Modeling Product Variety Induced Manufacturing Complexity for Assembly System Design

by

Xiaowei Zhu

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Mechanical Engineering)
in The University of Michigan
2009

Dedicated to my family who have loved and supported me

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my committee members, my friends, my family, and many others for their guidance, help, and support. Without them, I would never have been able to finish this dissertation.

First and foremost, I would like to thank my co-advisors, Professor S. Jack Hu and Professor Yoram Koren, for their great vision, vast knowledge, and continual encouragement. Professor Hu inspired me to explore on my own, and at the same time gave me the invaluable insights and strength to continue when my progress stops. Word by word, line by line, he taught me how to speak and write concisely and precisely. Professor Koren emphasizes on intuition and understanding the big picture, because he believes, this is the most valuable piece that graduate students should possess and could use for their lifetime.

I would also like to extend my thanks to my other committee members, Professor Amy M. Cohn and Dr. Ningjian Huang, for their constructive suggestions and valuable discussions. Professor Cohn is one of the best experts in Operations Research. I appreciate her inputs on how to efficiently solve the optimization problem. Dr. Huang not only serves on my committee, but was my mentor at General Motors as well. His industrial experience and expertise proved a integral part to this research.

I wish I could also thank my former committee member, Dr. Samuel P. Marin, who passed away recently. Dr. Marin was a mathematician and the former group manager at General Motors. He was the first who recognized the value of this research and bridged the theory to the real-world industrial practices. Vividly, I still remember his smiling face when he told me, "I think you may be able to apply this to our general assembly lines!".

I gratefully acknowledge the financial support from the NSF Engineering Research Center for Reconfigurable Manufacturing Systems (ERC/RMS) under Award Number EEC-9529125 and the General Motors Collaborative Research Laboratory (GMCRL) in Advanced Vehicle Manufacturing, both at The University of Michigan, as well as support from the NSF CMMI under grant 0825438.

My gratitude also goes to my friends and colleagues in the GMCRL, ERC, and at General Motors, with whom I had the privilege to work and spend time together. I want to thank them all for their help and support. The friendship made my life easier, happier, and warmer during the long-lasting Michigan winters.

I want to thank the ME, IOE, and Statistics departments' faculty, staff, and students.

I learned a lot during my time at Michigan both in and out of the classrooms.

Last but certainly not least, my greatest gratitude goes to my family for their love and unconditional support. My parents, Aiguan Meng and Chengming Zhu, are always proud of me and believe in me. From them, I received the most encouragement and strength in the hard times. My parents-in-law, Yongming Liao and Yinshan Xu, my brother-in-law, Bing Xu, understand our life in another country would never be easy. They tried their best to support us and let us feel confident. To my beloved wife, Hui Xu, I would like to say, “You bring me the joy along the journey and into the future. With you, I would never feel I am lonely.”. To my one-year-old daughter, Jiayi, “You are a gift to me and let me gain the momentum to never stop moving forward!”.

To all these people I will always be in debt. Thank you all!

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Modeling Product Variety Induced Manufacturing
Complexity for Assembly System Design

by
Xiaowei Zhu

Co-Chairs: Shixin Jack Hu and Yoram Koren

Mixed-model assembly systems have been recognized as a major enabler to handle product variety. However, the assembly process can become *quite* complex as the variety increases. The complexity may impact the system performance in terms of quality and productivity. This dissertation considers the variety induced manufacturing complexity in manual, mixed-model assembly lines, develops models for the propagation of complexity, and applies the models to assembly system design. A complexity measure called “Operator Choice Complexity” (OCC) is proposed to quantify human performance in making choices. The OCC is an entropy measure of the average randomness in a choice process. Meanwhile, empirical evidences support the proposed complexity measure. Based on the OCC, models are developed to evaluate the complexity at each station, and for the entire assembly system. The system level model, termed “Stream of Complexity” (SoC) model, addresses complexity propagation in multi-stage systems. By applying the SoC model, complexity can be mitigated with system design and operation decisions. The models are then applied to assembly sequence planning and build sequence scheduling.

Assembly sequence planning is to determine the order of assembly tasks. According to the SoC model, assembly sequence determines the directions of complexity flows. Thus proper sequence planning can reduce complexity. However, due to the difficulty of handling the directions of complexity flows in optimization, a transformed network flow model is formulated and solved based on dynamic programming.

Build sequence scheduling is to determine the order of products being built. It also determines the sequential dependencies between the choices. According to the complexity model, the knowledge of the sequential dependencies can help operators make choices, and scheduling proper build sequences can reduce complexity. However, deterministic models are *not* sufficient to study such sequence scheduling problems. A probabilistic model based on hidden Markov chains is proposed to formulate the scheduling problem with constraints.

Analytical solutions are obtained, and suggest that proportional production attains the maximum complexity while batch production attains the minimum.

The results of the research are highly applicable to all manufacturers who are interested in economically offering product variety without loss of quality and productivity.

CHAPTER 1

Introduction

1.1 Motivation

Mass customization has been recognized as a new paradigm for today's manufacturing [40]. Different industries are practicing mass customization to gain competitive advantages. In order to satisfy the finely targeted niche markets, the variety of products offered in these industries has increased dramatically over the last several decades. For example, in the automotive industry, BMW claims [1] that "Every vehicle that rolls off the belt is unique" and the number of possible automobile variations in the BMW 7 Series alone could reach 10^{17}. In the US market, the major US auto makers are competing with their Japanese rivals not only on closing the "quality gaps", but also the "variety gaps" [33]. Although statistics shows that US auto makers have far more product variety in terms of possible build combinations of options, gaps exist in how to efficiently and cost-effectively provide the variety.

Mixed-model assembly systems have been recognized as a major enabler to handle the increased variety. Such a system typically takes the form of a flow line, thus it is also called a mixed-model assembly line (MMAL). Fig. 1.1 illustrates an MMAL for automobiles. The line is capable of not only handling different vehicle models, but also a large number of customized options. The advantage of such a system is obvious: it saves equipment investment and absorbs demand fluctuations across the vehicle models.

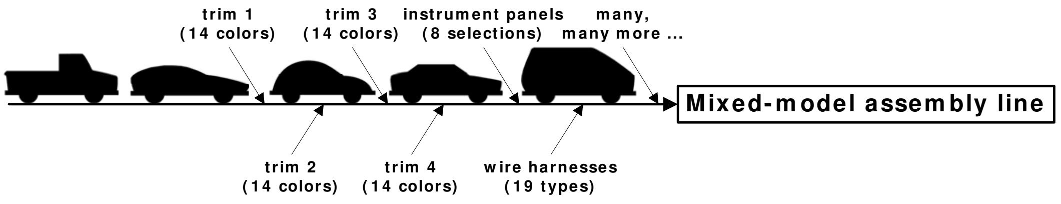

Figure 1.1: A Mixed-Model Assembly Line for Automobiles

[1] http://www.bmwgroup.com/e/nav/index.html.

One of the challenges in using the mixed-model assembly system to achieve mass customization is associated with the increased complexity induced by product variety. As variety increases, the manufacturing process can become quite complex. The complexity exists almost everywhere, from planning, part supply, to assembling. One of the causes for the complexity lies in the challenges of coordinating numerous small production steps as a large number of different parts are used for products with different options. It has been shown that such complexity has a significant negative impact on manufacturing performances, such as productivity and quality [15, 33, 14]. The obvious question is how to organize production in a way to allow the company to absorb high levels of complexity without sacrificing productivity and quality.

Although people realize the existence of complexity, the understanding of complexity is limited. From the dictionary [2], complexity is defined as "the state of having many different parts connected or related to each other in a complicated way". In people's mind, complexity is a subjective manner. For example, when one refers to something as being complex, he/she could not quantify the exact level of complexity. Oftentimes, he/she could state that something has more complexity than others, but may *not* be able to tell how much the complexities are. Therefore, the assessment of complexity is *incomplete*. The incompleteness is partly due to a lack of proper complexity measures.

For engineering systems design, a complete and accurate quantification of complexity is necessary. Thus, a measure of complexity has to be defined. Attempts have been made to define complexity measures. Some definitions deal with the complexity of a process, such as the computational effort required to solve a problem [7], the length of a shortest binary computer program to describe a random variable drawn according to a probability mass function (which is widely known as Kolmogorov complexity or algorithmic complexity). Complexity has also been defined to measure the bits of information it takes to describe a message in communication systems [45], which has led to an important concept – information entropy. Information entropy plays a strong role in information theory and other disciplines as measures of information, choice, uncertainty, and complexity.

For mixed-model assembly systems, the complexity measure should reflect the underlining "physics" of the assembly process, and answer the basic question about the mechanism through which variety causes complexity and impacts performance. Furthermore, the mixed-model system has multiple stages. The complexity will propagate across the stages, which makes the quantification of complexity more difficult. Therefore, models need to be developed to properly address the complexity propagation.

Some limited research work has been done in the literature on manufacturing system complexity. Such work includes manufacturing systems configuration complexity as well as variety induced complexity. However, the existing research does not provide sufficient

[2]Collins COBUILD Dictionaries and Reference Books, COBUILD is an acronym for Collins Birmingham University International Language Database.

understanding on the mechanism through which variety causes complexity and impacts performance for mixed-model assembly systems. For example, Deshmukh et al. [9, 10] defined static complexity of manufacturing systems due to part mix. But the research was focused on the station level or job shop setting, thus is not applicable for flow lines as in the mixed-model systems. Fujimoto et al. [17] proposed an information entropy based measure of complexity for assembly planning. However, the complexity measure is based on product configuration and does not incorporate the manufacturing system characteristics into the analysis. More recently, ElMaraghy et al. [13] applied entropy function to quantify the complexity of manufacturing systems and their configurations with examples in machining processes. However, their approach is not applicable for mixed-model assembly systems.

Hence, this dissertation is intended to develop proper understanding of product variety induced manufacturing complexity using new measures and models of complexity.

1.2 Research Objective and Tasks

The objective of the research is to develop new measures of complexity and mathematical models for the propagation of complexity in multi-stage, mixed-model assembly systems and to apply the developed models to assembly system design. Specifically, the research shall include the following tasks.

Task 1: To define measures of complexity reflecting the underlining "physics" of the assembly process in the mixed-model system. The measure should answer the question about the mechanism through which variety causes complexity and impacts performance.

Task 2: To develop models for understanding the mechanism of complexity propagation in multi-stage assembly systems, i.e., "Stream of Complexity" models. The models should be developed at both the stations and systems levels, integrating both product variety and manufacturing process information. Once developed, they could be applied for assembly system design and operations.

Task 3: To apply the Stream of Complexity models for system design and operations. When needed, specific models and algorithms should be developed to search for optimal system designs to minimize complexity.

1.3 Organization of the Dissertation

The dissertation is presented in a multiple manuscript format. Part of the work in Chapters 2, 3, and 4 has appeared as individual research papers. The organization of the dissertation is as follows.

Chapter 2 discusses the modeling of product variety induced manufacturing complexity.

A measure of complexity is proposed based on a careful observation of the choice processes in mixed-model assembly systems. As the simplest form, the measure is defined for independent and identically distributed (*i.i.d.*) sequences of choices. The *i.i.d.* based measure has a closed form for calculating complexity, and allows for the development of models for large systems analysis. Based on the measure, complexity models are developed for assembly stations and systems. The station level model captures the complexity calculation for each individual station, while the system level model reveals the interconnects between the stations. Moreover, in the system level model, a unique complexity propagation has been proposed. The propagation suggests the cause-and-effect relationship between variety and complexity, which enhances the understanding on the mechanism through which variety impacts manufacturing. Because of the stream-like flows in defining the relationship, the system level model is also called a "Stream of Complexity" (SoC) model. Finally, by using the complexity measure and models, various applications are suggested in an effort of minimizing complexity for the best performances.

Chapter 3 applies the SoC model for assembly sequence planning. The sequence planning is an important task in assembly system design. It is to determine the order of assembly tasks to be performed. According to the system level complexity model developed in Chapter 2 under the *i.i.d.* assumption, assembly sequence determines the directions in which complexity flows. Thus proper sequence planning helps reduce complexity. However, due to the difficulty of handling the directions of complexity flows in optimization, a transformed network flow model is formulated and solved based on dynamic programming.

Chapter 4 extends the complexity measure for *non-i.i.d.* sequences of choices, and applies the extended model for build sequence scheduling. The extension suggests that the knowledge of sequential dependencies could be utilized to reduce uncertainty and help operators to make choices. Since build sequence scheduling makes the decisions on the order of products being built, it determines the sequential dependencies between the choices. Thus, proper build sequence can help to reduce complexity. A model based on the Hidden Markov Chains is proposed and solved for the scheduling problem with constraints.

In summary, Chapter 2 answers the questions in Tasks 1 and 2. Chapters 3 and 4 tackle the problems in Task 3, and at the same time, they extend the complexity measure and model.

Finally, Chapter 5 concludes the thesis and suggests future research directions.

CHAPTER 2

Modeling Product Variety Induced Manufacturing Complexity

Abstract

Mixed-model assembly systems have been recognized as a major enabler to handle product variety. However, the assembly process becomes very complex when the number of product variants is high, which, in turn, may impact the system performance in terms of quality and productivity. This chapter [1] considers the variety induced manufacturing complexity in manual, mixed-model assembly lines where operators have to make choices for various assembly activities. A complexity measure called "Operator Choice Complexity" (OCC) is proposed to quantify human performance in making choices. The OCC takes an analytical form as an information-theoretic entropy measure of the average randomness in a choice process. Meanwhile, empirical evidences are provided to support the proposed complexity measure. Based on the OCC, models are developed to evaluate the complexity at each station, and for the entire assembly line. Consequently, complexity can be minimized by making systems design and operation decisions, such as error-proof strategies, assembly sequence planning, and build sequence scheduling.

2.1 Introduction

Traditional mass production was based on dedicated assembly lines where only one product model was produced in large quantities. Such systems can achieve high productivity by using principles of economies of scale and work division among assembly stations. However, in today's marketplace, where customers demand high product variety and short lead time, mass customization has been recognized as a new paradigm for manufacturing

[1]Part of work in this chapter has appeared on [54]: X. Zhu, S. J. Hu, Y. Koren, and S. P. Marin. Modeling of manufacturing complexity in mixed-model assembly lines. *Journal of Manufacturing Science and Engineering*, 130(5):051013-10, 2008. Also appears on the Proceedings of 2006 ASME International Conference on Manufacturing Science and Engineering.

[40, 27]. Mass customization promises individualized products at mass production cost. As a result of such paradigm change, assembly systems must be designed to be responsive to customer needs while at the same time achieving mass production's quality and productivity. Mixed-model assembly lines (MMAL) have been recognized as a major enabler to handle increased variety. An MMAL typically takes the form of a flow line. The topics of effectively assigning tasks to stations and balancing the lines for the multiple product types have been active research areas for MMAL in recent years [42].

Various industries are practicing mixed-model assembly lines. The variety of products offered in these lines has increased dramatically over the last decade. For example, in a typical automobile assembly plant, the number of different vehicles being assembled can reach tens of thousands in terms of the possible build-combinations of options. In fact, BMW claims that "Every vehicle that rolls off the belt is unique" and the number of possible automobile variations in the BMW 7 Series alone could reach 10^{17}.

Such an astronomical number of build-combinations undoubtedly presents enormous difficulties in the design and operation of the assembly systems. Using automobile assembly as an example, it has been shown by both empirical and simulation results [15, 33, 14] that increased vehicle product variety has significant negative impact on the performance of the mixed-model assembly process, such as quality and productivity. Such impact can result from assembly system design as well as people performance under high variety. The effect from the latter persists since only limited automation can be implemented in the automobile final assembly [39, 2]. Thus the questions presented here are two fold: how variety impacts people and system performance, and how to design assembly systems and organize production to allow high product variety without sacrificing quality and productivity.

One of the possible approaches to assess the impact of product variety on manufacturing system performance is to investigate how product variety *complicates* the mixed-model assembly process. However, only limited research has been done on defining manufacturing system complexity. For example, MacDuffie et al. [33] established *empirical* relationship between complexity and manufacturing system performance. They defined product mix complexity by looking at product variety (product mix and its structure) in assembly plants. According to the differences in the levels of product variety, three types of product mix complexity were defined in terms of empirical scores: Model Mix Complexity, Parts Complexity, and Option Complexity. By statistical analysis, significant negative correlation between the complexity measures and the manufacturing performance was found. The result was based on the data from 70 assembly plants worldwide who participated in the International Motor Vehicle Program at M.I.T.

Besides empirical studies, attempts have also been made to *analytically* define complexity in manufacturing. For instance, complexity has once been associated with the amount of effort needed to make a part. The effort was quantified by a logarithmic function of the probability of achieving a certain geometric precision and surface quality in *machining* [38].

The function is widely known as Shannon's Information Entropy [45]. Similarly, Fujimoto and Ahmed [16] defined a complexity index for *assembling*. The index takes the form of entropy in evaluating the assemblability of a product. The assemblability was defined as the uncertainty of gripping, positioning, and inserting parts in an assembly process. Also, complexity has been extended as a measure of uncertainty in achieving the specified functional requirements in axiomatic design [47].

Recently, complexity has been defined in an analytical form for manufacturing systems as a measure of how product variety complicates the process. Fujimoto et al. [17] introduced a complexity measure based on product structure using information entropy in different assembly process planning stages. By reducing the complexity, they claimed that impact of product variety on manufacturing systems could be reduced. However, the complexity measure does not incorporate the manufacturing system characteristics into the analysis. Deshmukh et al. [10] defined an entropic complexity measure for part mix in job shop scheduling. The complexity quantifies the difficulty associated with making scheduling decisions for the job shop, in which several types of products are manufactured simultaneously. An information-theoretic entropy measure of complexity is derived for a given combination and ratio of the part types. However, the complexity analysis is not applicable for mixed-model assembly, which has a flow line or various hybrid configurations.

In summary, there is a general agreement that (i) product variety does increase the complexity in manufacturing systems, and (ii) information entropy is an effective measure of complexity. However, in order to analyze the impact of variety on manufacturing complexity in mixed-model assembly systems, one has to take into consideration of the characteristics of the assembly system, such as system configuration, task to station assignment, and assembly sequences, etc. In addition, there is a lack of understanding on the mechanisms through which variety impacts manufacturing.

To address the above issues, this chapter defines a new measure of complexity which integrates both product variety and assembly process information, and then develops models for evaluating complexity in multi-stage mixed-model assembly systems. The chapter is organized as follows. Section 2.2 defines the measure of operator choice complexity which results from the analysis of choices and choice processes in mixed-model assembly operations. Moreover, the section also provides both theoretic and empirical justifications for the viability of the measure. Section 2.3 presents the modeling of complexity for mixed-model assembly lines, where models at the station and system levels are both investigated. Additionally, the influence of process flexibility is analyzed using numerical examples. Then potential applications for assembly system design by using the models are suggested in Section 2.4. Finally Section 2.5 summarizes the chapter.

2.2 Measure of Operator Choice Complexity

This section begins with a brief introduction to mixed-model assembly lines. Then it describes the choices and choice processes on the line to help theoretically define the measure of choice complexity. The measure is then justified by results from the cognitive ergonomics studies.

2.2.1 Mixed-Model Assembly Line

Figure 2.1 illustrates an example of a product structure and its corresponding mixed-model assembly line. The product has three functional features (F_i); each feature has several variants (e.g., V_{ij} is the j^{th} variant of F_i). The product structure is represented by a Product Family Architecture (PFA) [48].

