ADAPTIVE PRODUCTION PLANNING AND SCHEDULING FOR THE MAKE-TO-ORDER DNA MANUFACTURING SYSTEM

A Thesis
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Dan Song

In Partial Fulfillment of the Requirements
for the Degree of
MASTER OF SCIENCE

Major Department:
Industrial and Manufacturing Engineering

June 2010

Fargo, North Dakota

North Dakota State University
Graduate School

Title

ADAPTIVE PRODUCTION PLANNING AND SCHEDULING FOR THE

MAKE-TO-ORDER DNA MANUFACTURING SYSTEM

By

DAN SONG

The Supervisory Committee certifies that this ***disquisition*** complies with North Dakota State University's regulations and meets the accepted standards for the degree of

MASTER OF SCIENCE

ABSTRACT

Song, Dan, M.S., Department of Industrial and Manufacturing Engineering, College of Engineering and Architecture, North Dakota State University, June 2010. Adaptive Production Planning and Scheduling for the Make-to-order DNA Manufacturing System. Major Professor: Dr. Jun Zhang. Co-Advisor: Dr. Jing Shi.

This thesis develops an adaptive production planning and scheduling system for the make-to-order plasmid (DNA) manufacturing system. The system, which has stochastic nature and random demand, was represented by a mathematical programming model first. Then in order to solve it, discrete-event simulation models were developed to generate a feasible schedule that maximizes the production throughput in the planning horizon in a mix-product type environment. A special heuristic order selecting and splitting procedure was designed to aid the production planning and scheduling process. Experiments were conducted to evaluate the algorithm and results are compared with those obtained by using four classic dispatching rules, such as first come first served (FCFS) and shortest processing time (SPT).

To take advantage of simulation results, a rule-based expert system was created with pre-defined scheduling rules. Rules regarding production planning and scheduling can be used by human schedulers easily and the system is very flexible in further extension.

ACKNOWLEDGEMENTS

This thesis could not have been done without the help from the following persons that I would like to thank.

First, I wish to thank my major advisor, Dr. Jun Zhang. She has been supportive for years since I began working on the project. I remember the days that she encouraged me to continue with the research work. Her help is not only limited to the academic area but also applied to daily life. And during the tough time when writing this thesis, she gave me the great support and freedom that I needed to keep moving on.

Second, I would like to express my gratitude to my co-advisor, Dr. Jing Shi, who guided me through this research study.

I also wish to thank my committee members, Dr. Kambiz Farahmand, the chair of this department, who gave me opportunities to work with him and who was supportive during my master's study, and Dr. Chanchai Tangpong, for his kindness in being part of my committee and reviewing my thesis.

I would like to give the most gratitude to my parents, who gave birth to me, raised me, supported me and funded me to study in the US. I especially wish to thank my boyfriend, who has always been my soul mate back in China. Many thanks to my dear friends and those who have given me decent support during my days in the US.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1. INTRODUCTION

1.1. Overview

Production planning entails the acquisition and allocation of limited resources (machines, humans, production tools, and storage) to production activities so as to fulfill production objectives over a specified time horizon. Meanwhile, production scheduling determines an optimal sequence of jobs released for production. Both planning and scheduling enable the use of optimization techniques to reduce non-value added activities so as to increase productivity. Furthermore, those techniques are supportive in order to fulfill the primary production goal of meeting demand with minimization of the make-span, the total tardiness and the total costs or maximization of the total profitable margin (Pinedo, 2002). The underlying optimization problem varies due to differences in the manufacturing context. This thesis deals with a production planning and scheduling problem arising in a make-to-order (MTO) flexible microbiological flow shop.

Microbiological industry encompasses the use of microorganisms in producing food or industrial products, such as vaccines, antibiotics, gene-based medicines, etc. Specifically, this study focuses on the production of plasmid Deoxyribonucleic acid, a.k.a. plasmid DNA. With the fast evolution of clinical use and research of genes in the past decade, since the market demand of large-quantity plasmid products increases, industrial-scale plasmid DNA production has become a great interest for both researchers and bio-product manufacturers. Although plasmid DNA manufacturing is not a new-born industry, the research is rarely concerned with improving operations in plasmid manufacturing plant through simulation and mathematical modeling.

In this introductory chapter, a brief discussion of the plasmid manufacturing is presented from its basic elements, plasmid, E. coli and medium, to the production procedure. Then, it moves on to the description of the system including the characteristics of uncertain demand, the product type being manufactured and the processing features. With the consideration of the above characteristics and problems involved, the solution procedure is presented. The structure of the thesis follows terminology which will be used in the later sections and which will be clarified in the subsequent paragraph.

1.2. Plasmid, E. coli and medium

The production of plasmid DNA (pDNA) starts with plasmid samples. The term plasmid was first invented in 1952 (Lederberg), demonstrating a generic term for any extra-chromosomal heredity determinant. According to the scientific definition, plasmids are naturally occurring, stable genetic elements which may be composed of DNA or RNA, double-stranded or single-stranded, linear or circular and typically found in bacteria, fungi, and even in the mitochondria of some plants ("Plasmid-Types", n.d.). They sometimes are used as vectors for gene insertion or genetic engineering. Plasmids do not require hosts to be alive which means they can be isolated from the host cells and live out of the body for a short while.

With the size as small as 1 to 200 kbp (kilo base pairs), where a base-pair, or bp in short, is a pair of nucleotides connected via hydrogen bonds, plasmids are extremely suitable to carry certain gene information and transfer it among various sources of cells. They are being used as transferrable information-carriers, or "replicons", capable of autonomous replication in appropriate hosts. The replication takes place when a single host cell divides into two. Both offspring cells will contain the same plasmid.

To keep plasmids “alive” in host cells and produce more cells containing plasmids, Escherichia coli (commonly E. coli), one of several types of bacteria that normally inhabit the intestines of humans and animals, are used to provide plasmids accommodation. Although E. coli are well known for causing severe disease to human beings and other creatures, because of its capability to reproduce promptly and ability to survive for a short period outside the host body, it becomes the most appropriate host for producing plasmid DNA in either a laboratory or industrial environment. After plasmids have been inserted into E. coli cells with certain strain features, where a strain is a subdivision of the spiecies that has unique characteristics distinguishing it from other E. coli cells, a vitro holding them becomes a starter "growing" under the specified temperature and humidity for a relatively long period. Here, growing does not refer to growing bigger in physical measurement but the amount of cells. Along with E. coli expedition, a single plasmid inhabited in the E. coli cell splits into two and reproduces itself autonomously. This procedure is shown in Figure 1.

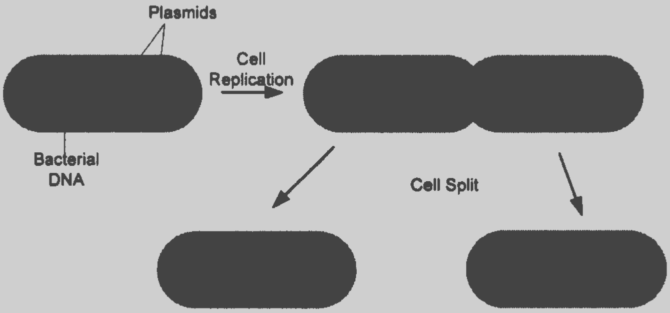

Figure 1. Overview of plasmid reproduction.

As E. coli bacteria provide inhabitancy for plasmids, the bacteria themselves also need nutrition supplies to stay active. Medium, also called growth medium, culture medium or nutrient broth in microbiology, is a liquid or gel usually comprised of water, some salts, a carbon source such as glucose, and amino acid or nitrogen (Danquah & Forde, 2007). It supplies the essential nutrients for the growth of plasmid and E. coli cells under the controllable environmental variables. In other words, both E. coli cells and the target plasmid replicate in the micro-environment provided by the medium. The most common growth media for microorganisms are nutrient broths and agar plates; specialized media are sometimes required (Growth medium, n.d.).

1.2.1. Plasmid production procedure

Plasmid replication can be done in a laboratory environment with the corporation of E-coli and nutrient broth/medium. Industrial-scale plasmid production transfers laboratory-based microbiology techniques into manufacturing plants, and amplifies it as well as makes the acquisition of final plasmids easier and more efficient compared with the traditional non-optimized lab condition (Ferreira, Monteiro, Prazeres, & Cabral, 2000). In the production of plasmid DNA serving for the market of therapeutic and pharmaceutical products, a "large-scale" always means 10mg of DNA to even several kilograms of plasmids (Danquah & Forde, 2007). Figure 2 generally outlines the procedure of producing plasmid DNA in a factory environment.

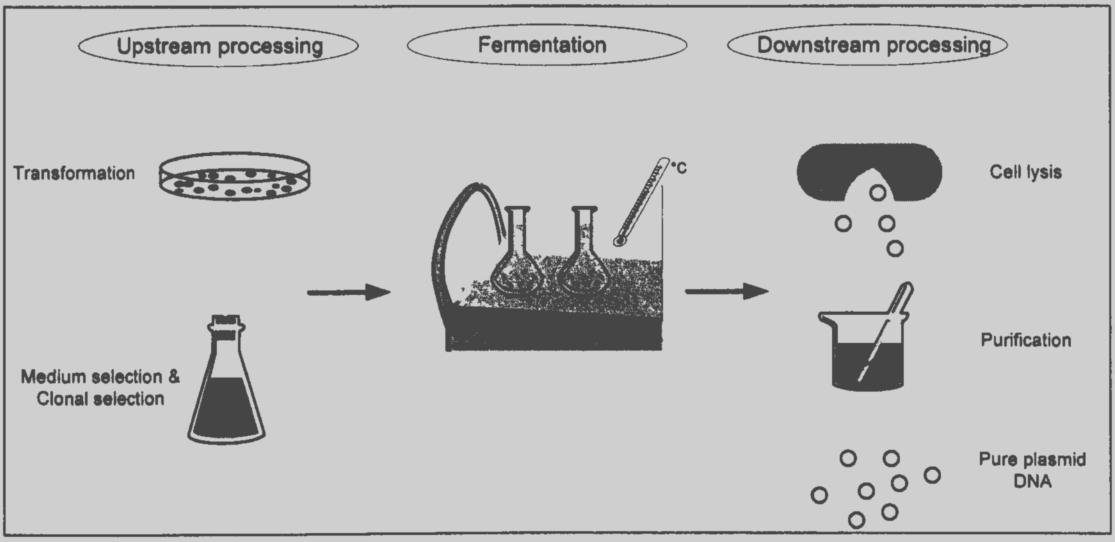

Figure 2. Process steps of the production of plasmid DNA.

There are three key steps involved in the processing of plasmid products: (1) upstream processing, (2) fermentation and (3) downstream processing.

Upstream processing of DNA plasmid production begins with the transformation of plasmids to competent E. coli cells. In molecular biology, bacteria transformation is the exchange of genetic material between the strains of bacteria by the transfer of a fragment of naked DNA from a donor cell to a recipient cell, followed by recombination in the recipient chromosome (Anonymous, n.d.). In plasmid production, the transformation refers to the uptake of plasmid DNA from the original cells to recipient E coli. Here, "naked DNA" are plasmids that do not encode necessary genetic materials for the transfer to new hosts. A heat shock step is adopted to allow the successful uptake of plasmid by bacteria. Subsequently, growth medium is selected and bacterial colonies grow under controlled conditions. Temperature and agitation are the typical elements explicitly controlled (Prather, Sagar, Murphy, & Chartrain, 2003). This is a culture process where a small volume of the cells are grown. A vial of selected medium along with the selected clonal, known as a starter, is used in the fermentation process.

Fermentation is a batch cultivation process where well-colonized starters are spiked into a large volume of medium, and begin exponential growth with the aid of nutrients in the shake fermentors. A key advantage of fermentation is that it is able to control and examine the conditions that influence cell growth, plasmid volumetric yield, quality and stability of growth. Those controllable variables include pH value, dissolved oxygen, temperature, composition of culture medium, as well as build-up of waste metabolites (Durland & Eastman, 1998). After fermentation, cells are harvested for further processing.

Downstream processing of plasmid DNA consists of a sequence of unit purification operations that are essentially aimed at eliminating impurities. An appropriate purification strategy begins with two major steps. The first step is cell lysis with alkaline solutions, where all molecular components including plasmid DNA, RNA, gDNA, endotoxins and proteins are released. Relatively large-scale contaminants such as cell debris, denatured proteins and nucleic acids are removed through a precipitation procedure using a solid-liquid unit operation, usually centrifugation. The second step is clarification and concentration. It is designed to further remove proteins and host nucleic acids and to increase the plasmid mass fraction, as well as prepare the extracts for the subsequent purification steps (Ferreira, Monteiro, Prazeres, & Cabral, 2000).

Anion exchange chromatography (AEX) is one of the methods of choice for large-scale purification of plasmid DNA, since it is well suited for plasmid separation with the objective of selectively isolating and purifying plasmid DNA from impurities. Owing to the high negative charge on DNA, AEX utilizes resins to carry positively-charged functional groups which interact with the negatively-charged phosphates of nucleic acids. In this way, bound molecules are eluted easily from the resin using concentrated salt solutions (Durland & Eastman, 1998).

Once plasmids have been sufficiently purified, it is usually necessary to concentrate them. It may also need to exchange unwanted salts or buffers for a preferred storage solution. At last, pure plasmids are stored at a temperature below 0°C which finishes the entire production.

1.2.2. Make-to-order plasmid manufacturing

Manufacturing of plasmid DNA is a highly customized production which is also known as make-to-order (MTO) manufacturing because each plasmid product has its individual quantity and characteristics. MTO companies make products according to the customer requirements; therefore, one customer may order products different from others in design, ingredients, packaging, etc. Basically, an MTO plasmid production system has the following characteristics and problems that have been usually concerned by researchers.

MTO plasmid production is order-driven but demand is uncertain. Production operation is scheduled and carried out in response to the orders received from end customers. As MTO products are built to orders, the demand of certain products is difficult to be predicted prior to receiving orders. Hence, manufacturers are susceptible to market demand fluctuations which leads to the reduced capacity utilization in manufacturing. Take plasmid production as an example. Products are highly customized compounds and each order is comprised of exact one type of plasmid. On customer's side, the source and the application of plasmids determine their production quantities and processing qualities known as demand to manufacturers. Moreover, when to receive orders, what to produce and how much to produce are all uncertain. These uncertainties create significant fluctuation in daily demand.

Production system accounts for multi-purpose highly-customer-configured products. They are made strictly through customer specifications and most likely to be different. However, all the MTO products produced by a specific company do share similar processing features. Some can be processed by using identical equipment; some are made from the same materials. Sometimes, the difference even shows in packaging stage only while all the other processes that jobs go through are unified. Unlike most of the MTO products, plasmids are produced with the provision of original samples by customers. The fact that the source and the use of plasmids are predefined brings uncontrollable factors into the production. For example, yield is one of the measurements of quality of plasmid which indicates the amount of a single product obtained in chemical reaction and associated with plasmid copy number; higher copy number results in better yield (Prather, Sagar, Murphy, & Chartrain, 2003). However, the copy number is an internal attribute of plasmid that cannot be modified during processing. If the low-copy-number plasmids are used, then the yield of final products is likely to drop and the processing time of a single lot of the product is to be extended by running multiple production cycles till the desired quantity is satisfied.

MTO enables the flexibility of production yet brings in issues to capacity utilization. This strategy is primarily suitable for the companies who are aiming at the market of low-volume high-value products. With a series of parallel or identical machines, these firms are able to manufacture a great variety of products and supply customers with the exact specification of products. A typical structure of plasmid manufacturing facility is a flexible flow shop where a series of batch and serial machines would be used to allow flow operation to be in motion. As it is known that the advantage of using batch machines is to allow aggregation of job processing and minimize the changeover between products, while the disadvantage is that jobs have to maintain homogeneous processing features. In case of processing plasmids, the batch fermentation process requires jobs to be grouped by temperature. In contrast to batch mode, serial machine with single capacity of each enables express operation and offers a great degree of freedom, and also, its disadvantage of processing long-lead-time jobs is obvious. Whilst deploying both batch and serial machines, to dispatch jobs over the capacity is apparently critical, and with random demand, capacity utilization sometimes has to compromise with other production goals such as shortening lead time, reducing overall costs, maximizing the total profit, etc.

The flexibility of plasmid manufacturing is not only reflected by the combination of batch and serial operation units but also by product mix strategy. The capacity of a MTO system is not precisely defined as it strongly relies on the product mix to be delivered. Production systems often face a problem of overloading capacity which is caused either by increase in demand or short of resources. Therefore, manufacturers would make decision on whether some products should be partially or completely excluded from the recent production plan due to the lack of capacity at some processing units (Henning & Cerda, 1996). Processing large order in a long production cycle tends to reduce changeover cost/time; however, it occupies a large amount of capacity and delays the latter ones in the sequence. Thus, the order can split to several sub-orders based on the desired production quantity to avoid the long delay for small orders. To decide which product and what amount to be rejected or canceled is a tough task. Theoretically, sub-orders can be identical or non-identical depending on the properties of machines and other production requirements. Thus, the completion time of the final product is the completion time of the last sub-order. Increasing the number of sub-orders can prolong the completion of the entire order while reducing the figure might also be a challenge of resource allocation with capacity constraints.

1.3. Research motivation and approach

Nowadays, owing to the rapid growth of plasmid product market, more and more emphasis has been placed on the industrial-scale manufacturing of plasmids. Plasmid manufacturers are facing a problem of balancing the capacity utilization and demand so as to achieve the higher level production goals and be competitive in the market. However, competition of shared and limited resources among different plasmid products, a large set of diverse constraints and multiple production objectives generate complexities to the scheduling problem. Although there are many research on industrial-scale plasmid manufacturing found in literature, either on quality control during processing (Durland & Eastman, 1998), or growth medium selection (Danquah & Forde, 2007), the operation optimization has rarely been discussed. Thus, an appropriate production planning and scheduling procedure for manufacturing such low-volume high-value product is to be developed.

Scheduling plasmid manufacturing is to deal with the following steps:

- Dispatching orders: determining whether the received order should be scheduled, using plant-specific heuristics such as shortest processing time first

(SPT), longest processing time first (LPT), etc.

- Assigning orders to processing resources: when parallel equipment is used, especially with unequal capacities, a proper organized assignment strategy is to be used to assign orders with top priority to the most appropriate machines.
- Determining order splitting strategy: as mentioned in Section 1.4, large orders tend to be divided into sub-orders to avoid the delay of subsequent orders.

The production system is a unique system with multiple stages where a set of parallel machines with unequal capacity are available to process biological orders in sequence. The primary scheduling and planning objective is to allocate single-period customer orders with various product quantities along planning horizon to optimally utilize the entire facilities or in other words, to level up production throughput. As scheduling problems in a flexible flow shop environment are combinatorial NP-hard, researchers often bypass the complex computational modeling procedure and seek for suitable heuristic algorithms with finite solution space. Due to the complex characteristics of the underlying problem, the purpose of this thesis is to present the formulation of mixed-integer programming and feasible simulation-based heuristic solution to the production planning and scheduling problem in the MTO plasmid DNA plant. Various dispatching rules such as SPT, LPT and first come first served (FCFS), are compared with the proposed heuristic approach.

The data was collected from a company that is doing business with gene therapy community. Required information includes order history and product yield quantity.

Another intention of this study is to provide an extendable knowledge-based system to aid scheduling process. A knowledge-based system, sometimes used as an individual expert system with a database which manages the relevant knowledge, is a programmed system designed to solve problems by mimicking the ability of human experts as giving reasons, explanations and conclusions based on a certain phenomenon or experience which will be regarded as a piece of knowledge through steps of analysis (Durkin, 1990). It has a rule-defining component that allows users to input knowledge representation with regard to application requirement, and an interference engine that performs analogy of human reasoning. In this paper, the proposed system, which works in a narrow domain, is able to store the rules that define scheduling policy, planning procedure and corresponding applicable phenomena using if-then relations. In reality, it acts as an intelligent information advisor aiding workers or employees in manufacturing or other decision-making process. It provides questions to users who work with order scheduling. By taking input responses, it searches for answers and give solution to users. For example, based on the order arrival data, the questions regarding whether the capacity is overloaded, whether the processing of orders utilizes the capacity effectively, etc. will be answered. Although it doesn't have self-learning components, once the rules are developed either through experience or simulation study, these knowledge rules are proven to be used in similar real-world situations solving problems as human experts.

1.4. Terms

To make the thesis clear, some definitions of terms are introduced.

Order: This term expresses the requirement of final products. Customers who are the end users of products define the processing-related specification, required due dates, required amounts, etc. There are two subclasses that represent the term order: final product and intermediate. The word *lot* is often used to demonstrate a single order requiring a set of operations during production. An order may consist of only one lot or several lots.

Transfer size: Products may be carried on processing in various sizes depending on the size of containers or the size of transfer units used. Transfer size refers to the volumetric measurement of containers to be used at each stage. Derived from the concept, transfer lots are intermediate products.

Processing unit: Processing unit is the description of resource (equipment, tools, human, etc.) used in the manufacturing system to perform a particular functioning process.

Production cycle: It expresses a sequence of serial, batch and ancillary operations that performs at processing units over a continuous time period to meet the requirements imposed by an order.

Serial production: a continuous operation process. Thus, a single cycle of serial operation consists of one order only.

Bach production: Orders are grouped to be processed in batch mode. It discontinues the production flow over orders. Thus, each cycle of batch production contains several orders.

Process flow: It specifies a routing along a number of machines or equipment, and possible auxiliary resource requirements at each processing stage.

Yield of product: As a quality scale, it measures the final amount of product to be obtained. If the final yield is less than the desired quantity, then the production goal of fulfilling the order cannot be satisfied.

1.5. Structure of the thesis

The thesis is organized as follows. Chapter 1 introduces the background information on plasmid manufacturing and research motivation. Chapter 2 provides an overall review of relevant literature to MTO scheduling approaches and applications of knowledge-based systems in manufacturing. In Chapter 3, a detailed description of plasmid processing characteristics and problem statement are provided. This is followed by the formulation of a mixed-integer linear analytical model and the description of major methodologies explained in Chapter 4. In Chapter 5, the analysis of historical data is provided and experiments based on simulation scenarios are presented. Finally, Chapter 6 summarizes the results of investigation and concludes the thesis.

CHAPTER 2. LITERATURE REVIEW

In this chapter, a comprehensive literature review is addressed. The review covers literature in three directions: (1) production planning and scheduling techniques in make-to-order manufacturing systems, (2) job splitting, selection or rejection/acceptance procedures, and (3) applications of expert systems or knowledge-based systems in flexible manufacturing environment.

2.1. Production planning and scheduling in make-to-order (MTO)

The simplest production system consists of a single processing unit manufacturing only one type of products. This kind of system is easy to capture and usually seen in small-size job shop. Most industries utilize parallel machines and multi-stage operations to increase the unit-time productivity. The system is shown in Figure 3.

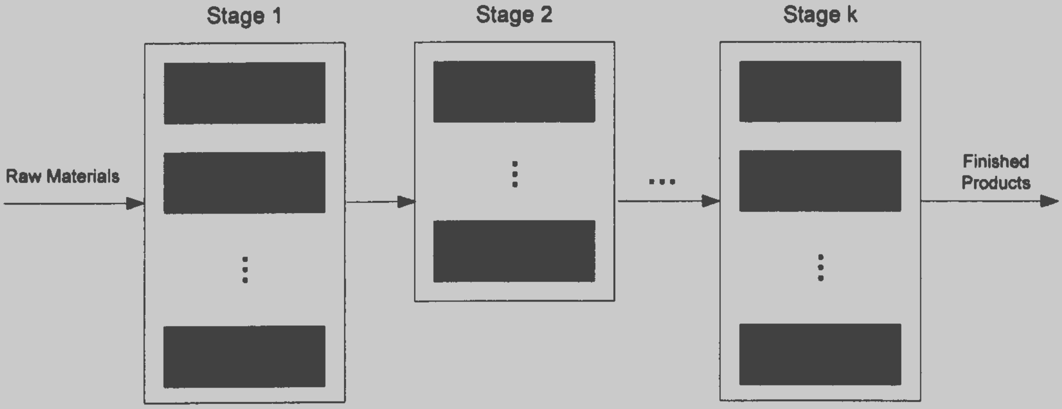

Figure 3. Multistage flow-shop with parallel units.

A set of jobs associated with product orders are required to run through sequential operations with the allowance of skipping one or two of them depending on the processing requirements. If orders follow the same sequence of processing on all the stages, then this type of production is called flow shop. To schedule a flow-shop production system is to specify the orders to be processed and the timing of the processing of jobs on machines or equipment in a specified time window, with an objective of minimizing make-span or total cost, or optimizing the utilization of the facility and so forth. A comprehensive review on flow-shop scheduling with make-span criterion is given by Hejazi and Saghafian (2005).

In general, there are two strategies of operating a manufacturing system: make-to-order (MTO) and make-to-stock (MTS). Make-to-order offers a deal of variety for high-profit products, while make-to-stock maintains inventory to respond to abrupt change in demand (Soman, Van Donk, & Gaalman, 2007). MTS has the advantage of demand forecasting over MTO, but MTO is perfect for orders exactly made to customer specifications. As an example found in plasmid manufacturing, customers are obligated to provide plasmid samples and specify growth and purification requirements to obtain final products from the company. Hence, the company is featured for MTO products only. There are challenges that MTO is always facing: (1) the large storage buffer required holding work-in-process (WIP) inventory, and (2) the changeover incurred due to the great variety of products to be produced. When it comes to receiving a large volume of orders mixed in variety and quantity, the limited capacity in storage and processing stages and the impact of a large number of setups on time-based factors are extremely significant. Hence, the need of planning production cannot be overlooked.

A lot of literature has addressed the issue of production planning and scheduling in MTO production systems which provides adequate insights (Kropp & Smunt, 1990; Sox, Jackson, Bowman, & Muckstadt, 1999; Kolisch, 2000; Arakawa, Fuyukia, & Inoueb, 2003; Neureuther, 2004; Soman, Van Donk, & Gaalman, 2004; Gomes, Barbosa-Povoa, & Novais, 2006; Jalora, 2006; Xuan & Tang, 2007; Chen, Mestry, Damodarana, & Wang, 2009). Their work and solution approaches are summarized in Table 1 (a similar table is also given by Soman et al. (2004)).

Table 1. Overview of literature on MTO production planning and scheduling

Authors/yr.	Subjects addressed	Demand-product and Processing features	Objective and Performance measures	Solution approach
Kropp and Smunt (1990)	Lot splitting in flows-hop Capacity planning	Demand is stochastic Single-job multi-machine flow shop environment	Minimizing make-span Minimizing WIP holding cost	Quadratic programming model
Sox et al. (1999)	Stochastic lot scheduling	Demand is Poisson Process Service time and setup times assumed to be independently distributed random variables	Maximum service level in terms of fraction of orders immediately filled	Simulation-based search algorithm with predefined production quantity upper bound and lower bound

Table 1. Overview of literature on MTO production planning and scheduling (continued)

Kolisch (2000)	Integrated project scheduling and part ordering Coordination between assembly and fabrication Multi-level capacitated lot sizing	MTO demand is random Parts can be purchased from market suppliers Assembly determines the schedule and number of parts to be fabricated Scarce resources No preemption	Minimizing inventory-holding cost in fabrication Minimizing setup-cost in fabrication and assembly	Mixed integer programming with list scheduling heuristics and backward lot-sizing generation scheme
Arakawaa et al. (2003)	Job shop scheduling Capacity adjustment	Demand is stochastic Supplemental capacity is allowed to be used	Minimizing total tardy jobs	Simulation-based optimization with capacity adjustment function (backward and forward simulations) Local search algorithm Parameter-space-search-improvement method
Neureuther (2004)	Aggregate planning Development of monthly production plan over a 12-month rolling horizon	Monthly demand is stochastic and has seasonal change Product has high variety (thousand) Operation sequence, setups and processing times vary among product types Single-stage operation	Minimizing total cost including inventory holding, regular-time and overtime, hiring and firing cost.	Demand forecasting function Linear programming scheduling model
Soman, Van Donk and Gaalman (2004)	Partitioning MTO and MTS Capacity planning and short-term scheduling	Customized perishable product with unpredictable demand for MTO-items No intermediate storage, single-equipment production system with sequence-dependent setups, and changeover times are less	Maximizing the expected profit while attaining the minimum service levels in terms of due-date performance for MTO products and line item fill rate for MTS products	Heuristic illustrative solution with hierarchical framework.

