
            YardInventory.Add(A15);
            YardInventory.Add(A16);
            YardInventory.Add(A17);
            YardInventory.Add(A18);
            YardInventory.Add(A19);
            YardInventory.Add(A20);
            #endregion


            //***making a copy of the yard inventory for the first day***
            List<YardSpace> ArtificialYard = new List<YardSpace>();
            foreach (var a in YardInventory)
                ArtificialYard.Add(a.DeepClone());

            ////////////////////////////////////////////////
            List<InventoryDatabase> ThirtyDayInventory = new
List<InventoryDatabase>();
            ThirtyDayInventory.Add(new
InventoryDatabase(1,this.ThirtydayInventory(YardInventory,1)));


            List<InventoryDatabase> ConstraintYardDataBase = new
List<InventoryDatabase>();
            List<InventoryDatabase> UpdatedYardDataBase = new
List<InventoryDatabase>();

            Random rnd = new Random();
            int counter = 0;
            int dayCounter1 = 1;
            int dayCounter2 = 1;
            // ***************** Initialize the constraint inventory
****************
            ConstraintYardDataBase.Add(new InventoryDatabase(1,
ThirtyDayInventory[0].SingleDayYardInventory));
            ArtificialYard.Clear();
            foreach (var pp in ConstraintYardDataBase[0].SingleDayYardInventory)
                ArtificialYard.Add(pp.DeepClone()); //Remember to clear this
temporary yard at the end
```

```
            //
**********************************************************************
            foreach (var t in IncomingMaterials)
            {

                if (t.date > dayCounter1)
                {
                    UpdatedYardDataBase.Add(new InventoryDatabase(t.date,
ArtificialYard));
                    ArtificialYard.Clear();


                    // ******** Draw material from the updated yard
*******************
                    //ThirtyDayInventory.Add(new
InventoryDatabase(t.date,ThirtydayInventory(UpdatedYardDataBase[t.date-
2].SingleDayYardInventory, OutgoingMaterials, t.date)));
                    ConstraintYardDataBase.Add(new InventoryDatabase(t.date,
ThirtydayInventory(UpdatedYardDataBase[t.date - 2].SingleDayYardInventory,
t.date)));
                    //
****************************************************************

                    dayCounter1 = t.date;

                    // ********************  Clear the empty yard cells
***************
                    foreach (var pp in ConstraintYardDataBase[t.date -
1].SingleDayYardInventory)
                    {
                        double counter5 = 0.0;
                        if (pp.MaterialsOntheYard.Count != 0)
                        {
                            foreach (var a in pp.MaterialsOntheYard)
                            {
                                counter5 = counter5 + a.quantity;

                            }
                            if (counter5 == 0)
                                pp.MaterialsOntheYard.Clear();
                        }
                    }

                }//End of two formula block

                //********** First update the yard based on constraintInventory
******
                // Here I want to make a clone of previously calculated
constraintinventory ****
                if (t.date > dayCounter2)
                {
                    foreach (var pp in ConstraintYardDataBase[t.date -
1].SingleDayYardInventory)
                        ArtificialYard.Add(pp.DeepClone()); //Remember to clear
this temporary yard at the end
                    dayCounter2 = t.date;
                }

                // *************** End of cloning *************************
                counter = 0;
                //Console.WriteLine("Placing material #: {0}\n", m.ID);
                bool placementStatus = false;
                while (!placementStatus)
                {
                    //m.segmentTag = (int)getlocalRandom[counter]; //number of the
segments to be 20
                    t.segmentTag = rnd.Next(1, 21);
                    foreach (var s in ArtificialYard)
                    {
                        if (t.segmentTag == s.ID)
                        {
```

```
                            if (s.Status == true)
                            {
                                if (s.MaterialsOntheYard.Count == 0)
                                {
                                    MaterialLibrary newMaterial = new
MaterialLibrary(t.TypeandQtyandLength.type, t.TypeandQtyandLength.quantity,
t.TypeandQtyandLength.length, t.TypeandQtyandLength.SecProperties.ToArray());
                                    s.MaterialsOntheYard.Add(newMaterial);
                                    placementStatus = true;
                                    //Genes[m.ID - 1] = n.ID;
                                }
                                else if (t.TypeandQtyandLength.type ==
s.MaterialsOntheYard[0].type)
                                {
                                    double tempCap = s.GetSegmentVolume();
                                    if (t.volume <= s.Capacity-tempCap)
                                    {
                                        placementStatus = true;
                                        //Genes[m.ID - 1] = n.ID;