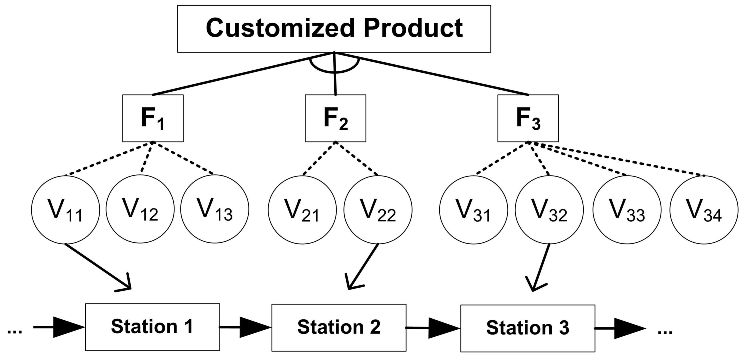

Figure 2.1: An illustration of a Product Family Architecture (PFA) and its mixed-model assembly line

The PFA illustrates all the possible build-combinations of the customized products by combining the variants of features. For example, in Fig. 2.1, the maximal number of different end products is 24 (i.e., $3 \times 2 \times 4$). Moreover, we represent the product mix information by a matrix $\mathbf{P}$, where p_{ij} is the demand (in percentage) of the j^{th} variant of the i^{th} feature. For instance, the $\mathbf{P}$ matrix for the product in Fig. 2.1 is the following:

$$\mathbf{P} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & 0 \\ p_{21} & p_{22} & 0 & 0 \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \tag{2.1}$$

Each row corresponds to the demand (in terms of mix ratio) of one feature, satisfying: $\sum_j p_{ij} = 1, \forall i$.

In the mixed-model assembly process, one of the variants from every feature is selected and assembled sequentially along the flow of the assembly line. For example, as depicted in Fig. 2.1, V_{11} is chosen for F_1, V_{22} for F_2, and V_{32} for F_3. Quite often, this assembly process is accomplished manually. Operators at every station must make correct choices among a number of alternatives. The choices include choosing the right part, tool, fixture, and assembly procedure for the variant.

2.2.2 Choices and Choice Processes

At each assembly station, the operator must choose the correct part from all possible variants according to customer orders. The specification of an order is usually written on a production tag/manifest attached on the partially completed assemblage. This process of selecting the right part is continuing during the day. To better understand the process, we define it as a *choice process*.

The choice process consists of a sequence of choices with respect to time. It can be modeled as a sequence of random variables, each of which represents choosing one of the possible alternatives. Mathematically, it can be considered as a discrete time discrete state stochastic process $\{X_t, t = 1, 2, \ldots\}$, on the state space (the choice set) $X_t \in \{1, 2, \ldots, M\}$, where t is the index of discrete time period, M is the total number of possible alternatives (parts) which could be chosen during each period. More specifically, $X_t = m, m \in \{1, 2, \ldots, M\}$, is the event of choosing the m^{th} alterative during period t.

In the simplest case, if the choice process is independent and identically distributed (*i.i.d.*), we then use a single random variable X (instead of X_t's) to describe the outcome of a choice. Furthermore, we know all the alternatives of X and their probabilities, i.e., the probability of a choice taking the m^{th} outcome is known as $p_m \triangleq \mathbf{P}(X = m)$, for $m = 1, 2, \ldots, M$. In the following discussions, we limit ourselves by assuming *i.i.d.* sequences.

2.2.3 Operator Choice Complexity

To characterize the operator performance in making choices, we define the term *operator choice complexity* (or *choice complexity*) as follows.

Definition: Choice complexity is the average uncertainty or randomness in a choice process, which can be described by a function H in the following form:

$$H(X) = H(p_1, p_2, \ldots, p_M) = -C \cdot \sum_{m=1}^{M} p_m \cdot \log p_m \tag{2.2}$$

where C is a constant depending on the base of the logarithm function chosen. If log_2 is selected, $C = 1$ and the unit of complexity is *bit*.

Theoretical Properties: The following seven properties of the function H as described in [45] make it suitable as a measure of choice complexity.

1. H is continuous in p_m, i.e., small changes in p_m should result in only small changes in choice complexity.

2. If p_m's are brought closer to each other, H would increase. Put alternatively, any change towards equalization of $p_1, p_2, \ldots, p_M$ should increase H. For a given M, H is a maximum and equal to $\log M$ when all p_i's are equal (i.e., $\frac{1}{M}$). In this case, H is a increasing function of M. This case is also intuitively the most uncertain situation to make a choice, since the operator is considered to be *non-informative* [25].

3. If a choice process is broken down into two successive stages, the original H is the weighted sum of the individual values of H. For example, $H(\frac{1}{2}, \frac{1}{3}, \frac{1}{6}) = H(\frac{1}{2}, \frac{1}{2}) + \frac{1}{2}H(\frac{2}{3}, \frac{1}{3})$.

4. $H = 0$ if and only if all the p_m's but one are zero, this one has the value of unity, i.e., $H(1, 0, \ldots, 0) = H(0, 1, \ldots, 0) = H(0, 0, \ldots, 1) = 0$. Thus only when we are certain of the outcome does H vanish and there exists no choice complexity. Otherwise H is positive.

5. H does not change when an additional alternative with no chance to happen is added into the original system.

6. H is a symmetrical function of $p_1, p_2, \ldots, p_M$, i.e., if the probabilities of choices are permuted among the alternatives, choice complexity does not change.

7. H is a sum of surprisal functions weighted by probability p_m's[25]. A surprisal function $\log \frac{1}{p_m}$ is defined to quantify how much surprise (uncertainty) is incurred for an individual choice. The higher the probability of the incoming alternative is, the less surprisal incurred, and vice versa. Thus, by weighting the surprisal with probabilities for the choice process, we obtain the entropy which characterizes the *average* randomness in the sequence.

Therefore the entropy function H possesses most of the desirable properties to be one of the possible measures of choice complexity.

2.2.4 Justifications for Choice Complexity Measure

There is a close similarity and connection between the theoretical properties of the complexity measure and the experimental results found in human cognitive studies. The experiments were conducted to assess human performance when making choices. Coincidentally, information entropy was found to be one of the effective measures. The performance of human choice-making activities was investigated by measuring average reaction times, i.e., how quickly a person can make a choice to a stimulus. One of the earliest studies was done by Merkei in 1885, described by Woodworth [52]. In the experiment, digits 1 through 5 were assigned to the fingers of the right hand and the Roman numbers I through V were assigned to the fingers of the left hand. On any given set of trials, the subject knew which of the set of stimuli would be possible (e.g., if there were three possible stimuli, they might be 3, 5, and V). Merkel studied the relationship between the number of possible stimuli and the choice reaction time (RT). His basic findings are presented in Fig. 2.2(a), where the relationship between choice RT and the number of alternatives was not linear.

This relationship in Fig. 2.2(a) has been further studied by a number of researchers since Merkel's original observations. Among them, the most widely known one was Hick [21]. He discovered that the choice RT is linearly proportional to the logarithm of the number of stimulus alternatives if all the alternatives are equal likely, see Fig. 2.2(b), i.e.,

$$\text{Mean Choice RT} = a + b \cdot [\log_2 n] \tag{2.3}$$

where n is the number of stimulus-response alternatives, a and b are constants, which can be determined empirically by fitting a line to the measured data. This relation came to be known as *Hick's Law*, which was regarded as one major milestone in the area of cognitive ergonomics.

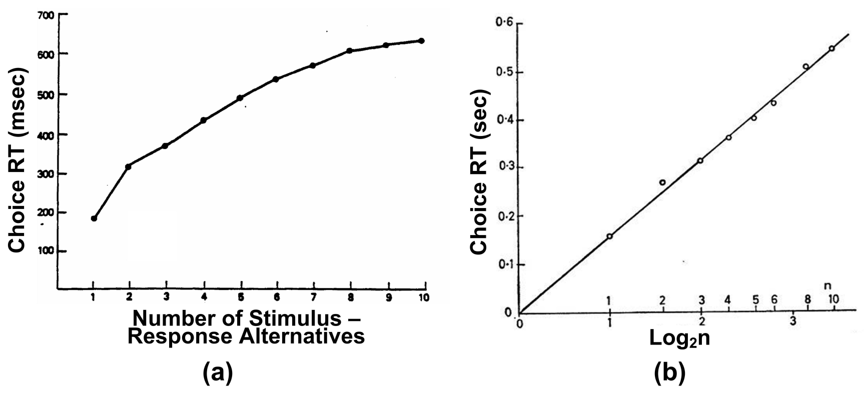

Figure 2.2: Mean choice RT as (a) a nonlinear function of the number of stimulus-response alternatives [52]; (b) a linear function of stimulus information, or $\log_2$ of the number of alternatives [21], reprinted from [51]

Coincidentally, the term $[\log_2 n]$ is exactly the information entropy calculated in Eq. (2.2) if all the p_m's are equal, which follows from the experiment setting that the choice process is *i.i.d.* and all the alternatives occur equal likely. The above analogy was first discovered by Hyman [23], where he concluded that, "The reaction time seems to behave, under certain conditions, in a manner analogous to the definition of information".

Hyman [23] also realized that, according to Shannon's definition of information entropy, he could change information content in the experiment by other means. Thus, in addition to varying the number of stimuli and letting each one of them occur equally likely in Hick's [21] experiment, he altered stimulus information content simply by (i) changing the probability of occurrence of particular choices, (ii) introducing sequential dependencies between successive choices of alternatives, see Fig. 2.3. Thus, naturally enough, we can use H to replace the $[\log_2 n]$ term, Eq. (2.3) becomes,

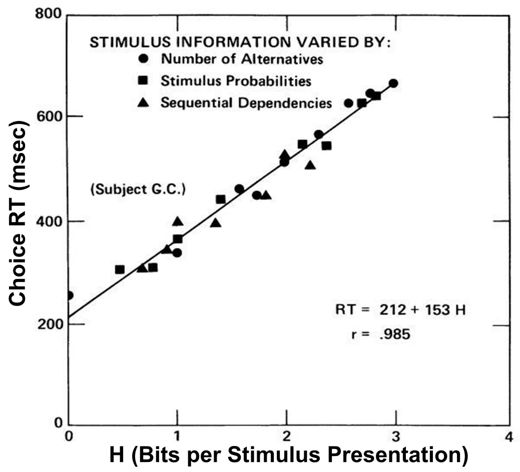

Figure 2.3: Choice RT for three different ways of manipulating the stimulus information H, reprinted from [6], using data from [23]

$$\text{Mean Choice RT} = a + b \cdot H \tag{2.4}$$

Because of the significance of this generalization, Hick's Law is also referred as the Hick-Hyman Law.

The H term in Eq. (2.4) is one of the variants of Shannon's information entropy [45] in the communication systems study. Thus, a fundamental assumption behind this analogue is that the mental process of human being is modeled as an information transmission process. In fact, this assumption is confirmed by the recent research in cognitive ergonomics on the queuing network modeling of elementary mental process. Liu [32] suggested that, at the level of mean RTs, a continuous-transmission fork-join network demonstrate the same logarithmic behavior as that of the experimental results in Hick-Hyman Law. Hence, the legitimacy of applying Eq. (2.4) is limited to the situation where the individuals are asked to response to the stimulus promptly, and the decision to be made is very simple, requiring little conscious thought. When analyzing mixed-model assembly process, we observe the very similar situation that the line operators are asked to handle variety in a very tight cycle time without time for deliberating over the decisions. However, if the subjects are given more time, the thinking process will not be as simple as merely an information transmission. Liu [32] also reviewed a class of more sophisticated queuing network models for RT.

Moreover, it was suggested in Welford [51] that the information measure is adequate to assess human performance, since it provides a valuable means for combining reaction time and *errors* (i.e., speed and accuracy) into a single score.

Practitioners in various fields have found the information entropic measure of human performance useful. One of the examples using the Hick-Hyman Law in assembly operation analysis comes from Bishu and Drury [4]. They used the amount of information, measured in bits, contained in a wiring assembly task to predict task completion time. The amount of information is a function of both the number of wires to choose from and the number of terminals to be wired. They found that task completion time was linearly related to the amount of information contained in the task. Additionally, they also found that the more the information gain was, the more likely errors will occur. That is, the total information content increases both the task completion time and errors. Gatchell [18] used the Choice Reaction Time technique and experimentally studied operator performance on part choices under part proliferation. Her findings suggest that operator with more part choices made more errors and needed more decision time.

According to both theoretical properties and empirical results, the entropy-based quantity H is suitable to measure *operator choice complexity.* Therefore, we propose to use the following form to quantify the value of choice complexity.

$$\text{Choice Complexity} = \alpha(a + b \cdot H), \alpha > 0 \tag{2.5}$$

The form is similar to that of the Hick-Hyman Law. It only differs in a positive scalar α, served as a weight to a specific choice process. In other words, the choice complexity is positive monotonic to the amount of uncertainty embedded in the choice process. Since Eq. (2.5) takes a simple linear form with constants α, a, and b, the only remaining part to be determined is the value of H when evaluating complexity. By incorporating information from product design, line design, and operation, one can develop models and methodologies to quantify the information content in terms of the various operator choices in a mixed-model assembly process.

2.3 Models of Complexity for Mixed-Model Assembly Lines

This section defines the operator choice complexity in the station level by simply extending the previous definition for a single assembly activity. Then complexity in the system level is examined after a unique propagation behavior of complexity is found. Moreover, process flexibility and commonality is taken into account when analyzing complexity. Finally a "Stream of Complexity" model is proposed for multi-stage assembly systems.

2.3.1 Station Level Complexity Model

On a station, in addition to the part choice mentioned in Section 2, the operator may perform other assembly activities as well in a sequential manner, and some examples of the corresponding choices are briefly described as follows, see Fig. 2.4.

Fixture choice: choose the right fixture according to the base part (i.e., the partially completed assemblage) to be mounted on as well as the added part to be assembled.

Tool choice: choose the right tool according to the added part to be assembled as well as the base part to be mounted on.

Procedure choice: choose the right procedure, e.g., part orientation, approach angle, or temporary unload of certain parts due to geometric conflicts/subassembly stabilities.

According to Eq. (2.5), we define the associated complexity at the station as part choice complexity, fixture choice complexity, tool choice complexity, and assembly procedure choice complexity respectively. All these choices contribute to the *operator choice complexity.*

Without loss of generality, we number the sequential assembly activities in Fig. 2.4 from 1 to K and denote C_j as the total complexity of station j, which is a weighted sum of the various types of choice complexity at the station.

$$C_j = \sum_{k=1}^{K} \alpha_j^k (a_j^k + b_j^k \cdot H_j^k), \alpha_j^k > 0, k = 1, 2, \dots, K \tag{2.6}$$

where α_j^k, are the weights related to the task difficulty of the k^{th} assembly activity at station j; a_j^k's and b_j^k's are empirical constants depending on the nominal human performance similar to that of the choice reaction time experiments; H_j^k is the entropy computed from the variant mix ratio relevant to the k^{th} activity at station j. For simplicity, we assume $a_j^k = 0, b_j^k = 1, \forall j, k$. Then Eq. (2.6) reduces to,

$$C_j = \sum_{k=1}^{K} \alpha_j^k H_j^k, \alpha_j^k > 0, k = 1, 2, \dots, K \tag{2.7}$$

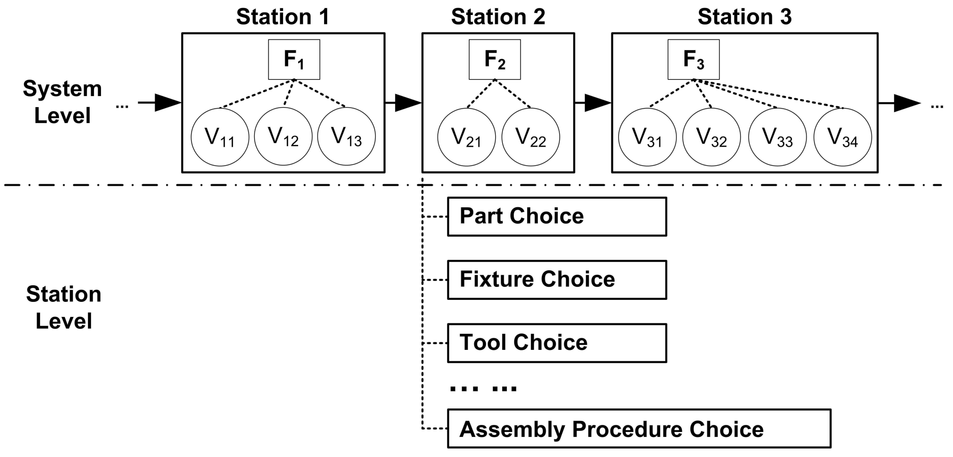

Figure 2.4: Choices in sequential assembly activities at one station

2.3.2 Propagation of Complexity

By Eq. (2.7), complexity on individual stations is considered as a weighted sum of complexities associated with every assembly activities. Among them, some activities are caused only by the feature variants at the current station, such as picking up a part, or making choices on tools for the selected part. The complexity associated with such assembly activity is defined as *feed complexity.* However, the choice of fixtures, tools, or assembly procedures at the current station may depend on the feature variant that has been added at an upstream station. This particular component of complexity is termed as *transfer complexity.*

A formal definition of the two types of complexity is given below. Assume a current station j:

Feed complexity: Choice complexity caused by the feature variants added at station j.

Transfer complexity : Choice complexity caused by the feature variants added at an upstream station, i.e., station i (i precedes j, denoted as $i \prec j$).

Transfer complexity exists because the feature variants added on the previous station i may affect the process of realizing the feature at station j, causing tool changeovers, fixture conversions, or procedure changes.

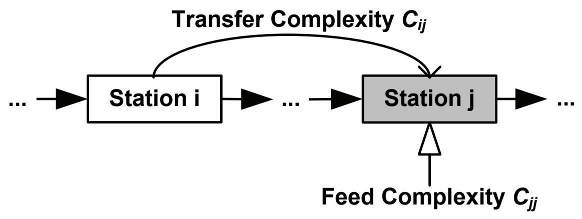

Figure 2.5: Complexity propagation scheme

The propagation behavior of the two types of complexity is depicted in Fig. 2.5, where, for station j, the feed complexity is denoted as C_{jj} (with two identical subscripts), and the transfer complexity is denoted as C_{ij} (with two distinct subscripts to represent the complexity of station j caused by an upstream station i). Thus the transfer complexity can flow from upstream to downstream, but not in the opposite direction. In contrast, the feed complexity can only be added at the current station with no flowing or transferring behavior.

Hence the total complexity at a station is simply the sum of the feed complexity at the station and the transfer complexity from all the upstream ones, i.e., for station j,

$$C_j = C_{jj} + \sum_{\forall i: i \prec j} C_{ij} \tag{2.8}$$

Compared with Eq. (2.7), we may find equivalence relationships term by term between the two sets of equations. We illustrate this in the following section with examples.

2.3.3 Examples of Complexity Calculation

In this section, by continuing the example in Fig. 2.1 which is redrawn in Fig. 2.6, we demonstrate the procedures of calculating complexity at a station. More specifically, we will consider examples with or without process flexibility respectively.