Table 1. Overview of literature on MTO production planning and scheduling (continued)

Gomes et al. (2006)	Determining the insertion point of new orders Categorizing non-reschedulable and reschedulable old orders	Demand is random New orders are allowed to be inserted in to current schedule. Products are manufactured in multi-purpose machines	Minimizing weighted total tardiness	Mixed-integer programming model (MILP) with reactive algorithms
Jalora (2006)	Order acceptance and scheduling in MTO First-party warehouse capacity planning Availability of third-party warehouse	Independent Order arrival processes between different classes Demand can be forecasted by probabilistic method Deterministic processing times Non-preemptive scheduling	Maximizing total profit	Revenue management model with simulation analysis
Xuan and Tang (2007)	Batch decoupling MTO scheduling in hybrid flow-shop Continuity of production without interruption during operation	Jobs are available at the beginning of scheduling horizon Multi-stage system with parallel identical machines Sequence-independent setup times Waiting between two adjacent processes incurs penalty cost to be used in the cost function No preemption	Minimizing total completion time Reducing WIP inventory	Improved dynamic programming for batch-level scheduling Batch-decoupling-based Lagrangian relaxation Algorithm to solve sub-problems
Chen et al. (2009)	Short-term capacity planning Product mix strategy Order acceptance and rejection Setting commitments of delivery to accepted orders	Demand is random Three-stage job shop with single or parallel machines Outsourcing or subcontracting is allowed	Maximizing total net profit	Mixed integer programming investigating scenarios with/without order acceptance optimization

2.1.1. Various MTO scheduling situations

Make-to-order scheduling problem is considered as stochastic scheduling problem with limited production capacity and random demand. Sox et al. (1999) comprehensively review a wide range of quantitative techniques that have been applied to this problem based on the segregation of discrete and continuous time control. Continuous-time representation is referred as Stochastic Economic Lot Scheduling Problem (SELSP). Its solution through simulation-based approach incorporates with heuristic search algorithms on the basis of queuing system. When the number of stages increase to n (n>2), it is commonly seen as a multistage problem although most research considers single or single-stage facilities with parallel machines only. As stated in the survey, regarding the bottleneck stage, the assumption of one bottleneck process in one production system is said to be realistic, because the bottleneck stage may be held up by a secondary bottleneck so that material is delayed at the secondary bottleneck and never arrives at the bottleneck process. Therefore, the cyclic scheduling that executes a prescribed sequence repeatedly is proved to be applicable in such problems. The determination of cyclic length takes into consideration of current and future states; thus, it is modeled as Markov Chain with random task times.

Kolisch (2000) divides a manufacturing scheduling and planning problem with two-level (in macro structure) processing stages into two sub-problems: (1) scheduling problem in the assembly stage and (2) capacitated lot-sizing problem in the fabrication stage. A mixed-integer programming model is formulated in the paper. When a job is determined by the graphical assembly network, with the list scheduling heuristic candidate, jobs are selected by predefined priority values associated with holding cost. As the objective of the problem is to meet the demand as soon as possible, the list of jobs then are transferred to a schedule according to their latest start time so that the production planning procedure can be triggered.

When an inefficient schedule results in tardy orders, schedulers often resolve the problem by either adding more processing units to provide more capacity, or altering the schedule by changing order sequence, or applying splitting technique (Dastidar & Nagi, 2007), or negotiating due dates with customers. The third one is scarcely used because it involves decision-making in upper-level departments. Therefore, production planning in lower-level decision-making departments would mainly deal with scheduling potential tardy jobs if the objective is relevant to tardiness performance metrics, or potential cost-ineffective jobs if associated with cost-based measures. Arakawa et al. (2003) propose a simulation-based optimization method incorporated with capacity adjustment function to minimize the total tardy jobs in an MTO production system. The method consists of two components: (1) a backward simulation which is used to estimate the starting time of jobs by their due dates and (2) a forward simulation that generates prioritized order sequence for each work center. In order to generate solution space for scheduling of jobs, the authors define two parameters, c_1 and c_2 representing due-date lateness and accumulated waiting time coefficients respectively. The sum of the product of variables and corresponding coefficients can be used to justify the generated schedule in each round of iteration during the job allocation procedure. Considering the capacity adjustment procedure independent of job allocation, the authors introduce other two parameters to define the solution space of capacity expansion. This makes the solution space of the entire problem four-dimensional. In this procedure, again, the schedules are generated by the backward and the forward simulations. Based on the above procedures, they further develop a local search method of merging both procedures together to shorten computational time. The method is proved, through experiments, to be effective in practical large-scale systems.

Compared with traditional offline method, online scheduling offers a great deal of capability of handling uncertain demand, unexpected rework and reprocessing during production. Predictive model enables monitoring during production and reactive model allows action to be taken when unexpected events occur. Lau et al. (2003) consider a similar bioprocess system in the manufacturing of penicillium. What makes the manufacturing of penicillium in common with plasmid production is that both require fermentation process, and both product yields become uncertain. Especially for penicillium, its yield is unstable and will degrade over time. Therefore, the authors propose a predictive scheduling model in which the yield of penicillium will be monitored during production. Obviously, the reuse of equipment during rework increases the flexibility of the processing plants, but resource allocation can be very difficult if demand is high and competition for the resource is significant. Gomes et al. (2006) consider the reuse of multipurpose machines in an MTO job-shop scheduling problem. It is called reactive scheduling because new orders are allowed to be inserted into the current schedule to adapt to the change in demand, and changes will be made to the new scenario to the old ones so that some of them can be rescheduled. Several scenarios are tested by applying the proposed algorithm and results show that, with medium-sized example, increasing the number of re-schedulable old orders will create significant raise in the number of operations to be changed.

Although forecasting of product demand in MTO industry is relatively difficult compared with MTS, if a seasonal pattern exists, then a forecasting function based on historical order data can be derived. Based on monthly demand data for a fabrication plant, Neureuther (2004) develops a weighted forecasting function, in which the weight of demand is determined by the ratio of monthly demand to the monthly production volume of fabrication parts.

To keep the inventory level satisfying relatively unpredictable demand and avoid costly setups, MTO, in some circumstances, has to be incorporated with make-to-stock (MTS) production system. Soman et al. (2004) propose a hierarchical scheduling framework that deals with the determination of the amount to be made to stock and the inventory level that has to be reserved for MTO considering the limited shelf-life for food products. Unlike pure MTS and MTO manufacturing systems, the authors point out that there are interactions between MTO and MTS production that is interesting yet unknown. To clearly describe the above framework, Soman et al. (2007) give an illustrative case study in the food industry. Without the aid of analytical methods, the authors discuss the heuristic procedure of solving the short-term scheduling starting from generating order candidates to a series of feasibility checks. These checks ensure whenever feasibility cannot be found there is an approach to remove the infeasibility.

Sawik (2007) develop an innovative lexicographic approach to solve a long-term scheduling problem in make-to-order manufacturing using integer programming formulation. Two objectives are considered, minimizing total tardiness and leveling up capacity utilization. With limited output buffer, the modeled system does not allow orders to be completed early than their customer-required shipping dates.

2.1.2. Capacity planning and order acceptance

Order scheduling and capacity planning are all dependent on the orders available at the facility. There is an adequate amount of literature research on the order acceptance procedures and principles.

Caloss et al. (2003) develop a negotiation platform for accepting orders on the basis of MTO environment in business-to-business commerce. Orders are selected through a bid evaluation that determines whether the system has enough capacity, profit margin of the product, operative cost, etc. In the example discussed in the paper, a mathematical model is presented.

Nandi and Rogers (2003) consider a hypothetical conceptual model. With the order arrival process following the Gamma distribution, orders arrived at the plant are classified into two categories that are regular and urgent respectively. Pairs of accept-then-reject simulation runs are conducted to create order acceptance rule mechanism. If the run of rejection outperforms that of acceptance, then the job should not be accepted. The performance measure of the pair-wise simulation is based on the revenue the orders contribute to the system with 0 as rejected and a positive value as accepted. Although the model is evaluated in the deterministic approach, the authors point out that, with more replications, one can find a suitable confidence interval of performance measure so that the model can be implemented in stochastic process. Any future orders are not allowed to enter the system because of uncertain order arrival.

Jalora (2006) proposes a revenue-management model to be used in planning capacity in the first-party warehouse in order to reduce inventory holding cost. The problem is derived taking into account the inventories on two sides to ensure the on-date delivery of finished products: the available third-party warehouse capacity which is the inventory held by raw-material suppliers and the first-party warehouse which is setup by manufacturers. The order acceptance and scheduling are both on the basis of capacity availability of inventory. Order is accepted only if its opportunity cost of scheduling is less than the profit earnings. Thus, the scheduling policy is defined as which period an order is scheduled to yields the least processing cost under the condition that the order is accepted.

Chen et. al (2009) present a short-term capacity planning mathematical model in which each order has a status indicating either it is accepted or rejected. Rejected orders will not go to production while the accepted have delivery commitment that has to be guaranteed. The model assumes that the orders are to be completed in more than one single period, i.e., a day, and thus, the order assignment variable requires an additional dimension of time. The utilization of resource capacity is rigorously evaluated under two different order acceptance policies: (1) orders are accepted optimally through optimization model and (2) all the orders are selected. The results show that the model can only be used to solve small-size problems as there is linear incremental relation between the amount of jobs and the number of binary variables.

2.1.3. Job splitting

During manufacturing, one often encounters the situation where a big lot should be split. There are two situations where splitting should be applied to a single order: 1) If a manufacturing system equipped with single-capacity parallel machines, when the size of a single lot exceeds the capacity of one of the parallel machines, one has to split the order into several sub-orders to satisfy the capacity constraint (as what has been done in plasmid manufacturing); 2) Orders with maximum completion time that prevents parallel machines from completing simultaneously can be split arbitrarily or equally into continuous sub-lots and processed independently on parallel machines (Xing & Zhang, 2000). Both of them have different objectives; the former deals with the limited capacity of machines, while the latter concentrates on creating smooth workflow in order to shorten make-span.

Splitting is particularly important when downstream processing stages involve batch operations and the delay of preceding orders will results in longer completion time of the entire batch. In batch production, tardy jobs even delay the transfer of all the orders within the batch. Thus, instead of processing the entire batch without splitting, the divided batch can give more flexibility to the system and overlapping operations on two consecutive processing units can be realized. Trietsch and Baker (1993) consider the intermittent idling of machines between processing of two adjacent batches. An example illustrates if the optimal number of sub-lots is employed, it will be able to shorten make-span to a great extent with fewer sub-lots and reduce the difficulty of tracking a large number of them. Integer constraint for the number of sub-lots is not desirable. It can be relaxed by applying fraction in the continuous model, considering the difficulty of solving integer programming in the discrete version. Various numbers of sub-lots are investigated, and a summary of models and their solutions is given.

Dellaert and Melo (1998) address a single-product MTO manufacturing lot-sizing problem where overtime hours are allowed as the extension of capacity so as to guarantee the promised delivery date. They consider a Markov decision process in which each production plan will cover demand in following periods. Four heuristic approaches are evaluated in the paper: (1) At least x orders are to be produced to cover next T periods' demand, and overtime hours are applied if necessary; (2) Overtime is only allowed to fulfill at most one-period demand which limits its usage compared with (1); (3) Production period is uncertain but demand is known and should be fulfilled on the basis of least-cost-per-period; (4) A fixed number of items (a batch) are produced to meet demand every T period.

Their results show that the performance of each heuristics relies on cost coefficients and demand parameters, and there is no unique solution for all the cases.

The problem of scheduling involved with splitting is stated as using a maximum-completion-time estimation procedure and the longest-processing-time-to-split (LPT) strategy to determine which order requires splitting (Xing & Zhang, 2000). In the given example, parallel machines have equal capacity and the processing time of each product unit is the same. What is worth noticing is that the processing of each sub-lot requires individual setup, thus, the total completion time of all the sub-lots form a single order is greater than that of processing the non-split order.

In order to overcome the disadvantage that Enterprise Resource Planning (ERP) system has which is using a fixed production time, Dastidar and Nagi (2007) present a mathematical model that deals with multistage batching splitting. They investigate the effect of move-size or transfer size, batch splitting strategy and batch overlapping which have not been intensively discussed prior to their study. Move-size determines the threshold of batch size in succeeding operation. Mathematical models with and without move-size effect have both been discussed in their paper. Batch is split only according to the lower-bound of maximum completion time of all the operations $\hat{C}_{max}$. For example, if m machines are present and n jobs are available, then the total completion of jobs in each machine cannot be greater than $\hat{C}_{max}$. The authors comment that this method maintains the minimum number of splits in a single job and prevents it from splitting over all the machines when LPT method is employed, especially with nonzero setups. Therefore, their results indicate the unnecessary setups have been effectively reduced.

Xuan and Tang (2007) present a hybrid flow-shop scheduling problem with batch operation in the last stage. The problem is modeled as $P \mid split \mid C_{max}$. The serial batch operation considered in their study restrains the jobs from being released before the last job leaves the stage. Considering transportation time separating from processing time, batching decoupling is realized through Lagrangian relaxation. The authors model the relaxed problem as multiple sub-problems each of which corresponds to a batch.

Shim and Kim (2008) extend the concept of sub-job to unit-job in batch production where each job corresponds to a production order. A sub-job is comprised of a set of unit-jobs. The unit-jobs are considered identical, while the sub-job can be with different sizes or involve various numbers of unit-jobs. Each sub-job is said to be processed on one machine only at a time. Thus, the jobs are classified into three types: completely-scheduled jobs in which all their sub-jobs are scheduled, partially-scheduled jobs with some of their sub-jobs scheduled and unscheduled jobs indicating none of their sub-jobs are scheduled. Different from what has been discussed in Xing and Zhang's (2000) paper, the setup is not required for processing sub-jobs if from the same job. Moreover, the number of unit-jobs within a job is predefined, and the allocation of the sub-jobs is actually to assign unit–jobs to each machine. The authors apply a branch-and-bound approach to solve the problem.

2.1.4. Task compatibility

Batch operation utilizes the advantage of machine capability of processing more than one order at a time. However, in some situations, there is an additional constraint to use batch operation; that is when orders in the same batch are compatible. The compatibility refers to similar processing features, such as time, humidity, agitation speed, etc. Oulamara et al. (2009) discuss a two-stage hybrid flow-shop with the consideration of batch task compatibilities which is defined as tasks sharing the same value of processing duration on batching stage. They consider each task in a batch has a different processing time from others, and the processing time of task j falls into the interval of $[a_j, b_j]$, where a_j and b_j are non-identical for all tasks. The compatibility relation is represented by a compatibility graph where each edge indicates that a pair of tasks is compatible. In another similar research, tasks are selected to be a batch by using the LPT rule on the discrete stage and full-compatibility-batch-largest-processing-time (FCBLPT) rule on the batching stage (Bellanger & Oulamara, 2009). The authors then investigate several scenarios where the number of machines in each stage is variable, and heuristics along with their worst-case analysis in which number of machines in each stage is variable.

2.1.5. Order assignment

Rim and Park (2008) consider the order priority during order assignment when the resource allocation is restricted. When there is a situation that resource orders requested is limited but the number of orders exceeds the resource limitation, the excessive orders are to be carried forward to the next planning period, i.e., day. Hence, as described in the context, the priority is given to the orders that have been transferred from previous periods, and the just-arrived orders will be assigned a lower priority. The method separates the old orders from the new ones and it reduces the interference dimension of the problem because the decision variables include new orders only. The authors use weighted performance metrics to differentiate the orders from important customers and those less important.

2.2. Knowledge-based scheduling

The use of knowledge-based scheduling approach has been identified as a mechanism to allow applications of public domain heuristic knowledge and design specific scheduling rules for a particular processing environment (Henning & Cerda, 1996).

From early years, the attempts of using integrated scheduling systems were fruitful. Brancaleoni et al. (1988) present an integration method of simulation and knowledge-based system in printed circuit board (PCB) manufacturing plant. This knowledge-based simulation system removes the drawbacks of conventional simulation models in the aspects of inflexibility, i.e., a set of a fixed number of parameters, and difficulty of application, i.e., requiring low-level programming language. The former is critical for manufacturing facility with a changeover period where the specification of product orders may change suddenly, and the latter concerns the user-friendly-application issue as essential, which is also the reason that knowledge-based system is adopted in various industries to aid decision-making process. As an extended work, Palaniswami and Jenicke (1992) illustrate their conceptual knowledge-based simulation scheduling model in a study of hypothetical manufacturing job shop where two sets of variables regarding processing characteristics are considered. The first set contains common processing variables, while the other one involves uncertainty that non-experts could not tackle with without considerable knowledge of a certain manufacturing system. Therefore, using rule-based knowledge system resolves the problem and provides the inputs for the simulation of manufacturing scheduling. In general, the decisions involved in the simulation model include job acceptance, increase of available processing time, reduction of batch sizes, etc.

The state-of-art of embedded knowledge-based scheduling systems with applications in flexible manufacturing system (FMS) is comprehensively reviewed, where a set of identical or complementary numerical controlled machines are present (Gonzalez, Garcia,

& Centeno, 1996). In such system, a simulator is usually connected with a data communication software and knowledge base or several knowledge bases are integrated in the controlled simulation environment. Therefore, the knowledge-based controller could monitor and detect problems in FMS during production. Whenever a problem occurs, the controller will be able to search for a suitable solution resided in the knowledge base. One big advantage of the knowledge-based controller is that it interacts with FMS as well as the database so that the on-line scheduling and planning control could be triggered and the problem detection is effective. Zeigler et al. (1996) give another example of embedding expert-system elements into the control function of object-oriented simulation environment.

Knowledge-based system is especially useful in predictive and reactive scheduling in process industries, i.e., chemical process systems (Henning & Cerda, 2000). When unexpected events occur, there is a need to adapt existing schedule to the instant change, and the reactive scheduling can handle the situation with explicitly-defined scheduling knowledge in the knowledge base. As a plant involves external manufacturing lines and packaging as the last stage, a carefully-designed schedule will help predict the delay and unavailability of intermediates in the packaging stage. When a scheduling problem has multiple preferred performance objectives, it is not usually easy to come up with a unique cost function. Thus, while finding the function is not available, the decomposition of the scheduling problem into sub-problems is a way to solve the difficulty. Although the decomposed system may not eventually reach an optimal solution, a satisfactory schedule can be generated. In case modifications are required to be made immediately, human experts can always adjust parameters through interactive module.

Dorin and Pănescu (2001) deploy an expert system in controlling a computer-integrated manufacturing system where intensive information flow increases the complexity of production system compared with classical systems. Instead of relying on non-heuristic algorithms and formulations, they set up two rules on the basis of human knowledge and experience to control processing stages. Workers do not require specific knowledge to determine the number of pieces to be manufactured on each machine, how many to be manufactured for each type, whether pieces should be placed on conveyors, etc. All of the rules with respect to the manufacturing decision are predefined in the knowledge-based control system. Hence, the expert system surpasses the difficulty of using mathematical optimization methods, which considered as user-friendly.

Short-term scheduling issues involved in flexible manufacturing system (FMS) could be influenced by the change in management of intermediates and tools. Ozbayrak and Bell (2003) present an example of knowledge-based scheduling and control of tools and parts in flow-shop in order to obtain the benefits of more efficient resource utilization, greater control of tools, and a more dependable rapid adjustment of production requirements due to unexpected malfunction and poor performance. They develop a step-by-step model which identifies the working environment and then selects the best strategy with the incorporation of rules and user criteria. The decisions of selecting the appropriate tool to process a particular job and whether to split a single batch are made by applying rules predefined and the sequence of processing jobs is also being managed by the knowledge-based scheduling module.

As far as Halevi and Wang (2007) concern in their paper, the provision of easy-to-use planning tools is very important to those decision-makers in manufacturing plants who are not experts in either economics or operation management. They propose a priority rule-based knowledge-based manufacturing system with "open" database that the process planner does not make decisions out of his field of expertise but generally generates "road map" where an appropriate routing of processes is defined. Rules of priority include data collected throughout the production on all stages with respect to the resource allocation, capacity constraint, marginal profit requirement and so forth. Based on the data acquired, the expert system can be used in resources planning, cost evaluation, profit forecasting, revenue and budget management, shop-floor scheduling, capacity planning, etc. Then the decisions are made by intelligent search method by matching optimum values of performance measure.

Conventional knowledge-bases or knowledge-based systems would simply neglect the effect of time during the application. However, a knowledge base is built on the data collected from real-life experiences which might be collected at a time point, a single period or even multiple periods. Some do change over the time, but some do not. Therefore, according to their evolution property with respect to time, they are classified into two categories: time dependent or time independent. Lorentzos, Yialouris and Sideridis (1999) consider the time evolving issue in a knowledge base. For time-dependent data, the authors present a validity time knowledge-base (VTKB) where the validity time of a piece of data must be represented in the rules with lower bound or upper bound or both. If the validity time changes in the future, the old version of validity time will be replaced by the new version by applying validity function so that the database can be maintained.

Although a great amount of research has been conducted in either production planning and scheduling in MTO environment or knowledge-based system in manufacturing application, the study underlying is different from them in many aspects. The production system is a hybrid flow-shop where discrete machines and batch machines are set up in sequence and jobs are processed in the same order. The objective considered is to maximize capacity utilization with respect to physical capacity of equipment and available processing time in manual-intensive stages. Order acceptance only deals with selecting orders from currently- and previously-received ones, and no rejection is allowed. The order sequence in the production is not considered as it does not influence the completion time and setups are not taken into account. Splitting an order is only considered when a job is "big" enough which will be given more detail in Chapter 4. In addition, the orders which not scheduled in the previous period will be offered higher priority than orders received today. The due-date constraint is relaxed since the customers tolerate a relatively long period for orders to be delivered. Consequently, the developed model and method are only to schedule orders to the resource capacity.

As for knowledge-based system, although much has been done in its integration with systematic control mechanism, the usage of knowledge-base is basically limited to provide inputs for control systems (Brancaleoni, Bugno, Cavalloro, Neuss, & McLaren, 1988; Palaniswami & Jenicke, 1992). The acquisition of knowledge through literature review, survey and human experiences are major means discussed in the literature (Mohamed & Celik, 2002; Brancaleoni, Bugno, Cavalloro, Neuss, & McLaren, 1988). However, not many of them emphasize on the two-way interaction of the knowledge base and control system; that is on one hand, the control system uses knowledge rules to determine the sequence and the route of jobs and resource allocation, etc., while on the other hand, the knowledge-base becomes evolved by receiving feedbacks from assessment of the performance of the manufacturing control system. Therefore, updating and maintaining knowledge base can be either done manually (Lorentzos, Yialouris, & Sideridis, 1999) or by creating a particular self-learning mechanism. This research reverses the above procedure of applying knowledge base to a manufacturing control system where the production output will provide input for creating rules and keeping them up to date. During the research, simulation will be used to evaluate the inputs and the outputs of production and offers expert system a good source of knowledge.

Therefore, the focus of this thesis is to help an MTO plasmid manufacturer to decide which order to be selected using the rules defined in the knowledge-based system and additionally set up a plan for the current production period.

CHAPTER 3. SYSTEM CONFIGURATION AND PROBLEM STATEMENT

This thesis investigates the order scheduling and capacity planning problem at the factory level in a plasmid manufacturing company, in which each production process corresponds to a unique shop floor comprising of multiple stages with parallel serial and batch machines. The objective is to maximize the throughput of production in terms of total production quantity over the given planning horizon.

This chapter addresses the configuration of the system in detail. The complexity of uncertain completion time, variance in yield, make-to-order characteristics, and fuzzily-defined capacity is further discussed in this chapter.

3.1. System configuration and process description

Plasmid manufacturing is one of the components in process industries. Other examples include chemical industry, food industry, etc. Prior to production, the company receives orders from customers with the provision of the original samples. The planning and scheduling procedure takes place when samples are gathered at the facility. Currently, the company is using a manual scheduling technique to determine the sequence and candidate product orders all based on the production manager's knowledge and experience. As demand is increasing, this type of method limits the overall performance of the plant, and there is a need of improved mechanism. The typical production process flow can be found in Figure 4.

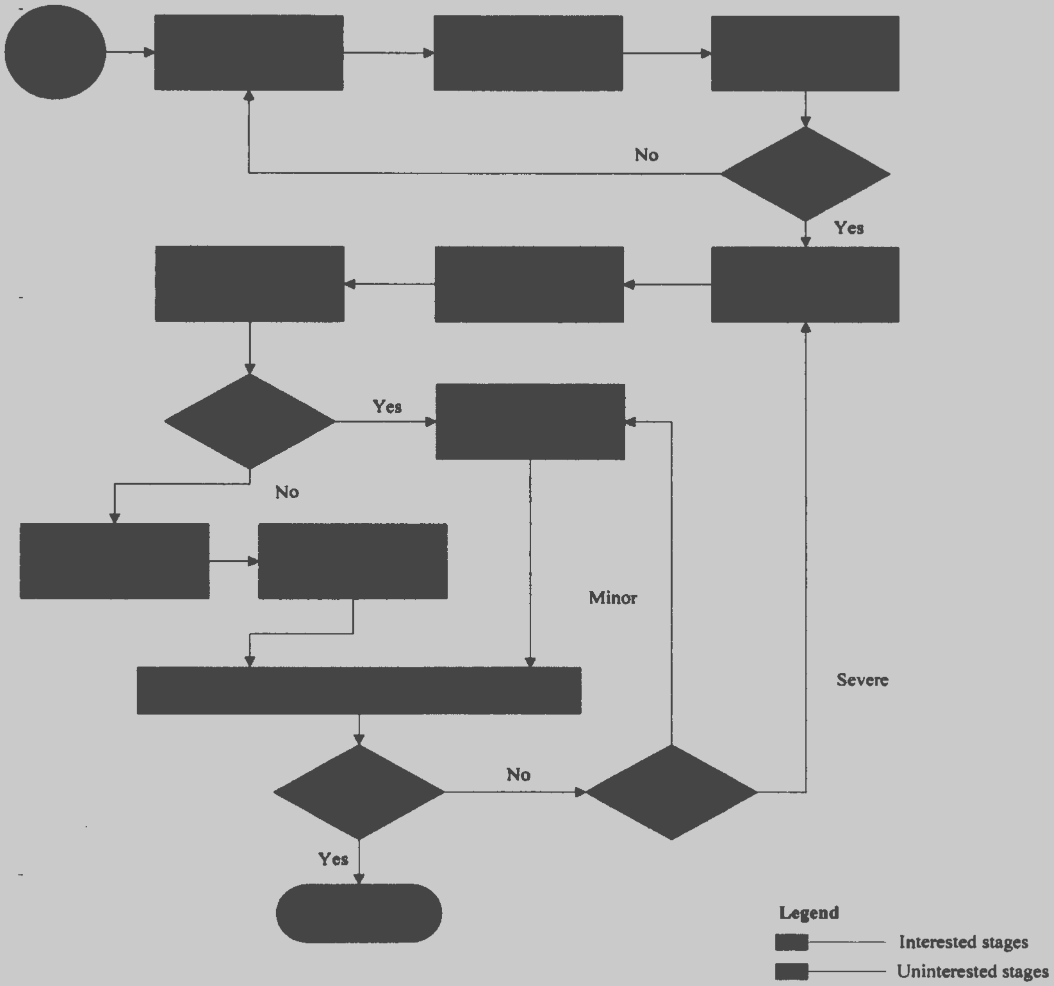

Figure 4. Process flow chart (stages interested are those involving uncertainty and in which planning and scheduling decisions are making great impact on).

To understand the flow of production, a detail description of sequential plasmid manufacturing processes is discussed in the following sections.

In some stage, operations are comprised of manual and machine work. Apparently, manual work involves the most serial operations where orders have to be processed one by one, which is known as single-capacity operation, whereas batch machines in most of stages can accommodate more than one order simultaneously so that the output of a batch machine is greater than one.

As mentioned in Chapter 1, the capacity of the system is not precisely defined because of the issue of product mix. The number of machines present in one stage may be distinct from its preceding and succeeding stages, and the machines are even unequal in capacity. From the plasmid sample to the final product, an order requires to pass at least 9 stages: transformation, starter culture, screening, inoculation, harvesting, lysis, concentration, polishing (conditional), quality control (QC) and shipping, in which uncertainty makes little impact on operations in the subsequent stages of Polishing.

As marked in Figure 4, the stages of interest are those stages in which either uncertain demand, or uncertain yield involved, while the uninterested stages involve less uncertainty.

3.1.1. Transformation

Transformation is the first process in which the operator transfers a fragment of received DNA samples into bacteria cells in order to make it grow effectively and efficiently under a proper environment. In this stage, temperature and humidity will be controlled consistently.

Certain amount of plasmids of each sample is first inserted into a tube to be separated from original samples pre-frozen in the fridge. Then, the plasmids stay in a warm-up hood for few minutes to adapt to a higher temperature. Small glass plates containing agar gel are used to accommodate the plasmids and bacterial host strains which will be later used in small incubators. Depending on which temperature is required, the operator selects a small incubator in which the plates stay in stacks for 12-16 hours, or until sufficient cell colonies have successfully obtained. Thereafter, the processing information will be put into a database, which includes the lot number, time transformed, number and size of tubes, and temperature setting. Meanwhile, the operator's login is also recorded. The general procedure is demonstrated in Figure 5.

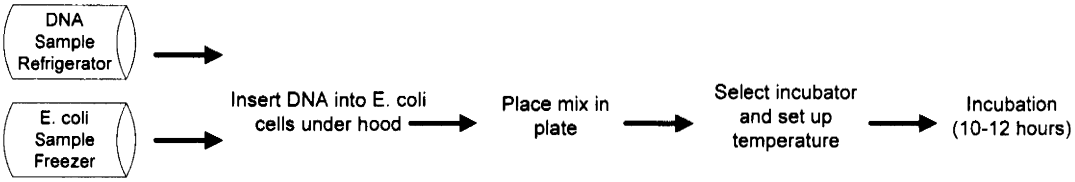

Figure 5. Transformation procedure.