                                        //Inventory update
                                        int flag = 0;
                                        foreach (var p in s.MaterialsOntheYard)
                                        {
                                            if (t.TypeandQtyandLength.ToString()
== p.ToString())

                                            {
                                                p.quantity = p.quantity +

t.TypeandQtyandLength.quantity;

                                                flag++;
                                            }
                                            //if(flag==false)

                                        }
                                        if (flag == 0)
                                        {
                                            s.MaterialsOntheYard.Add(new
MaterialLibrary(t.TypeandQtyandLength.type, t.TypeandQtyandLength.quantity,
t.TypeandQtyandLength.length, t.TypeandQtyandLength.SecProperties));
                                        }


                                    }

                                }
                            }
                        }
                    }
                    counter++;
                    if (counter == 100000) //One hunderd trials
                    {
                        Console.WriteLine("I am afraind there is no space on the
yard!!");

                        Console.WriteLine("the program attempted {0}

times",counter);

                        //return;
                    }
                }



            ThirtyDayInventory.Clear();
            UpdatedYardDataBase.Clear();
            ConstraintYardDataBase.Clear();
            YardInventory.Clear();
            foreach (var dd in IncomingMaterials)
            {
                Genes[dd.ID - 1] = dd.segmentTag;
            }
            for (int i = 0; i < Genes.Length; i++)
            {
```

```
                if (Genes[i] == 0)
                    throw new IndexOutOfRangeException("There are some zero genes
in the chromosomes");
            }

        }

        private bool PlacementVerifier(List<IncomingBatch> IncomingMaterials,
List<OutgoingBatch> OutgoingMaterials, int[] TestGenes)
        {

            List<YardSpace> YardInventory = new List<YardSpace>(); ;

            #region Initializing the yard

            //YardSpace A12 = new YardSpace(227.5, 250, 12, 3360.0, true, new
MaterialLibrary("W", 102, 60.0, "8", "24"), new MaterialLibrary("W", 400, 60,
"10", "30"), new MaterialLibrary("W", 400, 35, "14", "43"));
            YardSpace A1 = new YardSpace(32.5, 50, 1, 3360.0, true, new
MaterialLibrary("L", 215, 40.0, "8", "8", "1/8"));
            YardSpace A2 = new YardSpace(97.5, 50, 2, 3360.0, true, new
MaterialLibrary());
            YardSpace A3 = new YardSpace(162.5, 50, 3, 3360.0, true, new
MaterialLibrary());
            YardSpace A4 = new YardSpace(227.5, 50, 4, 3360.0, true, new
MaterialLibrary("W", 170, 60.0, "8", "24"));
            YardSpace A5 = new YardSpace(32.5, 150, 5, 3360.0, true, new
MaterialLibrary());
            YardSpace A6 = new YardSpace(97.5, 150, 6, 3360.0, true, new
MaterialLibrary());
            YardSpace A7 = new YardSpace(162.5, 150, 7, 3360, false);
            YardSpace A8 = new YardSpace(227.5, 150, 8, 3360.0, true, new
MaterialLibrary());
            YardSpace A9 = new YardSpace(32.5, 250, 9, 3360.0, false);
            YardSpace A10 = new YardSpace(97.5, 250, 10, 3360.0, true, new
MaterialLibrary());
            YardSpace A11 = new YardSpace(162.5, 250, 11, 3360.0, true, new
MaterialLibrary());
            YardSpace A12 = new YardSpace(227.5, 250, 12, 3360.0, true, new
MaterialLibrary("W", 102, 60.0, "8", "24"), new MaterialLibrary("W", 400, 60,
"10", "30"), new MaterialLibrary("W", 50, 35, "14", "43"));
            YardSpace A13 = new YardSpace(32.5, 350, 13, 3360.0, true, new
MaterialLibrary("C", 100, 60.0, "10", "15.3"), new MaterialLibrary("C", 100, 40,
"8", "13.75"), new MaterialLibrary("C", 100, 50, "15", "50"));
            YardSpace A14 = new YardSpace(97.5, 350, 14, 3360.0, true, new
MaterialLibrary());
            YardSpace A15 = new YardSpace(162.5, 350, 15, 3360.0, true, new
MaterialLibrary());
            YardSpace A16 = new YardSpace(227.5, 350, 16, 3360.0, true, new
MaterialLibrary("W", 300, 60.0, "8", "24"), new MaterialLibrary("W", 158, 60,
"10", "30"), new MaterialLibrary("W", 50, 35, "14", "43"));
            YardSpace A17 = new YardSpace(32.5, 450, 17, 3360.0, true, new
MaterialLibrary("PL", 88, 8.0, "3/8"), new MaterialLibrary("PL", 30, 8.0, "1"),
new MaterialLibrary("PL", 20, 8.0, "1/2"));
            YardSpace A18 = new YardSpace(97.5, 450, 18, 3360.0, true, new
MaterialLibrary("PL", 10, 8.0, "3/8"), new MaterialLibrary("PL", 10, 8.0, "1"),
new MaterialLibrary("PL", 10, 8.0, "1/2"));
            YardSpace A19 = new YardSpace(162.5, 450, 19, 3360.0, true, new
MaterialLibrary("PL", 10, 8.0, "3/8"), new MaterialLibrary("PL", 10, 8.0, "1"),
new MaterialLibrary("PL", 10, 8.0, "1/2"));
            YardSpace A20 = new YardSpace(227.5, 450, 20, 3360.0, true, new
MaterialLibrary());
            YardInventory.Add(A1);
            YardInventory.Add(A2);
            YardInventory.Add(A3);
            YardInventory.Add(A4);
            YardInventory.Add(A5);
            YardInventory.Add(A6);
            YardInventory.Add(A7);
            YardInventory.Add(A8);
            YardInventory.Add(A9);
```