Example without process flexibility

In Fig. 2.6, on the one hand, four sequential assembly activities are identified at station 3, complexity is expressed according to Eq. (2.7) by assigning superscripts 1 to 4 as part choice complexity, fixture choice complexity, tool choice complexity, and assembly procedure choice complexity respectively. Thus, according to the station level model, we have the following equation for station 3.

$$C_3 = \alpha_3^1 H_3^1 + \alpha_3^2 H_3^2 + \alpha_3^3 H_3^3 + \alpha_3^4 H_3^4 \tag{2.9}$$

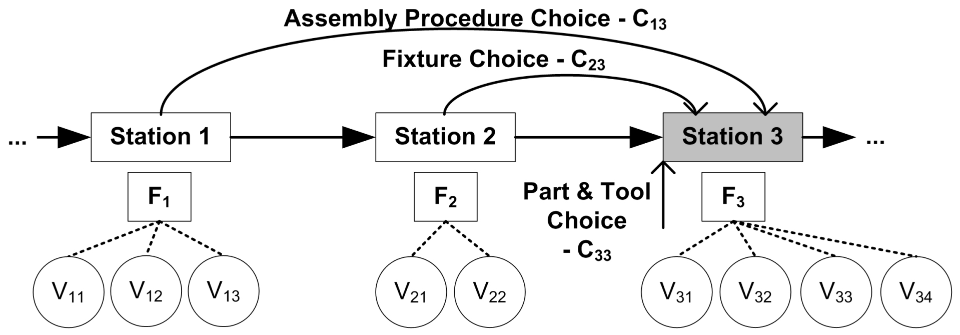

Figure 2.6: Complexity propagation of the example in Fig. 2.1

At the station, we also know the process requirement as follows.

1. One of the four parts, i.e., variants of F_3, is chosen according to customer order;
2. One of the four distinct tools is chosen according to the chosen variant of F_3;
3. One of the two distinct fixtures is chosen according to the variant of F_2 installed at station 2;
4. One of the three distinct assembly procedures is chosen according to the variant of F_1 installed at station 1.

On the other hand, the propagation scheme at the system level can be determined from the viewpoint of feed complexity (C_{33}) and transfer complexity (C_{13} and C_{23}), which is expressed according to Eq. (2.8) as follows.

$$C_3 = C_{33} + C_{13} + C_{23} \tag{2.10}$$

There exists an agreement between Eqs.(2.9) and (2.10), or equivalently, Eqs.(2.7) and (2.8), which is shown below.

Given process information, we identify the types of choice complexity in Eq. (2.10) as follows:

- Part choice complexity: $\alpha_3^1 H_3^1$
- Tool choice complexity: $\alpha_3^3 H_3^3$
- Fixture choice complexity: $\alpha_3^2 H_3^2$
- Procedure choice complexity: $\alpha_3^4 H_3^4$

By complexity propagation, we have:

- Feed complexity: $C_{33} = \alpha_3^1 H_3^1 + \alpha_3^3 H_3^3$
- Transfer complexity: $C_{23} = \alpha_3^2 H_3^2$, $C_{13} = \alpha_3^4 H_3^4$

From the agreement, the sources of complexity can be identified and the H terms are now easily calculated. That is, if an H term corresponds to the feed complexity, it is a function of the mix ratio of the current station; however, if an H corresponds to the transfer complexity, it is a function of the mix ratio of the station which is specified in the first subscript of its corresponding C_{ij}, i.e., station i. As a result, $H_3^1 = H_3^3 = H_3$, where H_3 is the entropy of the variants added at station 3; and similarly, $H_3^2 = H_2$, $H_3^4 = H_1$.

Now, let us consider numerical values for the example, assume the $\mathbf{P}$ matrix in Eq. (2.1) takes the following values.

$$\mathbf{P} = \begin{bmatrix} 0.5 & 0.2 & 0.3 & 0 \\ 0.5 & 0.5 & 0 & 0 \\ 0.3 & 0.3 & 0.2 & 0.2 \end{bmatrix}$$

Then,

$$\begin{aligned} H_3^1 &= H_3^3 = H_3 = H(0.3, 0.3, 0.2, 0.2) = 1.971 \text{ bits} \\ H_3^2 &= H_2 = H(0.5, 0.5) = 1 \text{ bit} \\ H_3^4 &= H_1 = H(0.5, 0.2, 0.3) = 1.485 \text{ bits} \end{aligned} \tag{2.11}$$

and,

$$\begin{aligned} C_3 &= C_{33} + C_{13} + C_{23} \\ &= 1.971\alpha_3^1 + 1.971\alpha_3^3 + \alpha_3^2 + 1.485\alpha_3^4 \end{aligned} \tag{2.12}$$

For simplicity, assuming $\alpha_3^1 = \alpha_3^2 = \alpha_3^3 = \alpha_3^4 = 1$, we finally obtain the total complexity at station 3.

$$C_3 = 1.971 + 1.971 + 1 + 1.485 = 6.427 \text{ bits} \tag{2.13}$$

Influence of process flexibility

So far, we have illustrated in Eqs.(2.11)–(2.13) an example of calculating choice complexity with no flexibility in the manual assembly process. However, flexibility is usually built into assembly systems such that common tools or fixtures can be used for different variants so as to simplify the process. That is, flexible tools, common fixtures, or shared assembly procedures are adopted to treat a set of variants so that choices (of the tools, fixtures, and assembly procedures) are eliminated. Since fewer choices are needed, complexity reduces. However, not all the assembly processes can be simplified by flexibility strategies. Sometimes, flexible tools, common fixtures, or shared assembly procedure may require significant changes or compromises in product design and process planning, which is usually costly if not impossible. To characterize the impact of flexibility, i.e., to establish the relationship between product feature variants and process requirements, a *product-process association matrix* (denoted as Δ-matrix) is defined in the following discussion.

We again use the example in Fig. 2.6. At station 3, we consider fixture changeover, and it is denoted as the k^{th} assembly activity. Which fixture should be used in assembling F_3 at station 3 is determined by the variant of F_2 assembled previously at station 2. If no flexibility is present, fixture choice is needed at station 3 by observing feature F_2 according to the following rules,

- Use fixture 1, if V_{21} is present;
- Use fixture 2, if V_{22} is present.

Thus there are two states in the fixture choice process; the mapping relationship can be expressed in a Δ-matrix as follows.

$$\Delta_{23}^2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{2.14}$$

where Δ_{23}^2 denotes the Δ-matrix for the 2^{nd} activity at station 3 associated with the variants added at station 2; the columns are the states of the 2^{nd} activity at station 3, rows are the variants of the feature F_2 affecting the activity. The ones in the cells establish associations between the state in the column and the variant in the row.

A general definition of the Δ-matrix for the k^{th} assembly activity at station j due to variety added at station i is given as follows.

$$\Delta_{ij}^{k} = \begin{bmatrix} \delta_{1,1} & \delta_{1,2} & \dots & \delta_{1,m} \\ \vdots & \vdots & \ddots & \vdots \\ \delta_{n,1} & \delta_{n,2} & \dots & \delta_{n,m} \end{bmatrix} \tag{2.15}$$

where,

$$\delta_{s,t} = \begin{cases} 1 & \text{Variant } s \text{ at station } i \text{ requires } k^{th} \text{ activity} \\ & \text{to be in state } t \text{ at station } j \\ 0 & \text{Otherwise} \end{cases}$$

m, n are the cardinality of states and variants respectively.

By definition, the Δ-matrix satisfies the following properties:

1. $\sum_{t=1}^{m} \delta_{s,t} = 1$, for $s = 1, 2, \dots, n$;
2. $\sum_{s=1}^{n} \delta_{s,t} \geq 1$, for $t = 1, 2, \dots, m$;
3. $n \geq m$.

Property 1 holds because one variant can lead to one and only one state. Property 2 holds because each state must be associated with at least one variant; otherwise, the column associated the empty state can be eliminated, and the size the matrix shrinks by 1. Lastly, property 3 holds because the maximal number of states cannot exceed the total number of variants. That is, in the extreme case of non-flexibility, each variant requires the characteristic to be in a distinct state, and the Δ-matrix becomes a unit matrix of dimension being the number of variants.

Consider the example in Fig. 2.6 again, however, if a common fixture is adopted, the same fixture can be used no matter V_{21} or V_{22} is mounted on station 2. Thus, by definition, the Δ-matrix becomes simply:

$$\Delta_{23}^{2} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}$$

which could be reduced to:

$$\Delta_{23}^{2} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \tag{2.16}$$

By using the Δ-matrix, we are now capable of calculating the H terms when flexibility is present in the process. Define a vector $\mathbf{q}_{ij}^{k} = [q_1, q_2, \dots, q_m]$, where $q_t, t \in \{1, 2, \dots, m\}$ is the probability of the k^{th} activity being in state t at station j due to the variants added at station i, satisfying $\sum_{s=1}^{m} q_s = 1$. By the definition of the product mix matrix P in Eq. (2.1) and the Δ-matrix in Eq. (2.15), the following relationship holds:

$$\mathbf{q}_{ij}^k = [q_1, q_2, \ldots, q_m] = \mathbf{P}_{i\cdot} \times \Delta_{ij}^k \tag{2.17}$$

where $\mathbf{P}_{i\cdot}$ is the i^{th} row of matrix $\mathbf{P}$, representing the mix ratio of the feature (i.e., F_2 in the example) assembled on station i. Thus, the corresponding H term is:

$$H_j^k = H(\mathbf{q}_{ij}^k) = -\sum_{t=1}^{m} q_t \cdot \log_2 q_t \tag{2.18}$$

Revisit the example in Fig. 2.6. When calculating the H term (H_3^2) corresponding to the fixture choice complexity at station 3, we have the following results:

Case 1 Use dedicated fixtures, i.e., a different fixture for each variant. By the Δ-matrix in Eq. (2.14), we have:

$\mathbf{q}_{23}^2 = \begin{bmatrix} q_1 & q_2 \end{bmatrix} = \mathbf{P}_{2\cdot} \times \Delta_{23}^2$

$= \begin{bmatrix} 0.5 & 0.5 \end{bmatrix} \times \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.5 & 0.5 \end{bmatrix}$

$\Rightarrow H_3^2 = \sum_{t=1}^{2} q_t \cdot \log_2 \frac{1}{q_t} = 2 \times 0.5 \log_2 \frac{1}{0.5} = 1$ bit

which duplicates exactly the result in Eq. (2.11).

Case 2 Common fixture is used. By the Δ-matrix in Eq. (2.16), we have:

$\mathbf{q}_{23}^2 = \begin{bmatrix} q_1 & q_2 \end{bmatrix} = \begin{bmatrix} 0.5 & 0.5 \end{bmatrix} \times \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \end{bmatrix}$

$\Rightarrow H_3^2 = \sum_{t=1}^{1} q_t \cdot \log_2 \frac{1}{q_t} = 1 \cdot \log_2 \frac{1}{1} = 0$ bit

Since fixture is common to the process of assembling F_3 with variants of F_2, no choice is needed.

Assume we have flexibility or commonality in fixture, tool, and assembly procedures respectively, which is expressed by the Δ-matrices in Table 1. As a summary, the table also demonstrates a detailed numerical example to calculate complexity at station 3. The results show a reduced value of choice complexity compared with Eq. (2.13) because of the additional process flexibility.

2.3.4 System Level "Stream of Complexity" Model

In general, consider an assembly line with n workstations, numbered 1 through n sequentially, see Fig. 2.7. The mix ratio defined in Eq. (2.1) is known. Using Eq. (2.2), we can obtain the entropy H for the variants at each station according to their mix ratios.

The propagation of complexity in a multi-stage system can be analyzed by considering how the complexity of assembly operations (choices) at a station is influenced by the variety added at its upstream stations (*incoming complexity*), as well as how variants added at the

Table 2.1: Numerical Example of Complexity Calculation

No.	Activity	Δ-matrix	q-vector	*H*-term
1	Part Pick-up	$\mathbf{\Delta}_{33}^{1}=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}$	$\mathbf{q}_{33}^{1}=\begin{bmatrix}.3\\.3\\.2\\.2\end{bmatrix}^{T}$	H_3^1=1.971
2	Fixture Conversion	$\mathbf{\Delta}_{23}^{2}=\begin{bmatrix}1\\1\end{bmatrix}$	$\mathbf{q}_{23}^{2}=[1]^{T}$	H_3^2=0
3	Tool Changeover	$\mathbf{\Delta}_{33}^{3}=\begin{bmatrix}1&0\\1&0\\0&1\\0&1\end{bmatrix}$	$\mathbf{q}_{33}^{3}=\begin{bmatrix}.6\\.4\end{bmatrix}^{T}$	H_3^3=0.971
4	Assembly Procedure Change	$\mathbf{\Delta}_{13}^{4}=\begin{bmatrix}1&0\\0&1\\1&0\end{bmatrix}$	$\mathbf{q}_{13}^{4}=\begin{bmatrix}.8\\.2\end{bmatrix}^{T}$	H_3^4=0.722
Total complexity at station 3 with equal weights				3.664

station impact the downstream stations (*outgoing complexity*). Because of the stream-like flows in defining the cause-and-effect between variety and complexity, the system level model is also called a "Stream of Complexity" model.

The incoming complexity at station j, C_j^{in} is the amount of complexity flowing into the station from its upstream stations, which can be calculated in the following way:

Station 1: $C_1^{\text{in}} = C_{01} = a_{01}H_0$

Station 2: $C_2^{\text{in}} = C_{02} + C_{12} = a_{02}H_0 + a_{12}H_1$

$\ldots\ldots$

Station j: $C_j^{\text{in}} = C_{0j} + C_{1j} + C_{2j} + \ldots + C_{j-1,j}$
$= a_{0j}H_0 + a_{1j}H_1 + a_{2j}H_2 + \ldots + a_{j-1,j}H_{j-1}$

$\ldots\ldots$

Station n: $C_n^{\text{in}} = C_{0n} + C_{1n} + C_{2n} + \ldots + C_{n-1,n}$
$= a_{0n}H_0 + a_{1n}H_1 + a_{2n}H_2 + \ldots + a_{n-1,n}H_{n-1}$

where,

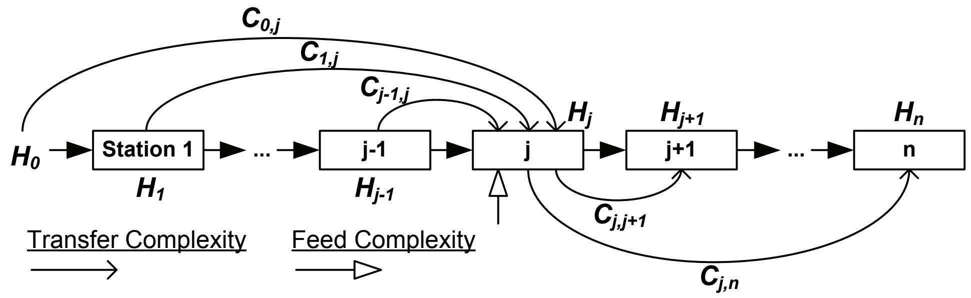

Figure 2.7: Propagation of complexity at the system level in a multi-stage assembly system

C_j^{in}	–	The incoming complexity of station j, $j = 1$ to n;
H_j	–	Entropy of variants added at station j;
H_0	–	Entropy of variants due to the base part;
a_{ij}	–	Coefficient of complexity impact on station j due to variety added at station i, i.e.,

$$a_{ij} = \begin{cases} \alpha_j^k & \text{Variants added at station } i \text{ has an impact on the } k^{th} \text{ assembly activity at station } j\text{, and } i < j\text{, } \alpha_j^k \text{ has been defined in Eq. (2.7).} \\ 0 & \text{Otherwise} \end{cases}$$

Or equivalently, by using a matrix representation, a comprehensive model can be obtained as follows:

$$\begin{bmatrix} C_1 \\ C_2 \\ \vdots \\ C_n \end{bmatrix}^{\text{in}} = \begin{bmatrix} a_{01} & 0 & \dots & 0 \\ a_{02} & a_{12} & \dots & 0 \\ \vdots & \vdots & \ddots & 0 \\ a_{0n} & a_{1n} & \dots & a_{n-1,n} \end{bmatrix} \times \begin{bmatrix} H_0 \\ H_1 \\ \vdots \\ H_{n-1} \end{bmatrix} \tag{2.19}$$

In short,

$$\mathbf{C}^{\text{in}} = \mathbf{A}^T \times \mathbf{H} \tag{2.20}$$

where $\mathbf{C}^{\text{in}}$ is the incoming complexity vector of size n for the system, with its i^{th} entry being the incoming complexity at station i, for $i = 1, 2, \dots, n$; $\mathbf{A}$ is the characteristics matrix of size $n \times n$, which characterizes the interactions between stations (due to the feature variants added on the stations) in term of choice complexity; and $\mathbf{H}$ encapsulates product variety information including the number of variants and their distributions.

The outgoing complexity at station j, C_j^{out} is the amount of complexity flowing out of the station. It is the amount of choice complexity caused by the variants added at the station, affecting the operations on the other stations downstream. Similarly, we have the following equations:

Station 1: $C_1^{\text{out}} = C_{12} + \ldots + C_{1n} = (a_{12} + \ldots + a_{1n}) \cdot H_1$

Station 2: $C_2^{\text{out}} = C_{23} + \ldots + C_{2n} = (a_{23} + \ldots + a_{2n}) \cdot H_2$

......

Station j: $C_j^{\text{out}} = C_{j,j+1} + \ldots + C_{jn}$
$= (a_{j,j+1} + \ldots + a_{jn}) \cdot H_j$

......

Station n: $C_{n-1}^{\text{out}} = C_{n-1,n} = a_{n-1,n} \cdot H_n$

where,

C_j^{out} – Outgoing complexity of station j, $j = 1$ to n;
In fact, by definition $C_n^{\text{out}} = 0$

Additionally, since the variety of the base part incurs transfer complexity as well, we denote it as C_0^{out}, i.e.,

$$\begin{aligned} \text{Base Part: } C_0^{\text{out}} &= C_{01} + C_{02} + \ldots + C_{0n} \\ &= (a_{01} + a_{02} + \ldots + a_{0n}) \cdot H_0 \end{aligned}$$

Using matrix form again, we obtain a comprehensive model for outgoing complexity as follows:

$$\begin{bmatrix} C_0 \\ C_1 \\ \vdots \\ C_{n-1} \end{bmatrix}^{\text{out}} = \left\{ \begin{bmatrix} a_{01} & a_{02} & \ldots & a_{0n} \\ 0 & a_{12} & \ldots & a_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & a_{n-1,n} \end{bmatrix} \times \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \right\}$$
$$.* \left[H_0, H_1, \ldots, H_{n-1} \right]^T \tag{2.21}$$

In short,

$$\mathbf{C}^{\text{out}} = [\mathbf{A} \times \mathbf{1}] .* \mathbf{H} \tag{2.22}$$

where $\mathbf{C}^{\text{out}}$ is the outgoing complexity vector of size n for the system, with its j^{th} entry being the outgoing complexity from station $j-1$, for $j = 1, 2, \dots, n$; $\mathbf{1}$ is a column vector of ones with size n; and $.*$ denotes the entry-by-entry product of two vectors with identical sizes.

2.3.5 Extension of the Model

The Stream of Complexity model can also be extended to incorporate the influence of process flexibility and commonality. In Section 2.3.3, we have demonstrated the use of *product-process association matrix* (i.e., the Δ-matrix) to deal with the situation where a common fixture was utilized for two different variants. Since the fixture helped to eliminate choices, the associated choice complexity was expected to decrease as well.

By the definition of Δ-matrix in Eq. (2.15), we safely drop k in the notation for convenience, and we rewrite Eq. (2.17):

$$\mathbf{q}_{ij} = \mathbf{P}_{i\cdot} \times \Delta_{ij} \tag{2.23}$$

In fact, the Δ-matrix in the above equations acts as a mathematical operator on the entropy of variants added at station i, we denote the operator in the form of a function.

$$\Delta_{ij}(H_i) \triangleq H(\mathbf{q}_{ij}) \tag{2.24}$$

Basically, what the operator does is to calculate the entropy value by incorporating the information of process flexibility (that may help eliminate choices) in the assembly process at station j regarding the variants assembled at station i.