Only a fragment of plasmid DNA will be inserted into one tube. A pDNA sample is split into several sub-samples, each of which is exactly the same as its parent plasmid. They are technically known as starters. As the candidates of the final product, all the starters will be treated equally, i.e., using the same temperature and the same medium, but none of them yields precisely the same amount. The use of starters could limit the variability of cultivation and provide more options if one of the starters fails in the first growth.

3.1.2. Starter Culture

Followed by transformation is the second stage, starter culture. After transformed plates have been incubated, the operator begins culturing starters. Under the same hood, the operator transfers colonies from the plates to falcon tubes containing liquid growing medium which are much larger than previous tubes used in transformation. These tubes are then placed into bigger incubators, distinguishing from those non-shakable incubators, which also create agitation to allow tubes shaking during the incubation. After 8-10 hour shaking and incubating, the tubed orders can be moved to screening process. The purpose of starter culture is to provide gradually growing environment for plasmids to adapt to the change in volume/size, particularly the change in medium (from gel to liquid).

3.1.3. Screening

Screening process is designed to investigate the quality of small-scale growth as well as the compatibility of plasmids with E. coli strains. Only a few of DNA is extracted from the E-coli cells using a small centrifuge with buffer liquids. A dye is added to the DNA and injected into a gel. An electric current passes through the DNA in a process called electrophoresis (O'Kennedy, Baldwin, & Keshavarz-Moore, 2000). Then a picture of the gel is taken to capture an image of the DNA string. Operator will upload the image into the database and make decision with respect to the quality of the growth. If one of the starters in an order fails to pass the screen inspection, then the starters with better growth rate will be chosen to be used in succeeding processes. If the starters all fail, then the order has to be re-transformed. Figure 6 represents the general procedure to perform Screening operation.

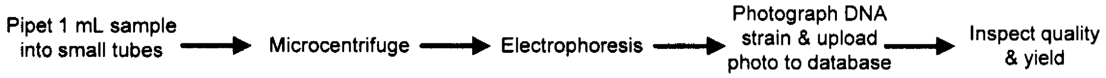
Pipet 1 mL sample into small tubes
Microcentrifuge
Electrophoresis
Photograph DNA strain & upload photo to database
Inspect quality & yield

Figure 6. Procedure of Screening.

3.1.4. Inoculation

Inoculation is the stag of fermentation. Successfully cultured plasmid starters are then selected to be used in fermentors. Only the best one will be selected and it is stored to become a banked cell. All the banked cells are stored in the freezer and supposed to be used when final yield is unsatisfied and more growth is needed. A small amount of sample extract (e.g. 1ml) is taken from the original starter, and spiked into a glass flask with specific volumetric size. Each starter corresponds to a flask filled with a medium of volume thousands of times larger than that of the sample itself. To determine the usage of a specific medium, one has to follow the technical instruction which will be discussed in Chapter 4. Growth again takes place in the fermentors. Inoculation takes around 16-24 hours depending on the sufficiency of growth. It is in this stage that large-scale yield is realized. Because the growth of plasmids is not always so perfect that media may not be fully utilized and neither is the sample of plasmid, the uncertain yield becomes an uncontrollable factor in plasmid production. If the yield is relatively low, then the final yield will decrease due to material loss in concentration and clarification resulting in under-satisfied demand. The process is presented in Figure 7.

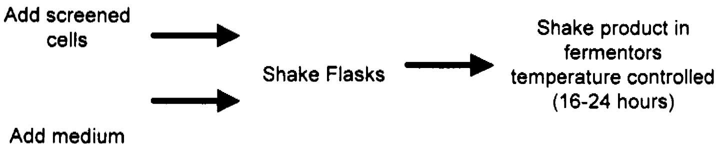
Add screened cells
Add medium
Shake Flasks
Shake product in fermentors temperature controlled (16-24 hours)

Figure 7. Inoculation procedure.

3.1.5. Harvesting

This is a key step where membranes of E-coli cells are fouled, surface area is destroyed and plasmid DNA begins separating from its host by centrifuging. The operator removes overnight-grown plasmids from a flask into a same-sized Beckman jar or bottle. Centrifuges are batch machines that process several bottles at the same time. Harvesting consists of a number of repeating runs, based on the overall number of bottles in the production cycle. Every run is made up of two sub-processes: spinning and cleaning. Spinning is a machine-only process where settled bottles are being spun and bacteria cell structures are falling apart. A few minutes later, the operator removes bottles from the centrifuge machine and begins to clean up it. Each order only goes through the spinning-cleaning cycle once. This process is described in Figure 8.

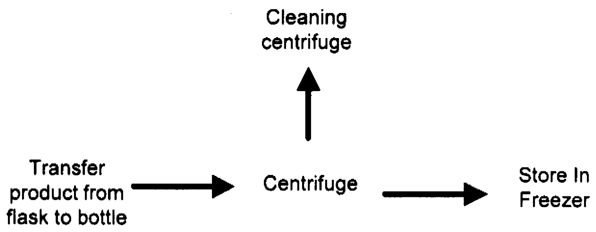

Figure 8. Harvesting procedure.

3.1.6. Downstream processing steps

Followed by harvesting, lysis is the first process of eliminating the disruptive elements brought by bacteria cells. Three buffering steps and one re-suspension step are involved. Harvested samples in bottles are highly concentrated mixture of solid and liquid with dead and broken E. coli cells and target plasmids. Thus, buffers P1, P2, and P3 are used to suspend and neutralize the mixture so that the plasmids and cells can be finally delaminated and become liquid again, which is supportive for the following purification steps. Figure 9 depicts the procedure of lysis.

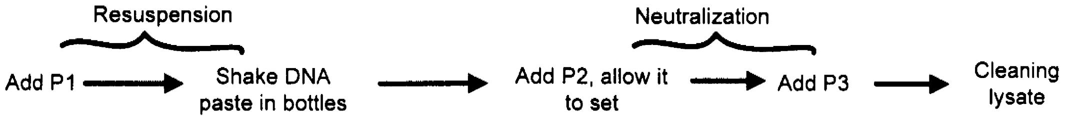

Figure 9. Lysis procedure.

AEX is short for Anion Exchange, the succeeding step of lysis, which is constituted of five sub-processes: clarification, gravity flow, precipitation, washing and drying. The solution from the previous processing stage must be stored in the fridge to keep plasmid active and ready to be used in AEX. Details are illustrated in Figure 10.

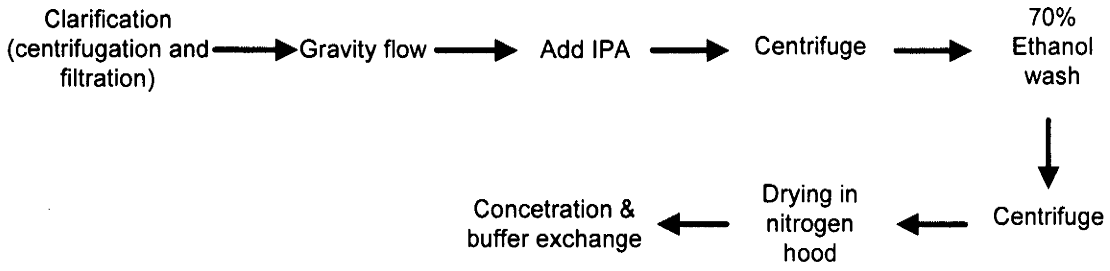

Figure 10. Procedure of AEX and its following processes.

Polishing is a step to further clean up endotoxins brought by bacterial cells, especially essential for plasmids grown in large-scale. To clarify, the endotoxins are toxins associated with certain bacteria which are structural components of the bacteria released mainly when bacteria are lysed. Technically, polishing is only required when large orders are present.

Concentration is a finishing process following successful purification which reduces the amount of liquid purification buffers and quantifies plasmid DNA. It is also necessary to exchange the unwanted salts or buffers for a desired storage solution.

Quality control ensures that the final product ready to be delivered meets the requirements of having sufficient quantity and satisfied quality.

3.2. System characteristics

Compiled from the observation of the plant, the following characteristics of the system are provided.

3.2.1. Plant characteristics

1. Multi-product flow-shop neglecting changeover time, sequence-dependent set-up and cleaning times (except for Harvesting) because of small product size and low variation in processing features.

2. Permutation flow-shop where jobs are processed in M stages (some will skip one or two stages) following the same sequence of operations.

3. Hybrid production system where stages comprised of serial and batch operations.

4. Infinite intermediate buffer and finished inventory due to the size of orders.

3.2.2. Product characteristics

1. Variation in source and quality of raw materials.

2. Product quantity or weight as the unit of measure in milligrams varies greatly.

3. Urgent orders may not desire a large quantity while small-quantity orders are adapted to be urgent.

3.2.3. Production characteristics

1. The plant receives sufficient amount of sample material for each order even though there are chances that some amount might be wasted or used as trial.

2. Processes have variable processing time for serial operations, while batch operations use the same processing time.

3. Special processing features, such as temperature, are required to be consistent and exclusive processing policy is applied indicating only orders with the same temperature will be processed in the same fermentation machine.

4. Processing is restrained to a specific length of time period. Any process starts at a particular time point, and it is fixed. The duration of each sub procedure varies but still a fixed length of time represents the availability of the entire process.

5. Re-processing is required to be done when the quality of orders is lower than expected.

6. Serial processes are labor intensive whereas batch operations involve machine work only. Each serial process can process no more than one order at a time, and each batch operation performs up to n orders simultaneously in batch mode. The processing of order j at serial stage requires unit time l_j, and the batch processing times for all orders in a batch are equal to p_i , where i is the stage index and p_i applies to all the orders regardless of the sizes, so that orders start and finish at the same time.

7. The production rate is mainly determined by the capacity and the total order quantity.

8. The sequence of scheduled orders does not influence the processing time in serial stages.

The development of the production planning and scheduling framework has to take into account all the factors above, even though sometimes only a subset of them is present.

3.3. Problems concerned

Now that the plasmid processing industry characteristics have been addressed, the problems concerned in the study with regard to decision making on scheduling and planning are given below.

3.3.1. High variation in demand

MTO is commonly adopted in a situation where demand variation is high and it is difficult to anticipate the pattern of future orders. The requests of plasmid products come every day from versatile sources aiming to be used in various circumstances. According to the data analysis on the ordering history, around 50% of customers were return customers in the past fiscal year. Even though, because the major raw materials, sample of plasmids, are supplied by customers, it is not able to anticipate the characteristics of genes, the time when next ordering will occur, the frequency of ordering next year, and the amount of orders that could be. On the other hand, the potential increase in both the number of customers and their needs of plasmids are totally unknown to manufacturers. Unable to conduct demand forecasting gives a big problem to the production planning and scheduling, since there could be the case when new scheduling process is about to start while capacity is not available.

3.3.2. Uncertain production duration and utilization of capacity

Plasmids products can be divided into several parts to be processed independently instead of being as a whole resulted in longer delay of subsequent orders. On one hand, the division or splitting gives more freedom in capacity utilization. On the other hand, the more sub-orders are divided, the longer processing of a single large order could be.

The splitting strategy continuously makes impact on capacity utilization. Capacity utilization is likely to be overlooked when processing large orders without splitting. However, although it is scarcely found in order history that large orders had short due dates, there could be the case that some of the urgent orders desires large amount. Therefore, from the management point of view, there is a need of switching splitting strategy to non-splitting practice when the urgency occurs.

3.3.3. Processing characteristics vary among orders

Processing characteristics refer to processing time, treatment requirement (i.e., heat), buffer usage, etc. In batch operations, orders sharing homogeneous processing characterizes are processed together when batch machine has enough capacity. These characteristics influence the yield-ability of bacteria and further make impact on plasmid yield. In the production system considered in the study, a special heat and agitation treatment, the use of medium broth and various bacteria strains are considered.

3.3.3.1. Temperature issue

Plasmid growth requires heat treatment. Plasmid samples will be assigned to optimal temperatures selected according to their biological features. Temperature-adjustable equipment is setup for the temperature that orders are required to be processed in. In real practice, optional temperature settings include more than one situation that provides tempered environment for the growth of bacteria. Technically, the optimal temperature is decided based on laboratory experimentations. Applicable temperature settings discussed in literature are 30 °C and 37°C (Durland & Eastman, 1998). To achieve adequate yield, temperatures other than optimal should be avoided. That is to say the temperature-controlled equipment and the orders grouped by temperatures are in one to one relation.

The assignment of order groups to equipment needs to be careful because the inappropriate use of temperatures would reduce the overall utilization of entire facility on two sides. First, not all stages have equal-sized temperature-controlled equipment. Some stage can process twelve 30°C orders, while its downstream can only do six. The decision to the problem is either to remove six from the upstream process so as to satisfy the downstream capacity requirement but decreases the utilization in the upstream, or to use additional runs that is likely to result in overtime work. Second, overloaded equipment is adapted to shift workload to other temperature-controlled machines. For example, assuming there are three groups of orders to be produced, 30°C, 34°C and 37°C. The total numbers of orders falling into these groups are 7, 2 and 4 respectively. Three identical machines are available at the time of planning and each has a capacity of processing 7 orders simultaneously. When there is enough capacity, all the groups can be processed each of which occupies an individual processing unit (Figure 11).

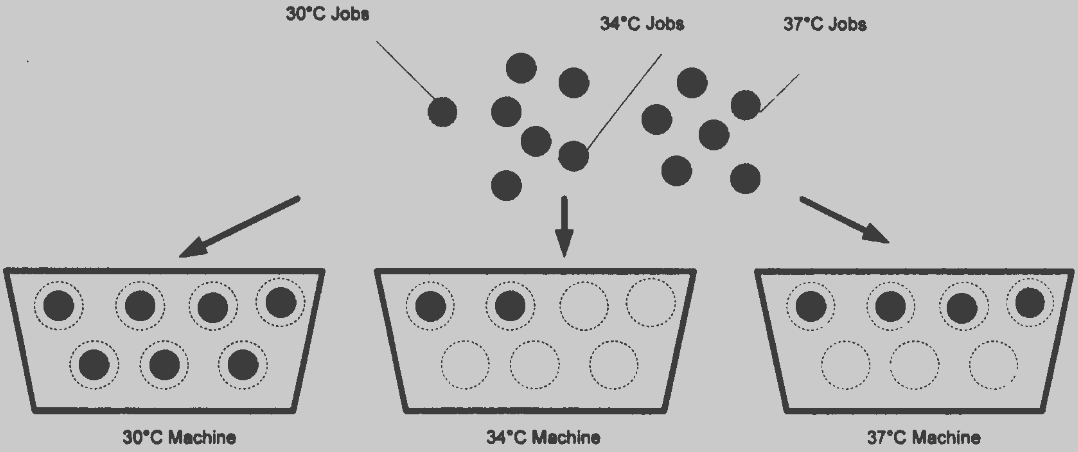

Figure 11. Representation of equipment assignment and distribution of orders (distributed orders in solid line). Uncolored circles in the machine indicate available capacity.

However, there is always the situation where certain group contains more orders than others and the number exceeds the capacity of a single machine. A decision has to be made whether to use extra machine to process the large batch instead of distributing the order groups evenly. The selection must be made based on the priority of these order groups (Figure 12).

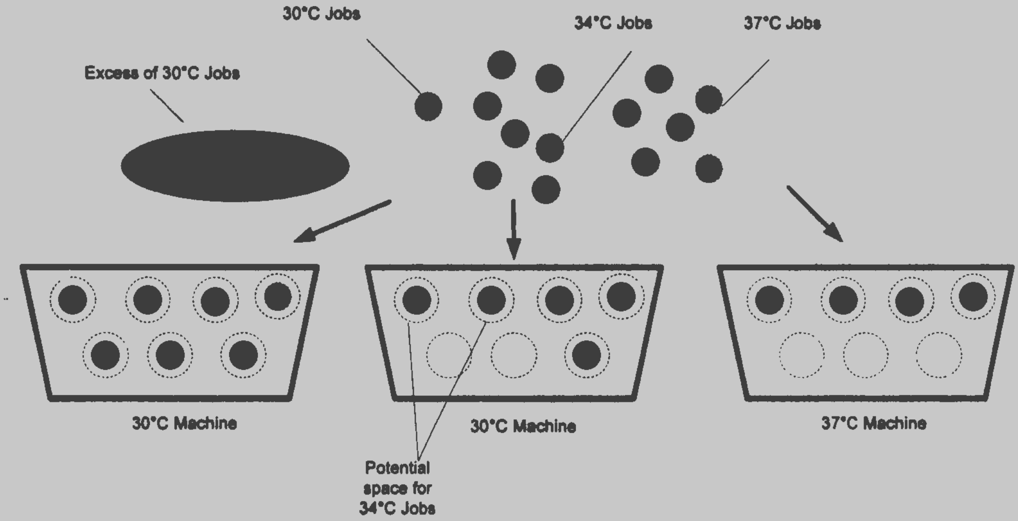

Figure 12. Uneven distribution of orders (distributed orders in solid line). Uncolored circles in the machine indicate available capacity.

3.3.3.2. Use of medium broth

Medium provides nutrient supplies to bacteria. The company uses various composited medium (i.e., LB) during production. The combination of temperature and medium type brings a great deal of variations to amount of yield. For example, LB is used frequently at 37°C because bacteria grow more rapidly at lower temperature setting compared with using other medium in the same condition, whereas other media, i.e., MY, might give better yield at 34°C but generates longer growth time. The yield rate captured from historical data and analysis of yield results are discussed in Chapter 4.

3.3.3.3. Various bacteria strains

In order to know the productivity of unidentified plasmid incorporating with bacteria which have particular structure, the company needs to use various strains from a single bacterium to make test of yield. A yield test, as its name tells, basically is rarely different from the regular production. The only thing that needs attention is that only a particular amount of plasmid as well as a mount of medium will be used to perform the yield test. For example, if there is one large order with deliver quantity of 1000mg, and it is not commonly produced in the plant, then the operator decides to make the yield test on 3 different types of bacteria strains first. Each time, only 1L growth medium will be used. Then, according to the final yield amount, the operator will decide which strain is suitable to go with the growth of plasmid. Because strains all have discrepancy in productivity, the yield test is the only way that identifies the difference and makes the best utilization of bacteria and plasmid samples. To select a proper strain to make starters require technical knowledge that is not able to be obtained by non-trained schedulers. Therefore, to simplify, in this thesis, only known strains found in the historical data will be investigated.

3.3.4. Time availability of labor

Manual work always has a time limit which is defined as the capacity of labors. For instance, the operator in transformation stage can handle maximum of 20 samples or orders within one hour. If the allowable maximum work length in transformation is one hour, then no more than 20 orders can be processed otherwise overtime is applied. Currently, the company uses the fixed processing time in manual stages. As it has been discussed earlier, it is not always ideal for orders varying in size as it limits the overall performance of the facility.

3.3.5. Leftover, throughput and lead-time

Capacity limitation and lack of scheduling rules in the current facility directly lead to the leftover of unscheduled orders. That is, the orders received but not yet scheduled become leftover and have to be scheduled in the next planning period.

It is easy to foresee that if the demand keeps growing, and there is not any efficient tool to deal with large number of leftovers, throughput and lead-time will become big obstacles to obtain financial goals.

Meanwhile, it is noticed that the large orders dominate the small ones when they are present in the manufacturing progress, because a large order takes more than twice the time when an order can be finished in a smaller amount. For example, a 10 mg order can be fulfilled in 5-7 days, but a 2000mg order may take as long as 3 months. Hence, when dealing with this type of orders, an appropriate algorithm that solves the issue by splitting large orders needs to be developed.

3.4. Problem statement

In real practice, the company has adopted an empirical scheduling method with which orders are scheduled manually based on the scheduler's knowledge and experience prior to and during the manufacturing process. Operators are required to input the information regarding each order into computer database so that orders can be monitored throughout production. In this way, any problem occurred during production will be handled immediately. However, manual schedulers could not solve the problem of matching capacity with demand perfectly by deciding order sequence and dispatching orders optimally so as to boost the overall performance of the system. Hence, there needs a scientific method to rigorously investigate the system and then a solution to the problem of the production planning and scheduling while demand is randomly distributed can be proposed.

The steps of conducting research on plasmid manufacturing include the followings:

- Understand the system in terms of process flow analysis, identification of system and product characteristics and capacity analysis.
- Collect historical data on ordering history and product yield.
- Analyze data and propose a reasonable heuristic approach.
- Evaluate the approach.

Considering the production objective and the requirement of bio-process system, in this thesis, production planning and scheduling problem with stochastic demand and product yield are considered. The objective is to maximize the production throughput, i.e., to complete as many jobs as possible in each period while not violating any capacity constraints. The objective is especially important in situations where a large number of jobs wait be to finished, each in a relatively low volume. Therefore, the perspective of this research is to activate the following things:

- Accept or reject orders to be scheduled: deciding whether the order should be scheduled based on the plant-specific heuristics at the time of receiving orders.
- Assign orders to process resources: when parallel equipment is used, especially with unequal-capacity constraint, a proper assignment strategy is to be used to assign the top priority orders to the most appropriate machines.
- Determine order splitting strategy: as mentioned in Chapter 1, large orders tend to split to avoid delays of subsequent orders and satisfy the capacity constraint.
- Estimate the throughput: by applying heuristic splitting and selecting rules, the assessment of throughput tells whether the system has improved or not.

CHAPTER 4. METHODOLOGY

In this chapter, a mathematical formulation is presented. Since the complexity of the problem requires multi-dimensional computing with the uncertainty of biological process, a heuristics approach is proposed. This includes the heuristic method of selecting of jobs to be processed, job splitting strategy and simulation approach to evaluate the function and results of these heuristic. From the management point of view, the ease of application of the model is essentially important as it may require particular knowledge to conduct planning. Therefore, based on the experiment results, a rule-based expert system generalizing similarity of scenarios is designed for manufacturers to make decision upon order entry.

4.1. Performance measure

The underlying study is to identify the optimal scheduling rules with the realistic environmental parameters. Hence the objective is to schedule as many orders as possible so as to maximize the throughput of the system. Since the output production quantity is reflected by the desired quantity of each customized order, the goal is transformed to maximize the overall desired product quantity in terms of plasmid weight in milligrams. Therefore, the performance measure is the total product quantity.

4.2. Assumptions

Several assumptions are made to clarify and simplify the manufacturing process. Because the plasmid production cannot be done in a single period, say one day, in order to model the scheduling system, the concept of planning period is used. A production cycle consists of several planning periods, therefore:

1. Planning periods are consecutive, equal-length divisions in planning horizon of a particular time duration;
2. Scheduling happens at the beginning of each planning period. It is done once per period only;
3. Orders are available at the beginning of each planning period only. Orders that arrive at the end of the period are carried forward to the following period;
4. Processing is carried on without interruption (non-preemptive operation mode);
5. The time needed to transfer materials to the succeeding operation units or storage units is included in the processing time and all intermediate products are transferred at the beginning and the end of a processing task respectively;
6. Labors are assigned according to the requirement of operation. Each operator is responsible for operations in one stage only;
7. Intermediate buffers are assumed to be infinite. Although there are predefined

intermediate buffers between two consecutive stages that allow intermediate orders to wait for the succeeding process, some stages do not have clearly-defined buffer storage. Those intermediate products have to be stored in the preceding equipment till the succeeding stage is accomplished. For instance, harvesting process is comprised of cyclic operations. It does not allow intermediates to stay in the equipment when the processing is finished. Therefore, in this case, unfinished products are kept in the fermentors until the centrifuges are available. The problem is modeled as assuming there is an infinite work-in-process buffer where unfinished products can be stored between inoculation and harvesting.

Similar assumptions were made in some of the literature (e.g., Blomer and Gunther (1998) and Domadaran et al. (2006)) on the problems similar to this study.

4.3. Formulation

A mixed-integer programming formulation has been employed in the mathematical modeling section. The decision variables included in the model are the ordered product quantity of each job, temperature assignment variable that establishes the relationship between machines and temperatures, and operation variables that define the volume of medium usage on daily basis. The mixed-integer programming model assumes the problem is deterministic, with demand and capacity constraints; it should fall into the following formulation.

4.3.1. Model

Indices

j – order, $j \in J$

i – processing stage, $i \in I$

c – processing feature of temperature, $c \in C$

g – processing feature of medium, $g \in G$

m – operation unit, $m \in M_i$ (machine or labor)

t – planning period

Sets

J – set of orders

J_i – subset of orders to be processed on stage i, $\{j \in J: p_{ij} > 0\}$

I – set of stages

I_c – subset of stages with requiremnt of temperature $\{j \in J: p_{ij} > 0\}$

M_i – set of machines at stage i

C – set of temperatures

G – set of medium

Variables

$Z_t -$ total product quantity

$h_j -$ volume of growth medium of order j

$VOLM_g -$ volume of medium g required

Binary variables

$$Y_j \quad = \begin{cases} 1, & \text{if order } j \text{ is scheduled to be processed} \\ 0, & \text{otherwise} \end{cases}$$

$$X_{mct} = \begin{cases} 1, & \text{if machine } m \text{ is set at temperature } c \text{ in period } t \\ 0, & \text{otherwise} \end{cases}$$

Parameters

$c_{kj} -$ cost weight of order j (associated with product priority)

$q_j -$ size of order j (e. g. quantity)

$p_i -$ unit machine processing time in serial stage i

$p_{ij} -$ batch machine processing time in batch stage i

$l_i -$ unit manual working time in stage i

$ST_i -$ starting time of processing in stage i

$b_{ij} -$ size of transfer unit of order j at stage i

$S_m -$ capacity of operation unit m

$R_{cg} -$ estimated yield rate in medium g at temperature c

$a_{jc} -$ indicating relationship of order j and temperature c

(0: non − related, 1: related)

d_{jg} – indicating the relationship of order j and temperature c

(0: non – related, 1: related)

Objective function: to maximize the total product quantity within planning period t

$$\text{Maximize} \quad Z_t = \textstyle\sum_{j \in J} c_{kj} q_j Y_j \qquad (4-1)$$

Subject to

Temperature assignment constraint:

$$\sum_{c \in C} X_{mc} = 1, \qquad \forall\, m \in M_i, i \in I_c \qquad (4-2)$$

Capacity constraints for non-temperature-controlled stages:

$$\sum_{j \in J} b_{ij} Y_j \leq \sum_{m \in M_i} S_m, \qquad \forall\, i \in I - I_c \qquad (4-3)$$

Capacity constraints for temperature-controlled stage:

$$\sum_{j \in J} \sum_{c \in C} b_{ij} Y_j a_{jc} \leq \sum_{m \in M_i} \sum_{c \in C} S_m X_{mc}, \qquad \forall\, i \in I - I_c \qquad (4-4)$$

Stage sequence constraints:

$$\mathit{Serial} \quad ST_{i'} \geq ST_i + (l_i + p_i) \textstyle\sum_{j \in J} Y_j, \ \forall\, i \in I \qquad (4-5)$$

$$\mathit{Batch} \quad ST_{i'} \geq ST_i + p_{ij}, \ \forall\, i \in I \qquad (4-6)$$

Order fulfillment constraint:

$$h_j \geq \frac{\sum_{g \in G} \sum_{c \in C} q_j Y_j a_{jc} d_{jg}}{R_{cg}}, \qquad \forall\, j \in J \qquad (4-7)$$

$$VOLM_g = \sum_{j \in J} \sum_{g \in G} h_j Y_j d_{jg}, \qquad \forall\, g \in G \qquad (4-8)$$

$$Y_j \in \{0,1\}, \ \forall\, j \in J \qquad (4-9)$$

$$h_j \geq 0, \ \forall\, j \in J \qquad (4-10)$$

$$X_{mc} \in \{0,1\}, \ \ \forall\, m \in M_i,\ c \in C \tag{4-11}$$

$$VOLM_g \geq 0, \ \ \forall\, g \in G \tag{4-12}$$

The objective function (4-1) is to maximize the weighted total production quantity within the planning horizon so that the output quantity will be maximized.

The temperature issue has been discussed in Chapter 3. It is known as the temperature should always be consistent throughout the production. Physically, a machine can only be set at one temperature (4-2), and the orders required to be processed at temperature *c* must be maintained the same temperature at any temperature-controlled stage throughout the production.

Constraint (4-3) and (4-4) are the generalized stage capacity constraints. The capacity of each stage is comprised of two elements: time and resource. Workload at each stage must be under or equal to the capability of resources such as manpower and machines. Take transformation stage as an example. The operator is allowed to work on the hood from 1pm to 2 pm only which means all the work has to be finished within 1 hour. The capacity of manpower is represented by the corresponding allowable processing time. Another capacity limit is the volumetric measure of machines, equipment and tools.

Plasmid production is a sequence of serial and batch operations. Hence, succeeding processing cannot start without finishing the preceding processing steps. This is indicated by constraint (4-5) and constraint (4-6).

Order fulfillment constraint (4-7) is to ensure that each final product satisfies the desired quantity and quality, which means, each ordered plasmid sample will have certain amount of excess so that the required amount of plasmids can be guaranteed.

The values of decision variables such as daily total dosage of media can be obtained by constraint (4-8).

4.3.2. Parameter descriptions

Cost coefficient-c_{kj}. Cost coefficient is associated with the importance of orders. In business practices, it is common to differentiate orders based on the sales amount, profit margin or due-date urgency (Rim & Park, 2008). The determination of c_{kj} depends on the company's marketing strategy.

The size of job- q_j. Quantity is the scale usually used to measure the amount of demanded final products.