```
            YardInventory.Add(A10);
            YardInventory.Add(A11);
            YardInventory.Add(A12);
            YardInventory.Add(A13);
            YardInventory.Add(A14);
            YardInventory.Add(A15);
            YardInventory.Add(A16);
            YardInventory.Add(A17);
            YardInventory.Add(A18);
            YardInventory.Add(A19);
            YardInventory.Add(A20);
            #endregion

            //***making a copy of the yard inventory for the first day***
            List<YardSpace> ArtificialYard = new List<YardSpace>();
            foreach (var a in YardInventory)
                ArtificialYard.Add(a.DeepClone());


            ///////////////////////////////////////////////
            List<InventoryDatabase> ThirtyDayInventory = new
List<InventoryDatabase>();
            ThirtyDayInventory.Add(new InventoryDatabase(1,
this.ThirtydayInventory(YardInventory, 1,false)));

            // ***********************************************************
            //          Placing the material on the constraint yard
            // ***********************************************************

            List<InventoryDatabase> ConstraintYardDataBase = new
List<InventoryDatabase>();
            List<InventoryDatabase> UpdatedYardDataBase = new
List<InventoryDatabase>();

            Random rnd = new Random();
            int counter = 0;
            int dayCounter1 = 1;
            int dayCounter2 = 1;
            // ***************** Initialize the constraint inventory
****************
            ConstraintYardDataBase.Add(new InventoryDatabase(1,
ThirtyDayInventory[0].SingleDayYardInventory));
            ArtificialYard.Clear();
            foreach (var pp in ConstraintYardDataBase[0].SingleDayYardInventory)
                ArtificialYard.Add(pp.DeepClone()); //Remember to clear this
temporary yard at the end

            //
**********************************************************************
            foreach (var t in IncomingMaterials)
            {

                if (t.date > dayCounter1)
                {
                    UpdatedYardDataBase.Add(new InventoryDatabase(t.date,
ArtificialYard));
                    ArtificialYard.Clear();


                    // ******** Draw material from the updated yard
*******************
                    //ThirtyDayInventory.Add(new
InventoryDatabase(t.date,ThirtydayInventory(UpdatedYardDataBase[t.date-
2].SingleDayYardInventory, OutgoingMaterials, t.date)));
                    ConstraintYardDataBase.Add(new InventoryDatabase(t.date,
ThirtydayInventory(UpdatedYardDataBase[t.date - 2].SingleDayYardInventory,
t.date,false)));
                    // ********************  Clear the empty yard cells
***************
                    foreach (var pp in ConstraintYardDataBase[t.date -
1].SingleDayYardInventory)
```