By using Δ-matrices for every flows of transfer complexity in the system, we have Eqs.(2.19) and (2.20) extended.

$$\begin{bmatrix} C_1 \\ C_2 \\ \vdots \\ C_n \end{bmatrix}^{\text{in}} = \begin{bmatrix} a_{01} \cdot \Delta_{01} & 0 & \dots & 0 \\ a_{02} \cdot \Delta_{02} & a_{12} \cdot \Delta_{12} & \dots & 0 \\ \vdots & \vdots & \ddots & 0 \\ a_{0n} \cdot \Delta_{0n} & a_{1n} \cdot \Delta_{1n} & \dots & a_{n-1,n} \cdot \Delta_{n-1,n} \end{bmatrix} \times \begin{bmatrix} H_0, H_1, \dots, H_{n-1} \end{bmatrix}^T$$

In short,

$$\mathbf{C}^{\text{in}} = (\mathbf{A} \cdot \Delta)^T \times \mathbf{H} \tag{2.25}$$

The matrix multiplication requires the entry-by-entry computation:

$$a_{ij} \cdot \Delta_{ij} \cdot H_i = a_{ij} \cdot \Delta_{ij}(H_i) = a_{ij} \cdot H(\mathbf{q}_{ij})$$

Similarly, the extended versions of Eqs.(2.21) and (2.22) for outgoing complexity are as follows.

$$\left\{\begin{bmatrix}C_0, C_1, \ldots, C_{n-1}\end{bmatrix}^T\right\}^{\text{out}} =$$
$$\left\{\begin{bmatrix} a_{01}\cdot\Delta_{01} & a_{02}\cdot\Delta_{02} & \ldots & a_{0n}\cdot\Delta_{0n} \\ 0 & a_{12}\cdot\Delta_{12} & \ldots & a_{1n}\cdot\Delta_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & a_{n-1,n}\cdot\Delta_{n-1,n} \end{bmatrix} \times \begin{bmatrix}1\\1\\\vdots\\1\end{bmatrix}\right\}$$
$$.* \begin{bmatrix}H_0, H_1, \ldots, H_{n-1}\end{bmatrix}^T$$

In short,

$$\mathbf{C}^{\text{out}} = [(\mathbf{A} \cdot \Delta) \times \mathbf{1}] .* \mathbf{H} \tag{2.26}$$

Therefore, the extended system level complexity model comprehensively incorporates both product (such as product architecture and mix) and process (such as system configuration, tooling, task to station assignment, flexibility, etc.) information.

2.4 Potential Applications

Once the propagation of complexity is understood and models developed, they can be applied to the design of mixed model assembly systems. Several potential applications are described below.

2.4.1 Performance Evaluation and Root Cause Identification Using Complexity Charts

Following the procedures in Section 2.3.4, we can analyze the incoming and outgoing complexity for each station and plot them against the station position in a multi-stage assembly system, see Fig. 2.8. As a result, the stations with high incoming complexity are the potential stations where error-proofing strategies need to be provided to mitigate the impact of variety induced complexity on operator and system performance.

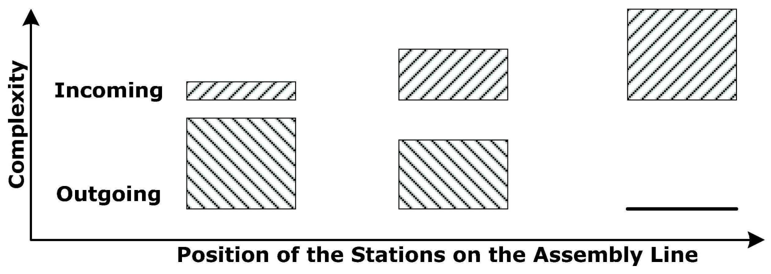

Figure 2.8: Incoming and Outgoing Complexity Charts

In Fig. 2.8, the outgoing complexity also shows how much influence the variants at one particular station have on its downstream operations. As a result, the outgoing complexity implies the root cause of the choice complexity in the system. Thus decisions from product design, such as process commonality strategies, option bundling policies need to be considered to moderate outgoing complexity.

2.4.2 Influence Index and Configuration Design

For any station j, once the values of incoming and outgoing complexity are found, we may define an index, called influence index, as follows:

$$I_j = \frac{C_j^{\text{out}}}{C_j^{\text{in}}} \tag{2.27}$$

The index quantifies how much relative influence the variants added at station j have on the operations of the other stations. To illustrate, in the Stream of Complexity model of Fig. 2.7, if every complexity streams have one unit of complexity, we can calculate the influence index for station j, $j = 1, 2, \ldots, n$ by simply counting the number of streams:

$$I_j = \frac{\text{\# of Outgoing Complexity Streams}}{\text{\# of Incoming Complexity Streams}} = \frac{n-j}{j} \tag{2.28}$$

Obviously,

- $I_1 = n - 1$, the first station *potentially* has the maximal influence on the others;
- $I_n = 0$, the last station has no influence on the others.

Thus, in such a sequential manufacturing process, the influence index is monotonically decreasing with respect to j. Hence we can conclude that operations at the later stations become vulnerable to be affected by the variants assembled at the previous ones. Therefore, by wisely assigning assembly tasks (i.e., the functional features) onto stations, it is possible to prevent complexity streams from propagating. One of the intuitive approaches is to assign features with more variants to the stations of smaller influence index (downstream stations), and vise versa. In this aspect, the proposed complexity model implies the principle of "delayed differentiation", which already has become a common practice in industry [30].

However, by Eq. (2.27), our model suggests that it is not sufficient by looking at the number of variants and the position where they are deployed according to the "delayed differentiation" principle. The evaluation of the impact of product variety on manufacturing complexity should also take into account the process flexibility built in the system. For instance, if all the variants from the upstream could be treated by the same flexible tools, common fixture, and shared assembly procedures in the downstream, variants can be introduced in the upstream without increasing system complexity. In this case, all the Δ-matrices for transfer complexity become column vectors with all ones in the column, indicating common process requirements for the feature variants in the product family. As a result, the transfer complexity vanishes.

Since different configurations have profound impact on the performance of the system [28], selecting an assembly system configuration other than a pure serial line may help reduce complexity. For instance, using parallel workstations at the later stages of a mixed-model assembly process reduces the number of choices on these stations if we can wisely route the variants at the joint of the ramified paths, see Fig. 2.9. However, balancing these types of manufacturing systems will be a challenge since the system configuration is no longer serial [26]. A novel method for task-machine assignment and system balancing needs to be developed to minimize complexity while maintaining manufacturing system efficiency.

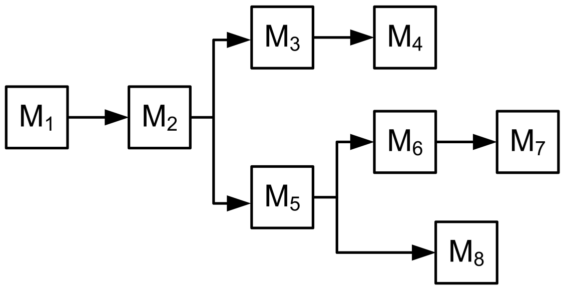

Figure 2.9: Possible configurations for mixed-model assembly systems. M_i's are machines in the system [26].

2.4.3 Assembly Sequence Planning to Minimize Complexity

Assembly sequence planning is an important task in assembly system design. It is to determine the order of assembly tasks to be performed. Since the assembly sequence determines the directions in which complexity flows, see Fig. 2.10, proper assembly sequence planning can reduce complexity.

Generally, suppose we have a product with n assembly tasks, and the tasks are to be carried out sequentially in an order subject to precedence constraints. By applying the

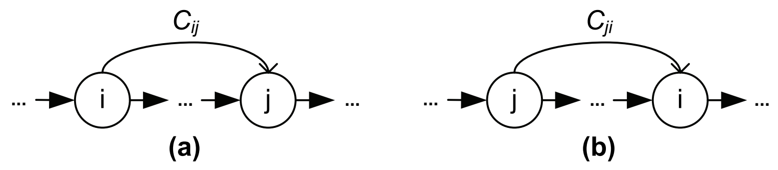

Figure 2.10: Differences in transfer complexity values for different assembly sequences: (a) Task i precedes j, which results in C_{ij}; (b) Task j precedes i, which results in C_{ji}, while C_{ij} and C_{ji} are not equal.

complexity model, we assume that the transfer complexity can be found between every two assembly tasks. Since only one of the two transfer complexity values in Fig. 2.10 is effective (because only the upstream task/station has influence on the downstream ones) for one particular assembly sequence, an optimization problem can be formulated to minimize the system complexity by finding an optimal assembly sequence while satisfying the precedence constraints. Chapter 3 discuss a detailed algorithm to solve the sequence planning problem.

2.4.4 Build Sequence Scheduling to Minimize Complexity

Build sequence scheduling is an important task in assembly system operation. It is to determine the order of products being built during the production. Since the build sequence determines the sequential dependencies between the choices, and the sequential dependencies reduce the degree of uncertainty, proper build sequence can help to reduce complexity.

In order to take into account the sequential dependencies, we need to extend the model discussed in this chapter to *non-i.i.d.* sequences. By the extended model, we are able to show that "conditioning reduces entropy", and the conditional entropy with sequential dependencies is generally smaller than that of the independent cases. Therefore, sequential dependencies provide additional information to the operators and help them to make choices. Chapter 4 discuss a detailed model to discuss the sequence scheduling problem.

2.5 Summary

The chapter proposes a measure of complexity based on the choices that the operator has to make at the station level. The measure incorporates both product mix and process information. Moreover, Stream of Complexity model is developed for the propagation of complexity at the system level. The significance of this research includes: (i) mathematical models that reveal the mechanisms that contribute to complexity and its propagation in multi-stage mixed-model assembly systems; (ii) understanding of the impact of manufacturing system complexity on performance; and (iii) guidelines for managing complexity in designing mixed-model assembly systems to optimize performance.

CHAPTER 3

Assembly Sequence Planning to Minimize Complexity

Abstract

Sequence planning is an important problem in assembly system design. It is to determine the order of assembly tasks to be performed sequentially. Significant research has been done to find good sequences based on various criteria, such as process time, investment cost, and product quality. This chapter [1] discusses the selection of optimal sequences based on the complexity measure and models developed in the previous chapter. According to the complexity models developed, assembly sequence determines the directions in which complexity flows. Thus proper assembly sequence planning can reduce complexity. However, due to the difficulty of handling the directions of complexity flows in optimization, a transformed network flow model is formulated and solved based on dynamic programming. Methodologies developed in this chapter extend the previous work on modeling complexity, and provide solution strategies for assembly sequence planning to minimize complexity.

3.1 Introduction

As an important step in assembly system design, *sequence planning* or sequence analysis, is to determine which assembly task should be done first, which should be done later. Proper determination of the sequence may help balance the line, reduce equipment investment, and ensure better product quality. Significant research has been done to find effective methodologies in search of good sequences [8].

The process of assembly sequence planning (ASP) begins with the representation of an assembled product, for example, by a graph or adjacency matrix. One type of graph, *liaison graph*, was first introduced by Bourjault in [5] to establish the relationships among component parts in an assembly. The liaison graph is a graphical network in which nodes

[1]Part of work in this chapter has appeared on [55]: X. Zhu, S. J. Hu, Y. Koren, S. P. Marin, and N. Huang. Sequence planning to minimize complexity in assembly mixed-model lines. In *Proceedings of the 2007 IEEE International Symposium on Assembly and Manufacturing*, 2007.

represent parts and lines (or arcs) between nodes represent liaisons. Each liaison represents an assembly feature where two parts join. The process of realizing such a liaison or liaisons is referred to as an *assembly task*. Hence, the problem of ASP is to find a proper order of realizing the liaisons, i.e., the sequence of tasks needed to completely assemble the product.

Another assembly representation is the precedence graph. A precedence graph is a network representation of precedence relations among assembly tasks. A sample graph is shown in Fig. 3.1. In the graph, nodes represent tasks, and there exists an arc (i, j) if task i is an *immediate predecessor* of task j.

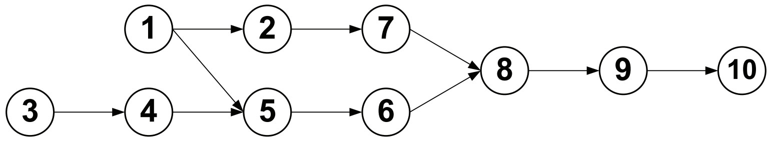

Figure 3.1: Precedence Graph of a Ten-Task Assembly (From [43], pp.5)

An immediate predecessor is defined as follows [37]. If $i \prec j$, then task i is known as a *predecessor* or *ancestor* of task j; and j is known as a *successor* or *descendent* of i. If i is a predecessor of j, and there is no other task which is a successor of i and a predecessor of j, then i is known as an immediate predecessor of j. For example, in Fig. 3.1, task 3 is a predecessor (but not an immediate predecessor) of task 5; task 5 has two immediate predecessors, tasks 1 and 4.

Two important concepts in a precedence graph are *unrelated* task pair and *transitivity*. A task may have more than one immediate predecessor and can be started as soon as all its immediate predecessors have been completed. By definition, if a task has two or more immediate predecessors, every pair of them must be *unrelated* (i.e., no precedence relationship) in the sense that neither of them is a predecessor of the other. Moreover, if i is a predecessor of j, and j is a predecessor of k, then obviously i is a predecessor of k. In notation, we write: $i \prec j, j \prec k \Rightarrow i \prec k$. This property of precedence relationships is called *transitivity*. By transitivity, therefore, we can determine the set of predecessors, or the set of successors of any task from the set of immediate predecessors of each task (i.e., from the precedence graph). We will use the above two concepts in the subsequent analysis.

Typically, one precedence graph corresponds to multiple, sometimes, a large number of sequences. With these candidate sequences, the best sequences are selected according to various criteria. One of such criteria is to balance the line. Scholl [43] presented a thorough treatment of assembly line balancing. Most of the efforts have been devoted to simple assembly line balancing problem with the restriction of one homogeneous product. Recently, Scholl and Becker [44] gave an up-to-date and comprehensive survey on the exact and heuristic solution procedures for simple assembly lines. In addition to balancing the line, other engineering knowledge could also be incorporated into the sequence selection process [8], for example, removing unstable subassembly states to avoid awkward assembly procedures and improve quality; eliminating refixturing & reorientation to reduce non-value added costs; imposing a subassembly to allow parallel processing; and so on.

Besides the attentions on a single product, researchers have also developed sequence planning methodologies for a group of similar products in a family. Gupta and Krishnan [19] showed that careful assembly sequence design for a product family helped to create genetic subassemblies which can reduce subassembly proliferation and the cost of offering product variety. Rekiek et al. [42] and Lit et al. [31] developed an integrated approach for designing product family (including assembly sequences) and the mixed-model assembly system at the same time so that multiple products could be assembled on the same line. Becker and Scholl [1] argued that mixed-model line balancing problems were often connected to sequencing problems in which one has to decide on the sequences of assembling the model units. For example, Yano and Rachamadugu [53] developed sequencing algorithms to minimize work overload caused by the differences in processing times among different models. Merengo et al. [34] proposed a concept of *horizontal balancing* to smooth the station times of the different models in balancing the line. Vilarinho and Simaria [49] considered balancing mixed-model assembly lines with parallel workstations and zoning constraints and developed a two-stage procedure to minimize the number of workstations for a given cycle time while at the same time balance the workloads between and within workstations.

In this chapter, we discuss sequence planning to reduce manufacturing complexity in manual, mixed-model assembly lines. The mixed-model assembly lines are widely used in various industries (such as appliances, automotive, and aerospace) to handle more than one product models. This flexibility helps share expensive equipment investments and absorb uncertain demand fluctuations. However, the mixed-model assembly process may become very *complex* when the number of product variants is high. This is because, during production, numerous small production steps need to be coordinated as many parts (e.g., for automobiles, more than three thousand different parts) are used for products with different options. Furthermore, only limited automation is economically available, and a significant portion of the process still relies on manual operations. Thus, the assembly process is subject to unpredictable human errors and performance (quality and productivity) degradations due to complexity. In order to better understand the manufacturing complexity and its impact on performance, a quantitative model has been developed in the previous chapter for evaluating complexity. The model is based on choices an operator has to make at a station, and complexity is measured as the uncertainty of making the choices. The details about the complexity measure and model will be reviewed in the next section. According to the complexity model developed, assembly sequence determines the directions in which complexity flows and thus proper assembly sequence planning can reduce complexity.

The objective of this chapter is to develop methodologies of finding the optimal assembly sequences to minimize system complexity. The chapter is structured as follows. Section 3.2 provides background information on the complexity measure and model needed in this chapter, and shows the opportunity of minimizing complexity by assembly sequence planning. Section 3.3 discusses the problem formulation and the preliminary attempts to solve the problem based on an integer program (IP). Due to the difficulties of handling constraints in the IP, Section 3.4 presents a network flow program formulation, which transforms the original problem into a manageable traveling salesman problem with precedence constraints represented by an extended precedence graph. During the transformation, the information from the precedence constraints are utilized to simplify the problem before actually solving the original optimization problem. Then, procedures are developed to solve the transformed problem using dynamic programming. Section 3.5 demonstrates numerical examples for the ten-task case study shown in Fig. 3.1. Finally, Section 3.6 summarizes the chapter.

3.2 Complexity Model for Sequence Planning

In this section, we review the complexity model developed for mixed-model assembly lines in Chapter 2, and demonstrates the opportunity of minimizing complexity by assembly sequence planning. The model considers the product variety induced manufacturing complexity in manual assembly lines where operators have to make choices according to the variants of parts, tools, fixtures, and assembly procedures.

A complexity measure called "Operator Choice Complexity" (OCC) was proposed to quantify the uncertainty in making the choices. The OCC takes an analytical form as an information-theoretic entropy measure of the average randomness (uncertainty) in a choice process. It is assumed that the more certain the operator is about what to choose in the upcoming assembly task, the less the complexity is, and the less chance the operator would make mistakes. Reducing the complexity may help to improve assembly system performance.

In fact, the definition of OCC is also similar to that of the cognitive measure of human performance in the Hicks-Hyman Law [21, 23], which has wide applications in various manual assembly operations [18, 4]. In a more general setting, Sivadasan et al. [46] defined entropy as the expected amount of information required to describe the state of the system. By using the entropy definition, they developed measures of the operational complexity for supplier-customer systems. The complexity measure takes into considerations of the uncertainty associated with managing the dynamic variations, in time or quantity, across information and material flows at the supplier-customer systems.

3.2.1 Measure of Complexity

In Eq. (2.7), the measure of complexity is defined as a linear function of the entropy of a choice process. The choice process consists of a sequence of random choices with respect to time. The choices are then represented by random variables, each of which represents choosing one of the possible alternatives from a choice set.

In fact, the choice process can be considered as a discrete time discrete state stochastic process $X' = \{X_t, t = 1, 2, \ldots\}$, on the state space (the choice set) $\{1, 2, \ldots, M\}$, where t is the index of discrete time period, M is the total number of possible alternatives which could be chosen during each period.

Based on the above notation, and if the random sequence is independent, identically distributed (*i.i.d.*), the entropy of the choice process can be calculated by the following H function:

$$H(X) = H(p_1, p_2, \ldots, p_M) = -C \cdot \sum_{m=1}^{M} p_m \cdot \log p_m \tag{3.1}$$

where C is a constant depending on the base of the logarithm function chosen, if $\log_2$ is selected, $C = 1$ and the unit of entropy is *bit*; $p_m \triangleq \mathbf{P}(X = m)$, for $m = 1, 2, \ldots, M$, which is the probability of choosing the m^{th} alternative; $(p_1, p_2, \ldots, p_M)$ determines the probability mass function of X, is also known as the *mix ratio* of the (component) variants associated with the choice process. The mix ratio reflects the variety information in the mixed-model assembly processes.