Temperature effect- a_{jc}. For each individual job, the growth temperature is determined based on the technical instructions. There are commonly three temperatures available: 30°C, 34°C and 37°C. The determination of temperature in this model is based on the probability of selecting each temperature found in historical data.

Growth media-g and d_{jg}. Growth medium is known as the nutrient support of growth of microorganisms and cells. For instance, there are two kinds of common media being applied in the factory: LB and MY. MY is used when order quantity is larger than the threshold, i.e., 40 mg, while LB is applied when small-quantity products are ordered. Compared to LB's fast catalyzing ability, plasmid growth in the MY medium requires relatively longer time.

The size of transfer unit/number of starters- b_{ij}. Transfer unit refers to starter, DNA in tube, bottle, and flask or any other forms. Theoretically, the starters are the divisions of a single plasmid sample. To select the appropriate number of divisions, one needs to have the microbiological background. For example, based on the empirical data, the number of starters of any order *j* can be 2, 3 or 4.

The criteria to determine the number of starters are as follows:

- If medium type is LB, then the probability of selecting 2, 3 and 4 starters are 2%, 5%, and 93% respectively.
- If medium type is MY, then the probability of selecting 2, 3 and 4 starters are 0%, 40%, and 60% respectively.

The starting time of stage i- ST_i. It represents several time points determined by the work schedule. Processing time of manual work are relatively flexible and they are batch-size dependent, while batch machine operate in a fixed length of time and the stage sequence constraint has to be maintained.

Yield rate- R_{cg}. Yield rate is estimated through historic data analysis with the combination of temperature and medium type. In reality, yield rate is a non-constant value that cannot be simply assumed. Based on a great mass of experiment and practice, the yield rate relatively strictly follows some statistical distribution. The analysis on the yield rate at different temperatures is discussed in Chapter 5.

The deterministic model has a lot uncertain parameters. For example, the yield rate follows statistical distribution but the MILP model cannot utilize the data. Furthermore, it does not deal with splitting large orders in dynamic mode which takes a number of iterations to evaluate if the capacity constraints are all satisfied. Therefore, the NP-hard scheduling problem cannot be solved in polynomial time. Hence, to find the optimal solution, a simulation method is proposed with the deployment of heuristic algorithms.

4.4. Simulation

Simulation is a versatile tool that has been used in various research areas from transportation, supply chain, manufacturing to finance. It has been proved to be a powerful means of understanding real-world systems and evaluating conceptual scenarios without inferring the stochastic and complex real-world applications. The study employs a discrete-event simulation technique and aims to identify the optimal scheduling rules with realistic environmental parameters.

4.4.1. Models

Simulation models are built for the manufacturing system using the simulation package AutoMod™ (Banks, 2004). This approach consists of the construction of two individual models. Models can be run independently or combined.

In order to model the scheduling procedure, the orders are generated by simulation Model 1 which creates a pool of data required in the initiation of simulation runs. It defines the processing characteristics of orders including the desired production quantity, temperature, growth medium and so on. By inputting daily order information into the model, decision-makers (i.e., schedulers or planners) can generate an initial production plan with parameters such as medium volume, and appropriate temperature defined in the earlier section. Then through assortment, orders are sorted according to their product quantity in decreasing order. The proposed selecting and splitting algorithm (SSA) then will be used to generate the candidates for a new schedule. If the candidates do not satisfy the capacity constraints, then the schedule will be rejected and another iteration of generating new schedule with SSA is needed.

Model 2 represents the manufacturing system. It simulates the multi-stage production system strictly follows the rules and requirements in the real system.

Model 1 and Model 2 can be combined to make decision of selecting jobs satisfying the production objective, or splitting large orders while maintaining the constraint of capacity. One can also use the combined model to evaluate the performance of current selection algorithm compared with conventional planning approaches.

4.4.2. Planning and scheduling procedure

As this work is aimed to plan and schedule the production of plasmid in a bioprocess plant consisting of serial and batch operations, two decisions are to be made:

First, as a set of assortment of current and new products with certain quantities as well as production requirement, such as the upper bound of production time, the process condition and so on, is given, the goal is to find out if the production operation is realized on the plant. If it is not the case, the planning technique, the proposed heuristic selecting and splitting algorithm is needed to optimize the production pattern without violating the constraints of capacity.

Second, because of the nature of batch production, the formation of batch with determination of unit jobs within it is required. The batch is formed at the beginning of the production and the sequence of orders will be maintained throughout the production. Then an optimal schedule of the batch is to be achieved in order to satisfy the production goal of maximizing the throughput.

Since the products are customized and the demand is random throughout the planning horizon, static long-term planning is not sufficient to reflect the real scenario with variability of demand. Thus, the production planning and scheduling considered is single-period planning and scheduling with a demand of N number of orders and each order is associated with a quantity/weight q_j.

To determine the orders to be processed, a heuristic procedure is proposed (Figure 13).

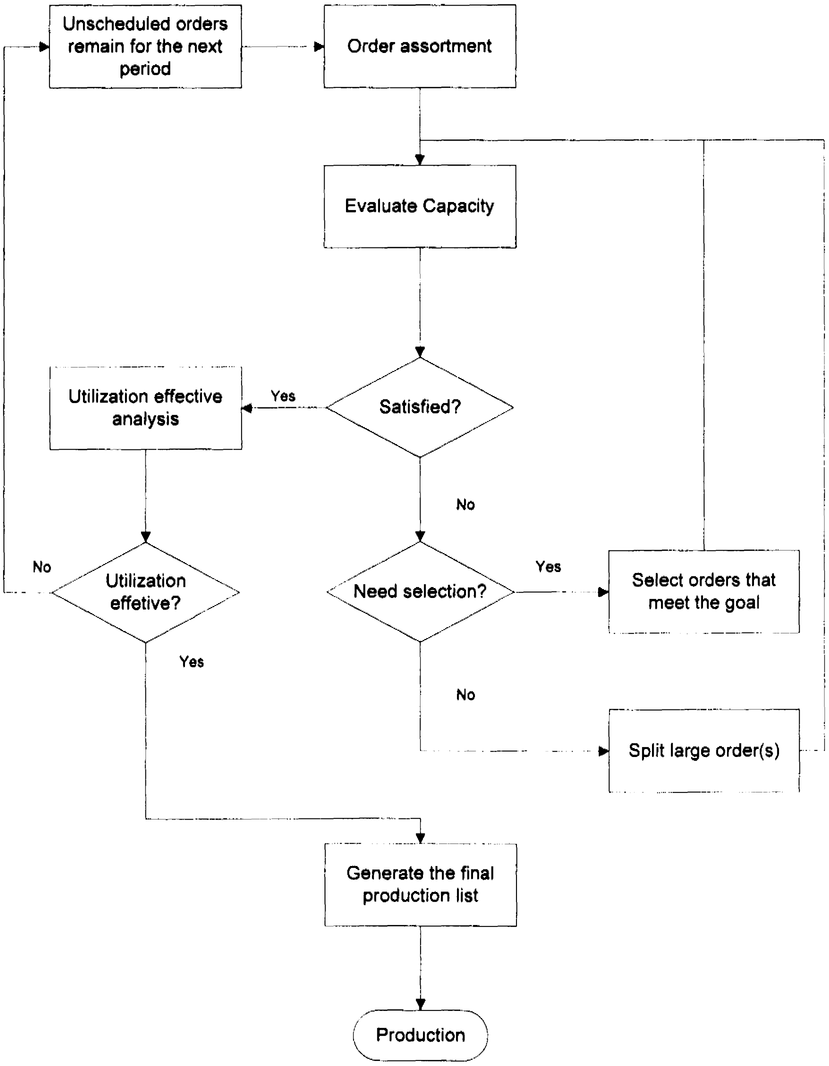
Unscheduled orders remain for the next period
Order assortment
Evaluate Capacity
Utilization effective analysis
Yes
Satisfied?
No
No
Utilization effetive?
Yes
Need selection?
Select orders that meet the goal
No
Split large order(s)
Yes
Generate the final production list
Production

Figure 13. Decision making procedure on production planning in a single period.

At time period t, upon the reception of new orders, an estimation of capacity in time or resource units is taken place. If the capacity constraint is not violated, then the utilization effectiveness is evaluated by using utilization function to see if the current production program is economic to operate. On the other hand, if the capacity constraint sustains, the decision is to select appropriate candidate orders to produce. This may require splitting jobs with relatively large quantities. Again, the large-quantity orders or large orders for short are those with a unit demanded quantity over 100mg. Selected orders will be considered as the final production list, and then be put onto manufacturing flow-line. Unselected jobs remain and will have to wait until the next planning period.

4.4.3. Utilization effectiveness analysis

The reason to conduct utilization effectiveness analysis is to estimate the financial effectiveness when production amount is relatively low, far away from reaching the capacity of the facility. The decision making is even critical when only 1 or 2 jobs present in the facility, the low-workload production leads to low-utilization of entire facility and eventually results in a large amount of variable cost.

Utilization is analyzed by the ratio of required usage of production facility or production time to the available capacity of facility or time at the planning point t.

$$U_i = \frac{Required\ capacity\ (time\ or\ resources)}{Available\ capacity\ (time\ or\ resources)} \qquad (4-13)$$

Where U_i is the utilization of resources or time in stage i.

Then the average system utilization can be computed as

$$AveU = \frac{\sum_{i=1}^{I} U_i}{I} \qquad (4-14)$$

Decision maker must setup his own threshold towards the minimum production level. In this study, it's postulated to be 25%. The reason to select 25% as the threshold is that with 25% average system utilization, the minimum utilization over the stages can reach at least 10% which is considered as a satisfactory value. The conclusion is drawn from the simulation tests which are discussed in Chapter 5.

4.4.4. Order selecting and splitting algorithm (SSA)

Order selection is applied when extra capacity is in need to process all orders. Splitting may be desired to deal with orders having large unit quantity. The reason for splitting is that when capacity is not enough but the throughput of the system must be maximized. This situation is extremely significant when the jobs are competing for the same resource. In this case, priority is given to the orders with larger product quantity because they have longer processing time and contribute more to the total output product quantity.

Decisions of order selection and splitting are all on the basis of capacity constraints. If demand exceeds the capacity of the system, then the operation of simulation-based order selecting and splitting rules under consideration can be stated as follows:

When a set of candidate orders arrive to the manufacturing facility, an evaluation run is executed to estimate the extent of the constraint violation. If capacity is not available for processing all orders, then orders are ranked by the product quantity. To ensure large orders will be processed quickly preventing subsequent orders from being delayed, orders are selected based on the rank, unless other priority requirement is present, such as urgent due date and high marginal profit. Then evaluation run is executed. If the result satisfies the all the requirements, then the selected orders are marked as scheduled. Otherwise, the splitting function is required. The procedure can be described by the following steps:

Step 1: set $k = 1$, generate the initial schedule S_0.

Step 2: generate a new schedule S_k with C_s number of orders. If C_s can't be satisfied, then go to Step 5, otherwise go to Step 3.

Step 3: evaluate the capacity requirements. If one of the constraints is violated, then go to step 4, otherwise return schedule S_k as the final schedule.

Step 4: select an order with the largest unit quantity, decrease the quantity by V_c , assign new attribute to the order, and go to step 3. Set $k = k + 1$.

Step 5: Evaluate the utilization effectiveness. If the result is less than the threshold, then return the schedule S_k as unscheduled. Wait until the next period.

C_S defines the capacity of the first stage which is a constant value implying the maximum number of orders allowed to be processed every period.

The value of V_C is the reduction in product quantity. If V_C is too small, then a number of iterations are needed to reach the optimal result. If V_C is too large, then the value of the objective function which maximizes the total production quantity may not be optimal. Therefore, the V_C is considered to be 50 which is considered large enough to make changes in the assigned attribute values, i.e. the number of starters, but yet moderate to gradually reduce the quantity of the single order.

4.5. Knowledge-based system

Through the simulation study, a set of heuristic rules of order selection and splitting are generated. Those rules will be stored in a knowledge-based scheduling system. Then the schedulers will input the orders' information, including the number of orders and the quantity in each order, into the expert system and recommended production plan will be provided by the knowledge-based scheduling system. The following sections will give detail discussion of the system.

A knowledge-based scheduling system, as discussed in Chapter 1, is a computer-based tool that provides intelligent scheduling decisions based on user inputs through ask-and-answer communication. Since the production of plasmid DNA requires versatile information with regard to the specific features of plasmid source, the planning and scheduling procedure strongly depends on the capacity of facility. With customized production recipes, the production capacity of a multi-product facility is not preciously defined because it has a strong dependency on the product mix to be delivered, as indicated in most of batching production lines (Henning & Cerda, 1996). Rules regarding decisions of planning and scheduling therefore need rigorously consideration to guarantee resource to be wisely utilized, cost to be minimized and production output to be maximized.

4.5.1. The structure of expert system

In this research, an object-oriented rule-based expert system is deployed to give decision makers a general guideline regarding the planning of plasmid DNA production. All rules are deductive rules acquired from simulation experiment represented in *if-then* relationship. The expert system for DNA manufacturing includes 5 rules that lead to 5 production plans in each planning period. Decision rules are created regarding whether certain orders should be added into the current production plan (selection), or "discarded" as not to produce, or split to respect the capacity constraint.

4.5.2. The development of rules

The rules used to aid scheduling decision-making are defined and tested by simulation Model 1. As there are four factors involved in the production of plasmids, total order quantity, amount of orders, quantity composition of orders, and temperature requirement, the experiment is carried out to find the lower-bound and upper-bound of every level of each factor. The detailed information on defining the levels of the factors is discussed in Chapter 5. Figure 14 outlines the structure of the expert system and Table 2 shows the levels of each factor.

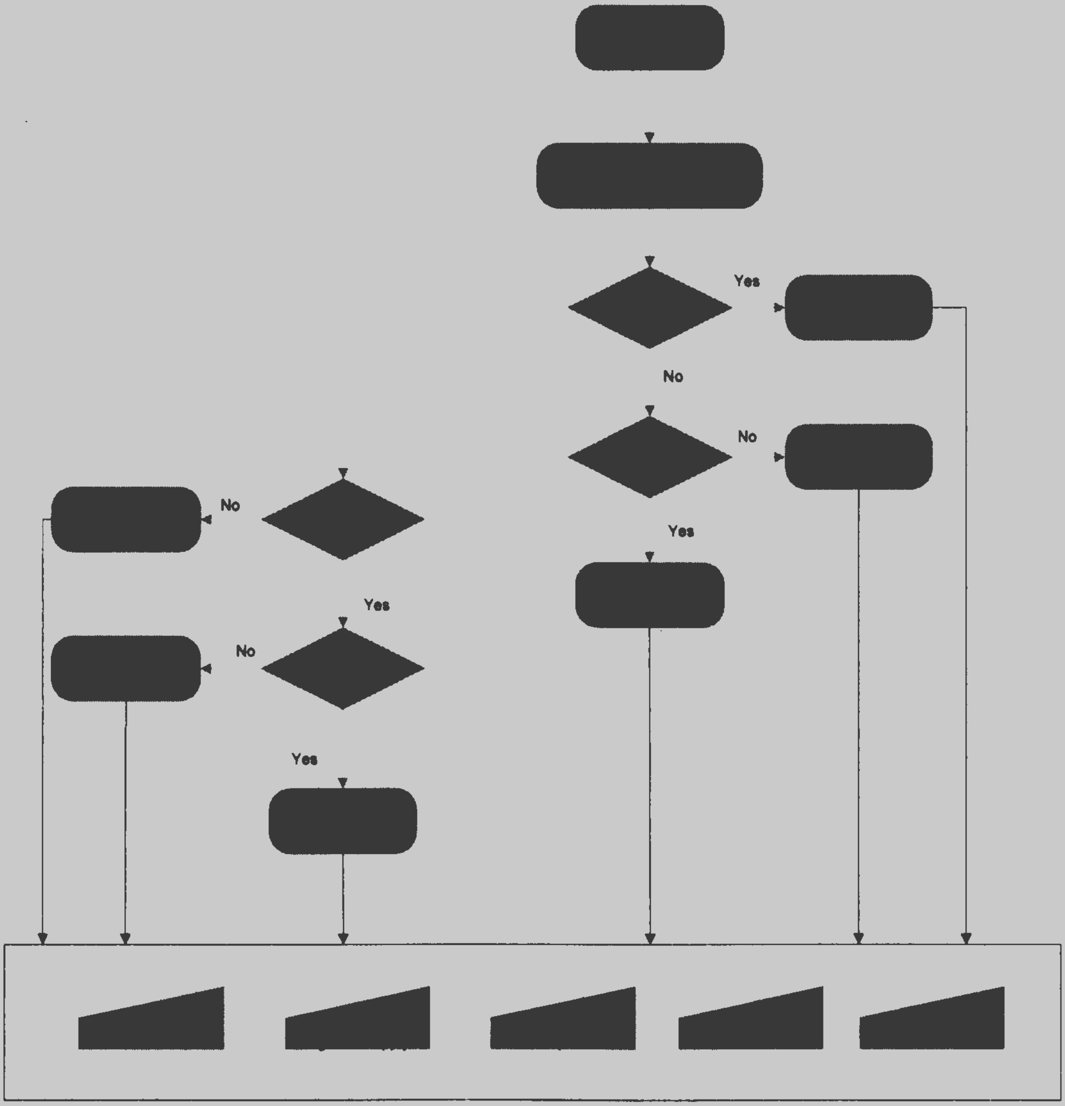

Figure 14. The structure and rules of the knowledge-based DNA manufacturing scheduling system.

Table 2. Factors and their levels

Index	Factor	Factor Type	Lower Level (-)	Upper Level (+)
A	Total quantity	Real	Less than or equal to critical value B	Greater than critical value B
B	Quantity composition	Percentage of large-quantity order	Small-quantity orders dominate	Large-quantity orders dominate
C	Temperature groups	Integer	Single (one temperature only)	Multiple (more than one temperature group)
D	Total number of orders	Integer	Less than or equal to 20 orders	Greater than 20 orders

CHAPTER 5. CASE STUDY AND EXPERIMENT

Due to the complexity of uncertain demand, the variability of yield, product mix and the requirement to split, in Chapter 4, a mixed integer programming model has been proposed to formulate the problem with various constraints and two simulation models are introduced. In this chapter, the simulation models are explained in detail and how they function to solve the scheduling problem derived from case study.

A case study is carried out based on the plasmid DNA manufacturing factory. The manufacturing facility includes one workshop with ten processing stages each of which is equipped with a series of parallel machines and discrete machines. Plasmid samples are required to go through transformation, starter culture, screening, inoculation, harvesting, lysis, AEX, polishing (conditional), concentration and QC to complete the production cycle.

The objective of the study is to schedule product orders prior to manufacturing to maximize the throughput of the entire facility. The throughput here is considered as the total production quantity obtained at the end of planning horizon in terms of total plasmid yield in milligrams. The average system utilization is an index that reflects the capacity of the system. If the ratio value is lower than 1, then the capacity is underutilized. On the other hand, if it is higher than 1, then the system is overloaded and capacity is not enough for the current newly-received orders. Orders are scheduled according to the available capacity. Unscheduled orders are carried forward to the next planning period and will be given the highest priority of processing.

5.1. Data

Demand data, yield data, processing times, system capacity and lot information were obtained from the company's database. The daily order amount, desired order quantity, and yield amount associated with each order represent the data collected. Analysis of these data was carried out.

5.1.1. Work schedule

The whole system is comprised of 7 processing stages (10 in total but only 7 are included in the capacity analysis). Table 3 shows the work schedule.

Table 3. Work Schedule

Process	6AM	7AM	8AM	9AM	10AM	11AM	12PM	1PM	2PM	3PM	4PM	5PM	6PM	7PM	8PM	9PM
Transformation	■						■	■ 3 hours	■	■	■	■	■	■	■	■
Starter Culture	■	■	■	■	■	■	■	■	■							
Screening								■	■ 4 hours	■	■					
Inoculation										■	■ 3 hours	■	■	■	■	■
Harvesting	■	■	■	■ 7 hours	■	■	■									
Lysis		■	■	■	■	■	■	■	■ 13 hours	■	■	■	■	■	■	
AEX & Concentration					■	■	■	■	■ 8 hours	■	■	■				
QC			■	■	■	■	■ 9 hours	■	■	■	■					

■ Labor required ■ Machine only

In each stage, the processing task is divided into manual work subtask and machining subtask. Manual work is serial whereas task on machine is in batch mode. The company uses a fixed work schedule currently which is considered as capacity constraints in terms of time. Work schedule are used to control the operator working hours in simulation model. Available processing time of each stage and corresponding capacity can be found in Appendix C.

5.1.2. Demand data analysis

5.1.2.1. Production quantity analysis

Daily demand for 221 days in fiscal year 2007-2008 is also provided (see Figure 15).

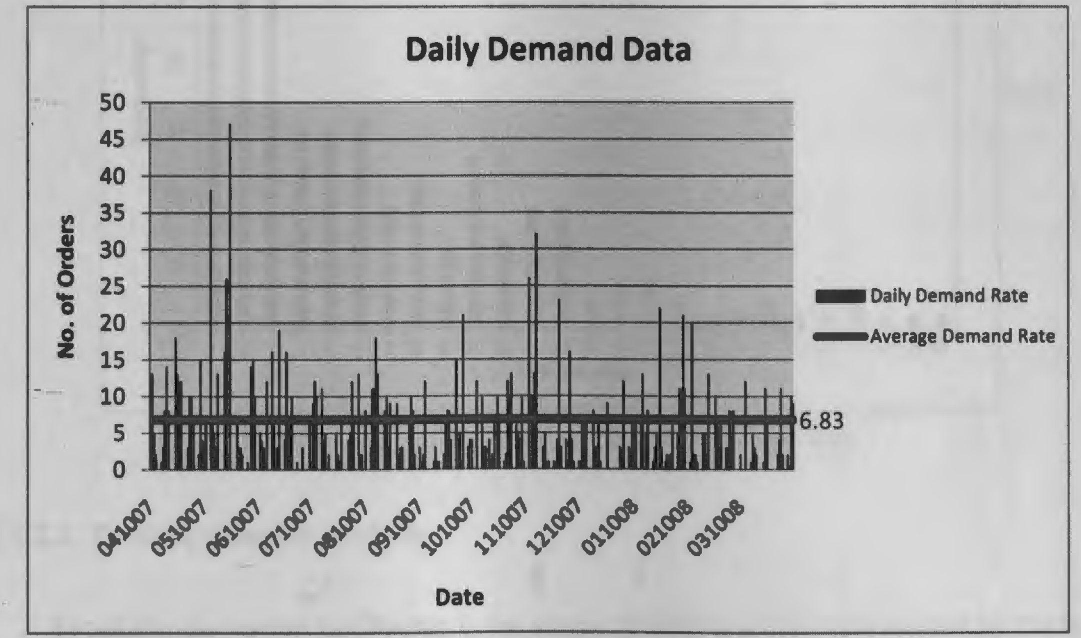

Figure 15. Daily demand data.

Within one year time window, from April, 2007 to April, 2008, the company received 1516 orders, each with different size (demanded quantity).

As shown in the figure, it becomes clear that the number of customized orders varies every day, from the maximum of 47 to the minimum of 1 only, with the average of 6.8 orders per day. However, since the purpose of the study is to design a robust system that handles increased demand and schedules production in short-term period, say on daily basis, the targeted daily demand is determined as from 20 to 47 orders as shown in the upper tail of the frequency plot in Figure 16.

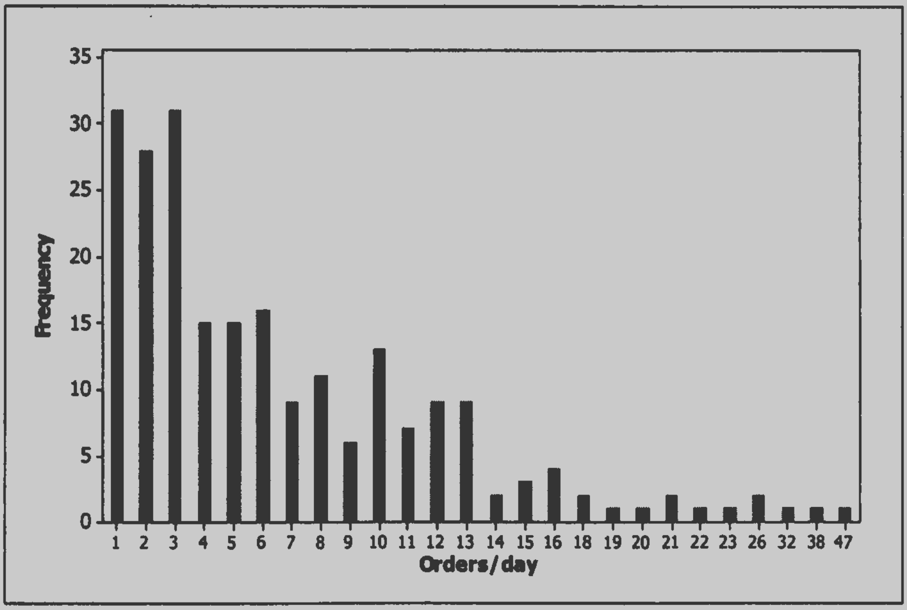

Figure 16. The frequency of No. of orders per day.

5.1.2.2. Product quantity analysis

Recall the discussion in Chapter 1, the single product quantity represented by the total weight of obtained plasmids in milligrams varies from order to order. Because the demand is not known prior to production, it is difficult to predict the total product quantity or order quantity. Hence, this study focuses on anticipating order quantity based on order history.

Since each order is independent of previous purchase and has either no interrelationship with others that come at the same time, product quantity data consists of a number of independent random variables. The frequency of each ordered quantity is shown in Figure 17.

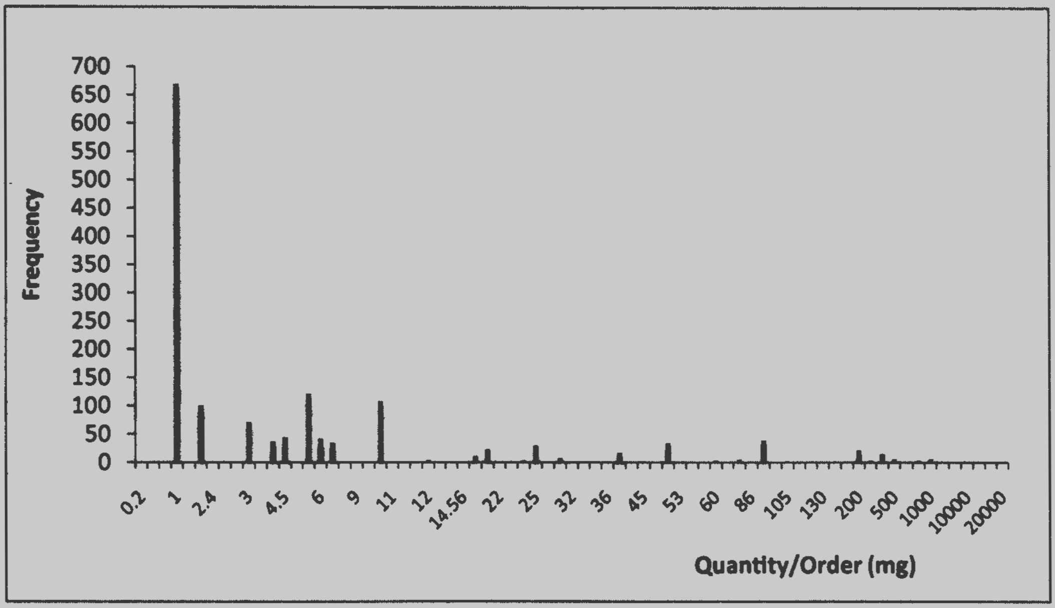

Figure 17. Frequency of single order quantity.

It is obvious that during year 2007-2008, small-quantity products were ordered more frequently than those large-quantity products.

5.1.2.3. Proportion of large orders

As defined in Chapter 3, large orders are those with quantity over 100mg. They require longer processing times in comparison with small orders. Yet, the processing of large orders might delay the subsequent orders and result in longer lead time of the entire batch. However, since they contribute more to the throughput, if the capacity is utilized wisely, the shortcoming they bring to the system can be compensated by applying appropriate scheduling rules. Among 1517 orders, only 7% around are large orders. Figure 18 shows the result.

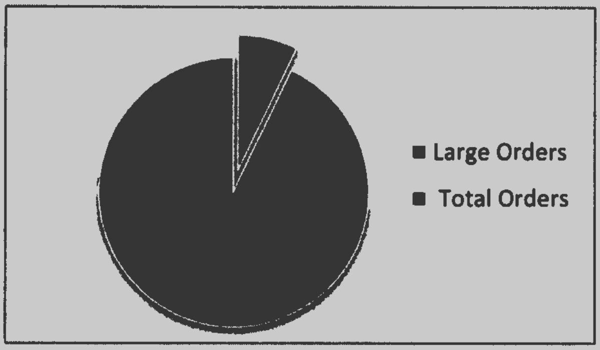

Figure 18. Proportion of large orders.

5.1.2.4. Daily quantity

Daily quantity analysis was conducted to see if there was an existing pattern. The quantity varies greatly from 20000mg to 0.2mg (Figure 19).