```
                    {
                        double counter5 = 0.0;
                        if (pp.MaterialsOntheYard.Count != 0)
                        {
                            foreach (var a in pp.MaterialsOntheYard)
                            {
                                counter5 = counter5 + a.quantity;

                            }
                            if (counter5 == 0)
                                pp.MaterialsOntheYard.Clear();
                        }
                    }
                    // *******************************************
                    dayCounter1 = t.date;
                }//End of two formula block

                //********** First update the yard based on constraintInventory
******
                // Here I want to make a clone of previously calculated
constraintinventory ****
                if (t.date > dayCounter2)
                {
                    foreach (var pp in ConstraintYardDataBase[t.date -
1].SingleDayYardInventory)
                        ArtificialYard.Add(pp.DeepClone()); //Remember to clear
this temporary yard at the end
                    dayCounter2 = t.date;
                }
                // *************** End of cloning *************************
                counter = 0;
                t.segmentTag = TestGenes[counter];
                foreach (var s in ArtificialYard)
                {
                    if (t.segmentTag == s.ID)
                    {
                        if (s.Status == true)
                        {
                            if (s.MaterialsOntheYard.Count == 0)
                            {
                                MaterialLibrary newMaterial = new
MaterialLibrary(t.TypeandQtyandLength.type, t.TypeandQtyandLength.quantity,
t.TypeandQtyandLength.length, t.TypeandQtyandLength.SecProperties.ToArray());
                                s.MaterialsOntheYard.Add(newMaterial);
                                //placementStatus = true;
                                //Genes[m.ID - 1] = n.ID;
                            }
                            else if (t.TypeandQtyandLength.type ==
s.MaterialsOntheYard[0].type)
                            {
                                double tempCap = s.GetSegmentVolume();
                                if (t.volume <= s.Capacity - tempCap)
                                {
                                    //placementStatus = true;
                                    //Genes[m.ID - 1] = n.ID;

                                    //Inventory update
                                    int flag = 0;
                                    foreach (var p in s.MaterialsOntheYard)
                                    {

                                        if (t.TypeandQtyandLength.ToString() ==
p.ToString())

                                        {
                                            p.quantity = p.quantity +
t.TypeandQtyandLength.quantity;
                                            flag++;

                                        }
```

```
                                            //s.MaterialsOntheYard.Add(new 
MaterialLibrary(t.TypeandQtyandLength.type, t.TypeandQtyandLength.quantity, 
t.TypeandQtyandLength.length, t.TypeandQtyandLength.SecProperties));
                                    }
                                    if(flag==0)
                                        s.MaterialsOntheYard.Add(new 
MaterialLibrary(t.TypeandQtyandLength.type, t.TypeandQtyandLength.quantity, 
t.TypeandQtyandLength.length, t.TypeandQtyandLength.SecProperties));

                                }
                                else
                                {
                                   // Console.WriteLine(" FALSE 
FALSE!**********************");
                                    ThirtyDayInventory.Clear();
                                    UpdatedYardDataBase.Clear();
                                    ConstraintYardDataBase.Clear();
                                    YardInventory.Clear();
                                    return false;
                                }

                            }
                            else
                            {
                               // Console.WriteLine(" FALSE 
FALSE!**********************");
                                ThirtyDayInventory.Clear();
                                UpdatedYardDataBase.Clear();
                                ConstraintYardDataBase.Clear();
                                YardInventory.Clear();
                                return false;
                            }
                        }
                        else
                        {
                            //Console.WriteLine(" FALSE 
FALSE!**********************");
                            ThirtyDayInventory.Clear();
                            UpdatedYardDataBase.Clear();
                            ConstraintYardDataBase.Clear();
                            YardInventory.Clear();
                            return false;
                        }
                    }
                }
                counter++;

            }//End of material placement for all the days

            ThirtyDayInventory.Clear();
            UpdatedYardDataBase.Clear();
            ConstraintYardDataBase.Clear();
            YardInventory.Clear();
            Console.WriteLine(" TRUE TRUE!**********************");
            return true;
        }

        #endregion

        #region public methods

        double GetDistance(double x1, double x2, double y1, double y2)
        {
            double dist = 0.0;

            dist = Math.Sqrt(Math.Pow((x2-x1),2)+Math.Pow((y2-y1),2));
            return dist;
        }

        static IEnumerable<int> UniqueRandom(int minInclusive, int maxInclusive)
        {
```