With known entropy values of various choice processes at a station, the complexity for the station is calculated by the summation of all the entropy values associated with the choices. The choices are related to variety, and determining where the variety are added on the assembly line is key to the complexity calculation. We will show how the complexity calculation is accomplished in the context of complexity propagation and system level model.

3.2.2 Complexity Propagation

Base on the idea that variety causes complexity in a multi-stage manufacturing system, we define two types of complexity for each station:

- **Feed complexity:** Choice complexity caused by the feature variants added at the current station.
- **Transfer complexity:** Choice complexity of the current station caused by the feature variants previously added at an upstream station.

The propagation scheme of the two types of complexity is depicted in Fig. 3.2, where, for station j, the feed complexity is denoted as C_{jj} (with two identical subscripts), and the transfer complexity is denoted as C_{ij} (with two distinct subscripts to represent the complexity of station j caused by the variants added at an upstream station i). Transfer complexity exists because the feature variant added on the upstream station i has been carried down to station j, and may affect the process of realizing other features at station j. The effects can cause, for example, accessory part selections, tool changeovers, fixture conversions, or assembly procedure changes. By definition, the transfer complexity can *only* flow from upstream to downstream, but not in the opposite direction. In contrast, the feed complexity can *only* be added at the current station with no "transfer" behavior.

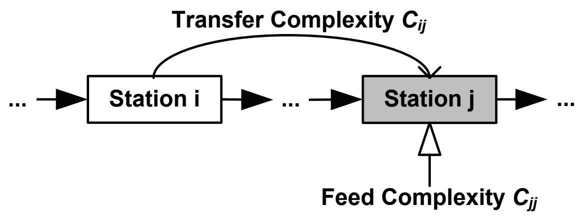

Figure 3.2: Types of Complexity

3.2.3 System Level "Stream of Complexity" Model

With the two types of complexity defined, we can derive a system level complexity model to characterize the interactions among multiple sequentially arranged stations. Consider an assembly line having n workstations shown in Fig. 3.3. The stations are numbered 1 through n sequentially from the beginning of the line to the end. The mix ratio, i.e., the percentages of component variants added at each station, is known. Using Eq. (3.1), we can obtain the entropy H for the variants at each station according to their mix ratios (by assuming an *i.i.d.* component build sequence).

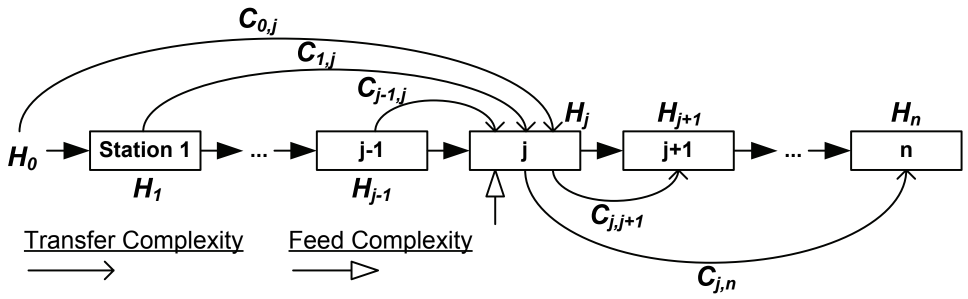

Figure 3.3: Propagation of Complexity at the System Level in a Multi-Stage Assembly System

In Fig. 3.3, each directed arc stands for a stream of transfer complexity, C_{ij}, flowing from station i to j (C_{ij} can be zero). Hence, we also call this model as the "Stream of Complexity" model. The total complexity at a station is simply the sum of the feed complexity at the station and all transfer complexity from the upstream ones. For station j, the total complexity is:

$$C_j = C_{jj} + \sum_{\forall i: i \prec j} C_{ij} \tag{3.2}$$

According to the definition of transfer complexity, if component variants added at station i cause choices during the assembly operations at station j, we have:

$$C_{ij} = a_{ij} \cdot H_i, \text{ for } i = 0, 1, 2, \ldots, n-1; j = 1, 2, \ldots, n \tag{3.3}$$

where,

H_i	–	Entropy of component variants added at station i;
H_0	–	Entropy of variants of the base part;
a_{ij}	–	Coefficient of interaction between assembly operations at station j and variants added at station i, i.e.,

$$a_{ij} = \begin{cases} 1 & \text{Variants added at station } i \text{ have impacts on station } j, \text{ and } i < j, \text{ assuming choice difficulties are identical.} \\ 0 & \text{Otherwise} \end{cases}$$

Therefore, the values of C_{ij}'s are determined by the following two steps.

Step 1: Determine the value of H_i, which depends on the mix ratio of component variants added at station i. As we have mentioned earlier, for component variants with an *i.i.d.* build sequence, Eq. (3.1) can be used to calculate the H_i levels. In other words, H_i is determined by the assignment of the assembly task at station i.

Step 2: Determine the value of a_{ij}, which depends on the relationship between the component variants added at station i and the process requirements at station j, which, in turn, is related to the assignment of assembly task at station j.

The two-step procedure of determining C_{ij} makes it difficult to formulate an optimization problem to minimize total system complexity since different sequences will result in different C_{ij}'s. In addition, the number of candidate sequences can be quite large and it is computationally prohibitive to exhaustively evaluate all of them in finding the one with minimum system complexity. Therefore, methodologies and algorithms are needed to *efficiently* search for the optimal sequences.

To begin with, we set up a sequence planning problem as follows. We consider an assembly system, having n assembly tasks, denoted as 1 to n. Tasks are to be arranged sequentially in an order subject to precedence constraints, such as the constraints expressed by the precedence graph in Fig. 3.1, where $n = 10$. Additionally, to make the problem comparable to the original problem in Fig. 3.3, we assume each task corresponds to one and only one station, and vice versa.

According to the multi-stage complexity model in Fig. 3.3, transfer complexity may be found between every two tasks. The complexity becomes *effective* only from upstream tasks to downstream ones. For example, in Fig. 3.4, when task i precedes task j (denoted as $i \prec j$), there is transfer complexity flowing from tasks i to j (denoted as a directed arc from nodes i to j). In other words, since task j is performed after task i, it is possible that the assembly process for task j requires choices of parts/tools/fixtures/assembly procedures (to be made) according to the variants previously installed by task i. Using the notations of transfer complexity in Fig. 3.2, we know that the amount of complexity incurred in the above scenario is C_{ij}. Alternatively, it is also possible for transfer complexity, C_{ji}, to exist and flow from tasks j to i if $j \prec i$. Obviously, only one of the two scenarios will take place at one time. Thus, for each assembly sequence, although transfer complexity can exist in either directions, one and only one of the values in the pair (C_{ij}, C_{ji}) is effective. We define this value as the "effective complexity"

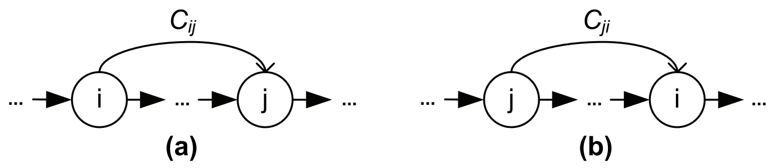

Figure 3.4: Transfer Complexity between Two Assembly Tasks i and j, (a) C_{ij} if $i \prec j$, (b) C_{ji} if $j \prec i$

Hence, the assembly sequence determines the directions in which transfer complexity flows, and hence the total system complexity. In general, C_{ij} and C_{ji} are not equal. Therefore, an assembly sequence planning (ASP) problem needs to be formulated to find the optimal sequences, which result in the minimum total system complexity. In the following, we discuss the formulation of the ASP problem.

3.3 Problem Formulation

In this section, we discuss the assumption and formulation of the sequence planning problem proposed above. In addition, an integer program has been formulated as the first attempt to solve the problem. Although the attempt was not successful, it suggested the further discussions in the next section.

3.3.1 Assumption: Position Independent Choice Complexity

For simplicity and practical reasons, we assume *position independence* for transfer complexity. That is, the values of C_{ij} and C_{ji} depend *solely* on the relative positions of the two tasks i and j, but not the tasks in between, nor their absolute positions in the assembly sequence. Although the assumption seems restrictive, it is applicable for assembling customized products with highly modularized components such as the final assembly process of automobiles, home appliances, and electronics. Because of the simplification, the computational effort to find the optimal solution is greatly reduced and the problem becomes manageable as well.

Under the above assumption, we are able to determine all the values of transfer complexity between every pair of assembly tasks by Eq. (3.3). For the example of the ten-task assembly system described in Fig. 3.1, a node-node cost array can be formed as shown in Fig. 3.5 to record all the transfer complexity values, where C_{ij} corresponds to the cell (i, j) at row i and column j. For each feasible assembly sequence, the system complexity then can be found by summing all the complexity values that exist, i.e., $\sum C_{ij}$, where $(i, j) \in \{(i, j)|i \prec j\}$. If no precedence constraint is present, there exist $n!$ feasible assembly sequences. It is obvious that when the constraints are less stringent, the number of feasible sequences can become quite large.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	■	C_{12}	C_{13}	C_{14}	C_{15}	C_{16}	C_{17}	C_{18}	C_{19}	$C_{1,10}$
2	C_{21}	■	C_{23}	C_{24}	C_{25}	C_{26}	C_{27}	C_{28}	C_{29}	$C_{2,10}$
3	C_{31}	C_{32}	■	C_{34}	C_{35}	C_{36}	C_{37}	C_{38}	C_{39}	$C_{3,10}$
4	C_{41}	C_{42}	C_{43}	■	C_{45}	C_{46}	C_{47}	C_{48}	C_{49}	$C_{4,10}$
5	C_{51}	C_{52}	C_{53}	C_{54}	■	C_{56}	C_{57}	C_{58}	C_{59}	$C_{5,10}$
6	C_{61}	C_{62}	C_{63}	C_{64}	C_{65}	■	C_{67}	C_{68}	C_{69}	$C_{6,10}$
7	C_{71}	C_{72}	C_{73}	C_{74}	C_{75}	C_{76}	■	C_{78}	C_{79}	$C_{7,10}$
8	C_{81}	C_{82}	C_{83}	C_{84}	C_{85}	C_{86}	C_{87}	■	C_{89}	$C_{8,10}$
9	C_{91}	C_{92}	C_{93}	C_{94}	C_{95}	C_{96}	C_{97}	C_{98}	■	$C_{9,10}$
10	$C_{10,1}$	$C_{10,2}$	$C_{10,3}$	$C_{10,4}$	$C_{10,5}$	$C_{10,6}$	$C_{10,7}$	$C_{10,8}$	$C_{10,9}$	■

Figure 3.5: Transfer Complexity Values between any Two of the Ten Tasks

Among all the feasible assembly sequences, our objective is to find the sequence with minimum system complexity, which we define to be the optimal sequence. Notice that, the feed complexity C_{jj} in Eq. (3.2) is associated with the feature variants added at the current station, thus it does not change with sequences. As a result, for simplification, we could possibly set $C_{jj} = 0$. This simplification does not affect the procedure of finding optimal sequences, and it only changes the total system complexity by a constant. Put differently, in formulating the assembly sequence planning (ASP), only transfer complexity matters.

Therefore, the optimization problem for the ASP can be written as follows.

Program 1:

Minimize:

System (transfer) complexity $Z = \sum_{(i,j)\in\{(i,j)|i\prec j\}} C_{ij}$

With respect to: Assembly sequence

Subject to: Precedence constraints

3.3.2 Problems with Integer Program Formulation

Because of the imposed precedence constraints, Program 1 can be viewed equivalently as finding a node covering chain (or tour) in the precedence graph of Fig. 3.1 with minimum sum of effective complexity flowing from nodes i to j, where $i \prec j$. Accordingly, an integer program can be formulated as follows.

Program 2:

Decision variables:

$$x_{ij} = \begin{cases} 1 & \text{Task } i \text{ is assigned prior to task } j \\ 0 & \text{Otherwise} \end{cases}$$

Minimize $Z(\mathbf{x}) = \sum_{\forall i,j} C_{ij} x_{ij}$

Subject to:

(i) $x_{ij} + x_{ji} = 1, \forall i, j$

(ii) $x_{ij} \in \{0, 1\}$

(iii) If $(i_1, i_2), i_2, (i_2, i_3), \ldots, i_{n-1}, (i_{n-1}, i_n)$ forms a tour in the precedence graph, then

$$\begin{cases} x_{i_1,i_2} = x_{i_1,i_3} = \ldots = x_{i_1,i_{n-1}} = x_{i_1,i_n} = 1 \\ \qquad x_{i_2,i_3} = \ldots = x_{i_2,i_{n-1}} = x_{i_2,i_n} = 1 \\ \ldots \qquad \ldots \qquad \ldots \qquad \ldots \\ \qquad\qquad\qquad x_{i_{n-1},i_n} = 1 \end{cases}$$

Constraints (i) and (ii) ensure one and only one task is assigned to a station, constraint (iii) determines the directions of complexity flows. Because of the difficulties in handling constraint (iii) of Program 2, the ASP problem in the above formulation is difficult to solve directly. Here, we use the methods from network flow modeling to simplify the above problem. That is, we add all the complexity values as flows on the arcs of the precedence graph, such as the one in Fig. 3.1, then solve a minimum flow problem accordingly.

3.4 A Network Flow Program Formulation

To begin with, we assume the values of transfer complexity between every pair of assembly tasks are known. They are recorded in the array shown in Fig. 3.5. However, due to precedence constraints, some of the cells in the array are *inadmissible*, i.e., if it is not possible for task i to precede task j, we denote the cell (i, j) as inadmissible, and assign it an ∞ complexity value. Finding all these inadmissible cells and purging them can greatly simplify the original problem. Thus, a procedure is developed below.

3.4.1 Purging Inadmissible Cells

It takes three steps to find and purge inadmissible cells using the transitivity property of precedence mentioned in Section 3.1.

Step 1: For row i, mark with X in the j^{th} column for all j, where $j \in J_i = \{j | i \prec j\}$ (J_i is the set of nodes with a precedence relationship with node i, including implicit transitivity relationships).

Step 2: For row i, $\{i, j\}$ is a pair of unrelated elements if the j^{th} row is unmarked. The meaning of unrelated pair is that either task i could be assigned before task j, or vice versa. The incurred complexity of these two scenarios is C_{ij} or C_{ji} respectively, and their cell locations are found and marked with Y. Fig. 3.6 shows the resulting array after Steps 1 and 2.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	■	X	Y	Y	X	X	X	X	X	X
2		■	Y	Y	Y	Y	X	X	X	X
3	Y	Y	■	X	X	X	Y	X	X	X
4	Y	Y		■	X	X	Y	X	X	X
5		Y			■	X	Y	X	X	X
6		Y				■	Y	X	X	X
7			Y	Y	Y	Y	■	X	X	X
8								■	X	X
9									■	X
10										■

Figure 3.6: Resulting Array after Steps 1 and 2

Step 3: By now, all the unmarked cells are inadmissible cells. Mark them with ∞ and restore the corresponding cost coefficients with appropriate subscripts to the rest of cells, and *shadow* the cells which were marked with X for later use.

3.4.2 Equivalent Network Flow Model

It is observed that the complexity values in the *non-shadowed* cells are formed in pairs; in each one of the feasible solutions, one and only one of them are *effective* and counted in the total system complexity cost function. Conversely, the complexity values in the *shadowed* cells are imposed by precedence constraints either explicitly or implicitly; therefore, all of them are by all means counted in every feasible solutions.

The above observation implies one of the ways to simplify the complexity cost array without changing the original problem. That is, we simply set all the *shadowed* cells to zero, see Fig. 3.7. Then the only change to the objective function of the original optimization problem in Program 2 is by a constant, which does not affect the optimal solutions. In fact, the equivalent argument for the simplification is to set complexity from i to j to zero if the precedence constraint requires i to precede j either explicitly or implicitly (through transitivity), i.e., $C_{ij} = 0$ for $i \prec j$.

The cells with denoted complexity cost in Fig. 3.7 are the ones forming an unrelated pair. On the original precedence graph as shown in Fig. 3.1, directed, dotted arcs between i and j are drawn in both directions if $\{i, j\}$ is such an unrelated pair, and flow values

	1	2	3	4	5	6	7	8	9	10
1	■	0	C_{13}	C_{14}	0	0	0	0	0	0
2	∞	■	C_{23}	C_{24}	C_{25}	C_{26}	0	0	0	0
3	C_{31}	C_{32}	■	0	0	0	C_{37}	0	0	0
4	C_{41}	C_{42}	∞	■	0	0	C_{47}	0	0	0
5	∞	C_{52}	∞	∞	■	0	C_{57}	0	0	0
6	∞	C_{62}	∞	∞	∞	■	C_{67}	0	0	0
7	∞	∞	C_{73}	C_{74}	C_{75}	C_{76}	■	0	0	0
8	∞	∞	∞	∞	∞	∞	∞	■	0	0
9	∞	∞	∞	∞	∞	∞	∞	∞	■	0
10	∞	∞	∞	∞	∞	∞	∞	∞	∞	■

Figure 3.7: Reduced Complexity Cost Array

C_{ij}, C_{ji} are assigned to the associated arcs respectively. An extended precedence graph is obtained as shown in Fig. 3.8. Notice that the flows on the solid arcs are the complexity values between two tasks with explicit precedence relationships, which are *all zero* due to the simplification made in the previous paragraph. However the flows on the dotted arcs are the complexity values between every *unrelated* pair of tasks, which may take on *non-zero* values.

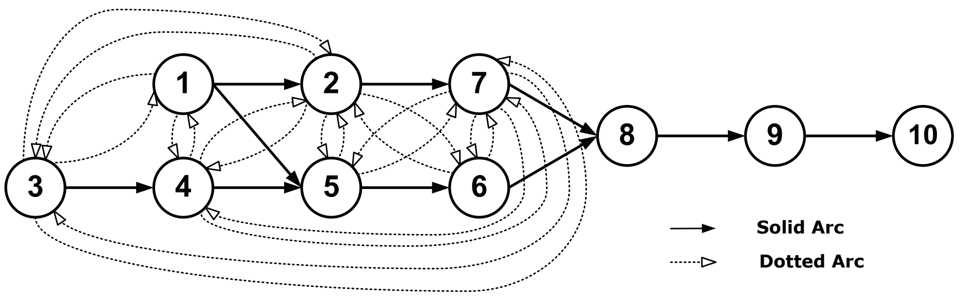

Figure 3.8: Extended Precedence Graph

The graph in Fig. 3.8 is now an equivalent network flow model where the objective is to find a tour which has the least flow cost. This is because each of the feasible solutions corresponds to a path satisfying the following properties in the extended precedence graph.

1. The path is directed, i.e., the travel must follow the direction of the arrows;
2. Every node must be visited once and only once;
3. The path must have the solid arcs directed forward.

Properties 1 and 2 are required simply due to the definition of a precedence graph. In other words, the path is *Hamiltonian*. Property 3 is imposed because the solid arcs are the explicit precedence relationships (ten in total for the example) which must be satisfied. Once these precedence are satisfied, all the implicit precedence relations hold automatically, i.e., the precedence constraints specified by the original precedence graph are satisfied. For example, by inspection, one of the feasible solutions is 1-2-3-4-5-6-7-8-9-10. By stretching the path from the extended precedence graph, we find that all the solid arcs are directed forward, see Fig. 3.9(a). For comparison, an infeasible solution violating property 3 is illustrated in Fig. 3.9(b): a path satisfying properties 1 and 2 is taken with the sequence of nodes being 1-4-7-3-2-5-6-8-9-10. By stretching the path again, the solid arcs of $(2, 7)$ and $(3, 4)$ are found to be in the opposite direction. Thus property 3, i.e., the constraint in the original precedence graph, has been violated.