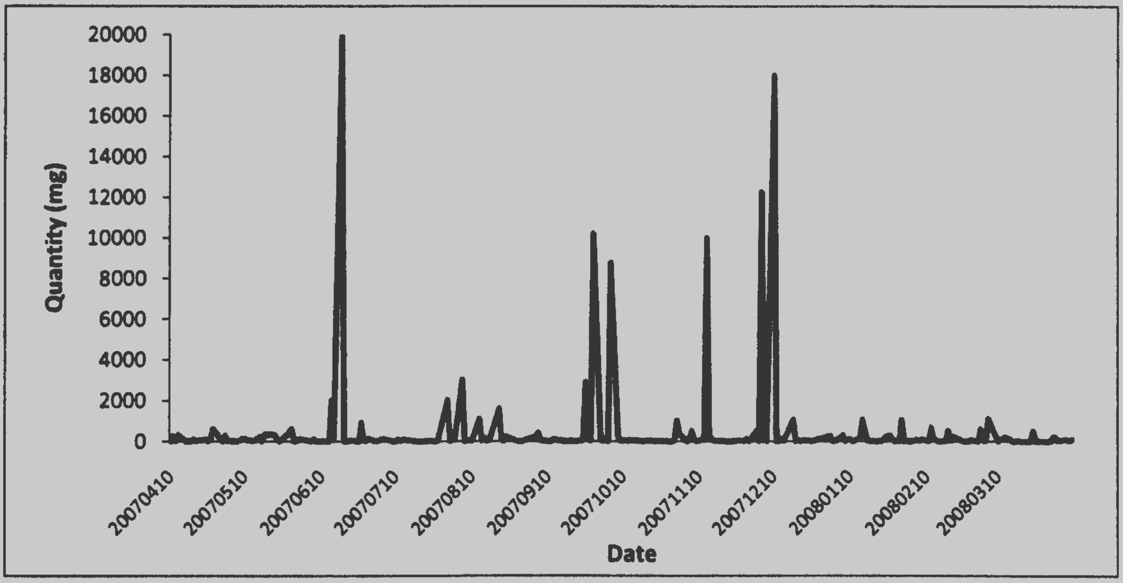

Figure 19. Daily quantity data.

Figure 19 shows the demand daily quantity data found in fiscal year 2007-2008. Historical data analysis indicates that previously the demand was not high enough to conduct simulation study monitoring the situation of increased demand. Therefore, the concept of large-order rate which implies the percentage of occupancy is introduced.

5.1.3. Yield data analysis

The samples of orders provided by customers determine the yield of products whereas the latter makes impact on the factory stay time of the product. If the yield of a single product order is low, then it might need several production cycles to be fulfilled and the throughput of the system becomes restricted. However, because samples are given by the customers while bacteria strains are available in the factory, the yield of samples and its interaction with bacteria cannot be controlled by human power. Hence, the study underlying concerns only historical yield to estimate the growth of plasmid during the simulation.

To be general, there are two types of culture media, LB and MY, and three temperature settings, 30°C, 34°C and 37°C, deployed in the model. The yield of bacteria strains cultured in each medium varies under each temperature setting. Therefore, data series are divided into two categories based on medium selected and yield amount is analyzed associated with each temperature setting. Specifically, the yield data refers to the milligram yield of product per liter of medium used. In total, there are 266 data points found and among them, 170 are derived from orders cultured with LB and 96 from those with MY.

5.1.3.1. Yield with MY at 30 °C

There are too few items found in this category. With unknown population mean and variance, it is unable to make fitness assumption with any existing statistical distribution. Therefore, in the simulation, a constant mean value is used. Table 4 shows the descriptive statistics with MY yield at 30 °C.

Table 4. Descriptive Statistics: Yield with MY at 30 °C

Variable	Total	Mean	Standard Deviation	Minimum	Maximum
Yield with MY at 30°C	5	16.77	6.31	10.93	25.80

5.1.3.2. Yield with MY at 34 °C

In total, 34 data points are plotted in the probability graph. They follow normal distribution at 95% confidence interval with the mean of 13.90 and standard deviation of 5.275.

5.1.3.3. Yield with MY at 37 °C

After performing goodness of fit test, the sample of 57 data points fits into the normal distribution with a mean of 10.49 and standard deviation of 2.145.

The results of analysis of yield data at 34 °C and37 °C are depicted in Figure 20 and Figure 21 respectively.

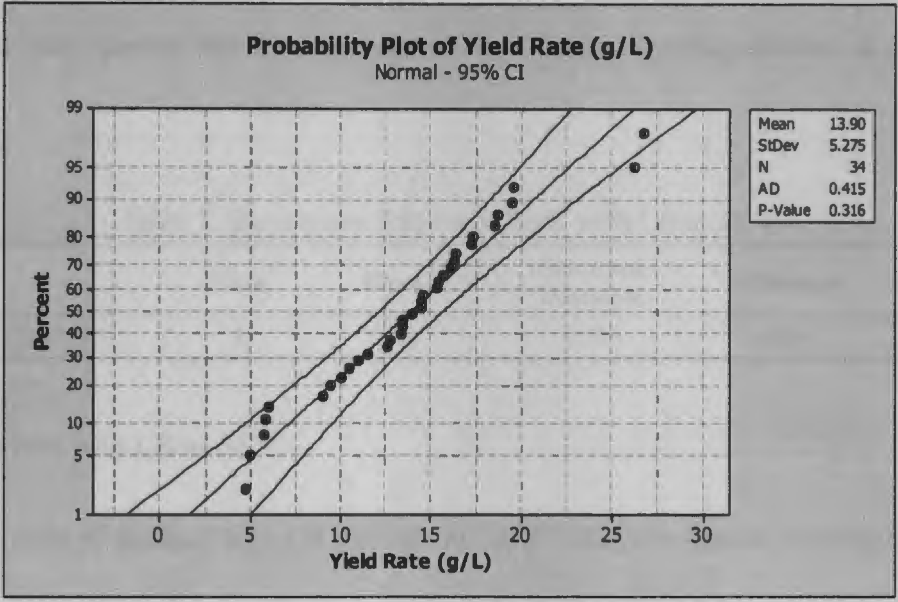

Figure 20. Probability Plot of Yield Rate with MY at 34 °C.

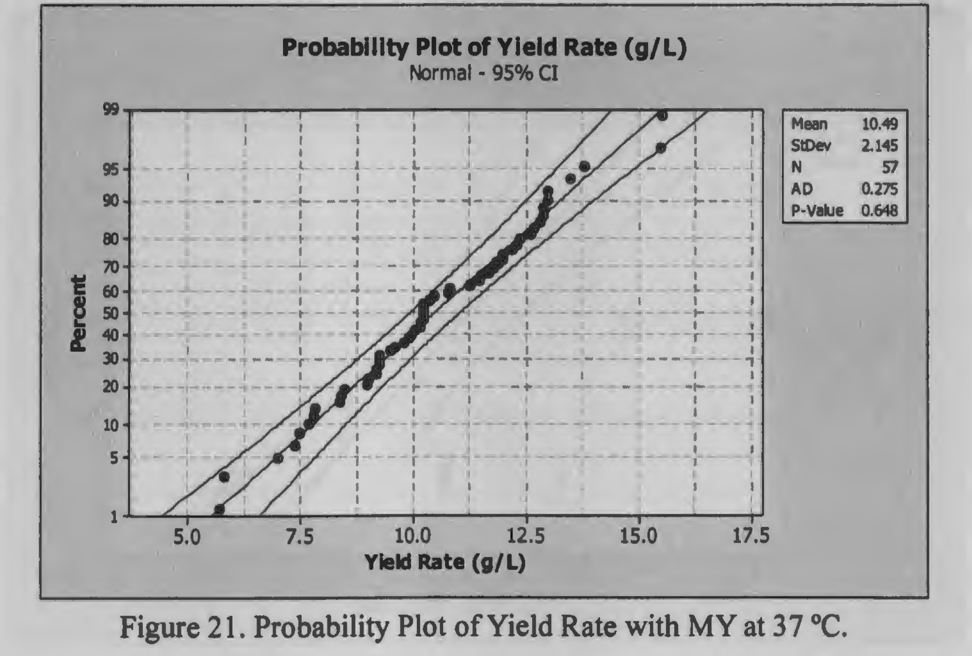

Figure 21. Probability Plot of Yield Rate with MY at 37 °C.

5.1.3.4. Yield with LB at 30 °C

Similarly, the record of yield of LB at 30 °C is scarce. Thus, a simple assumption is made that these sample data have a mean value of 7.548 regardless of size of population (Table 5).

Table 5. Descriptive Statistics: Yield with LB at 30 °C

Variable	Total	Mean	Standard Deviation	Minimum	Maximum
Yield with LB at 30°C	3	7.548	0.750	6.694	8.100

5.1.3.5. Yield with LB at 34 °C

The yield of product with LB medium at 34 °C follows a normal distribution nicely. The mean value is 6.466 and standard deviation is 2.139. The result is shown in Figure 22.

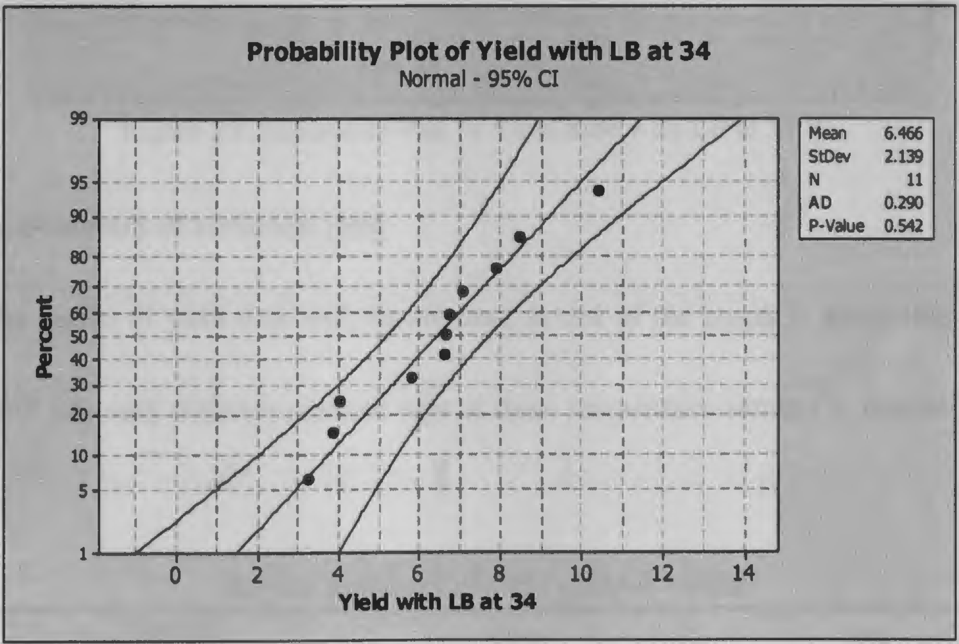

Figure 22. Probability Plot of Yield Rate with LB at 34 °C.

5.1.3.6. Yield with LB at 37 °C

Through goodness of fit test, the data associated with yield at 37 °C with LB medium fits into a normal distribution with mean value of 6.428 and standard deviation of 1.848. The result is shown in Figure 23.

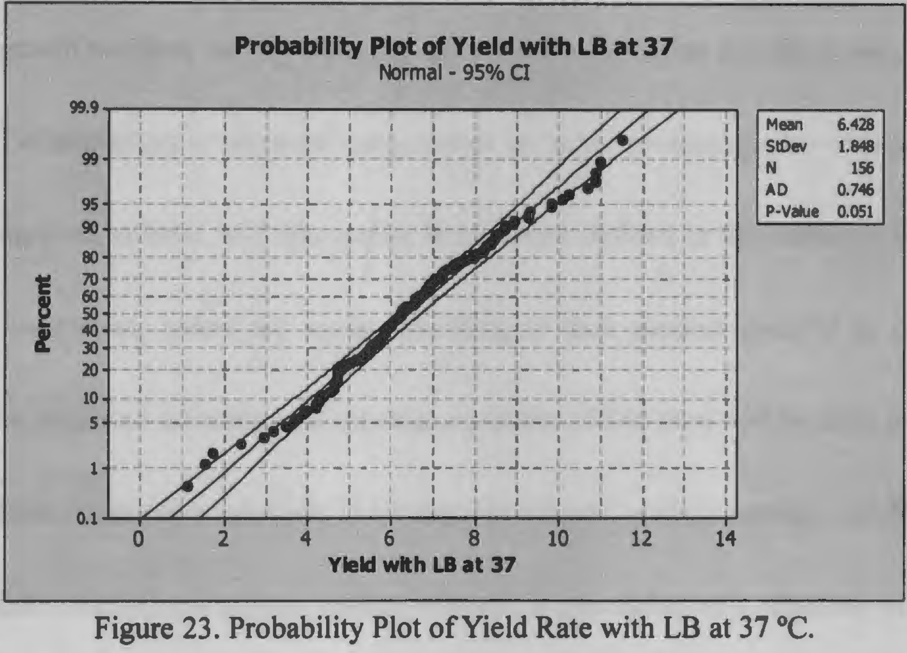

Figure 23. Probability Plot of Yield Rate with LB at 37 °C.

5.1.3.7. Summary of historical yield

The results of yield data analysis are used as one of the inputs in simulation model. The yield rate with different medium type at three temperature settings is summarized in Table 6.

Table 6. Summary of yield analysis results

Medium Temperature	LB	MY
30 °C	Mean 7.548	Mean 16.667
34 °C	Normal (6.466, 2.139)	Normal (13.90, 5.275)
37 °C	Normal (6.428, 1.848)	Normal (10.49, 2.145)

5.2. Model verification and validation

In order to model the scheduling procedure, orders are generated by simulation Model 1 which creates a pool of data required in the initiation of simulation runs. It defines the processing characteristics of orders including the desired production quantity, temperature, type of growth medium, etc. By inputting daily order information into the model, decision-makers (i.e. schedulers or planners) can generate an initial production plan with parameters such as medium volume, and appropriate temperature defined in the earlier section. Then through assortment, orders are sorted according to their product quantity in decreasing order. The proposed selecting and splitting algorithm (SSA) then will be used to generate the candidate for the new schedule. If the candidate doesn't satisfy capacity constraint, then the schedule will be rejected and another iteration of generating new schedule with SSA is needed.

Model 2 represents the manufacturing system. It simulates the multi-stage production system strictly follows the rules and requirements in the real system.

Model 1 and Model 2 can be combined to make decision of selecting jobs satisfying the production objective, or splitting large orders while maintaining the constraint of capacity. One can also use the combined model to evaluate the performance of current selection algorithm compared with conventional planning approach.

Drawings of the layout are not made to scale since the movement of resources are not considered as in the scope of the study. The snap shot of the system is just used to display the manufacturing flow shop (see Figure 24).

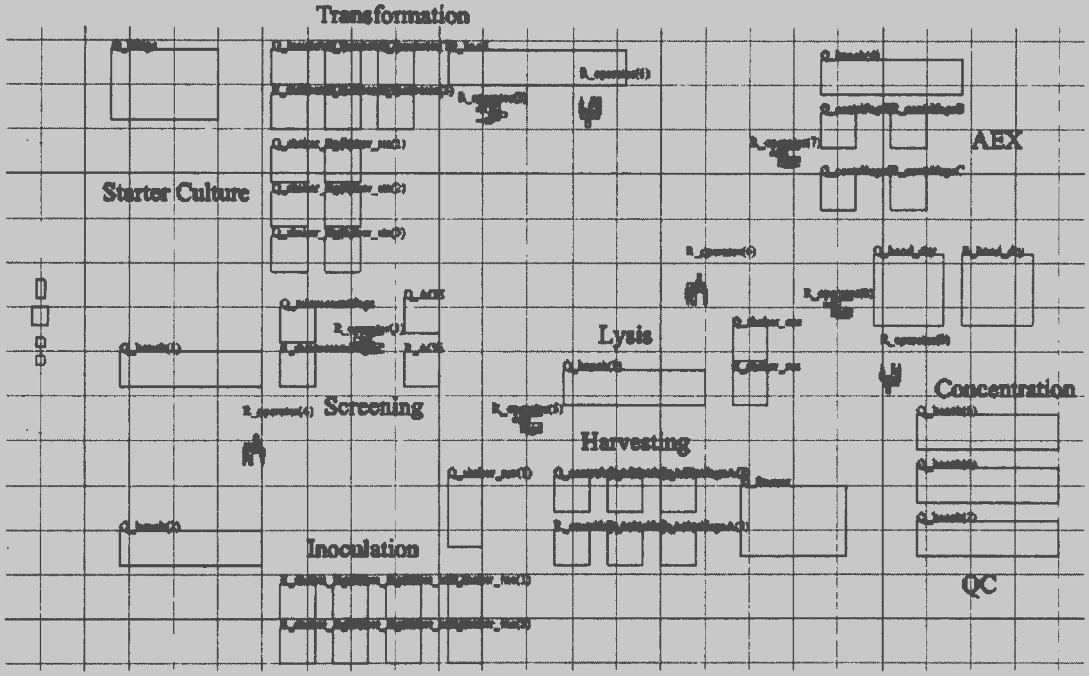

Figure 24. The snapshot of the simulation environment.

Verification is generally used to determine whether the model is the correct representation of the conceptual model, while validation determines whether model is true to the real system for the purpose of experimentation. There are a number of techniques that can be used to verify and validate a simulation model, such as watching model animation for verification and sensitivity analysis or using historic data for validation. Since the model employs the splitting and selecting mechanism, the historical output may not match that derived from artificial input. Therefore, sensitivity analysis is utilized to examine the accuracy and consistency of the model. Input parameters include order arrival rate and proportion of large orders.

All experiments are carried out through AutoMod™ simulation system on PC with configurations of Intel® Core™ 2 CPU 6330 1.86 GHz chipset, 2GB RAM, 160GB HDD, and 256MB ATI® Radeon™ X1300 PRO discrete graphic card.

Firstly, the behavior of some performance measures was examined by adjusting the order arrival rate. Incremental daily order arrival rate from 10 to 50 are observed in 30-day period and the impact on the throughput and the average lead time (ALT) are investigated, as shown in Figure 25.

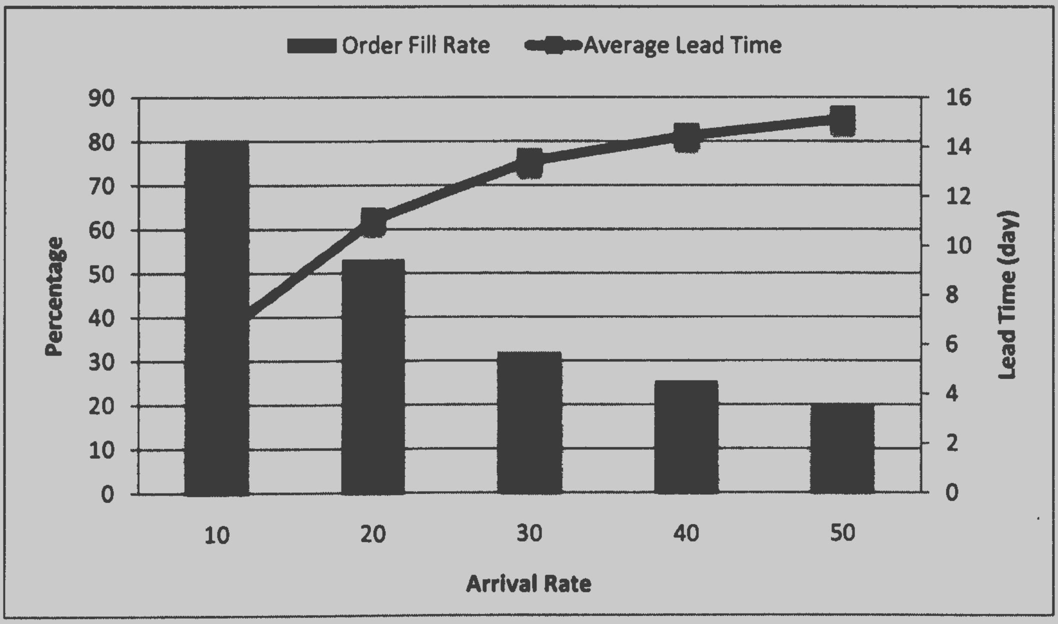

Figure 25. The impact of order amount.

The above figure shows that the ALT and the order fill rate both increase with incremental arrival. That is because high arrival rate causes bottleneck in the system, and the time orders spent in the system queues increases. The direction of increase matches the expectation and the consistency of the model is held. Additionally, the throughput of the system is restricted by the capacity. Thus when order arrival rate increases beyond 20, the drop of order fill rate becomes less significant because the maximum number of orders to be scheduled is set to be 20 (as the capacity of Transformation).

Secondly, the sensitivity to product mix was tested by increasing the number of large orders incrementally and keeping the total number unchanged. Twenty percent increment of large orders in the total received orders from 20% to 100% was considered (Figure 26).

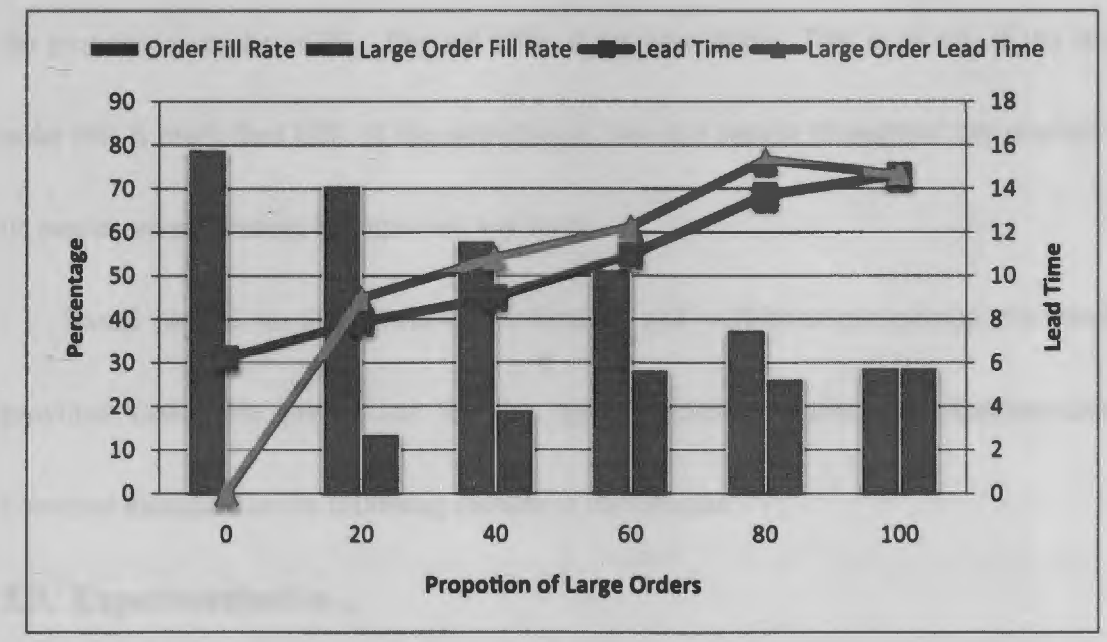

Figure 26. The impact of large orders.

The total number of orders is set to be 20 in which capacity violation occurs with higher large order rate.

As expected, the throughput of the system decreases when increasing the proportion of large orders in total orders received. The difference between fulfilled large orders and total finished orders reduces as well. Since all the orders are large, the gap no longer exists at 100% level. The average manufacturing lead time (AMLT) increases consistently as the system continuously inputs more large orders. When large orders present, the large order lead time (LOLT) is always larger than the average lead time (ALT). That tells the system is sensitive to large orders and the existing large orders dominate the determination of system performance. Moreover, the impact of large orders is on the maximum level when the proportion reaches 60%. Beyond 60%, it remains stable. That is to say, if the large order rate is more than 60% of the entire batch, then the system throughput has reached to its maximum and cannot be improved any more.

Based on the results of above verification and validation procedures, the model provided reasonable predictions for the system behavior under the experimentation scenarios discussed in the following section of this chapter.

5.3. Experimentation

In this section, a set of scenarios are defined by varying total number of orders and the proportion of large orders. Results are compared with four classic dispatching rules, First Come First Served (FCFS), First Come Last Served (FCLS), Longest Processing Time (LPT), and Shortest Processing Time (SPT).

1) FCFS: Regardless of the quantity, the available capacity is assigned to the next arriving order. The order will be put into production immediately when resources become available. The unscheduled orders due to capacity shortage are carried forward to the next day, with the highest priority to avoid excessively long delay.
2) FCLS: Orders are collected before production. The capacity is assigned to the orders arrived last first without considering the order quantity.
3) SPT: Capacity is assigned to the order with the smallest quantity first. Tie breaks by selecting the order arrived first.
4) LPT: Unlike SPT, orders are ranked by their quantity in decreasing order. Then capacity is assigned to those with the largest quantity first.
5) SSA: Consider the impact of orders with quantity larger than 100mg, the capacity is first assigned to those orders. If the required capacity is more than available capacity, then one will consider performing selecting or splitting or both to the entire batch.

The simulation model was executed for 30 days with order generation repeatedly on daily basis. From the results of model verification and validation test runs, one can see that the impact of large orders on the throughput cannot be overlooked. Therefore, we test the scenarios with high demand rate and relatively large rate of large orders. Since from previous data analysis, we found that 20 to 47 orders account for high demand rate and the purpose of the study, again, is to design a scheduling rule for increased demand. Thus demand rate ranging from 20 to 50 are selected as one parameter. Another parameter, the proportion of large orders, is chosen as 20% to 40% as indicated in the model verification and validation section. For 4 levels of number of orders times 2 levels of large order proportion times five methods, with single replication, it yields 40 runs in total. The experiment results are shown in Table 7.

Table 7. Results of experimentation on varying the number of orders

Large Order Rate	Rules	Throughput (milligram)			
		n=20	n=30	n=40	n=50
20%	SSA	43,355	27,740	31,010	33,680
	FCFS	40,270	20210	20,505	22,475
	FCLS	41,420	21,995	21,250	22,015
	SPT	42,935	20,904	21,390	21,520
	LPT	42,775	22,930	23,380	22,180
40%	SSA	46,355	33,050	35,145	35,900
	FCFS	41,450	26,405	25,650	25,900
	FCLS	44,065	28,195	28,670	27,370
	SPT	42,025	26,030	26,860	23,865
	LPT	43,750	27,145	27,015	27,075

Performance measures here are the throughput of the system in terms of total finished product quantity and the percentage order fulfill rate which is the rate of the throughput to the demand.

The proposed algorithm outperforms all rules in all the cases. However, when the number of orders is relatively low, 20 or below, and the proportion of large orders is not large either, at 20% level, the result does not show significantly difference. As the order amount continuously increases from 30 to 50, the difference among SSA and four rules becomes more significant. The biggest gap occurs at the level of 50. Classic methods seem to yield similar results. Figure 27 shows the comparison results.

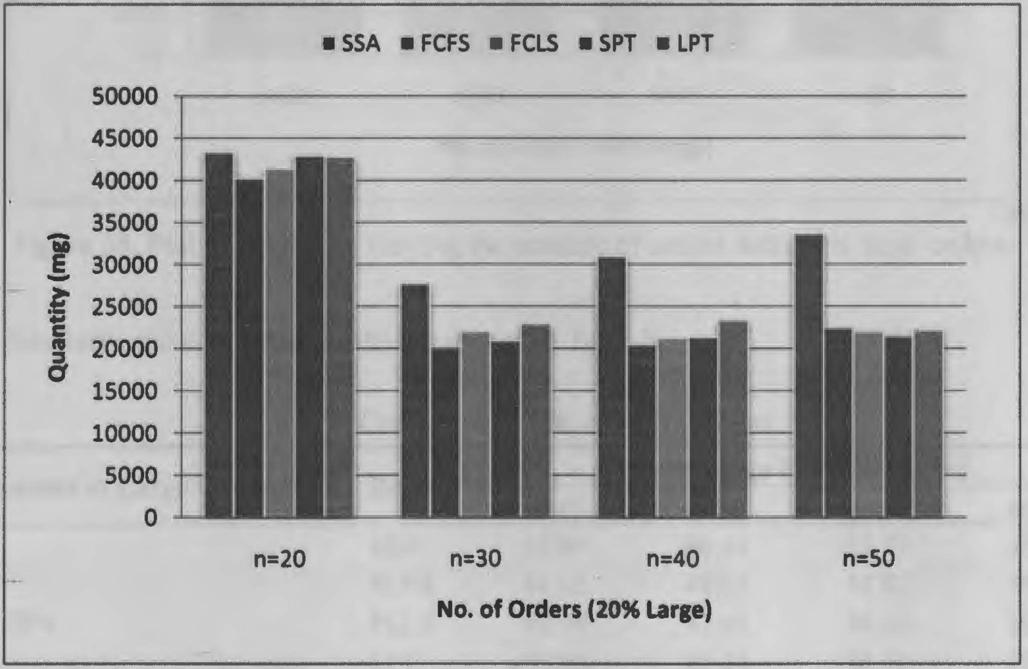

Figure 27. Plot of results on varying the number of orders with 20% large orders.

With 40% large orders, the total production quantity increases as the number of orders arises. This follows the trend found in the 20% scenario. Still, the performance of SSA method is over the classic methods. However, the advantage of using SSA is revealed when the total order amount is large enough, more than 30 per day. Figure 28 shows the detail results.