```
            List<int> candidates = new List<int>();
            for (int i = minInclusive; i <= maxInclusive; i++)
            {
                candidates.Add(i);
            }
            Random rnd = new Random();
            while (candidates.Count > 0)
            {
                int index = rnd.Next(candidates.Count);
                yield return candidates[index];
                candidates.RemoveAt(index);
            }
        }

        public void Crossover(ref Chromosome chromosome2, out Chromosome child1,
out Chromosome child2,List<IncomingBatch> in1, List<OutgoingBatch> out1,
List<YardSpace> myYard1)
        {

            child1 = new Chromosome(Length);
            child2 = new Chromosome(Length);
            bool IsPlacementOK = false,child1Placed=false,child2Placed=false;

            List<int> getlocalRandom = new List<int>();
            foreach (var a in UniqueRandom(0, in1.Count-1))
            {
                getlocalRandom.Add(a);
            }
            int counter = 0;
            while (!IsPlacementOK)
            {
                int pos = (int)getlocalRandom[counter];
                counter++;
                for (int i = 0; i < Length; i++)
                {
                    if (i < pos)
                    {
                        child1.Genes[i] = Genes[i];
                        child2.Genes[i] = chromosome2.Genes[i];

                    }
                    else
                    {
                        child1.Genes[i] = chromosome2.Genes[i];
                        child2.Genes[i] = Genes[i];
                    }
                }

                child1Placed = child1.PlacementVerifier(in1, out1,  child1.Genes);
                child2Placed = child1.PlacementVerifier(in1, out1,  child2.Genes);

                if (child1Placed && child1Placed)
                {
                    IsPlacementOK = true;
                }
                else if (/*counter == in1.Count*/counter==2&&IsPlacementOK==false)
                {
                    child1 = new Chromosome(this.Length,true,  in1,   out1);
                    child2 = new Chromosome(this.Length,true,  in1,  out1);
                    IsPlacementOK = true;

                }
            }
        }

        public void Mutate( List<IncomingBatch> in1,List<OutgoingBatch> out1,
double MutationRate)
        {
            bool IsPlacementOK = false;
            int counter = 0;
```

```
            while (!IsPlacementOK)
            {
                counter++;
                for (int pos = 0; pos < Length; pos++)
                {
                    if (_random.NextDouble() < MutationRate)
                    {
                        this.Genes[pos] = (int)((this.Genes[pos] + _random.Next(1,
20)) / 2.0);

                    }
                }
                IsPlacementOK = this.PlacementVerifier( in1,  out1, this.Genes);
                if (/*counter == this.Length*/counter==2)
                {

                    this.CreateGenes(in1, out1);
                    IsPlacementOK = true;
                }

            }
        }

        public void GetValues(ref double[] values)
        {
            for (int i = 0; i < Length; i++)
                values[i] = Genes[i];
        }

        public override string ToString()
        {
            string results = "";
            for (int i = 0; i < Length; i++)
            {
                results += string.Format("{0;F4}", Genes[i]);
            }
            return results;
        }


        #endregion


    }
}
```

Class IncomingBatches:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace BasicGA
{
    public class IncomingBatch
    {
        #region Constructors

        /// <summary>
        /// Default contstructor
        /// </summary>
        public IncomingBatch(MaterialLibrary matIn)
        {
            this.X = 0.0;
            this.Y = 0.0;
            this.TypeandQtyandLength = matIn;
```

```
        }
        #endregion

        #region Public and private properties
        public double X { get; set; }
        public double Y { get; set; }
        public MaterialLibrary TypeandQtyandLength;
        public int ID { get; set; }
        public int date { get; set; }
        public int segmentTag { get; set; }
        public double distance { get; set; }
        public double volume { get; set; }



        #endregion

        #region Public methods

        #endregion
    }
}
```

Class OutgoingBatches:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace BasicGA
{
    public class OutgoingBatch
    {
        #region Constructors
        /// <summary>
        /// Default construtor
        /// </summary>

        public OutgoingBatch(MaterialLibrary matOut)
        {

            this.TypeandQtyandLength = matOut;