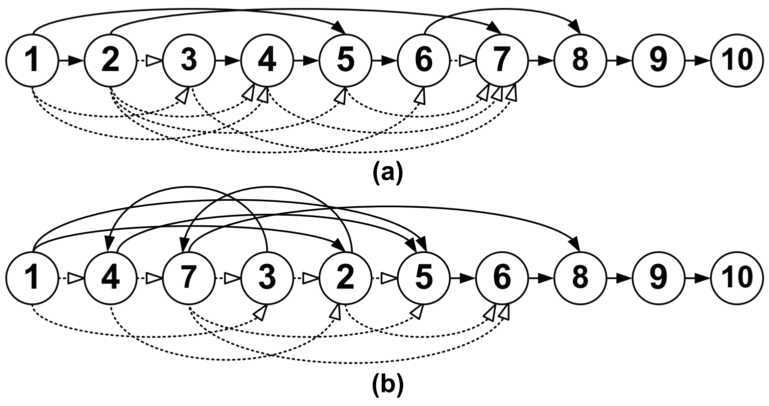

Figure 3.9: Stretched Path of (a) a Feasible Solution, (b) an Infeasible Solution

Once the constraints are satisfied, the objective value is computed by counting all the effective flows. The effective flow is defined as the flow on the dotted arc whose direction is the same as that of the path. In fact, the effective flow represents the effective transfer complexity in the unrelated pair. Take the example in Fig. 3.9(a), the effective flows are C_{13}, C_{14}, C_{23}, C_{24}, C_{25}, C_{26}, C_{37}, C_{47}, C_{57}, and C_{67}. Therefore, the objective value of the solution is,

$$\begin{aligned} Z = {} & C_{13} + C_{14} + C_{23} + C_{24} + C_{25} + C_{26} \\ & + C_{37} + C_{47} + C_{57} + C_{67} + \text{constant} \end{aligned} \tag{3.4}$$

The reason for adding the constant in the above expression follows from the arguments (of simplification) made for the use of the reduced complexity cost array in Fig. 3.7.

Similarly, in Fig. 3.9(b), the effective flows are C_{13}, C_{14}, C_{42}, C_{47}, C_{73}, C_{75}, C_{76}, C_{32}, C_{25}, and C_{26}, and the objective value of the infeasible solution (not achievable) is,

$$\begin{aligned} Z = {} & C_{13} + C_{14} + C_{42} + C_{47} + C_{73} + C_{75} \\ & + C_{76} + C_{32} + C_{25} + C_{26} + \text{constant} \end{aligned} \tag{3.5}$$

In conclusion, by combining properties 1 and 2 with property 3, the equivalent network flow model transforms the original formulation (Program 2) into a problem of finding a Hamiltonian tour in the *extended precedence graph*, and at the same time, subject to the constraints imposed by the *original precedence graph.* The problem is then similar to what is widely known as the traveling salesman problem with precedence constraints (TSP-PC) [3], which can be solved using Dynamic Programming (DP) with some modifications.

In addition, by using the transformation, we gain the advantage of greatly reducing the number of non-zero, finite cells in the complexity cost array. For the purpose of assembly sequence planning, this reduction significantly simplifies the work of evaluating the transfer complexity values by the procedures discussed about Eq. (3.1). For the ten-task example, only the transfer complexity of the pairs $\{1,3\}$, $\{1,4\}$, $\{2,3\}$, $\{2,4\}$, $\{2,5\}$, $\{2,6\}$, $\{3,7\}$, $\{4,7\}$, $\{5,7\}$, and $\{6,7\}$ needs to be evaluated. Moreover, as we shall see shortly, the transformation also provides the means of finding state transition costs in DP, which makes our problem comparable to the classical TSP-PC with C_{ij} being the arc length.

3.4.3 Solution Procedures by Dynamic Programming

Here we reformulate our problem and restate it with the notations similar to that of the classical TSP-PC. First, we append a "dummy" node 0 that has no transfer complexity from or to the other nodes, and connect it with the starting nodes (ending nodes) with forward (backward) arcs according to the precedence constraints. The starting node (ending node) is defined as the node having no predecessor (successor). In our example, the starting nodes are 1 and 3, and the ending node is 10. Thus, we add node 0 with forward arcs $(0,1)$, $(0,3)$, and backward arc $(10,0)$ to the graph in Fig. 3.8, and set $C_{01} = C_{03} = C_{10,0} = 0$. Next, let the extended precedence graph be $G = (\mathcal{N}, \mathcal{A})$, which is a directed graph, where $\mathcal{N} = \{0, 1, 2, \ldots, n\}$ is the node set, $\mathcal{A}$ is the arc set. $C_{ij} \geq 0, (i,j) \in \mathcal{A}$ is the complexity incurred if task i precedes task j, i.e., node i is visited prior to j (denoted as $i \prec j$). By convention, let $C_{ii} = \infty, \forall i \in \mathcal{N}$ to eliminate self-loops. (Notice that, setting $C_{ii} = \infty$ has different purpose from setting $C_{jj} = 0$ in Section 3.3.1. They are not contradictory.) Finally, for each node $i \in \mathcal{N}$, the precedence relationships defined by the original precedence graph (or, equivalently the solid arcs in Fig. 3.8) can be expressed by means of a set of nodes ($\Pi_i^{-1} \subset \mathcal{N}$) that must be visited *before* node i, or a set of nodes ($\Pi_i \subset \mathcal{N}$) that must be visited *after* node i.

The ASP problem is then cast as one of the variants of the classical TSP-PC in [3] as to find a Hamiltonian tour starting from node 0, visiting every node in $\Pi_i \subset \mathcal{N}$ before entering node i ($i \in \{1, 2, \ldots, n\}$), and finally returning to node 0. The objective is to find a feasible tour that minimizes the sum of the complexity incurred. However, it is important to note that instead of calculating the sum of the costs on its arcs (along the traveling path) as in the classical problem, we compute the sum of C_{ij}'s, where $(i,j) \in \mathcal{A}$ and $i \prec j$. This presents difficulties in handling the state transition cost in developing DP procedures. For that, we need to ensure the following two conditions:

- **Condition 1:** The decision space for state transition, i.e., going from a state (say $State^*$) to another state depends on $State^*$, not the path coming into $State^*$.
- **Condition 2:** The state transition cost depends on $State^*$, not the path coming into $State^*$.

We will show how to satisfy these conditions in the following the DP procedures.

Define state (S, i) as the state being at node i ($i \in S, S$ is the set of nodes visited so far) and visited every node in Π_j^{-1} before passing through node j ($\forall j \in S$), and further define the objective value function $f(S, i)$ to be the least complexity cost "determined" (explained later on the state transition cost structure) on a path starting at node i, legitimately visiting the rest of $(n + 1 - |S|)$ nodes, i.e., all the nodes in the set $\mathcal{N} \backslash S$, and finally finishing at the dummy node 0.

State transition takes place from state (S, i) to $(S \bigcup \{j\}, j)$ by visiting node j (where $j \in D(S)$) at the next step, where $D(S)$ is the decision space, consists of the set of nodes that can be visited after acquiring state (S, i).

Theorem 1: $D(S)$ is a function of solely S.

Proof: First, denote $Y_k = \{i : |\Pi_i^{-1}| + 1 \leq k \leq n + 1 - |\Pi_i|\}$ as the set of nodes that may stay in position k ($k \in \{1, 2, \dots, n + 1\}$) in any feasible tour. Next, note that, if $|S| = k$, then $j \in Y_k$, i.e., by definition, $D(S) = Y_k \backslash S$. Since Y_k is determined by the precedence graph $G = (\mathcal{N}, \mathcal{A})$, thus $D(S)$ relies only on S.

Put alternatively, $D(S)$ is determined by the set of the nodes we have visited not the node where we are at, nor the path coming into the node. Thus, Theorem 1 shows that Condition 1 has been satisfied.

As we have mentioned earlier, the state transition cost structure of our ASP problem is *quite* different from that of the classical TSP-PC, which causes the difficulty of solving the problem. However the equivalent transformation in the previous section gains us the insight to transfer the state transition cost to that of the classical TSP-PC as being the arc length from node i to node j.

Theorem 2: The complexity incurred by choosing to visit node j at state (S, i) is a function of the state and node j.

Proof: First of all, we notice that by choosing node j as the next node to be visited, we "determine" the complexity flowing from node j to all the other nodes in the set $\mathcal{N} \backslash (S \bigcup \{j\})$ but not in the opposite direction. The "determined" complexity flows are the finite C_{jk}'s with node $k \in \mathcal{N} \backslash (S \bigcup \{j\})$. To express explicitly, the transition cost from state (S, i) to state $(S \bigcup \{j\}, j)$ is $\sum_{\forall k \in K(S,j)} C_{jk}$, where $K(S, j) = \{\mathcal{N} \backslash (S \bigcup \{j\})\} \bigcap \{k | 0 \leq C_{jk} < \infty\}$. Therefore, the state transition cost depends only on state (S, i) and node j.

Put alternatively, the effective flows are "determined" by selecting finite values in the columns corresponding to the un-visited nodes, and in row j of the reduced complexity array in Fig. 3.7. Therefore, by Theorems 1 and 2, Condition 2 has been satisfied.

Corollary 3: The state transition cost from state (S, i) to $(S \bigcup \{j\}, j)$ is zero, if j is the only candidate decision in $D(S)$, i.e., $D(S) = \{j\}$.

Proof: Since $D(S) = \{j\}$, $\forall k \in K(S, j)$ we have strictly $j \prec k$, which is imposed by precedence constraints. Because of the simplification shown in Fig. 3.7 (where $C_{ij} = 0$ if $i \prec j$), we have $C_{jk} = 0$. Therefore, by Theorem 2, the state transition cost $\sum_{\forall k \in K(S,j)} C_{jk}$ is zero.

The result of Corollary 3 helps us to simplify the calculation of state transition costs in DP recursions.

Now the functional equation for the exact solution follows. Moreover, to fully utilize the easily-found decision space $D(S)$, the DP recursion is intentionally developed to be backwards.

Program 3:

$$f(S, i) = \begin{cases} \min_{j | j \in D(S)} \{\sum_{\forall k \in K(S,j)} C_{jk} + f(S \bigcup \{j\}, j)\}, \\ \quad \text{for } \{0\} \subseteq S \subset \mathcal{N}, i \in S; \text{for } S = \mathcal{N}, i \in S \backslash \{0\} \\ \\ 0, \text{ for } S = \mathcal{N}, i = 0 \end{cases}$$

Answer:$f(\{0\}, 0)$

The classical TSP-PC is known to be NP-hard. Using the DP, the computational complexity of the unconstrained TSP with n nodes is $O(n^2 2^n)$ [20]. For Program 3 with no precedence constraints, due to the unique structure in handling state transition costs, the computational complexity is higher than $O(n^2 2^n)$. The computational complexity can be evaluated as follows.

- The number of states in the DPN corresponding to Program 3 is at most $\sum_{k=1}^{n} k \binom{n}{k}$.
- Correspondingly, the number of arcs is at most is at most:$\sum_{k=2}^{n} k(k-1)\binom{n}{k} + n$.
- For each arc in the above, one has to obtain the state transition cost from state (S, i) to state $(S \bigcup \{j\}, j)$ by $\sum_{\forall k \in K(S,j)} C_{jk}$, where $K(S, j) = \{\mathcal{N} \backslash (S \bigcup \{j\})\} \bigcap \{k | 0 \leq C_{jk} <$

$\infty\}$. The calculations required are at most $(n-k)$ additions. Furthermore, due to the comparisons needed in $\min_{j|j\in D(S)}\{\sum_{\forall k\in K(S,j)} C_{jk} + f(S\bigcup\{j\},j)\}$, each arc corresponds to $(n-k)$ additions and 1 comparison.

Therefore, total number of additions or comparisons needed for Program 3 is at most:

$$\begin{aligned}&\textstyle\sum_{k=2}^{n}(k(k-1)\binom{n}{k}+n)\times(n-k+1)\\&\quad\le n\textstyle\sum_{k=2}^{n}k(k-1)\binom{n}{k}+n^2\\&\quad= n^2(n-1)2^{n-2}+n^2\end{aligned}$$

Thus, $O(n^3 2^n)$ can be accepted as an upper bound for the computational complexity of Program 3. Although this number grows exponentially with the number of nodes, it is still much lower than $O(n^2 n!)$, which is the computational complexity using the exhaustive search. To better comprehend the computational load in magnitude, we take the following example. If $n = 30$, $O(n^3 2^n)$ is equivalent to 2.9×10^{13} additions or comparisons, which equals to 4.03 hours on a 2 GHz CPU.

In practice, since the number of assembly tasks for sequence planning is moderate, which is around 30 stations for one of the line segments in a typical automobile plant, the exact algorithm based on DP is still applicable. In the long-term strategic planning, such as ASP, it is practically manageable to solve the problem in a reasonable amount of time, i.e., as a matter of several hours. Additionally, in most of the cases, there exist precedence constraints. As we have seen in Section 3.4, through the simplification and transformation, we are able to significantly reduce the problem size, and in turn the computational load, by utilizing the information from the precedence constraints. If further computational improvements are needed, heuristics in [3] or more recent algorithms based on ant colony optimization [11] can be investigated.

3.5 Numerical Examples

By continuing the ten-task example, we demonstrate the numerical results solved by Program 3. We examine the original precedence relationships (network of the solid arcs in Fig. 3.8), and by Theorem 1, we can find the decision space $D(S)$ for every feasible node set S, see Table 3.1.

Then the complete Dynamic Programming Network (DPN) is drawn in Fig. 3.10, where nodes represent states (refer to Table 3.1 for the details of the states), and arcs represent possible state transitions (refer to Table 3.2 for transition costs, where node in the first column (rows) is the starting (ending) node of the arc, and an ∞ value denotes no arc between the two nodes).

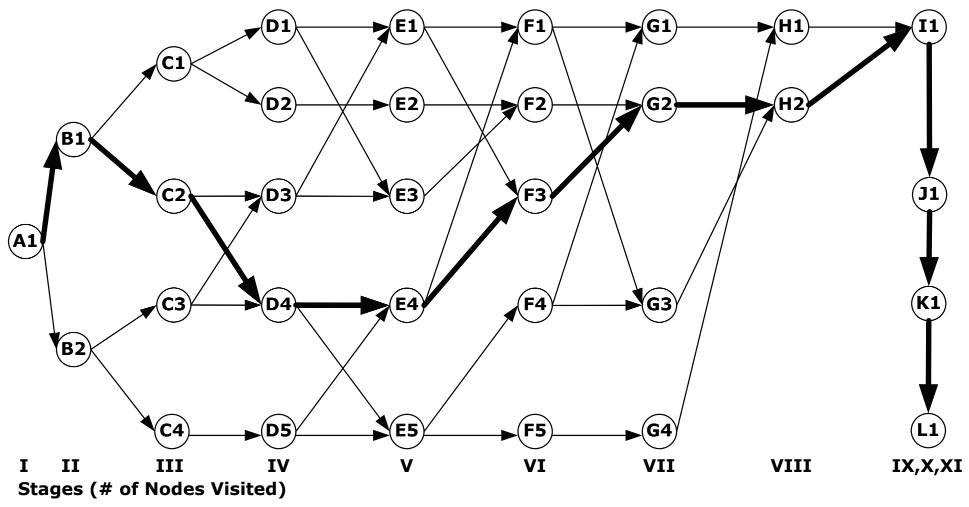

Figure 3.10: Complete DPN for the Ten-Task Example

3.5.1 Example 1

In this example, we demonstrate the effect of precedence constraints by using special numerical values for the cost array.

First, notice that in Fig. 3.9(b), we illustrate an infeasible solution with the sequence 1-4-7-3-2-5-6-8-9-10. The corresponding system complexity can be calculated by Eq. (3.5). In this example, we intentionally assign zeros to all C_{ij}'s in Eq. (3.5), and assign ones to the remaining C_{ij}'s. Put alternatively, in Fig. 3.7, we assign 0's to the cells $(1,3)$, $(1,4)$, $(4,2)$, $(4,7)$, $(7,3)$, $(7,5)$, $(7,6)$, $(3,2)$, $(2,5)$, $(2,6)$, and 1's to the cells $(2,3)$, $(2,4)$, $(3,1)$, $(3,7)$, $(4,1)$, $(5,2)$, $(5,7)$, $(6,2)$, $(6,7)$, $(7,4)$.

Based on this special setup, if no precedence constraints are imposed, we can easily conclude that system complexity (neglecting feed complexity) $Z = 0$ (with constant $= 0$), and the sequence 1-4-7-3-2-5-6-8-9-10 is the optimal. However, if the precedence constraints in Fig. 3.1 are imposed, the above optimal is no longer achievable.

As discussed in Section 3.4.2, one of the arbitrary feasible solution is the sequence 1-2-3-4-5-6-7-8-9-10. By Eq. (3.5), the corresponding system complexity is $Z = 5$ bits.

By Program 3, we find that the optimal sequence is 1-3-4-2-7-5-6-8-9-10, and the corresponding system complexity is $Z = 1$ bit. The optimal solution corresponds to a shortest path in the DPN, which is shown in Fig. 3.10 by the thick arcs.

To conclude the above comparisons, we can see that the resulting optimal solution gives a system complexity ($Z = 1$ bit) lower than that of the arbitrary feasible solution ($Z = 5$ bits) but higher than that of the infeasible solution ($Z = 0$). This comparison demonstrates that the optimization finds a optimal solution with higher objective value due to active precedence constraints, which verifies the effectiveness of Program 3. To verify, Table 3.3 lists all feasible solutions ($*$ denotes the optimal sequence).

Table 3.1: Label, State, and Corresponding Decision Space $D(S)$ for the Nodes of the DPN

Label	State	D(S)	Label	State	D(S)
A1	({0},0)	1,3	E5	({0,1,3,4,5},5	2,6
B1	({0,1},1)	2,3	F1	({0,1,2,3,4,5},5)	6,7
B2	({0,3},3)	1,4	F2	({0,1,2,3,4,7},4)	5
C1	({0,1,2},2)	3,7	F3	({0,1,2,3,4,7},7)	5
C2	({0,1,3},3)	2,4	F4	({0,1,2,3,4,5},2)	6,7
C3	({0,1,3},1)	2,4	F5	({0,1,3,4,5,6},6)	2
C4	({0,3,4},4)	1	G1	({0,1,2,3,4,5,6},6)	7
D1	({0,1,2,3},3)	4,7	G2	({0,1,2,3,4,5,7},5)	6
D2	({0,1,2,7},7)	3	G3	({0,1,2,3,4,5,7},7)	6
D3	({0,1,2,3},2)	4,7	G4	({0,1,2,3,4,5,6},2)	7
D4	({0,1,3,4},4)	2,5	H1	({0,1,2,3,4,5,6,7},7)	8
D5	({0,1,3,4},1)	2,5	H2	({0,1,2,3,4,5,6,7},6)	8
E1	({0,1,2,3,4},4)	5,7	I1	({0,1,2,3,4,5,6,7,8},8)	9
E2	({0,1,2,3,7},3)	4	J1	({0,1,2,3,4,5,6,7,8,9},9)	10
E3	({0,1,2,3,7},7)	4	K1	({0,1,2,3,4,5,6,7,8,9,10},10)	0
E4	({0,1,2,3,4},2)	5,7	L1	({0,1,2,3,4,5,6,7,8,9,10},0)	0

3.5.2 Example 2

In this example, we demonstrate a complete numerical example, which includes both the complexity calculation and the assembly sequence planning.