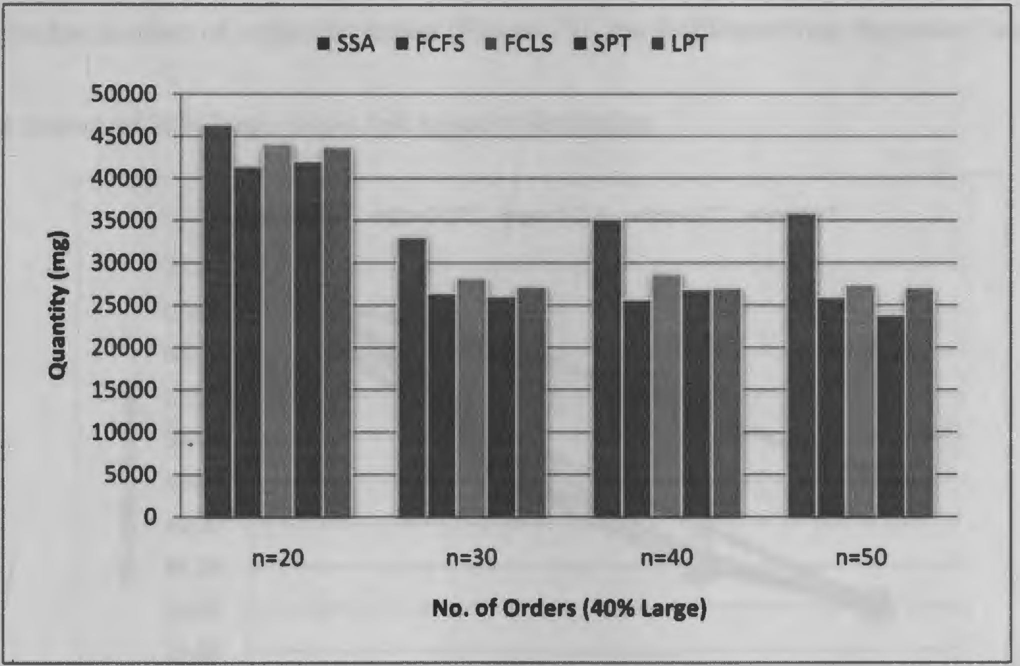

Figure 28. Plot of results on varying the number of orders with 40% large orders.

Similarly, the converted results are shown in Table 8.

Table 8. Converted results of % fulfill rate

Proportion of Large Orders	Rules	Percentage Fulfill Rate (%)			
		n=20	n=30	n=40	n=50
20%	SSA	65.69	60.44	52.74	45.82
	FCFS	61.02	44.03	34.87	30.58
	FCLS	62.76	47.92	36.14	29.95
	SPT	65.05	45.54	36.38	29.28
	LPT	64.81	49.96	39.76	30.18
40%	SSA	60.83	54.81	43.71	35.72
	FCFS	54.40	43.79	31.90	25.77
	FCLS	57.83	46.76	35.66	27.23
	SPT	55.15	43.17	33.41	23.75
	LPT	57.41	45.02	33.60	26.94

Because the total demand quantities in those scenarios are different, they are converted into percentage fulfill rate which is the rate of total fulfilled amount (in milligrams) vs. the total demand quantity.

As the number of orders increases (Figure 29), the fulfillment rate decreases because of the impact of 20% large orders and capacity limitation.

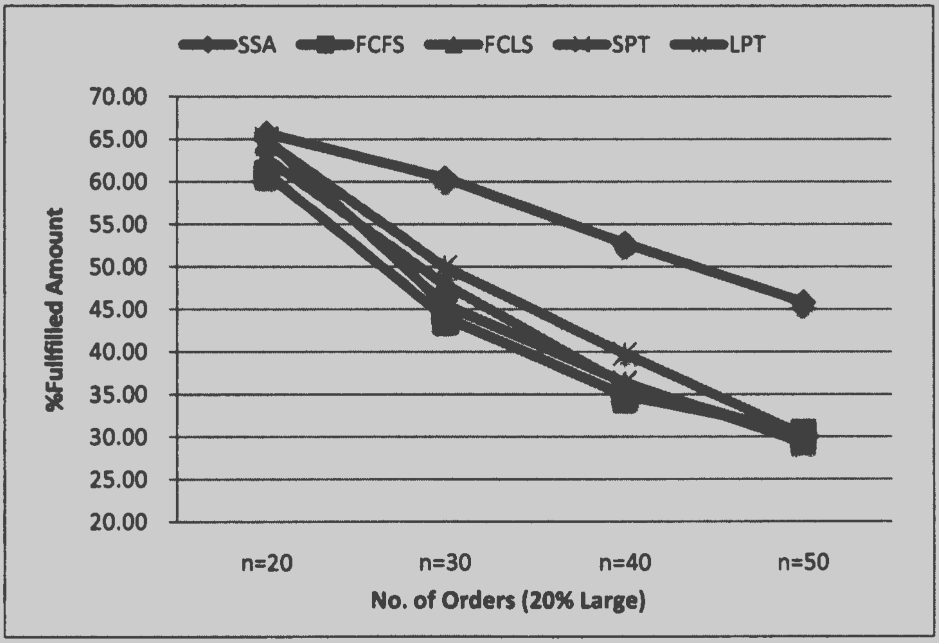

Figure 29. The %fulfilled amount with 20% large orders.

Similar reduction found with other four methods. The decreasing rate varies slightly but close to a line for SSA, while it shows differently with the four classic rules. However, the difference resulted from using the five methods can be neglected. That could be because when the amount of large orders and total order amount are both small, the impact of large orders on the system capacity has not revealed yet. At level 50, the four rules are outperformed by SSA. SSA yields a result around 15% higher than any others.

Results for SSA are still better than other four methods. Figure 30 shows that the reduction in 40% cases as the total number of orders increases follows the same direction of 20% case.

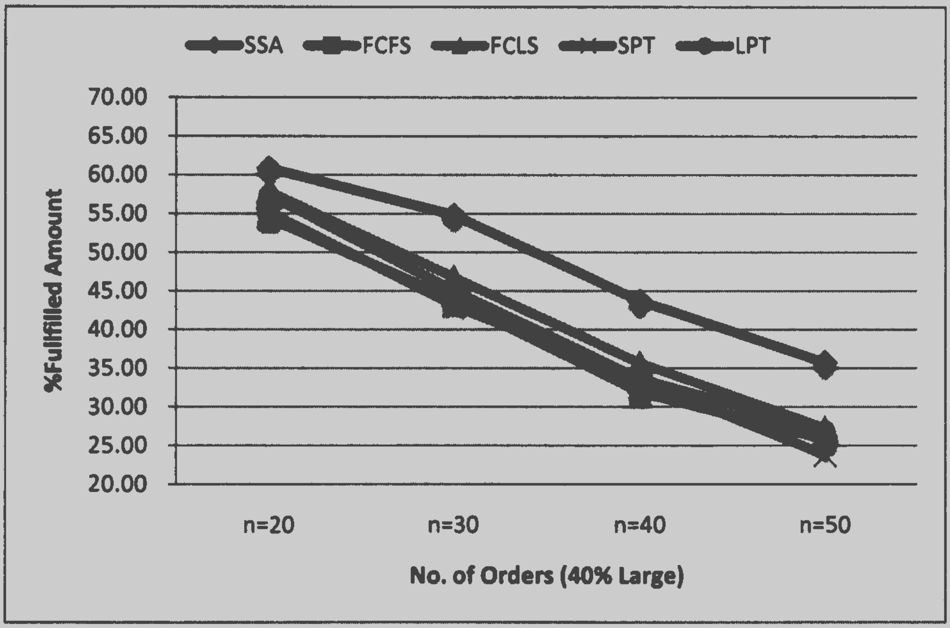

Figure 30. The %fulfilled amount with 40% large orders.

It confirms the previous finding that at the level 20, the effectiveness of SSA does not show significant difference from the four rules, but when it increases to 30, SSA is the best of all. Although the decreasing trend in the percentage fulfilled amount is unavoidable, scheduling with SSA, the impact of large orders can be reduced to minimum. Especially, it gives nearly 10% improvement in the figure compared with other methods.

5.4. Discussion of simulation results

Among all five methods, SSA always yields the nearly best results (except in the case

20 orders with 20% large orders). The decreasing of fulfill rate is almost the same for both scenarios, since the impact of large orders becomes more and more severe when the demand rate increases. As when the total number of orders reaches to 40 from 30, the slope of reduction in 40% case is sharper than that of 20% (Figure 31).

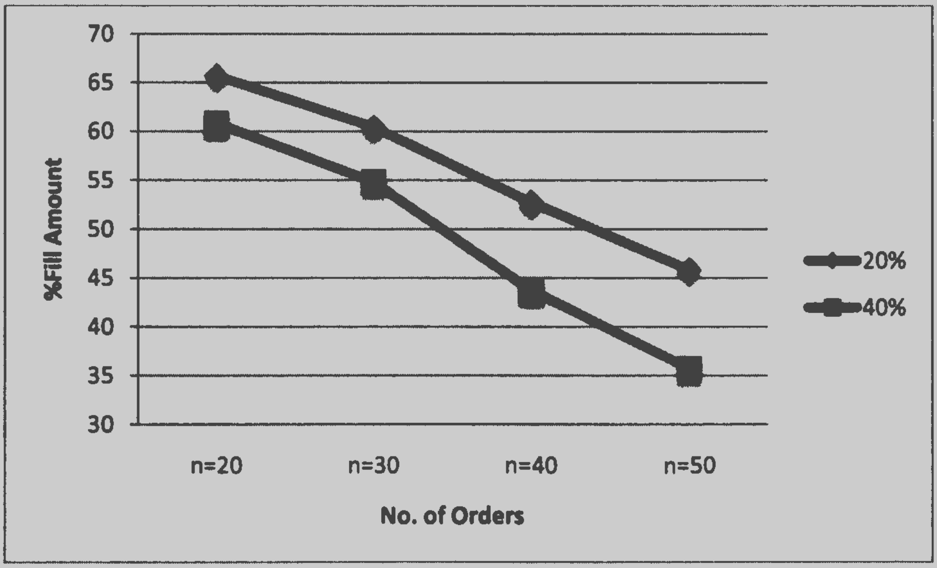

Figure 31. The plot of results obtained with SSA.

SSA shows its advantage when demand is higher than 20 with the presence of large order effect in either 20% or 40% cases, which has met the primary purpose of investigating increased demand.

Another finding is that among all four dispatching rules, LPT yields the best results in the 20-order case. While in the 40-order case, even the strategy of producing large orders first cannot compensate the impact of large order on the throughput. Although FCFS yields the lowest consistently in the two graph, it shows in the 40% case, at level 20 and 30 it yields nearly the same as SPT does. The gap increases when the number of orders reaches to 40 and FCFS outperforms SPT again when the number is 50. The reason for the crossover is probably regarding the previous leftovers; capacity has been assigned to orders left from the previous periods and the remaining capacity for newly-received orders decreases. Thus, even with more small-quantity fast-running orders may not overcome the limitation of capacity. We can conclude that to develop a sorting strategy is very important when the demand is high and single order quantity is large.

5.5. Designing rules for the expert system

SSA has been intensively used during simulation runs. The results coming from each run of splitting orders or making selection of candidates are printed into an individual file at the time of executing simulation runs. A rule-based expert system was designed to generalize scheduling rules based on the results so that human schedulers can take the advantage of simulation results. This is especially useful to make quick scheduling decision at the time of receiving orders with no need of technical knowledge. Since the four factors have been already defined in Chapter 4, the levels of factors are further specified in the following sections.

In the previous chapter, the four factors that make impact on the decision of order scheduling have been introduced, total quantity, quantity composition, temperature group and total number of orders. Each of them has two levels. To find out in which situation the received or unscheduled orders need to be triaged or split, or in other words, to examine the conditions that system capacity is overloaded, the first step is to investigate the combination of the levels of these factors. Here, the levels of factors are re-defined and the interactions among factors are identified.

5.5.1. Factors

5.5.1.1. Finding the critical value of factor A

The results of generation tests for finding the exact level of factor A are summarized in Table 9.

Table 9. Determination of single-order total quantity

Run Index	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C8**	**C9**	**C10**	**C11**
Total Quantity (mg)	100	400	1000	10000	5500	3500	4500	4000	4250	4125	4200
						Calculated Utilization (%)					
Stage											
Transformation	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
Incubation	2.56	2.56	2.56	2.56	2.56	2.56	2.56	2.56	2.56	2.56	2.56
Starter Culture	5.56	5.56	5.56	5.56	5.56	5.56	5.56	5.56	5.56	5.56	5.56
Inoculation	2.36	9.43	23.59	239.15	131.13	83.49	107.55	95.28	101.42	98.59	100.00
Lysis	3.94	7.87	19.68	199.12	109.40	69.26	89.72	79.49	84.21	81.85	83.43
AEX	7.97	9.43	13.82	80.48	47.15	32.24	39.84	36.04	37.79	36.92	37.50
Average system utilization	4.56	6.64	11.70	88.65	50.13	33.02	41.71	37.32	39.42	38.41	39.01

Factor A, total quantity, an integer value, has two levels: >critical value B (+), <=critical value B (-). The boundary between two levels with regard to the total quantity is defined as a critical value B. Random order generation, with various total quantity values while fixing the values of other three factors, is performed in Model 1 to find this critical value. Only one order at a time will be generated.

As it can be seen from the table, although in some cases, the average system utilization is below 100%, the utilization has exceeded 100% at some stages. Case 11 gives the optimal value of B, which is 4200mg, compared with other cases studied. Therefore, the two levels of factor A are re-defined: >4200 (+), <=4200 (-).

Moreover, Primary Rule A is also obtained:

Primary Rule A: If only one order is to be scheduled, as long as its individual quantity is less than or equal to 4200mg), the order can be fulfilled without splitting.

5.5.1.2. Specifying the levels of factor B

The levels of factor B are initially defined as whether large orders dominate in the entire batch of orders. To be specific and clear, large-quantity orders have been defined as those with desired quantity over or equal to 100mg. The domination characteristic is further specified as the percentage of total quantity of large order over the total quantity of all the orders; 50% indicates the domination.

5.5.1.3. Determining the levels of factor C

Temperature group forms when the orders are to be processed in the same temperature. The level Single indicates only one temperature is involved, whereas Multiple can have 2 or 3 temperatures.

5.5.1.4. Redefining factor D

Factor D represents the total amount of orders to be scheduled. As factor D is the only factor that affects the capacity on the first stage, Transformation, therefore, the lower bound for factor D at higher level is 20. That defines another primary rule B.

Primary Rule B: if the number of the unscheduled orders is more than 20, then the facility is over-capacitated and decisions of making selection among selecting, selecting or splitting need to be made.

Since all the factors have been specified, the refined factor levels are summarized in Table 10.

Table 10. Re-defined factor levels

Index	**Factor**	**Factor Type**	**Lower Level (-)**	**Upper Level (+)**
A	Total quantity	Real	Less than 4200 mg	Greater than 4200mg
B	Quantity composition	Percentage	Less than 50%	More than 50%
C	Temperature groups	Integer	Single (only 1)	Multiple (2 or 3)
D	Total number of orders	Integer	Less than 20 orders	Greater than 20 orders

5.5.2. Combined effects of factors

Although levels of factors are predefined, their interactions among levels are not clearly identified. In some scenario, there could be the case that only three of them have impact on decision making, or there exists the situation only one of them actually influences the capacity utilization. To find the combination which leads to the decision of processing or not processing, splitting or not splitting, or selecting and splitting, more simulation tests are performed. Experiment is carried out in simulation Model 1 where orders are generated randomly. It is similar to the concept of factorial design, while the difference is that we are finding the combinations that will make the system overloaded or underutilized instead of investigating the significance of effect of factors. Response value is the capacity utilization of each processing stage. Table 11 shows the experiment design table.

Table 11. Design of experiments

Combination	A	B	C	D
1	-	-	-	-
B	-	+	-	-
AB	+	+	-	-
C	-	-	+	-
BC	-	+	+	-
ABC	+	+	+	-
D	-	-	-	+
AD	+	-	-	+
BD	-	+	-	+
ABD	+	+	-	+
CD	-	-	+	+
ACD	+	-	+	+
BCD	-	+	+	+
ABCD	+	+	+	+

However, the combination A and AC do not exist due to the conflict caused by the higher level of factor A, and the lower level of factor B and factor D. Therefore, they have been removed from the original table. Each combination is replicated 4 times. Therefore, in total, there should be $(2^4 - 2) \times 4 = 56$ runs. An example of rule generation results is given in Table 12.

Table 12. Example of data obtained through rule development

Run Index	Transformation	Incubation	Culture	Inoculation	Lysis	AEX	Average system utilization	Decision
1	25.00	12.82	27.78	7.55	12.59	16.30	17.01	Not to process
B	50.00	19.87	43.06	13.68	22.82	25.95	29.23	Process all
AB	25.00	8.33	18.06	101.89	86.57	43.35	47.20	Select (SSA)
BC	50.00	16.03	34.72	8.49	14.17	22.37	24.30	Not to process
C	80.00	21.80	47.22	8.96	14.95	29.61	33.76	Process all
ABC	25.00	7.69	16.67	92.45	78.70	40.42	43.49	Process all
BD	125.00	41.67	90.28	40.57	53.52	54.83	67.64	Apply SSA
D	200.00	95.51	206.94	53.77	89.72	88.60	122.43	Apply SSA
AD	200.00	102.56	222.22	93.87	136.16	106.51	143.55	Apply SSA
ABD	200.00	82.69	179.17	356.60	314.03	170.91	217.23	Apply SSA
BCD	125.00	39.10	84.72	20.28	33.84	47.44	58.40	Apply SSA
ACD	250.00	128.21	277.78	87.74	146.39	135.53	170.94	Apply SSA
ABCD	125.00	46.15	100.00	210.38	192.82	107.16	130.25	Apply SSA

5.5.3. Generation of deductive Rules

According to the decision-making procedure defined in Chapter 4, the decisions regarding scheduling consist of Process All, Not to Process, Split, Select and Apply SSA. The batch of orders including more large orders adapts to need more capacity to be processed, while that contains less number of orders trend to be rejected for being processed if the system performance is lower than expected. The threshold associated with the average system utilization is postulated as 25% in this study. However, if the users have a higher or lower threshold, it can be modified. Based on the experimentation, rules are obtained as follows (Table 13) (if levels of a factor are not specified, any level of the factor applies):

Table 13. Rules and descriptions

Rule	Description
Rule 1	If the total number of orders is less than or equal to 10, and total quantity is less than or equal to 4200mg, then the batch of orders should not be processed.
Rule 2	If the total number of orders is less than or equal to 10, more than one temperature is required, and total quantity is more than 4200mg, then the batch of orders should be processed.
Rule 3	If the total number of orders is less than 10, only one temperature is required and total quantity is more than 4200mg, then some orders should be split.
Rule 4	If the total number of orders is less than or equal to 10, more than one temperature is required, large orders are more than 80% and total quantity is more than 4200mg, then one of the orders should be split.
Rule 5	If the total number of orders is greater than or equal to 10 but less than or equal to 20, and total quantity is less than 4200mg, then some order should be split.
Rule 6	If the total number of orders is greater than or equal to 10 but less than or equal to 20, and total quantity is more than 4200mg, then some orders in the batch should be split.
Rule 7	If the total number of orders is greater than 20, one should use Model 1 to select orders in the batch to schedule.

Rules are evaluated by simulation tests. They have been put into the expert system developed by CLIPS™. If rules indicate that splitting and selecting needs to be done, then one can use SSA to find the final solution. An example is given below.

If in the case that total number of orders is greater than 20 and then the knowledge base reveals that some order should be split. The simulation models are capable of developing the final production list based on the input of the initial order list including order quantity amount, required temperature setting, etc. A set of orders are generated as in Table 14 and the final production list is provided in Table 15.

Table 14. Initial production list

Lot index	Order Quantity(mg)	No. of starters	Temperature (°C)	Medium (type)	Medium Vol. (L)
1	13	2	37	LB	1
2	496	4	34	MY	22
3	218	3	37	MY	9
4	37	2	37	LB	3
5	487	3	34	MY	13
6	269	3	34	MY	8
7	275	3	37	MY	15
8	64	3	34	MY	1
9	73	3	37	MY	4
10	85	4	37	MY	3
11	86	4	37	MY	3
12	476	3	37	MY	19
13	480	4	34	MY	17
14	220	4	37	MY	9
15	23	4	30	LB	1
16	492	4	34	MY	13
17	39	2	37	LB	2
18	47	3	37	LB	4
19	27	2	37	LB	2
20	488	4	34	MY	14
21	247	4	34	MY	5
22	24	4	37	LB	1
22	24	2	37	LB	2

Total quantity
4690

Total number of orders: 23

No. 30 degree orders: 1

No. 34 degree orders: 8

No. 37 degree orders: 14

Utilization estimation

Stage	Calculated Utilization (%)
Transformation	115.00
Incubation	47.44
Starter Culture	102.78
Inoculation	80.66
Lysis	88.15
AEX	43.08
Average system utilization	79.52

Table 15. Final production list

Lot index	Order quantity(mg)	No. of starters	Temperature (°C)	Medium (type)	Medium Vol.(L)
2	496	4	34	MY	22
16	492	4	34	MY	13
20	488	4	34	MY	14
5	487	3	34	MY	13
13	480	4	34	MY	17
12	476	3	37	MY	19
7	275	3	37	MY	15
6	269	3	34	MY	8
21	247	4	34	MY	5
14	220	4	37	MY	9
3	218	3	37	MY	9
11	86	4	37	MY	3
10	85	4	37	MY	3
9	73	3	37	MY	4
8	64	3	34	MY	1
18	47	3	37	LB	4
17	39	2	37	LB	2
4	37	2	37	LB	3
19	27	2	37	LB	2
22	24	4	37	LB	1

Total quantity: 4630

Total number of orders: 20

No. 30 degree orders: 0

No. 34 degree orders: 8

No. 37 degree orders: 12

Utilization estimation

Stage	Calculated Utilization (%)
Transformation	100.00
Incubation	42.31
Starter Culture	91.67
Inoculation	78.77
Lysis	85.00
AEX	39.18
Average system utilization	72.82

Total number of orders scheduled: 20	Total number of orders remain unscheduled: 3
Required usage of LB medium: 13 L	Required usage of MY medium: 155 L

CHAPTER 6. CONCLUSION AND FUTURE DIRECTION

6.1. Conclusion

Production planning and scheduling techniques are extremely useful in improving the efficiency of make-to-order manufacturing systems. Faced with a great deal of uncertainty, it becomes difficult to solve the NP-hard problem in a process system, i.e., plasmid industry. The variety involved in the product mix results in various scheduling strategies. However, the manual scheduling strategy currently used by the company is not optimal, since the company has suffered from the pain of overloaded system and lack of efficiency. In order to deal with the uncertainties as well as plan the production adaptive to changes in the system, a simulate-based short-term planning and scheduling tool with heuristic selecting and splitting algorithms (SSA) is proposed. Four different dispatching rules are tested in comparison with the proposed solution procedure. The results show that under the combination of high demand rate and the presence of more large orders, the proposed method outperforms the others. In order to store the findings from simulation experimentation and allow schedulers who know little or lack the experience working with simulation models to make quick scheduling decision at the time of receiving orders, an expandable knowledge-base is designed. Deductive rules are created with the help of the expert system tool.

Although a certain number of assumptions were made in order to simplify the system as well as reduce the effect of uncertain parameters, the research work that strictly follows the processing requirements is practical to some degree and schedulers can utilize the knowledge-base system make quick response to order scheduling. Additionally, the logic-structured scheduling system is extremely flexible. By modifying the values of some input parameters, the system can be updated to match any similar system settings. The independent expert system, although is not equipped with a database and self-learning module, can be expanded and incorporated with other systems since it is compatible with C language. As the programming language is straightforward and the logic has already been defined, anyone who is able to identify the structure of the knowledge rules can modify it so that the up-to-date information can be used in the future. Especially, simulation is a powerful tool that allows users to conduct virtual experiment without changing the parameters in the current system when considering system redesign. Furthermore, the entire simulation-based scheduling and planning system combined with the expert module can be applied to the similar process systems manufacturing customized products, such as chemical, pharmaceutical, food, etc.

6.2. Future directions

Since plasmid industry is relatively new to industrial engineering technological world, it carries the characteristics of both biological and manufacturing processes. There are some issues that have not been done in this research but merit attentions.

6.2.1. Long-term scheduling issue

This research focuses on developing a short-term planning and scheduling solution without considering its subsequent impact on the next planning period. At present, with relatively low demand rate, the short-term algorithm is adequate enough to deal with a small amount of leftovers and increasing amount of new orders. However, if taking into account the potential dramatic increase in demand and cost-profit balance, then the short-term scheduling procedure is myopic and med-term or long-term scheduling procedure would be required.

6.2.2. Design a robotic knowledge-based system

Nowadays the use of knowledge-based supporting system has grown rapidly. More sophisticated and computerized knowledge-based scheduling system or manufacturing system can vastly enlarge the scopes of human schedulers so that more possible scenarios can be developed and more simulation runs can be defined. Additionally, the biological decision making criteria, such as the determination of the number of starters and type of bacteria strains to be used with each order, can be integrated into the rule-based expert system so that users do not need to acquire the particular knowledge in order to plan production and schedule orders. Moreover, by evaluating and employing the historic processing experience into the knowledge-based system, the system can get evolved over time.

6.2.3. Employ LP-based search algorithms

Even though LP seems hard to solve NP-hard problems in large scale because the dimension of the problem creates a great number of variables, the implementation of LP-based search techniques, i.e., Genetic Algorithm, can be used.

6.2.4. Introducing additional constraints

The due-date requirement is not discussed in this thesis. However, there could be the case where due-date cannot be met and penalty occurs. Thus, for those orders urgent or with penalty cost, cost or priority coefficient associated with each order will have to be developed. Also, the work schedule may be modified to provide more capacity to the system and new workforce plan that balances the capacity and demand can be developed.

REFERENCES

1. Anonymous. (n.d.). *Bacterial Transformation*. Retrieved April 2010, from Merck Source: http://www.mercksource.com/pp/us/cns/cns_hl_dorlands_split.jsp?pg=/ppdocs/us/common/dorlands/dorland/eight/000110198.htm
2. Arakawa, M., Fuyukia, M., & Inoueb, I. (2003). An optimization-oriented method for simulation-based job shop scheduling incorporating capacity adjustment function. *International Journal of Production Economics* , *85*, 359–369.
3. Banks, J. (2004). *Getting Started With AutoMod (Second Edition).* Chelmsford, MA, USA: Brooks Automation, Inc.
4. Bellanger, A., & Oulamara, A. (2009). Scheduling hybrid flowshop with parallel batching machines and compatibilities. *Computers & Operation Research* , *36*, 1982-1992.
5. Blomer, F., & Gunther, H. O. (1998). Scheduling of a multi-product batch process in the chemical industry. *Computers in Industry* , *36*, pp. 245-259.
6. Brancaleoni, C., Bugno, L., Cavalloro, P., Neuss, P., & McLaren, B. (1988). A Knowledge-Based Simulation of a PCB Manufacturing Plant. *4th International Conference on Simulation in Manufacturing.* London, England.

7. Caloss, T., Cantamessaa, M., Vub, D., & Villaa, A. (2003). Production planning and order acceptance in business to business electronic commerce. *International Journal of Production Economics , 85*, 233–249.

8. Chen, C., Mestry, S., Damodarana, P., & Wang, C. (2009). The capacity planning problem in make-to-order enterprises. *Mathematical and Computer Modelling , 50*, 1461-1473.

9. Damodaran, P., Manjeshwar, P. K., & Srihari, K. (2006). Minimizing makespan on a batch-processing machine with non-identical job sizes using genetic algorithms. *International Journal of Production Economics , 103*, 882-891.

10. Danquah, M. K., & Forde, G. M. (2007). Growth Medium Selection and Its Economic Impact on Plasmid DNA Production. *Journal Of Bioscience And Bioengineering , 104(6)*, 490–497.

11. Dastidar, S. G., & Nagi, R. (2007). Batch splitting in an assembly scheduling environment. *International Journal of Production Economics , 105*, 372–384.

12. Dellaert, N. P., & Melo, M. T. (1998). Make-to-order policies for a stochastic lot-sizing problem using overtime. *International Journal of Produciton Economics , 56-57*, 79-97.

13. Dorin, V., & Pănescu, D. A. (2001). *Flexible manufacturing systems management using expert systems.* Retrieved March 09, 2010, from Department of Automatic Control and

Industrial Informatics, Technical University of Iasi, Romania: www.ac.tuiasi.ro/grant_CNFIS0006/s94.pdf

14. Durkin, J. (1990). Application of Expert Systems in the Sciences. *Expert Systems In The Sciences , 90* (5), 171-179.

15. Durland, R. H., & Eastman, E. M. (1998). Manufacturing and quality control of plasmid-based gene expression system. *Advanced Drug Delivery Reviews , 30*, 33-48.

16. Ferreira, G. N., Monteiro, G. A., Prazeres, D. M., & Cabral, J. M. (2000). Downstream processing of plasmid DNA for gene therapy and DNA vaccine applications. *Trends in Biotechnology , 18*, 380-388.