        }
        #endregion

        #region Public properties

        //public double qty { get; set; }
        //public string type { get; set; }
        public MaterialLibrary TypeandQtyandLength;
        public int ID { get; set; }
        public int date { get; set; }
        public int segmentTag { get; set; }
        public double length { get; set; }
        public double volume { get; set; }
        #endregion

        #region Public methods
        public OutgoingBatch DeepClone()
        {
            var clone = (OutgoingBatch)this.MemberwiseClone();
            clone.TypeandQtyandLength = new
MaterialLibrary(clone.TypeandQtyandLength.type,
clone.TypeandQtyandLength.quantity, clone.TypeandQtyandLength.length,
clone.TypeandQtyandLength.SecProperties);

            return clone;
        }
```

```
        #endregion
    }
}
```

Class YardSpace:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace BasicGA
{
    public class YardSpace
    {
        #region Constructors
        public YardSpace()
        {
            x = 0;
            y = 0;
            Id = 0;
            cap = 0;
            Status = true;
        }
        /// <summary>
        /// Initializes the (Empty)Yard space objects based on constant site
status.
        /// </summary>
        /// <param name="X"> X coordinate of the segment center with respect to
global coordinate system in foot</param>
        /// <param name="Y"> Y coordinate of the segment center with respect to
global coordinate system in foot</param>
        /// <param name="ID"> ID number of the segment center </param>
        /// <param name="cap"> Capacity of the yard in ft^3 </param>
        /// <param name="Date"> Inventory date</param>
        /// <param name="SegmentStatus"> determines if the site is busy with
special jobs are not if sets to true it is available otherwise it is busy</param>
        public YardSpace(double X, double Y, int ID, double cap, bool
SegmentStatus)
        {
            this.x = X;
            this.y = Y;
            this.Id = ID;
            this.cap = cap;

            this.Status = SegmentStatus;

        }
        /// <summary>
        /// Initializes the filled yard space objects based on changing site
status.
        /// </summary>
        /// <param name="X">X coordinate of the segment center with respect to
global coordinate system in foot</param>
        /// <param name="Y">Y coordinate of the segment center with respect to
global coordinate system in foot</param>
        /// <param name="ID">ID number of the segment center</param>
        /// <param name="cap">Capacity of the yard in foot^3</param>
        /// <param name="Date"> Inventory date</param>
        /// <param name="SegmentStatus">determines if the site is busy with
special jobs are not if sets to true it is available otherwise it is busy</param>
        /// <param name="AvailableMaterials">Contains information about the
available materials on the yard such as length, type and Quantity</param>
        public YardSpace(double X, double Y, int ID, double cap, bool
SegmentStatus, params MaterialLibrary[] AvailableMaterials)
        {
            this.x = X;
            this.y = Y;
            this.Id = ID;
            this.cap = cap;
            this.Status = true;
```

```
            for (int i = 0; i < AvailableMaterials.Length; i++)
            {
                if (AvailableMaterials[i].length != 0)
                    this.MaterialsOntheYard.Add(AvailableMaterials[i]);
                else
                {
                    //this.MaterialsOntheYard[0].type
                }

            }


        }

        public YardSpace DeepClone()
        {
            var clone = (YardSpace)this.MemberwiseClone();
            clone.MaterialsOntheYard = new List<MaterialLibrary>();
            foreach (var m in this.MaterialsOntheYard)
            {
                clone.MaterialsOntheYard.Add(m.Clone());
            }

            return clone;
        }
        public YardSpace ShallowClone()
        {
            return (YardSpace)this.MemberwiseClone();
        }


        #endregion

        #region fields
        int Id;
        private double x, y, cap;
        public bool Status;
        public List<MaterialLibrary> MaterialsOntheYard = new
List<MaterialLibrary>();
        #endregion

        #region Public properties
        public double X
        {
            get
            {
                return x;
            }
        }

        public double Y
        {
            get
            {
                return y;
            }
        }

        public int ID
        {
            get
            {
                return Id;
            }
        }

        public double Capacity
        {
            get
            {
                return cap;
```

```
            }
        }



        #endregion

        #region Public methods
        public double GetSegmentVolume()
        {
            double volu = 0.0;

            foreach (var m in MaterialsOntheYard)
            {
                volu = volu + m.GetVolume(m.ToString(), m.length);

            }

            return volu;
        }
        #endregion
    }
}
```