In Fig. 3.11, we redraw the precedence graph and denote the number of variants that the corresponding task has to handle. We assume all variants are equal likely to take place in the choice processes. Moreover, the variants in the upstream stations are assumed to influence any assembly task in the downstream stations, i.e., all $a_{ij} = 1$.

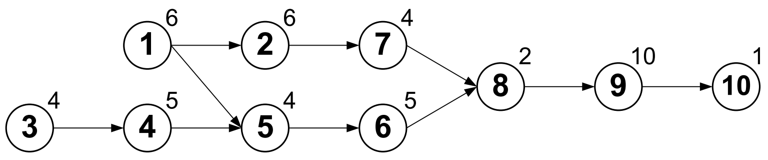

Figure 3.11: Precedence Graph of a Ten-Task Assembly with Number of Variants Indicated (From [1])

With the above assumptions, we can calculate every C_{ij} in Fig. 3.5 by Eq. (3.1). However, thanks to the transformation made in Section 3.4, only the values in Fig. 3.7 needs to be evaluated, i.e.,

- $C_{13} = C_{14} = \log_2 6$
- $C_{23} = C_{24} = C_{25} = C_{26} = \log_2 6$
- $C_{31} = C_{32} = C_{37} = \log_2 4$

- $C_{41} = C_{42} = C_{47} = \log_2 5$
- $C_{52} = C_{57} = \log_2 4$
- $C_{62} = C_{67} = \log_2 5$
- $C_{73} = C_{74} = C_{75} = C_{76} = \log_2 4$

Notice that, due to the particular setup of the example, the complexity C_{ij} is calculated based on the number of variants of the station indicated by the first subscript i.

Using Program 3, we are able to perform the sequence planning to minimize complexity. The optimal sequence is found to be 3-4-1-5-2-7-6-8-9-10, and the corresponding system complexity is $Z = 22.55$ bits.

3.6 Summary

In this chapter, we have demonstrated the opportunity of minimizing complexity for manufacturing systems by assembly sequence planning. The complexity is defined as operator choice complexity, which indirectly measures the human performance in making choices, such as selecting parts, tools, fixtures, and assembly procedures in a multi-product, multi-stage, manual assembly environment.

Methodologies developed in this chapter extend the previous work on modeling complexity and provide solution strategies for assembly sequence planning to minimize complexity. The solution strategies overcome the difficulty of handling the directions of complexity flows in optimization and effectively simplify the original problem through equivalent transformation into a network flow model. This makes the problem comparable to the traveling salesman problem with precedence constraints. By a careful construction of the state transition cost structure, we obtain the exact optimal solution through recursions based on dynamic programming. Such solution strategy is also generally applicable to problems in multi-stage systems where complex interactions between stages are considered.

However, due to the restrictive assumption on position independence for complexity, the application of the methodology is still limited. Moreover, the exponentially growing computational complexity is also not satisfactory for large problems with the number of assembly tasks far beyond 30. Hence, the further improvement should address the above limitations by developing approximations and heuristics without significantly sacrificing the accuracy of the solutions.

Table 3.2: State Transition Costs on the Arcs of the DPN

	B1	B2				
A1	$C_{13}+C_{14}$	$C_{31}+C_{32}+C_{37}$				
	C1	C2	C3	C4		
B1	$C_{23}+C_{24}+C_{25}+C_{26}$	$C_{32}+C_{37}$	∞	∞		
B2	∞	∞	C_{14}	$C_{41}+C_{42}+C_{47}$		
	D1	D2	D3	D4	D5	
C1	C_{37}	$C_{73}+C_{74}+C_{75}+C_{76}$	∞	∞	∞	
C2	∞	∞	$C_{24}+C_{25}+C_{26}$	$C_{42}+C_{47}$	∞	
C3	∞	∞	$C_{24}+C_{25}+C_{26}$	$C_{42}+C_{47}$	∞	
C4	∞	∞	∞	∞	0	
	E1	E2	E3	E4	E5	
D1	C_{47}	∞	$C_{74}+C_{75}+C_{76}$	∞	∞	
D2	0	∞	∞	∞	∞	
D3	C_{47}	∞	$C_{74}+C_{75}+C_{76}$	∞	∞	
D4	∞	∞	∞	$C_{25}+C_{26}$	$C_{52}+C_{57}$	
D5	∞	∞	∞	$C_{25}+C_{26}$	$C_{52}+C_{57}$	
	F1	F2	F3	F4	F5	
E1	C_{57}	∞	$C_{75}+C_{76}$	∞	∞	
E2	∞	0	∞	∞	∞	
E3	∞	0	∞	∞	∞	
E4	C_{57}	∞	$C_{75}+C_{76}$	∞	∞	
E5	∞	∞	∞	C_{26}	$C_{62}+C_{67}$	
	G1	G2	G3	G4		
F1	C_{67}	∞	C_{76}	∞		
F2	∞	0	∞	∞		
F3	∞	0	∞	∞		
F4	C_{67}	∞	C_{76}	∞		
F5	∞	∞	∞	0		
	H1	H2	I1	J1	K1	L1
G1	0	∞	∞	∞	∞	∞
G2	∞	0	∞	∞	∞	∞
G3	∞	0	∞	∞	∞	∞
G4	0	∞	∞	∞	∞	∞
H1	∞	∞	0	∞	∞	∞
H2	∞	∞	0	∞	∞	∞
I1	∞	∞	∞	0	∞	∞
J1	∞	∞	∞	∞	0	∞
K1	∞	∞	∞	∞	∞	0

Table 3.3: All Feasible Solutions for Example 1

Assembly Sequence	Complexity (bit)
1-2-3-4-5-6-7-8-9-10	5
1-2-3-4-5-7-6-8-9-10	4
1-2-3-4-7-5-6-8-9-10	3
1-2-3-7-4-5-6-8-9-10	4
1-2-7-3-4-5-6-8-9-10	3
1-3-2-4-5-6-7-8-9-10	4
1-3-2-4-5-7-6-8-9-10	3
1-3-2-4-7-5-6-8-9-10	2
1-3-2-7-4-5-6-8-9-10	3
1-3-4-2-5-6-7-8-9-10	3
1-3-4-2-5-7-6-8-9-10	2
1-3-4-2-7-5-6-8-9-10 $*$	1
1-3-4-5-2-6-7-8-9-10	4
1-3-4-5-2-7-6-8-9-10	3
1-3-4-5-6-2-7-8-9-10	5
3-1-2-4-5-6-7-8-9-10	5
3-1-2-4-5-7-6-8-9-10	4
3-1-2-4-7-5-6-8-9-10	3
3-1-2-7-4-5-6-8-9-10	4
3-1-4-2-5-6-7-8-9-10	4
3-1-4-2-5-7-6-8-9-10	3
3-1-4-2-7-5-6-8-9-10	2
3-1-4-5-2-6-7-8-9-10	5
3-1-4-5-2-7-6-8-9-10	4
3-1-4-5-6-2-7-8-9-10	6
3-4-1-2-5-6-7-8-9-10	5
3-4-1-2-5-7-6-8-9-10	4
3-4-1-2-7-5-6-8-9-10	3
3-4-1-5-2-6-7-8-9-10	6
3-4-1-5-2-7-6-8-9-10	5
3-4-1-5-6-2-7-8-9-10	7

CHAPTER 4

Build Sequence Scheduling to Minimize Complexity

Abstract

Build sequence scheduling is an important topic in mixed-model assembly. It is to determine the order of products being built in the assembly line. Significant research has been conducted to determine good sequences based on various criteria. For example, in Just-In-Time production systems, optimal sequences are searched to minimize the variation in the rate at which different parts were utilized. This chapter discusses the selection of optimal build sequences based on complexity introduced by product variety in mixed-model assembly lines. The complexity was defined as the information entropy that operator processes during assembly, which indirectly measures the human performance in making choices, such as selecting parts, tools, fixtures, and assembly procedures in a multi-product, multi-stage, manual assembly environment. In Chapter 2, a simple version of complexity measure has been developed for independent identically distributed (*i.i.d.*) sequences. This chapter [1] extends the concept by taking into account the sequential dependence of the choices and its impact on build sequence schedules. A model based on the Hidden Markov Chains is proposed for the sequence scheduling problem with constraints. Through a two-model two-part example, the results from the model indicates that proportional production causes the most complexity, while the batch production the least. Methodologies developed in this chapter enhance the previous work on modeling complexity, and provide solution strategies for build sequence scheduling to minimize complexity.

4.1 Introduction

An assembly process in a mixed-model line can be viewed from different perspectives. Besides the procedure of mounting parts to a partially completed assemblage, it is also a part

[1]Part of work in this chapter has appeared on [56]: X. Zhu, H. Wang, S. J. Hu, and Y. Koren. Build sequence scheduling to minimize complexity in mixed-model assembly lines. In *Proceedings of the 9th Biennial ASME Conference on Engineering Systems Design and Analysis*, 2008.

supply process. Particularly, in Just-In-Time mixed-model assembly lines, different types, shapes, and colors of materials have to be provided at the right time, at the right place, and with the right quantity. Among the materials, some are sequenced pre-hand, others are un-sequenced and stored in line-sided bins. In the latter case, the role of operators is to choose the correct parts from the bins, and assemble them to the correct model. Fig. 4.1 illustrates how operators choose variants of parts C and D and assemble them to models A and B according to customer requirements. In a view, one of the objectives of the assembly operation is to ensure the correct part sequence to match with the model sequence.

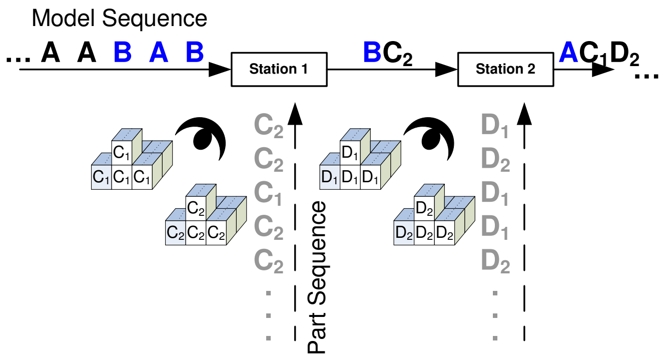

Figure 4.1: Model and Part Sequences in a Mixed-Model Assembly Line

The build sequence is one of the important scheduling decisions made during the operations of a mixed-model assembly line. For example, in automobile assembly lines, one needs to decide which vehicle is the first, which one is the next in the continuous product flow. In this chapter, we refer to the build sequence as the model sequence shown in Fig. 4.1. The objectives of build sequence scheduling could be to level work content [53], or to smooth material usage [36, 29], or both [12]. However, to avoid work overload at the bottleneck stations, some spacing rules are usually enforced for build sequence decisions [24], with typical examples being:

- Maximum of 3 model I's in a row;
- Minimum of 3 non-model II's between any two model II's;
- Minimum of 1 non-model III between any two model III's;
- Minimum of 2 non-model IV between any two model IV's;
- Minimum of 10 non-exports between any two exports.

Obviously, the build sequences corresponding to these spacing rules cause sequential dependencies in both model and part sequences. The sequential dependencies will be utilized later in the article.

In studying mixed-model assembly process, a deterministic model is *usually* not sufficient to describe build sequences with spacing rules. For example, consider a constrained binary sequence [7] having at least one 0 and at most two 0's between any pair of 1's. Thus, sequences like 101001 and 0101001 are valid, but 0110010 and 0000101 are not. Hence, the resulting build sequences are generated by arbitrarily inserting one 0 or two 0's between every two 1's.

To model such random sequences, probabilistic models are needed. In the above example, we can use a Markov chain to generate the legitimate sequences. We define State 1 as being in "1", State 2 as being in "0" for the first time, and State 3 as being in "0" for the second time. The following state transition diagram in Fig. 4.2 can be used to describe all possible transitions and their chances.

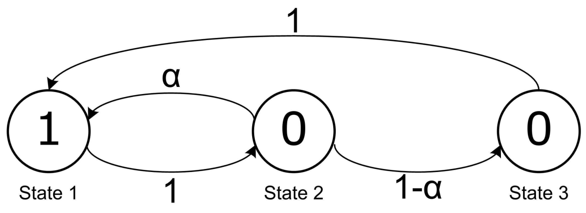

Figure 4.2: State Transition Diagram of a Markov Chain

Correspondingly, we define a state transition matrix $\mathbf{\Lambda}$ for the chain as follows:

$$\Lambda = \{\lambda_{ij}\} = \begin{bmatrix} 0 & 1 & 0 \\ \alpha & 0 & 1-\alpha \\ 1 & 0 & 0 \end{bmatrix} \tag{4.1}$$

where $\lambda_{ij} = P(X_t = j | X_{t-1} = i)$ is the transition probability from State i to State j at time t, and $\lambda_{21} = \alpha$ means after attaining the first "0" we can either obtain the second "0" with probability $1-\alpha$ or obtain a "1" with probability α.

Furthermore, based on a probabilistic model (such as a Markov chain) rather than a deterministic one, one is capable of studying the statistical characteristics of the constrained build sequences. The reason is that the former can generate enough number of sequences instead of an individual one for statistical inference.

The complexity associated with properly coordinating the part supply process as described in Fig. 4.1 increases if the product variety goes high. In mass production, only one kind of material is fed into each station. Thus, the assembly operation is rather straightforward: same part, same procedure, every cycle. In mixed-model lines, however, more than one types of materials are to be chosen and assembled. The assembly operation becomes more complex. The complexity is related with how operators handle variety. It has been reported by both empirical and simulation studies [15, 33, 14] that increased product variety has significant negative impact on the performance of the mixed-model assembly process.

Such impact can result from assembly system design as well as people performance under high variety.

To evaluate the impact, complexity needs to be defined and measured. In Fig. 4.1, one has observed that both model and part sequences are random, thus, complexity can be defined as the average randomness in a sequence. In other words, since the operators must make choices of parts, the process of making the choices is equivalent to handling uncertainty of the random sequence. Based on this idea, a complexity measure called "Operator Choice Complexity" (OCC) was proposed in Chapter 2 to quantify the uncertainty in making the choices. The OCC takes an analytical form as an information-theoretic entropy measure of the average uncertainty in a choice process, i.e., the random sequence of parts. It can be inferred that the more certain an operator is about what to choose in the upcoming assembly task, the less the complexity is. In fact, the definition of OCC is similar to that of the cognitive measure of human performance in the Hicks-Hyman Law [21, 23]. Reducing the complexity may help to improve assembly system performance.

In Chapters 2 and 3 [55, 54], complexity measure has been used for assembly sequence design under the *i.i.d.* (independent identically distributed) conditions. In this chapter, assembly processes with sequential dependencies from the *non-i.i.d.* choices are considered. The study of the sequential dependencies enables the application of the complexity measure for build sequence scheduling problems.

The objective of this chapter is to develop methodologies for finding the optimal build sequences to minimize complexity. The complexity measure and its models for assembly systems were developed in Chapter 2 and will be reviewed in the next section. According to the complexity models developed, build sequence determines the sequential dependencies of the random choices and thus proper sequence scheduling can reduce complexity.

The chapter is structured as follows. Section 4.2 provides background information on the complexity measure and shows the opportunity of minimizing complexity by build sequence scheduling. Section 4.3 discusses the problem formulation based on a Hidden Markov model and Section 4.4 provide the general solution strategy. Section 4.5 presents a two-model two-part numerical example and discusses results. A trade-off factor is proposed to demonstrate the use of the model for a balanced decision on complexity and responsiveness. Section 4.6 summarizes the chapter.

4.2 Complexity Model for Sequence Scheduling

In this section, we first review the complexity measure and its models discussed in Chapter 2. Based on a general form of the complexity measure, we discuss the sequential dependencies of the *non-i.i.d.* sequences. Due to the principle of "conditioning reduces entropy", the sequential dependencies provide an avenue of reducing complexity.

4.2.1 General Form of Complexity Measure

In a general form, the measure of complexity, OCC is a linear function of the *entropy rate* of a stochastic process (choice process). The choice process consists of a sequence of random choices with respect to time. The choices are then modeled as a sequence of random variables, each of which represents choosing one of the possible alternatives from a choice set. In fact, the choice process can be considered as a discrete time discrete state stochastic process $X' = \{X_t, t = 1, 2, \ldots\}$, on the state space (the choice set) $\{1, 2, \ldots, M\}$, where t is the index of discrete time period, M is the total number of possible alternatives which could be chosen during each period. More specifically, $X_t = m, m \in \{1, 2, \ldots, M\}$, is the event of choosing the m^{th} alterative during period t. With the above notation, the general form of OCC is:

$$\begin{aligned} H(X') &= \lim_{N\to\infty} \frac{1}{N} H(X_1, X_2, \ldots, X_N) \\ &= \lim_{N\to\infty} H(X_N | X_{N-1}, X_{N-2}, \ldots, X_1) \end{aligned} \tag{4.2}$$

The second equivalence sign holds if the process is stationary [7].

In the simplest case, if the sequence is independent, identically distributed (*i.i.d.*), Eq. (4.2) can be reduced to an analytical entropy function H as we have discussed in Eq. (2.2) of Chapter 2. The H function takes the following form:

$$H(X) = H(p_1, p_2, \ldots, p_M) = -C \cdot \sum_{m=1}^{M} p_m \cdot \log p_m \tag{4.3}$$

where $p_m \triangleq \mathbf{P}(X = m)$, for $m = 1, 2, \ldots, M$, which is the probability of choosing the m^{th} alternative; $(p_1, p_2, \ldots, p_M)$ determines the probability mass function of X, is also known as the *mix ratio* of the (component) variants associated with the choice process.

4.2.2 Principle of "Conditioning Reduces Entropy"

Sequential dependencies provide additional "information" to the operators, thus potentially reduce uncertainty and complexity. Suppose we have two successive choices. To simplify the subsequent notations, we denote X as the first choice, and Y the second. Both choices have M alternatives (numbered from 1 to M). Let $p(x_i, y_j)$ be the probability of the joint event $\{X = x_i, Y = y_j\}$, where $i, j \in \{1, 2, \ldots, M\}$. The complexity of the joint choice is:

$$H(X, Y) = -\sum_{i=1}^{M} \sum_{j=1}^{M} p(x_i, y_j) \log p(x_i, y_j)$$

While,

$$\begin{aligned} H(X) &= -\sum_{i=1}^{M}\sum_{j=1}^{M} p(x_i, y_j) \log \sum_{j=1}^{M} p(x_i, y_j) \\ H(Y) &= -\sum_{j=1}^{M}\sum_{i=1}^{M} p(x_i, y_j) \log \sum_{i=1}^{M} p(x_i, y_j) \end{aligned}$$

It is easy to show that,

$$H(X, Y) \leq H(X) + H(Y) \tag{4.4}$$

with equality achieved only if the two choices are independent [45], i.e., $p(x_i, y_j) = p(x_i)p(y_j)$, where $p(x_i) \triangleq P(X = x_i)$, and $p(y_j) \triangleq P(Y = y_j)$.