17. Gomes, M. C., Barbosa-Povoa, A. P., & Novais, A. Q. (2006). Optimal reactive scheduling of multi-purpose make-to-order industries. *16th European Symposium on Computer Aided Process Engineering and 9th International Symposium on Process System Engineering*, (pp. 1587-1592).

18. Gonzalez, L. R., Garcia, M. L., & Centeno, M. A. (1996). On-line knowledge-based simulation for fms: a state of the art surviey. *Proceedings of the 1996 Winter Simulation Conference*, (pp. 1057-1061).

19. *Growth medium-Wikipedia, the free encyclopedia.* (n.d.). Retrieved from http://en.wikipedia.org/wiki/Growth_medium

20. Halevi, G., & Wang, K. (2007). Knowledge based manufacturing system (KBMS). *Journal of Intelligent Manufacturing*, *18*, 467–474.

21. Hejazi, S. R., & Saghafian, S. (2005). Flowshop-scheduling problems with makespan criterion: a review. *International Journal of Production Research*, *43(14)*, 2895-2929.

22. Henning, G. P., & Cerda, J. (1996). A knowledge-based approach to production scheduling for. *Computers & Chemical Engineering*, *20*, S1295-S1300.

23. Henning, G. P., & Cerda, J. (2000). Knowledge-based predictive and reactive scheduling in industrial environments. *Computers and Chemical Engineering*, *24*, 2315-2338.

24. Jalora, A. (2006). *Order acceptance and scheduling at a make-to-order system using revenue management.* Retrieved from Texas A&M University: http://hdl.handle.net/1969.1/4421

25. Kolisch, R. (2000). Integration of assembly and fabrication for make-to-order production. *Int. J. Production Economics*, *68*, 287-306.

26. Kropp, D. H., & Smunt, T. L. (1990). Optimal and heuristic models for lot splitting in a flow shop. *Decision Sciences*, *21* (4), 691.

27. Lau, S., Willis, M., Montague, G., & Glassey, J. (2003). Predictive scheduling of a bioprocess plant. *Proceeding of the IASTED International Conference, Modeling,*

Simulation, and Optimization, (pp. 296-301). Banff, Alberta, Cananda.

28. Lederberg, J. (1952). Cell genetics and hereditary symbiosis. *Physiological Reviews* , *32(4)*, 403-430.

29. Lorentzos, N. A., Yialouris, C. P., & Sideridis, A. B. (1999). Time-evolving rule-based knowledge bases. *Data & Knowledge Engineerin* , *29*, 313-335.

30. Mohamed, A., & Celik, T. (2002). Knowledge-based system for alternative design, cost estimating and scheduling. *Knowledge-Based Systems* , *15*, 177-188.

31. Nandi, A., & Rogers, P. (2003). Simulation-based order acceptance in make-to-order manufacturing systems. *Summer Computer Simulation Conference.*

32. Neureuther, B. D. (2004). Aggregate planning in make-to-order environments. *Proceedings of the Second World Conference on POM and 15th Annual POM Conference.* Cancun, Mexico.

33. O'Kennedy, R. D., Baldwin, C., & Keshavarz-Moore, E. (2000). Effects of growth medium selection on plasmid DNA production and initial processing steps. *Journal of Biotechnology* , *76*, 175-183.

34. Oulamara, A., Finke, G., & Kuiteing, A. K. (2009). Flwoshop scheduling problem with a batching machine and task compatibilities. *Computer & Operation Research* , *36*, 391-401.

35. Ozbayrak, M., & Bell, R. (2003). A knowledge-based decision support system for the management of parts and tools in FMS. *Decision Support Systems*, *35*, 487– 515.

36. Palaniswami, S., & Jenicke, L. (1992). A knowledge-based simulation system for manufacturing scheduling. *International Journal of Operations & Production Management*, *12* (11), 4-11.

37. Pinedo, M. (2002). *Scheduling theory, algorithms, and systems* (2nd ed.). Upper Saddle River, New Jersey: Prentice Hall Inc.

38. *Plasmid - Types Of Plasmids, Replication, Use In Research And Technology.* (n.d.). Retrieved from http://medicine.jrank.org/pages/2653/Plasmid.html

39. Prather, K. J., Sagar, S., Murphy, J., & Chartrain, M. (2003). Industrial scale production of plasmid DNA for vaccine and gene therapy: plasmid design, produciton, and purification. *Enzyme and Microbial Technology*, *33*, 865-883.

40. Rim, S., & Park, I. (2008). Order picking plan to maximize the order fill rate. *Computers & Industrial Engineering*, *55*, 557-566.

41. Sawik, T. (2007). A lexicographic approach to bi-objective scheduling of single-period orders in make-to-order manufacturing. *European Journal of Operational Research*, *180*, 1060–1075.

42. Shim, S., & Kim, Y. (2008). A branch and bound algorithm for an identical parallel

machine scheduling problem with a job splitting property. *Computers & Operations Research* , *35*, 863-875.

43. Soman, C. A., Van Donk, D. P., & Gaalman, G. (2004). Combined make-to-order and make-to-stock in a food production system. *International Journal of Production Economics* , *90*, 223-235.

44. Soman, C., Van Donk, D., & Gaalman, G. C. (2007). Capacitated planning and scheduling for combined make-to-order and make-to-stock production in the food indsutry: an illustrative case study. *International Journal of Production Economics* , *108*, 191-199.

45. Sox, C. R., Jackson, P. L., Bowman, A., & Muckstadt, J. A. (1999). A review of the stochastic lot scheduling problem. *Int. J. Production Economics* , *62*, 181-200.

46. Trietsch, D., & Baker, K. R. (1993). Basic techniques for lot streaming. *Operations Research* , *41* (6), 1065.

47. Xing, W., & Zhang, J. (2000). Parallel machine scheduling with splitting jobs. *Discrete Applied Mathematics* , *103*, 259-269.

48. Xuan, H., & Tang, L. (2007). Scheduling a hybrid flowshop with batch production at the last stage. *Computers & Operations Research* , 2718 – 2733.

49. Zeigler, B. P., Cho, T. H., & Reozenblit, J. W. (1996). A knowledge-based simulation

environment for hierarchical flexible manufacturing. *IEEE Transactions On Systems, Man, And Cybernetics-Part A: Systems And Humans* , *26* (1), 81-90.

APPENDIX A. SIMULATION MODEL CODE

```
/*system begins operating at 6:00am everyday*/
begin model initialization function
        create 1 load of load type L_control to P_load        /*L_load: to control the system
runs*/
/*      create 1 load of load type L_control to P_read*/  /*for reading historic data only*/
        create 1 load of type L_dummy to P_control
        take down R_operator(1)                    /*at the first beginning, no labor is working*/
        take down R_operator(2)
        take down R_operator(3)
        take down R_operator(4)
        take down R_operator(5)
        take down R_operator(6)
        take down R_operator(7)
        take down R_operator(8)
        take down R_operator(9)
        take down R_operator(10)
        take down R_operator(11)
        return true
end

/*for reading historic data only**************************************************/
/*begin P_read arriving
        open "arc/data.txt" for reading save result as V_file
        read V_headers from "arc/data.txt" with delimiter "\n"
        while V_file eof is false do
        begin
                read A_date,A_quant,A_lot from "arc/data.txt" with delimiter "\t"
                if V_date=null then
                begin
                        set V_date=A_date
                        clone 1 load to P_lot nlt L_order
                end
                else
                        if V_date=A_date then clone 1 load to P_lot nlt L_order
                        else
                        begin
                                print "Date\t", "lot index\t", "Order quantity\t",
"No.starters\t", "Temperature\t", "Medium\t", "Medium Vol.\t" to "arc/list.txt"
                                set V_date=A_date
                                wait for 24 hr
                                clone 1 load to P_lot nlt L_order
                                inc V_day by 1
                        end
        end
end

begin P_lot arriving   /*generater orders for that particular day*/
        if A_rework=0 then
        begin
                inc V_index by 1
                set A_index=V_index
        end
        else
        begin
                set V_flask(A_index)=0
                set V_jar(A_index)=0
                set V_tips(A_index)=0
                set V_bottle(A_index)=0
        end
        if A_quant<50 then
        begin
                set A_medium="LB"
                set A_temp=oneof(2:30,5:34,93:37)
                if A_temp=30 then set A_paste=7.548   /*paste yield rate (g/L)*/
                else if A_temp=34 then
                begin
                        set A_paste=n 6.466, 2.139
                        if A_paste<3.27 then set A_paste=3.27
                end
                else
        begin
            set A_paste= n 6.428, 1.848
            if A_paste<1.167 then set A_paste=1.167
        end
    end

    else
    begin
        if A_rework=0 then
            if A_quant>=100 then set A_YT=1
        set load type to L_large
        set A_medium="MY"
        set A_temp=oneof(1:30,44:34,55:37)
    if A_temp=30 then set A_paste=16.667  /*paste yield rate (g/L)*/
        else if A_temp=34 then
        begin
            set A_paste=n 13.900, 5.275
            if A_paste<4.75 then set A_paste=4.75
        end
        else
        begin
            set A_paste=n 10.49,2.145
            if A_paste<5.7 then set A_paste=5.7
        end
    end

    set A_time to ac   /*set time when order enters the system */

    set V_LossRate=0.998 /*loss rate from paste to final product*/

    call S_assign

    if A_rework=0 then print V_day"\t", A_index"\t", A_quant"\t", A_starter"\t",
A_temp"\t", A_medium"\t", A_Mvol"\t" to "arc/list.txt"

    send to P_pretransformation
```

```
end
*/

/*read from files********************************************************/
/*begin P_read arriving
    open "arc/data.txt" for reading save result as V_file
    read V_headers from "arc/data.txt" with delimiter "\n"
    while V_file eof is false do
    begin
        read A_day,A_index,A_quant,A_starter, A_temp, A_medium, A_Mvol from
"arc/data.txt" with delimiter "\t"
        if V_day=0 then
        begin
            set V_day=A_day
            clone 1 load to P_lot nlt L_order
        end
        else if V_day=A_day then clone 1 load to P_lot nlt L_order
        else
        begin
            print "Day\t", "lot index\t", "Order quantity\t", "No.starters\t",
"Temperature\t", "Medium\t", "Medium Vol." to "arc/list.txt"
         set A_last=1
            set V_day=A_day
            clone 1 load to P_lot nlt L_order
         wait for 24 hr
         set A_last=0
            clone 1 load to P_lot nlt L_order
        end
    end
end*/


/*control the operators break*****************************************************/
begin P_control arriving
    clone 1 load to P_break(1) nlt L_break /*L_break: to control the break of labors*/
    clone 1 load to P_break(2) nlt L_break
```

```
        clone 1 load to P_break(3) nlt L_break
        clone 1 load to P_break(4) nlt L_break
        clone 1 load to P_break(5) nlt L_break
        clone 1 load to P_break(6) nlt L_break
        clone 1 load to P_break(7) nlt L_break
        clone 1 load to P_break(8) nlt L_break
        clone 1 load to P_break(9) nlt L_break
        clone 1 load to P_break(10) nlt L_break
        clone 1 load to P_break(11) nlt L_break
        send to die
end


/*load generation******************************************************/
begin P_load arriving
        set V_day=1
        print "Date\t", "lot index\t", "Order quantity\t", "No.starters\t", "Temperature\t",
"Medium\t", "Medium Vol.\t" to "arc/list.txt"
        print "Lot index\t", "Order quantity(mg)\t", "Manufacturing Lead Time" to
"arc/time.txt"
        while 1=1 do
        begin
                set V_No=30  /*20,30,40,50 scenarios*/
                set V_c=50
                set V_new=V_No

                clone V_No loads to P_generate nlt L_order  /*L_order: to generate
individual orders*/

                print "+++++++++++++++" to "arc/case.txt"
                print "Day" V_day"\n" to "arc/case.txt"
                print "lot index\t", "Order quantity(mg)\t", "No.starters\t",
"Temperature(C)\t", "Medium (type)\t", "Medium Vol.(L)\t", "Split?(0=non-
split,1=split,2=leftover)\t", "Leftover?(0=non-leftover,other=days)\t", "tip" to "arc/case.txt"
                wait for 24 hr
                inc V_day by 1
                call S_reset
```

```
            set V_lot=0
            set V_new=0
        end
        send to die
end


begin P_generate arriving
        while V_No>0 do
        begin
            if V_No=V_new then
            begin
                set A_first=1
                clone 1 load to P_store
                set A_first=0
            end
            inc V_index by 1
            set A_index=V_index /*A_index: the numeric value corresponding to lot
number/order ID(unique)*/
            dec V_No by 1
            if V_No=0 then set A_last=1 /*if the load is the last one to be scheduled*/
            clone 1 load to P_lot
        end
end


begin P_lot arriving   /*generater orders for that particular day*/
/*      set A_quant to nextof(100,200,50,90,90)  /*replication 1*/
/*      set A_quant to nextof(300,700,50,900,90)   /*replication 2*/ */
/*      set A_quant to nextof(10,500,5,100,2)  /*replication 3*/   */
/*      set A_quant to nextof(1000,20,50,100,5)  /*replication 4*/*/
        set V_LossRate=0.998 /*loss rate from paste to final product*/
        if A_quant<=50 then
        begin
            set A_medium="LB"
            set A_temp=oneof(2:30,5:34,93:37)
        set A_starter to oneof(66:2,28:3,6:4)
        if A_temp=30 then set A_paste=7.548   /*paste yield rate (g/L)*/
        else if A_temp=34 then
        begin
            set A_paste=n 6.466, 2.139
            if A_paste<3.27 then set A_paste=3.27
        end
        else
        begin
            set A_paste= n 6.428, 1.848
            if A_paste<1.167 then set A_paste=1.167
        end
    end
    else
    begin
        set A_medium="MY"
        set A_temp=oneof(1:30,44:34,55:37)
        set A_starter to oneof(40:3,60:4)
        set load type to L_large
    if A_temp=30 then set A_paste=16.667  /*paste yield rate (g/L)*/
        else if A_temp=34 then
        begin
            set A_paste=n 13.900, 5.275
            if A_paste<4.75 then set A_paste=4.75
        end
        else
        begin
            set A_paste=n 10.49,2.145
            if A_paste<5.7 then set A_paste=5.7
        end
    end

    set A_yield=A_paste*1000*(1-V_LossRate)   /*final yield rate*/
set A_Mvol=A_quant/A_yield
if A_Mvol<1 then set A_Mvol=1
    if A_YT=1 then set A_Mvol=1
```

```
          set A_flask=F_flask(A_Mvol)
          set A_jar=F_jar(A_Mvol)
          set A_falcon=F_falcon(A_paste)
          set A_tip=F_tip(A_paste)

          call S_gather

          set A_time to ac   /*set time when order enters the system */

         print V_day"\t", A_index"\t", A_quant"\t", A_starter"\t", A_temp"\t", A_medium"\t",
A_Mvol"\t" to "arc/list.txt"

/*        if A_last=1 then clone 1 load to P_CapPrint
          wait to be ordered on OL_orders*/
          send to P_store   /*for experimentation*/
end

/*order selection and splitting algorithm*****************************/
begin P_CapPrint arriving
          call S_capacity
          print "\n" to "arc/case.txt"
          print "Total quantity\t", V_total as .2"\n" to "arc/case.txt"
          print "Total number of orders:\t", V_TotalNo to "arc/case.txt"
          print "No. 30 degree orders\t", V_30 to "arc/case.txt"
          print "No. 34 degree orders\t", V_34 to "arc/case.txt"
          print "No. 37 degree orders\t", V_37"\n" to "arc/case.txt"

          print "Utilization estimation\n" to "arc/case.txt"
          print "Stage\t", "Calculated Utilization(%)\t" to "arc/case.txt"
          print "Transformation\t", V_utilization(1)*100 as .3"\t" to "arc/case.txt"
          print "Incubation\t", V_utilization(2)*100 as .3"\t" to "arc/case.txt"
          print "\t",V_utilization(21)*100 as .3"\t", "(process 30-degree orders only)"  to
"arc/case.txt"
          print "\t",V_utilization(22)*100 as .3"\t", "(process 34-degree orders only)" to
"arc/case.txt"
```

```
          print "\t",V_utilization(23)*100 as .3"\t", "(process 37-degree orders only)\n" to
"arc/case.txt"

          print "Starter Culture\t", V_utilization(3)*100 as .3"\t" to "arc/case.txt"
          print "\t",V_utilization(31)*100 as .3"\t", "(process 30-degree orders only)" to
"arc/case.txt"
          print "\t",V_utilization(32)*100 as .3"\t", "(process 34-degree orders only)"  to
"arc/case.txt"
          print "\t",V_utilization(33)*100 as .3"\t", "(process 37-degree orders only)\n" to
"arc/case.txt"

          print "Inoculation\t", V_utilization(4)*100 as .3"\t" to "arc/case.txt"
          print "\t",V_utilization(41)*100 as .3"\t", "(process 30-degree orders only)" to
"arc/case.txt"
          print "\t",V_utilization(42)*100 as .3"\t", "(process 34-degree orders only)"  to
"arc/case.txt"
          print "\t",V_utilization(43)*100 as .3"\t", "(process 37-degree orders only)\n" to
"arc/case.txt"

          print "Lysis\t", V_utilization(5)*100 as .3"\t" to "arc/case.txt"
          print "AEX\t", V_utilization(6)*100 as .3"\t" to "arc/case.txt"
          print "Average system utilization\t", V_AveU*100 as .3"\n"  to "arc/case.txt"

          send to P_decision
end

begin P_decision arriving
          if V_utilization(1)<=1 and V_utilization(2)<=1 and V_utilization(3)<=1
                    and V_utilization(4)<=1 and V_utilization(5)<=1 and V_utilization(6)<=1
then
          begin
                    print "Total number of orders scheduled:",OL_orders current loads"\t",
"Total number of orders remain unscheduled:", OL_remove current loads to "arc/case.txt"
                    print "Required usage of LB medium:", V_LBvol" L\t", "Required usage of
MY medium:", V_MYvol" L" to "arc/case.txt"
                    if A_split=1 and A_quant>0 then
```

```
        begin
            set A_split=0
            clone 1 load to P_temp
        end
        order all loads from OL_orders to P_store
        for each V_load in OL_remove load list do
        begin
            if V_load A_split=2 then set V_load A_split=0
            inc V_load A_leftover by 1
            inc V_load priority by 1
        end
        send to die
    end
    else
    begin
        if V_TotalNo>20 then send to P_transit(1)   /*selecting transit*/
        else send to P_transit(2)   /*splitting transit*/
    end
end

begin P_transit arriving
    if procindex=1 then
    begin
        call S_reset
        order 20 loads from OL_orders to P_select
        order all loads from OL_orders to OL_remove
    end
    else
    begin
        call S_reset
        set V_current=OL_orders current loads
        if V_lot=0 then order 1 load from OL_orders to P_split
        else order 1 load satisfying A_index=V_lot from OL_orders to P_split
    end

    print "=============================\n" to "arc/case.txt"
```

```
        print "New order list\n" to "arc/case.txt"
        print "lot index\t", "Order quantity(mg)\t", "No.starters\t", "Temperature(C)\t",
"Medium (type)\t", "Medium Vol.(L)\t", "Split?(0=non-split,1=split,2=leftover)\t",
"Leftover?(0=non-leftover,other=days)" to "arc/case.txt"

        send to die
end

begin P_select arriving
        call S_gather
        inc V_counter by 1
        if V_counter=20 then
        begin
                set V_counter=0
                clone 1 load to P_CapPrint
        end
        wait to be ordered on OL_orders
end

begin P_split arriving
        if A_split=0 then set A_quant_1=A_quant
        set A_split=1
        set V_lot=A_index   /*the lot to be split*/
        dec A_quant by 50 /*quantity after splitting*/
        inc A_quant_2 by 50 /*quantity split*/
        if A_quant<=0 then set A_split=2   /*remove the order from the list*/
        else call S_assign
        order all loads from OL_orders to P_recheck
        send to P_recheck
end

begin P_recheck arriving
        if A_split<>2 then call S_gather
        if A_index<>V_lot then wait to be ordered on OL_orders
        else if A_split=2 then
        begin
```

```
                set V_lot=0
                clone 1 load to P_CapPrint
                send to P_temp
        end
        else
        begin
                clone 1 load to P_CapPrint
                wait to be ordered on OL_orders
        end
end

begin P_temp arriving
        if A_split=2 then set A_quant to A_quant_1
        else set A_quant=A_quant_2
        wait to be ordered on OL_remove
end

/*manufacturing process starts from
here******************************************************/
begin P_rework arriving
        if A_quant<=50 then
        begin
                set A_medium="LB"
                set A_temp=oneof(2:30,5:34,93:37)
                set A_starter to oneof(66:2,28:3,6:4)
                if A_temp=30 then set A_paste=7.548   /*paste yield rate (g/L)*/
                else if A_temp=34 then
                begin
                        set A_paste=n 6.466, 2.139
                        if A_paste<3.27 then set A_paste=3.27
                end
                else
                begin
                        set A_paste= n 6.428, 1.848
                        if A_paste<1.167 then set A_paste=1.167
                end
        end
        else
        begin
                set A_medium="MY"
                set A_temp=oneof(1:30,44:34,55:37)
                set A_starter to oneof(40:3,60:4)
                set load type to L_large
                set A_medium="MY"
                set A_temp=oneof(1:30,44:34,55:37)
        if A_temp=30 then set A_paste=16.667  /*paste yield rate (g/L)*/
                else if A_temp=34 then
                begin
                        set A_paste=n 13.900, 5.275
                        if A_paste<4.75 then set A_paste=4.75
                end
                else
                begin
                        set A_paste=n 10.49,2.145
                        if A_paste<5.7 then set A_paste=5.7
                end
        end
        set V_flask(A_index)=0
        set V_jar(A_index)=0
        set V_tips(A_index)=0
        set V_bottle(A_index)=0

        set A_yield=A_paste*1000*(1-V_LossRate)  /*final yield rate*/
    set A_Mvol=A_quant/A_yield
    if A_Mvol<1 then set A_Mvol=1
        if A_YT=1 then set A_Mvol=1
        set A_flask=F_flask(A_Mvol)
        set A_jar=F_jar(A_Mvol)
        set A_falcon=F_falcon(A_paste)
        set A_tip=F_tip(A_paste)
        send to P_pretransformation
end
```

```
begin P_store arriving
/*      if A_first=1 then
        begin
                order all loads from OL_remove to P_pretransformation
                send to die
        end
        else send to P_pretransformation
*/
        if A_last=1 then order all loads from OL_FCFS to P_pretransformation   /*FCFS*/
        else wait to be ordered on OL_FCFS
        send to die

/*      if A_last=1 then order all loads from OL_FCLS to P_pretransformation   /*FCLS*/
        else wait to be ordered on OL_FCLS
        send to die*/

/*      if A_last=1 then order all loads from OL_SPT to P_pretransformation   /*SPT*/
        else wait to be ordered on OL_SPT
        send to die
        */
/*      if A_last=1 then order all loads from OL_LPT to P_pretransformation   /*LPT*/
        else wait to be ordered on OL_LPT
        send to die*/
end

begin P_pretransformation arriving
        move into Q_fridge
        get R_operator(1)
        move into Q_fridge_1  /*virtual queue with capacity of 20*/
        move into Q_hood
        set A_WT to V_day-A_day
        wait for 60 min    /*transfer samples to plates*/  /*single-capacity*/
        wait for 30 min              /*warm-up stay*/
        wait for 90 min
        free R_operator(1)   /*transfer samples from plates to tubes*/
```

```
   clone A_starter loads to P_transformation
   send to die
end

begin P_transformation arriving
         move into Q_dummy   /*store unattended samples*/
         set A_mindex=0
   get R_temp
   set V_mindex=1
   while V_mindex<=3 and A_mindex=0 do
   begin
         if Q_incubator(V_mindex) current loads=0 then
         begin
                set V_temp(V_mindex)=A_temp
                set A_mindex=V_mindex
         end
         else if Q_incubator(V_mindex) current loads=Q_incubator(V_mindex) capacity
then inc V_mindex by 1
         else
         begin
                if A_temp=V_temp(V_mindex) then set A_mindex=V_mindex
                else inc V_mindex by 1
         end
   end
         free R_temp
   if A_mindex=0 then
   begin
         wait for 1 min
         send to P_transformation
   end
   move into Q_incubator(A_mindex)
   wait until R_operator(1) active state=Off
         use R_incubator(A_mindex) for 14 hr   /*12-16hr*/
         send to P_culture
end
```

```
begin P_culture arriving
        move into Q_dummy
        get R_operator(2)
        move into Q_hood_2
        wait for 1.5 min  /*transfer plasmids from plates to tubes 1.5min/sample*/
        free R_operator(2)
        move into Q_dummy
        set A_mindex=0
  get R_temp
  set V_mindex=1
  while V_mindex<=3 and A_mindex=0 do
  begin
        if Q_shaker_sta(V_mindex) current loads=0 then
        begin
                set V_temp(V_mindex)=A_temp
                set A_mindex=V_mindex
        end
        else if Q_shaker_sta(V_mindex) current loads=Q_shaker_sta(V_mindex) capacity
then inc V_mindex by 1
        else
        begin
                if A_temp=V_temp(V_mindex) then set A_mindex=V_mindex
                else inc V_mindex by 1
        end
  end
  free R_temp
  if A_mindex=0 then
  begin
        wait for 10 min
        set V_delay=1
        send to P_culture
  end
  else set V_delay=0
        move into Q_shaker_sta(A_mindex)
        wait until R_operator(2) active state=Off
  use R_shaker_sta(A_mindex) for 9 hr  /*8-10 hours*/  /*shaking tubed plasmids*/
```

```
    send to P_screen
end

begin P_screen arriving
    move into Q_bench(1)
    use R_operator(3) for 1.5 min/*preparing screening minipreps*/
    move into Q_microcentrifuge
    use R_microcentrifuge for 30 min
     move into Q_bench(1)
     use R_operator(3) for 1 min   /*buffering*/
    move into Q_AGE
    use R_AGE for 30 min
    move into Q_bench(1)
    use R_operator(3) for 1.5 min  /*quality inspection*/
    set A_quality to oneof(95:"good",5:"bad")  /*quality of each starter*/
    send to P_quality
end

begin P_quality arriving  /*screening quality checking*/
    move into Q_dummy
    inc V_starter(A_index) by 1
    if A_quality="good" then
    begin
        inc V_good(A_index) by 1 /*if no less than one starter per order has passed
screening, other starters belonging to this order leave system*/
        if V_good(A_index)=1 then
        begin
            clone 1 loads to P_preinoculation
            wait to be ordered on OL_MCB
        end
    end
    else
    begin
        inc V_bad(A_index) by 1
        if V_bad(A_index)=A_starter then
        begin
```

```
                set A_rework=1
                clone 1 load to P_pretransformation /*if all the starters of one
sample has failed, it will grow again*/
            end
        end
        if V_starter(A_index)=A_starter then
        begin
            set V_good(A_index)=0
            set V_bad(A_index)=0
            set V_starter(A_index)=0
        end
        send to die
end

begin P_reassign arriving
        set A_YT=0
        inc A_quantMax by A_yield*A_Mvol
        set V_flask(A_index)=0
        set V_jar(A_index)=0
        set V_tips(A_index)=0
        set V_bottle(A_index)=0
        call S_assign
        clone 1 load to P_preinoculation
        wait to be ordered on OL_MCB
end


begin P_preinoculation arriving  /*insert starter into growth medium in flasks*/
        move into Q_bench(2)     /*infinite queue*/
        use R_operator(4) for 2.0*A_flask min
        move into Q_dummy
        clone A_flask loads to P_inoculation
        send to die
end

begin P_inoculation arriving
```

```
        set A_mindex to 0
        get R_ino
        if load type=L_large then call S_132
        else call S_72n8
        free R_ino
        if A_mindex=0 then
        begin
                wait for 1 min
                send to P_inoculation
        end
        move into Q_shaker_ino(A_mindex)
        wait until R_operator(4) active state=Off
        use R_shaker_ino(A_mindex) for 12 hr
        send to P_harvest
end

begin P_harvest arriving
        use R_operator(5) for 1 min /*loading*/
        set A_mindex=0
       choose a queue from among Q_centrifugeA(1), Q_centrifugeA(2),Q_centrifugeA(3)
whose remaining space is maximum
                save choice as A_queue
        set A_mindex to A_queue index
  move into Q_centrifugeA(A_mindex)
        wait until Q_centrifugeA(A_mindex) current loads=Q_centrifugeA(A_mindex)
capacity or R_operator(5) active state=Idling
  use R_centrifugeA(A_mindex) for 12 min
        use R_centrifugeA(A_mindex) for 8 min   /*cleaning cycle*/
        use R_operator(5) for 1 min /*unloading*/
        move into Q_freezer
        send to P_lysis
end

begin P_lysis arriving  /*processing L_order*/
        get R_operator(6)
        move into Q_bench(3)
        wait for 1 min   /*buffering P1*/
        free R_operator(6)
        move into Q_shaker_sus
        use R_shaker_sus for 15 min  /*resuspension*/
        move into Q_bench(3)
        wait for 5 min  /*buffering P2*/
        get R_operator(6)
        wait for 1 min   /*buffering P3*/
        free R_operator(6)
        inc V_flask(A_index) by 1
        if V_flask(A_index)=A_flask then clone A_jar loads to P_clarification
        send to die
end

begin P_clarification arriving
        move into Q_bench(4)
        set A_mindex to 0
       choose a queue from among Q_centrifugeA(1), Q_centrifugeA(2),Q_centrifugeA(3)
whose remaining space is maximum
               save choice as A_queue
        set A_mindex to A_queue index
        get R_operator(7) /*clarification*/
  move into Q_centrifugeA(A_mindex)
  use R_centrifugeA(A_mindex) for 12 min /*spinning cycle*/
        use R_centrifugeA(A_mindex) for 8 min  /*cleaning cycle*/
        free R_operator(7)
        inc V_jar(A_index) by 1
        if V_jar(A_index)=A_jar then clone A_tip loads to P_gravity
        send to die
end

begin P_gravity arriving
        move into Q_bench(4)
        use R_operator(8) for 30 min  /*gravity flow*/
        inc V_tips(A_index) by 1
        if V_tips(A_index)=A_tip then clone A_falcon loads to P_AEX
        send to die
end

begin P_AEX arriving
        move into Q_centrifugeB  /*precipitation*/
        use R_centrifugeB for 30 min
        move into Q_bench(4)
        get R_operator(9)
        wait for 10 min    /*washing*/
        free R_operator(9)
        move into Q_centrifugeC
        use R_centrifugeC for 20 min
        move into Q_bench(4)
        get R_operator(9)
        wait for 5 min  /*labor working cycle*/
        free R_operator(9)
        move into Q_hood_dry       /*drying*/
        use R_hood_dry for 30 min
        if A_quant>=40 then send to P_polishing
        else send to P_concentration
end

begin P_polishing arriving
        move into Q_bench(5)
        use R_operator(10) for 1 hr
        send to P_concentration
end

begin P_concentration arriving
        move into Q_bench(10)
        use R_operator(10) for 2 hr
        send to P_QC
end

begin P_QC arriving
        move into Q_bench(11)
```

```
    use R_operator(11) for 2 hr
    send to P_kill
end

begin P_kill arriving  /*to calculate the time in system for each load*/
    move into Q_dummy /*dummy queue makes the loads leave the processing
system*/
    inc V_bottle(A_index) by 1  /*the last L_flask replaces all the flask loads
comprising of the lot or part*/
    if V_bottle(A_index)=A_falcon then send to P_yield  /*if all the flask of this lot
have arrived*/
    else send to die
end

begin P_yield arriving
    if A_YT=0 then
    begin
        inc A_quantMax by A_yield*A_Mvol
        if A_quantMax<A_quant
        begin
            set V_YT=A_index
            inc A_rework by 1
            order 1 load satisfying A_index=V_YT from OL_MCB to
P_reassign
        end
        else send to P_time
    end
    else
    begin
        if A_yield>=3 then
        begin
            set V_YT=A_index
            order 1 load satisfying A_index=V_YT from OL_MCB to
P_reassign
        end
        else
        begin
            inc A_rework by 1
            send to P_rework
        end
    end
    send to die
end

begin P_time arriving
    if A_quant<A_quant_1 then
    begin
        set V_time=A_index
        order 1 load satisfying A_index=V_time from OL_time to continue in case
order not filled backorder on OL_time
        wait to be ordered on OL_time
        set A_quant=A_quant_1
    end
    inc V_throughput by A_quant
 set A_time to ac-A_time
    tabulate A_time/3600/24 in T_MLT  /*tabulate time in system in days*/
    print A_index"\t", A_quant"\t",A_time/3600/24"\t", A_WT to "arc/time.txt"
    send to die
end

/*subrountines***********************************************************/
begin S_72n8
    set V_machine=1
    while V_machine<=8 and A_mindex=0 do
    begin
        if Q_shaker_ino(V_machine) current loads=0 then
        begin
            set A_mindex=V_machine
            set V_m(V_machine)=A_temp
        end
        else if Q_shaker_ino(V_machine) current loads=Q_shaker_ino(V_machine)
capacity then inc V_machine by 1
```

```
                else
                begin
                        if A_temp=V_m(V_machine) then set A_mindex=V_machine
                        else inc V_machine by 1
                end
        end
end

begin S_132
        if Q_shaker_ino(8) current loads=0 then
        begin
                set A_mindex=8
                set V_m(8)=A_temp
        end
        else if Q_shaker_ino(8) current loads=Q_shaker_ino(8) capacity then call S_72n8
        else
        begin
                if A_temp=V_m(8) then set A_mindex=8
                else call S_72n8
        end
end

begin S_assign
        /*determine number of starters,medium type, temperature, paste weight, yield rate
and medium volume for each order*/
        if A_quant<=50 then set A_starter to oneof(66:2,28:3,6:4)
        else set A_starter to oneof(40:3,60:4)
        set A_yield=A_paste*1000*(1-V_LossRate)   /*final yield rate*/
        set A_Mvol=A_quant/A_yield
        if A_Mvol<1 then set A_Mvol=1
        if A_YT=1 then set A_Mvol=1
        set A_flask=F_flask(A_Mvol)
        set A_jar=F_jar(A_Mvol)
        set A_falcon=F_falcon(A_paste)
        set A_tip=F_tip(A_paste)
end
```

```
begin S_gather
        inc V_total by A_quant
        inc V_TotalNo by 1
        if A_medium="LB" then inc V_LBvol by A_Mvol
        else inc V_MYvol by A_Mvol
        if A_tip=1000 then inc V_RunTime by A_Mvol*1000/200*30
        else if A_tip=2000 then inc V_RunTime by A_Mvol*1000/944*30
        else if A_tip=3000 then inc V_RunTime by A_Mvol*1000/(944+200)*30
        else inc V_tip by A_tip
        if A_temp=30 then
        begin
                inc V_30 by 1
                inc V_usage(21) by A_starter
                set V_usage(31)=V_usage(21)
                inc V_usage(41) by A_flask
        end
        else if A_temp=34 then
        begin
                inc V_34 by 1
                inc V_usage(22) by A_starter
                set V_usage(32)=V_usage(22)
                inc V_usage(42) by A_flask
        end
        else
        begin
                inc V_37 by 1
                inc V_usage(23) by A_starter
                set V_usage(33)=V_usage(23)
                inc V_usage(43) by A_flask
        end
        print A_index"\t",A_quant
as .2"\t",A_starter"\t",A_temp"\t",A_medium"\t",A_Mvol"\t",A_split"\t",
A_leftover"\t",A_tip to "arc/case.txt"
        inc V_usage(1) by 1        /*transformation prep*/
        inc V_usage(2) by A_starter    /*incubation*/
        set V_usage(3) to V_usage(2)  /*starter culture */
        inc V_usage(4) by A_flask      /*inoculation*/
        inc V_usage(5) by A_jar     /*lysis*/
        set V_usage(6) to V_tip     /*AEX*/ /*gravity flow*/
        inc V_usage(61) by A_falcon  /*AEX*/ /*precipitation*/
end

begin S_capacity
        if A_medium="LB" then inc V_LBvol by A_Mvol
        else inc V_MYvol by A_Mvol
        set V_utilization(1)=V_usage(1)/20 /*capacity of stage 1*/
        set V_utilization(2)=V_usage(2)/(3*R_incubator(1) capacity)/*capacity of stage 2*/
        set V_utilization(21)=V_usage(21)/(3*R_incubator(1) capacity)
        set V_utilization(22)=V_usage(22)/(3*R_incubator(1) capacity)
        set V_utilization(23)=V_usage(23)/(3*R_incubator(1) capacity)
        set V_utilization(3)=V_usage(3)/(3*R_shaker_sta(1) capacity) /*capacity of stage
3*/
        set V_utilization(31)=V_usage(31)/(3*R_shaker_sta(1) capacity)
        set V_utilization(32)=V_usage(32)/(3*R_shaker_sta(1) capacity)
        set V_utilization(33)=V_usage(33)/(3*R_shaker_sta(1) capacity)    /*capacity of
stage 4*/

        set V_utilization(4)=V_usage(4)/(R_shaker_ino(1) capacity+R_shaker_ino(2)
capacity+R_shaker_ino(3) capacity+R_shaker_ino(4) capacity+R_shaker_ino(5)
capacity+R_shaker_ino(6) capacity+R_shaker_ino(7) capacity+R_shaker_ino(8) capacity)
        set V_utilization(41)=V_usage(41)/(R_shaker_ino(1) capacity+R_shaker_ino(2)
capacity+R_shaker_ino(3) capacity+R_shaker_ino(4) capacity+R_shaker_ino(5)
capacity+R_shaker_ino(6) capacity+R_shaker_ino(7) capacity+R_shaker_ino(8) capacity)
        set V_utilization(42)=V_usage(42)/(R_shaker_ino(1) capacity+R_shaker_ino(2)
capacity+R_shaker_ino(3) capacity+R_shaker_ino(4) capacity+R_shaker_ino(5)
capacity+R_shaker_ino(6) capacity+R_shaker_ino(7) capacity+R_shaker_ino(8) capacity)
        set V_utilization(43)=V_usage(43)/(R_shaker_ino(1) capacity+R_shaker_ino(2)
capacity+R_shaker_ino(3) capacity+R_shaker_ino(4) capacity+R_shaker_ino(5)
capacity+R_shaker_ino(6) capacity+R_shaker_ino(7) capacity+R_shaker_ino(8) capacity)
        set V_utilization(5)=(V_usage(5)/8.0*(1+5+1)+V_usage(5)/12.0*15)/(4.5*60)
/*capacity of stage 5*/
```

```
        if V_RunTime>V_usage(6)/(24*3)*30 then
                set V_utilization(6) = (V_usage(5)/36.0*(12+8) +V_RunTime+
V_usage(61)/6.0*(30) +V_usage(61)/16.0*(20)+30)/(9.5*60) /*capacity of stage 6*/
        else
                set V_utilization(6) = (V_usage(5)/36.0*(12+8)+ V_usage(6)/(24*3)*30+
V_usage(61)/6.0*(30)+V_usage(61)/16.0*(20)+30)/(9.5*60) /*capacity of stage 6*/
        set V_AveU =(V_utilization(1)+ V_utilization(2)+ V_utilization(3)+
V_utilization(4)+V_utilization(5)+V_utilization(6))/6
end

begin S_reset
        set V_30=0
        set V_34=0
        set V_37=0
        set V_LBvol=0
        set V_MYvol=0
        set V_TotalNo=0
        set V_total=0
        set V_tip=0
        set V_RunTime=0
        set V_usage(1)=0
        set V_usage(2)=0
        set V_usage(3)=0
        set V_usage(4)=0
        set V_usage(5)=0
        set V_usage(6)=0
        set V_usage(61)=0
        set V_usage(21)=0
        set V_usage(22)=0
        set V_usage(23)=0
        set V_usage(31)=0
        set V_usage(32)=0
        set V_usage(33)=0
        set V_usage(42)=0
        set V_usage(41)=0
        set V_usage(43)=0
```

```
end

/*functions******************************************************************/
begin F_flask function          /*calculates the number of flasks needed for each order in
inoculation*/ /*integer function*/
        if vol/0.5>1 then return (vol/1) /*"1" is equivalent to 1L (the capacity of each
inoculation flask*/
        else return (1)
end

begin F_jar function /*computes the number of beckman jars needed in harvesting*/
        if vol<10 then return (vol/1)
        else return (vol/2)
end

begin F_falcon function   /*number of falcon tubes*/
        if paste*15*0.5/225.0<1 then return (1)
        else return (paste*15*0.5/225.0)
end

begin F_tip function /*computes the number of tips needed in AEX*/
        if paste*15*3<200 then return 1 /*1*8cc tip is used*/
        else if paste*15*3>=200 and paste*15*3<400 then return 1  /*1*16cc tip is used*/
        else if paste*15*3>=400 and paste*15*3<500 then return 2  /*2*16cc tip is used*/
        else if paste*15*3>=500 and paste*15*3<600 then return 3  /*3*16cc tip is used*/
        else if paste*15*3>=600 and paste*15*3<900 then return 4  /*4*16cc tip is used*/
        else if paste*15*3>=900 and paste*15*3<1500 then return 1000 /*White DMAE is
used*/
        else if paste*15*3>=1500 and paste*15*3<10000 then return 2000 /*Green DMAE
is used*/
        else return 3000 /*Green+White DMAE is used*/
end

/*operator-break-control processes start from here**********************/
begin P_break arriving
        if procindex=1 then
```

```
begin
        set V_break(1)=6
        set V_duration(1)=3
        set V_off(1)=15
end
else if procindex=2 then
begin
        set V_break(2)=0
        set V_duration(2)=1
        set V_off(2)=23
end
else if procindex=3 then
begin
        set V_break(3)=7
        set V_duration(3)=3
        set V_off(3)=14
end
else if procindex=4 then
begin
        set V_break(4)=9
        set V_duration(4)=3
        set V_off(4)=12
end
else if procindex=5 then
begin
        set V_break(5)=1
        set V_duration(5)=5.5
        set V_off(5)=17.5
end
else if procindex=6 then
begin
        set V_break(6)=2
        set V_duration(6)=4.5
        set V_off(6)=17.5
end
else if procindex=7 then
```

```
begin
          set V_break(7)=6
          set V_duration(7)=9
          set V_off(7)=9
end
else if procindex=8 then
begin
          set V_break(8)=6
          set V_duration(8)=9
          set V_off(8)=9
end
else if procindex=9 then
begin
          set V_break(9)=6
          set V_duration(9)=9
          set V_off(9)=9
end
else if procindex=10 then
begin
          set V_break(10)=2
          set V_duration(10)=9
          set V_off(10)=13
end
else if procindex=11 then
begin
          set V_break(11)=2
          set V_duration(11)=9
          set V_off(11)=13
end
wait for V_break(procindex) hr
while 1=1 do
begin
          bring up R_operator(procindex)
          set R_operator(procindex) active state=Working
          wait for V_duration(procindex) hr
          if procindex=2 then wait until Q_hood_2 current loads=0  or V_delay=1
```

```
        if procindex=4 then wait until R_operator(3) active state=Off and Q_bench(2) current loads=0
        if procindex=3 then wait until Q_microcentrifuge current loads=0 and Q_AGE current loads=0 and R_operator(procindex) active state=Idling
        if procindex=5 then wait until Q_centrifugeA(1)current loads=0 and Q_centrifugeA(2)current loads=0 and Q_centrifugeA(3)current loads=0
        take down R_operator(procindex)
        set R_operator(procindex) active state=Off
        wait for V_off(procindex)+V_break(procindex) hr
    end
    send to die
end
```

APPENDIX B. EXPERT SYSTEM CODE

```
;;;==================================================
;;;  DNA Production Expert System
;;;
;;;  This expert system helps make decision of DNA production
;;;
;;;   CLIPS Version 6.3
;;;
;;;   To execute, please load, reset and run.
;;;==================================================

;;****************
;;* DEFFUNCTIONS *
;;****************

(deffunction ask-question (?question)    ;A function defines the ask-and-answer
  (printout t ?question)
  (bind ?answer (read))
  (if (integerp ?answer)
        then TRUE)
  (while (not (integerp ?answer)) do
      (printout t "Invalid Answer. Plsease answer with a numeric number." crlf)
        (printout t ?question)
        (bind ?answer (read))
        (if (integerp ?answer)
           then TRUE))
?answer)

(deffunction yes-no (?question $?allowed-values)    ;A function defines the answers of yes-
or-no-p as yes, or y, or n, or no only
  (printout t ?question)
  (bind ?input (read))
  (if (lexemep ?input)
     then (bind ?input (lowcase ?input)))
  (while (not (member ?input ?allowed-values)) do
    (printout t "Invalid Answer. Please answer \"yes\" or \"no\"." crlf)
    (printout t ?question)
    (bind ?input (read))
    (if (lexemep ?input)
       then (bind ?input (lowcase ?input))))
?input)

(deffunction yes-or-no-p (?question)    ;A function defines the response of yes-or-no-p
  (bind ?response (yes-no ?question yes no y n))
  (if (or (eq ?response yes) (eq ?response y))
     then TRUE
     else FALSE))

;;;***************
;;;* QUERY RULES *
;;;***************

(defglobal ?*leftovers* = 100000)  ;Defines global variable ?*leftover*
 (defglobal ?*new-orders* = 100000)  ;Defines global variable ?*new-orders*
 (defglobal ?*total-orders* = 100000) ;Defines global variable ?*total-orders*
 (defrule determine-leftovers""     ;Determines if there is leftover and the total amount of
leftovers
(not (leftover-is ?))
  =>
  (if (yes-or-no-p "Any leftover orders(yes/no)?")
     then
        (bind ?response(ask-question "What is the total number of leftovers?"))
        (bind ?*leftovers* ?response)
        (assert (leftover-is present))
        else (bind ?*leftovers* 0)
            (assert(leftover-is zero))))


(defrule determine-new-order""    ;Determines if there is new order and the total amount of
new orders
(not (new-order-is ?))
```

```
  =>
  (if (yes-or-no-p "Any new orders(yes/no)?")
    then
       (bind ?response(ask-question "How many new orders recieved?"))
       (bind ?*new-orders* ?response)
       (assert (new-order-is present))
       else (bind ?*new-orders* 0)
          (assert(new-order-is zero))))

(defrule determine-total-amount   ;Determines the total number of orders to be
scheduled  ;Determine the total number of orders to be scheduled
(leftover-is ?)
(new-order-is ?)
(not (total-number-is ?))
=>
   (bind ?*total-orders* (+ ?*new-orders* ?*leftovers*))
   (if (>= ?*total-orders* 20)
       then (assert (total-number-is large))
       else (if (and (> ?*total-orders* 5) (< ?*total-orders* 20))
              then (assert (total-number-is moderate))
              else (assert (total-number-is small)))))

(defrule determine-quantity-composition  ;Determines the quantity composition of total
orders
(leftover-is ?)
(new-order-is ?)
(not (total-quantity-is ?))
=>
  (if (yes-or-no-p "Total quantity over 400mg(yes/no)?")
       then (bind ?answer(ask-question "How many percentage of orders are greater than
or equal to 100mg?"))
          (if (> ?answer 80)
               then (assert (total-quantity-is large))
               else (assert (total-quantity-is moderate)))
       else (assert (total-quantity-is small))))
```

```
(defrule determine-temperature-composition ;Determines the temperature composition of
total oders
(leftover-is ?)
(new-order-is ?)
(not (temperature-group-is ?))
=>
  (bind ?answer(ask-question "How many temperature groups are there among orders?"))
  (assert (temperature-group-is ?answer)))


;;;**********************
;;;* PRODUCTION PLANNING RULES*
;;;**********************

(defrule produce-all-conclusions ""  ;Rule 1
  (declare (salience 10))
  (temperature-group-is ?)
  (and (total-number-is moderate)
       (total-quantity-is small))
  (not (select ?))
  =>
  (assert (select "Select all orders to process.")))

(defrule not-to-produce-conclusions ""  ;Rule 2
  (declare (salience 10))
  (temperature-group-is ?)
  (or (and (total-number-is small)(total-quantity-is moderate))
     (and (total-number-is small)(total-quantity-is small)))
  (not (select ?))
  =>
  (assert (select "Not to produce. Wait until next day.")))

(defrule selection-conclusion ""  ;Rule 3
  (declare (salience 10))
  (temperature-group-is ?)
  (or (and (total-number-is moderate)(total-quantity-is moderate))
    (and (total-number-is large)(total-quantity-is small)))
  (not (select ?))
=>
  (assert (select "Select orders based on temperatuer requirement.")))

(defrule selection-and-split-conclusion ""   ;Rule 4
  (declare (salience 10))
  (temperature-group-is ?)
  (and (total-number-is large)
        (or (total-quantity-is moderate)(total-quantity-is large)))
  (not (select ?))
=>
  (assert (select "Select orders based on temperatuer requirement and split the largest order
if necessary.")))

(defrule split-conclusion ""   ;Rule 5
  (declare (salience 10))
  (temperature-group-is ?)
  (and (or (total-number-is moderate) (total-number-is small))
        (total-quantity-is large))
  (not (select ?))
=>
  (assert (select "Split the largest order.")))


;;;****************************
;;;* STARTUP AND REPAIR RULES *
;;;****************************

(defrule system-banner ""
 (declare (salience 10))
 =>
 (printout t crlf crlf)
 (printout t "The DNA Production Expert System")
 (printout t crlf crlf))
```

```
(defrule print-production-plan ""
 (declare (salience 10))
 (select ?item)
 =>
 (printout t crlf crlf)
 (printout t "Suggested Production Plan:")
 (printout t crlf crlf)
 (format t " %s%n%n%n" ?item))
```

APPENDIX C. CAPACITY AND PROCESSING OF EACH STAGE

Process		Capacity (No. of units/time)	Equipment	Processing time/each order		Labor	Process type	Unit
				Equipment	Labor			
Transformation	Preparation	20	Hood	-	1 hr/20 units(3 min/unit)	A	Serial	sample
	Incubation	52*3 (52 each)	Incubator	12-16 hr	-	-	Batch	starter/plate
Starter culture	Preparation	1	Hood	-	1-2 min/unit	B	Serial	starter/tube
	Growth	24*3 (24 each)	Shaker	8-10 hr	-	-	Batch	starter/tube
Screening	Preparation	1	Bench(1)	-	1-2 min/unit	C	Serial	starter/tip
	Centrifuging	24*2 (24 each)	Micro-centrifuge	60 min/(24*2) units		-	Batch	starter/tip
	AGE	20*2 (20 each)	AGE	60 min/(20*2) units	1-2 min/unit	-	Serial+ Batch	starter/tip
Inoculation	Preparation	1	Bench(2)	-	1.8-2 min/unit	D	Serial	flask
	Fermentation	Variable(8,12,132)	Shakers	14-24 hr	-	-	Batch	flask
Harvesting	Centrifuging	12*3 (12 each)	Centrifuge-A	12 min/12 units	-	-	Batch	bottle
	Cleaning	12*3 (12 each)	Centrifuge-A	15 min/12 units	Included	E	Batch	-
Lysis	Buffer P1	4*2 (4 each)	Bench(3)	-	2 min/(4 *2units) (0.25 min/unit)	F*2**	Serial	bottle
	Re-suspension	15	Shaker	15 min/12 units	-	-	Batch	bottle
	Buffer P2	4*2 (4 each)	Bench(3)	-	2 min/(4*2 units) (0.25 min/unit)	F*2	Serial	bottle
	Stay	4*2 (4 each)	Bench(3)	5 min/(4 units*2)			Batch	bottle
	Buffer P3	4*2 (4 each)	Bench(3)	-	2 min/(4*2 units) (0.25 min/unit)	F*2	Serial	bottle
AEX	Clarification	12*3 (12 each)	Centrifuge-A	12 min/12 units	-	-	Batch	bottle
	Cleaning	12*3 (12 each)	Centrifuge-A	-	15 min/(12*3) units	G*3	Batch	bottle
	Gravity flow	24*3	Bench (4)	-	30 min/(24*3) units	G*3	Serial	tip
	Precipitation	6	Centrifuge-B	30 min/6 units		-	Batch	tube
	Cleaning	6	Centrifuge-A	-	5 min/6 units	G*3	Batch	tube
	Preparation	16	Centrifuge-C	-	10 min /unit	G*3	Serial	tube
	Washing	16	Centrifuge-C	20 min/16 units	-	-	Batch	tube
	Preparation	1	Bench(4)	-	5 min /unit	G*3	Serial	tube
	Drying	20	Hood_dry	30 min /20 units	-	G*3	Batch	tube
Polishing		N/A	HIC columns	1-2 hr	20 min	H	Serial	columns
Concentration		N/A	Diafiltration	2 hr	-	H	Serial	diafiltration
QC		N/A	-	-	2 hr	I	Serial	miscellaneous

** The stage has two operators in real life. For simplicity, their capacity is doubled to make a single operator.

APPENDIX D. EXPERIMENT RESULTS FOR GENERATION OF RULES

Run	Transformation	Incubation	Starter Culture	Inoculation	Lysis	AEX	Average system utilization	Decision
1	50.00	16.03	34.72	9.43	15.74	23.17	24.85	Not to process
1	25.00	12.82	27.78	7.55	12.59	16.30	17.01	Not to process
1	75.00	26.92	58.33	14.15	23.61	31.80	38.30	Process all
1	100.00	42.31	91.67	22.64	37.78	43.57	56.33	Process all
A	-	-	-	-	-	-	-	N/A
B	50.00	19.87	43.06	13.68	22.82	25.95	29.23	Process all
B	75.00	27.56	59.72	23.59	30.69	34.94	41.92	Process all
B	25.00	12.82	27.78	20.76	23.61	20.40	21.73	Not to process
B	90.00	39.74	86.11	41.04	59.03	49.12	60.84	Process all
AB	25.00	12.82	27.78	156.13	132.22	60.75	69.12	Split (SSA)
AB	75.00	35.90	77.78	204.72	177.08	89.91	110.07	Split (SSA)
AB	100.00	42.31	91.67	102.83	88.15	62.65	81.27	Split (SSA)
AB	50.00	20.51	44.44	109.91	92.87	52.05	61.63	Split (SSA)
BC	50.00	23.08	50.00	27.83	41.71	33.19	37.64	Process all
BC	75.00	36.54	79.17	44.81	60.60	46.13	57.04	Process all
BC	25.00	12.82	27.78	21.23	22.82	19.96	21.60	Not to process
BC	100.00	51.28	99.34	64.62	90.51	64.11	80.27	Process all
C	80.00	21.80	47.22	8.96	14.95	29.61	33.76	Process all
C	25.00	10.90	23.61	9.43	15.74	17.11	16.97	Not to process
C	100.00	48.08	98.03	26.89	44.86	47.22	61.87	Process all
C	50.00	24.36	52.78	14.15	23.61	26.83	31.95	Process all
AC	-	-	-	-	-	-	-	N/A
ABC	25.00	7.69	16.67	92.45	78.70	40.42	43.49	Process all

ABC	75.00	25.64	55.56	204.25	177.87	89.25	104.59	Split (SSA)
ABC	25.00	11.54	25.00	136.32	114.91	54.02	61.13	Split (SSA)
ABC	50.00	21.80	47.22	144.34	121.20	62.28	74.47	Split (SSA)
BD	125.00	54.49	118.06	93.40	88.94	69.23	91.52	Apply SSA
BD	275.00	117.31	254.17	75.47	125.93	120.10	161.33	Apply SSA
BD	175.00	82.05	177.78	85.38	107.04	88.96	119.37	Apply SSA
BD	200.00	93.59	202.78	87.26	122.78	101.02	134.57	Apply SSA
D	200.00	95.51	206.94	53.77	89.72	88.60	122.43	Apply SSA
D	250.00	76.92	166.67	35.38	59.03	86.33	112.39	Apply SSA
D	225.00	90.39	195.83	43.40	72.41	86.48	118.92	Apply SSA
D	250.00	119.23	258.33	84.43	122.78	113.82	158.10	Apply SSA
AD	250.00	128.21	277.78	116.04	167.64	130.99	178.44	Apply SSA
AD	200.00	102.56	222.22	93.87	136.16	106.51	143.55	Apply SSA
AD	175.00	89.74	194.44	91.51	126.71	96.71	129.02	Apply SSA
AD	175.00	89.74	194.44	97.17	132.22	98.76	131.22	Apply SSA
ABD	200.00	93.59	202.78	254.25	245.56	146.64	190.47	Apply SSA
ABD	125.00	58.97	127.78	366.04	317.18	154.53	191.58	Apply SSA
ABD	175.00	82.05	177.78	522.64	453.33	217.62	271.40	Apply SSA
ABD	200.00	82.69	179.17	356.60	314.03	170.91	217.23	Apply SSA
CD	110.00	56.41	122.22	38.68	64.54	74.42	77.71	Apply SSA
CD	125.00	64.10	138.89	45.76	76.34	82.09	88.70	Apply SSA
CD	200.00	102.56	222.22	71.70	119.63	114.62	138.46	Apply SSA
CD	150.00	76.92	166.67	55.66	92.87	93.71	105.97	Apply SSA
BCD	125.00	54.49	118.06	50.00	73.98	102.27	87.30	Apply SSA
BCD	175.00	89.74	194.44	77.83	129.86	102.41	128.22	Apply SSA
BCD	250.00	109.62	237.50	70.28	117.27	124.71	151.56	Apply SSA
BCD	110.00	50.64	109.72	87.26	101.53	70.69	88.31	Apply SSA
ACD	250.00	128.21	277.78	87.74	146.39	135.53	170.94	Apply SSA

ACD	200.00	102.56	222.22	105.19	137.73	110.82	146.42	Apply SSA
ACD	150.00	76.92	166.67	113.21	132.22	97.81	122.80	Apply SSA
ACD	300.00	153.85	333.33	103.77	173.15	156.43	203.42	Apply SSA
ABCD	110.00	50.64	109.72	206.60	191.25	104.02	128.71	Apply SSA
ABCD	125.00	55.77	120.83	172.64	162.13	96.86	122.21	Apply SSA
ABCD	250.00	114.10	247.22	221.23	222.73	150.88	201.03	Apply SSA
ABCD	200.00	90.39	195.83	150.00	176.30	121.35	155.64	Apply SSA