Consider the two choices X and Y again, assume they are not independent. For any particular value x_i that X can take, assume there is a conditional probability given that Y has the value of y_j, i.e., $p(y_j|x_i) \triangleq P(X = x_i|Y = y_j)$. By Bayes' rule:

$$p(y_j|x_i) = \frac{p(x_i, y_j)}{\sum_{j=1}^{M} p(x_i, y_j)} \tag{4.5}$$

We define conditional entropy of Y, $H(Y|X)$ as the expected entropy of Y for knowing the values of X, i.e.,

$$\begin{aligned} H(Y|X) &= -\sum_{i=1}^{M} p(x_i) \sum_{j=1}^{M} p(y_j|x_i) \log p(y_j|x_i) \\ &= -\sum_{i=1}^{M}\sum_{j=1}^{M} p(x_i, y_j) \log p(y_j|x_i) \end{aligned} \tag{4.6}$$

The quantity measures how uncertain we are of Y on the average (how complex we make choices by responding to Y) when we know X. Substituting the value of $p(y_j|x_i)$ in Eq. (4.5), we obtain:

$$\begin{aligned} H(Y|X) &= -\sum_{i=1}^{M}\sum_{j=1}^{M} p(x_i, y_j) \log p(x_i, y_j) \\ &+ \sum_{i=1}^{M}\sum_{j=1}^{M} p(x_i, y_j) \log \sum_{j=1}^{M} p(x_i, y_j) \\ &= H(X, Y) - H(X)) \end{aligned}$$

or,

$$H(X,Y) = H(X) + H(Y|X) \tag{4.7}$$

The uncertainty (or choice complexity) of the joint choice X and Y is the uncertainty of X plus the uncertainty of Y when X is known.

From Eqs. (4.4) and (4.7) we have,

$$H(X) + H(Y) \geq H(X,Y) = H(X) + H(Y|X)$$

Hence,

$$H(Y) \geq H(Y|X) \tag{4.8}$$

The uncertainty (or choice complexity) of the second choice Y is *never* increased by the knowledge of the first choice X. $H(Y|X)$ will reach the maximum as $H(Y)$ if X and Y are independent choices. This is known as the principle of "conditioning reduces entropy" [7].

4.3 Build Sequence Problem Formulation

Let us consider a customized vehicle with N types of models (e.g., chassis frames). At a station, one of the M types of parts is to be assembled. Also, assume the type of the model is *not* recognizable by the operator. However, the parts being installed are observable. In the meantime, the operator tries to predict the next part type based on the information given on the current part. By using the information and due to the principle of "*conditioning reduces uncertainty*", the operator may reduce the uncertainty (thus complexity) of making choices, i.e., $H(Y|X) \leq H(X)$ as suggested in Eq. (4.8).

In order to calculate the complexity $H(Y|X)$ (in the form of conditional entropy), we need to compute the probabilities of the observations of the parts. The probabilities can be obtained via a Hidden Markov Model (HMM) described as follows.

4.3.1 The Hidden Markov Model

The model has two major parts, an unobservable (thus hidden) Markov chain and a set of observations.

The Markov chain incorporates the spacing rules and is capable of probabilistically generating build sequences. The chain has N' ($N' \geq N$ as in the example of Fig. 4.2, where $N = 2$ and $N' = 3$) states, and each of which corresponds to a type of models. Assume the chain is *ergodic*, which means the states are interconnected in such a way that any one of them is reachable from any of the others. Here, ergodicity is a reasonable assumption since every model will be built in the plant. We denote the set of the states as $S = \{s_1, s_2, \ldots, s_{N'}\}$, and the generated state sequence as $\mathbf{Q} = Q_1 Q_2 \ldots$, $Q_t \in S$, for

$t = 1, 2, \ldots$ The state transition probability distribution $\Lambda = \{\lambda_{ij}\}$, where λ_{ij} is probability of moving into state j while previously in state i, i.e., $\lambda_{ij} = P(Q_t = s_j | Q_{t-1} = s_i)$.

A set of observations correspond to the parts assembled by the operator. Let M be the number of observations per state, which suggests all parts are possible to be installed on any model. We denote the set of the observations as $V = \{v_1, v_2, \ldots, v_M\}$, and the generated observation sequence as $\mathbf{O} = O_1 O_2 \ldots$, $O_t \in V$, for $t = 1, 2, \ldots$ The observation probability distribution is $B = \{b_{ij}\}$, where b_{ij} is the probability of attaining observation j while in state i, i.e., $b_{ij} = P(O_t = v_j | Q_t = s_i)$.

Furthermore, we define $\Pi^{(t)} = \{\pi_i^{(t)}\}$ as the probability vector of states at time t (i.e., after $(t-1)$ transitions), where $\pi_i^{(t)}$ is the probability of being in state i, i.e., $\pi_i^{(t)} = P(Q_t = s_i)$. As an initial condition, we set the values for $\Pi^{(1)} = \Pi = \{\pi_i\}$, where $\pi_i = P(Q_1 = s_i)$.

Given appropriate values of N', M, Λ, B, and Π. The HMM can generate a random sequence of observations with its corresponding sequence of states. Let the length of the sequence be T. The procedure of sequence generation can be described as follows [41].

1. Choose an initial state $Q_1 = s_i$ according to the initial state distribution Π;
2. Set $t = 1$;
3. Choose $O_t = v_k$ according to the observation probability distribution in state s_i, i.e., b_{ki} in B;
4. Transit to a new state $Q_{t+1} = s_j$ according to the state transition probability distribution for state s_j, i.e., λ_{ij} in Λ;
5. Set $t = t + 1$; return to step 3 if $t < T$; otherwise terminate the procedure.

The above sequence generation procedure simulates the build sequence scheduling for the mixed-model assembly line. The model sequence corresponds to the state sequence, and the part sequence corresponds to the observation sequence as shown in Fig. 4.2. At a station, once the type of chassis is determined, a type of part is decided accordingly as in step 3. Thus, the generated observation sequence mimics the sequence of parts handled by the operator at the station.

4.3.2 The Optimization Problem

To minimize complexity, we need to make decisions on the parameters of how to generate the state sequence, i.e., the transition probability distribution Λ. Once the decisions are made, the state sequence drives the generation of the observation sequence, hence, the complexity can be determined in the form of conditional entropy rate as defined in Eq. (4.6). In short, we can construct the following optimization problem.

$$\begin{array}{lll} \text{Minimize} & : & \text{Complexity} = \lim_{t\to\infty} H(O_t|O_{t-1}) \\ \text{With respect to} & : & \Lambda \\ \text{Subject to} & : & \text{Spacing rules and other constraints} \end{array} \tag{4.9}$$

Note, according to the objective function, we need to evaluate $H(O_t|O_{t-1})$ in the long run and preferably in the steady state. Thus numerical methods are required to find $H(O_t|O_{t-1})$ when $t \to \infty$. Also, the spacing rules and other constraints (such as mix ratios) should be incorporated into the structure of the Markov chain.

4.4 Solutions

In order to calculate complexity, we need to know the probability of the observations, which can be derived from the probability of the first observation and the conditional probabilities of the other observations. Let $\phi_j^{(t)}$ be the probability of observation in state j at time t, i.e.,

$$\phi_j^{(t)} = P(O_t = v_j) \tag{4.10}$$

Let $\theta_{ij}^{(t-1)}$ be the conditional probability of attaining observation j at time t given that the previous observation is i, i.e.,

$$\theta_{ij}^{(t-1)} = P(O_t = v_j|O_{t-1} = v_i) \tag{4.11}$$

Once values of $\phi_j^{(t)}$ and $\theta_{ij}^{(t-1)}$ are determined, the complexity can be calculated according to the definition of conditional entropy in Eq. (4.6) as follows:
For $t = 1$,

$$\begin{aligned} H(O_1) &= \sum_{j=1}^{M} P(O_1 = v_j) \cdot \log \frac{1}{P(O_1 = v_j)} \\ &= \sum_{j=1}^{M} \phi_j^{(1)} \cdot \log \frac{1}{\phi_j^{(1)}} \end{aligned}$$

For $t = 2, 3, \ldots$,

$$\begin{aligned} H(O_t|O_{t-1}) &= \sum_{i=1}^{M} P(O_{t-1}=v_i) \cdot \sum_{j=1}^{M} P(O_t=v_j|O_{t-1}=v_i) \\ &\cdot \log \frac{1}{P(O_t=v_j|O_{t-1}=v_i)} \\ &= \sum_{i=1}^{M} \phi_i^{(t-1)} \sum_{j=1}^{M} \theta_{ij}^{(t-1)} \cdot \log \frac{1}{\theta_{ij}^{(t-1)}} \end{aligned} \tag{4.12}$$

The following discussions illustrate the procedures of deriving $\phi_j^{(t)}$ and $\theta_{ij}^{(t-1)}$.

For the first observation, i.e., $t = 1$, we can directly obtain the probability of observations from the initial conditions in Π.

$$\begin{aligned} \phi_j^{(1)} &= P(O_1 = v_j) \\ &= \sum_{l=1}^{N'} P(O_1 = v_j|Q_1 = s_l) \cdot P(Q_1 = s_l) \\ &= \sum_{l=1}^{N'} \pi_l \cdot b_{lj} \end{aligned}$$

For the subsequent observations, i.e., $t = 2, 3, \ldots$, we obtain the conditional probability $\theta_{ij}^{(t-1)}$ by the derivation from the first observation according to the state transition probability distribution Λ and the observation probability distribution B.

$$\begin{aligned} \theta_{ij}^{(t-1)} &= P(O_t = v_j|O_{t-1} = v_i) \\ &= \sum_{l=1}^{N'} P(O_t = v_j, Q_{t-1} = s_l|O_{t-1} = v_i) \\ &= \sum_{l=1}^{N'} P(O_t = v_j|O_{t-1} = v_i, Q_{t-1} = s_l) \cdot P(Q_{t-1} = s_l|O_{t-1} = v_i) \end{aligned}$$

where,

$$\begin{aligned}
&P(O_t = v_j | O_{t-1} = v_i, Q_{t-1} = s_l) \\
&\quad = \sum_{r=1}^{N'} P(O_t = v_j | O_{t-1} = v_i, Q_{t-1} = s_l, Q_t = s_r) \\
&\quad\quad \cdot P(Q_t = s_r | O_{t-1} = v_i, Q_{t-1} = s_l) \\
&\quad = \sum_{r=1}^{N'} P(O_t = v_j | Q_t = s_r) \cdot P(Q_t = s_r | Q_{t-1} = s_l) \\
&\quad = \sum_{r=1}^{N'} \lambda_{lr} \cdot b_{rj}
\end{aligned}$$

and,

$$\begin{aligned}
P(Q_{t-1} = s_l | O_{t-1} = v_i) &= \frac{P(Q_{t-1} = s_l, O_{t-1} = v_i)}{P(O_{t-1} = v_i)} \\
&= \frac{P(O_{t-1} = v_i | Q_{t-1} = s_l) \cdot P(Q_{t-1} = s_l)}{\sum_{k=1}^{N'} P(O_{t-1} = v_i | Q_{t-1} = s_k) \cdot P(Q_{t-1} = s_k)} \\
&= \frac{\pi_l^{(t-1)} \cdot b_{li}}{\sum_{k=1}^{N'} \pi_k^{(t-1)} \cdot b_{ki}}
\end{aligned}$$

Therefore,

$$\theta_{ij}^{(t-1)} = \sum_{l=1}^{N'} \left[\sum_{r=1}^{N'} \lambda_{lr} \cdot b_{rj} \right] \cdot \frac{\pi_l^{(t-1)} \cdot b_{li}}{\sum_{k=1}^{N'} \pi_k^{(t-1)} \cdot b_{ki}}$$

where π_i^t can be obtained from $\Pi^{(t)} = \Pi \times \Lambda^{t-1}$ with $\Lambda^t = \overbrace{\Lambda \times \Lambda \times \ldots \Lambda}^{t}$.

By the definitions of $\phi^{(t)}$ and $\theta_{ij}^{(t-1)}$ in Eqs.(4.10) and (4.11), for $t = 2, 3, \ldots$,

$$\begin{aligned}
\phi_j^{(t)} &= \sum_{i=1}^{M} P(O_t = v_j | O_{t-1} = v_i) \cdot P(O_{t-1} = v_i) \\
&= \sum_{i=1}^{M} \theta_{ij}^{(t-1)} \cdot \phi_i^{(t-1)}
\end{aligned}$$

By substituting $\theta_{ij}^{(t-1)}$ and $\phi_j^{(t)}$ into Eq. (4.12), we obtain the values of $H(O_t|O_{t-1})$ for $t = 2, 3, ...$

To simplify the notations, we use the following matrix operations for the above algebra. Note, '$\times$' denotes matrix multiplication; '$\cdot$' denotes dot product of two matrices with identical sizes.

$$\theta_{ij}^{(t-1)} = \frac{\Pi^{(t-1)} \cdot \mathbf{B}_{\cdot i}^{T}}{\Pi^{(t-1)} \times \mathbf{B}_{\cdot i}} \times \Lambda \times \mathbf{B}_{\cdot j}$$

Since $\phi_i^{(t-1)} = \Pi^{(t-1)} \times \mathbf{B}_{\cdot i}$, we have:

$$H(O_t|O_{t-1}) = \Pi^{(t-1)} \times \mathbf{B} \times \left[(\theta^{(t-1)} \cdot \log \frac{1}{\theta^{(t-1)}}) \times \mathbf{1} \right]^T$$

where $\mathbf{1}$ is a column vector of ones with size M.

4.5 Numerical Example and Discussions

We use an example of two models and two parts to demonstrate the use of above procedures. The results of the example are discussed. Based on the results, we introduce responsiveness as an additional factor to achieve a balanced decision on system performance.

4.5.1 Example Setup

The two-model two-part system is given as follows.

1. Two models, i.e., $S = \{s_1, s_2\}$, $N = 2$; two parts, i.e., $V = \{v_1, v_2\}$, $M = 2$.
2. The spacing rules are expressed by a two-state Markov chain in Fig. 4.3, with the following state transition probability matrix:

$$\Lambda = \begin{bmatrix} 1-\alpha & \alpha \\ \beta & 1-\beta \end{bmatrix}$$

3. The observation probability distribution matrix B is as follows:

$$\mathbf{B} = \begin{bmatrix} b_1 & 1-b_1 \\ b_2 & 1-b_2 \end{bmatrix}$$

4. As an additional constraint, the mix ratio of the chassis is $\mu_1 : \mu_2$, where $\mu_1 + \mu_2 = 1$. This ratio is also known as the steady state probability mass function of the two-state Markov chain.
5. The state sequence starts with $Q_1 = s_1$, thus, $\Pi = [1 \quad 0]$.

Because of the additional constraint in item 4, it is easy to verify that β is simply a function of α and μ_1, i.e.,

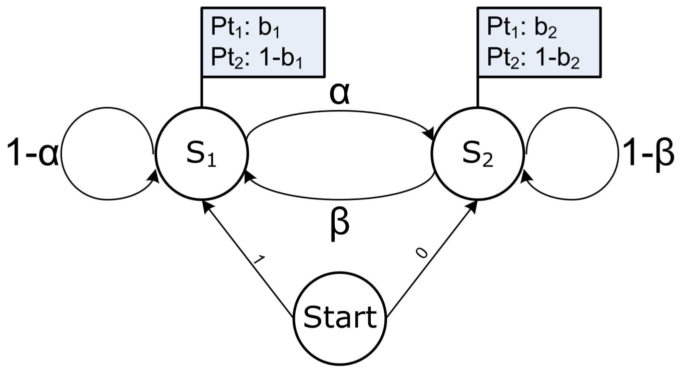

Figure 4.3: HMM Example with Two Models and Two Parts

$$\beta = \frac{\mu_1}{1-\mu_1}\alpha \tag{4.13}$$

Hence, the build sequence scheduling formulation in Eq. (4.9) is reduced to the following problem:

$$\begin{aligned} \text{Minimize} &\ : \ h(\mu_1, b_1, b_2, \alpha) = \lim_{t\to\infty} H(O_t|O_{t-1}) \\ \text{With respect to} &\ : \ \alpha \\ \text{Subject to} &\ : \ \text{Spacing rules} \end{aligned} \tag{4.14}$$

The objective is to minimize the choice complexity in the form of conditional entropy rate as defined in Eq. (4.6). However, we are more interested in the long run steady state of the quantity, i.e., $\lim_{t\to\infty} H(O_t|O_{t-1})$. In calculating the quantity, convergence is observed by the numerical examples; it seems that the convergence is closely related to the convergence of the Markov chain. The convergence rate is different for different α's.

4.5.2 Result Discussions

We use parameters $\mu_1 : \mu_2 = 0.3 : 0.7$, $b_1 = 0.9$, and $b_2 = 0.5$ to run the model for T sufficiently large (i.e., steady state). Here are the results:

1. **Optimality**: The maximum entropy $h^*(\mu_1, b_1, b_2, \alpha^*)$ is obtained at $\alpha^* = 1 - \mu_1$ (and $\beta^* = \mu_1$), where the underlining Markov process starts with its long run probability distributions; as α deviates from α^*, the magnitude of decrease on entropy is closely related to the observation probability distribution, i.e., the decreasing trend is not necessarily symmetric, see Fig. 4.4.

2. **Symmetry**: Function h is symmetric about the axes $b_1 = b_2 = 0.5$, i.e., $h(\mu_1, b_1, b_2, \alpha) = h(\mu_1, 1-b_1, 1-b_2, \alpha)$. In particular, if $b_1 = b_2$, g stays constantly at 1. This property

is again consistent with the intuition behind the structure of the example.

3. **Loss of influence from mixing**: If $b_1 = b_2$, values of g is uniform for all α's, i.e., $h(\mu_1, b_1, b_1, \alpha) = H(b_1, 1 - b_1)$.

4. **Mixing overwhelms distinction**: If $\mu_1 = \mu_2 = 0.5$ and $b_1 + b_2 = 1$, the value of maximum h is 1, i.e., $h^* = H(0.5, 0.5) = 1$.

5. **Sensitivity**: If $\mu_1 < \mu_2$ and $|b_1 - 0.5| < |b_2 - 0.5|$, then entropy reduction by deviating α from $1 - \mu_1$ (to either directions) is more effective, since the state can stay in state 2 for longer time.

Properties 1 and 5 suggest that batch production results in the least complexity. The intuition behind this result is that batch production takes the most distinctive transition probabilities, and the minimum entropy is achieved at the extreme points. On the contrary, proportional production causes the most complexity.

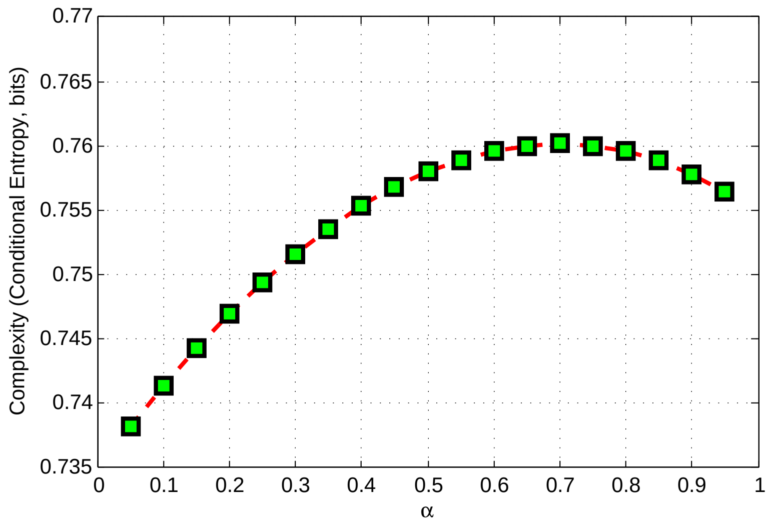

Figure 4.4: HMM Example Output with Changing α (Model Parameters: $\mu_1 = 0.3$, $b_1 = 0.9$, $b_2 = 0.5$)

From the outcomes of the example, we observe that proportional production results in the maximum complexity. Furthermore, from Fig. 4.4, we find that the curve is concave. In other words, if we are interested in minimizing complexity, we will find the optimal on the boundary. This solution is *trivial* although it is consistent with our intuition. That is, complexity is reduced if we tend to follow batch production. On the other hand, proportional production does have advantages in terms of responsiveness. Therefore, we can incorporate responsiveness as a trading factor with complexity.

4.5.3 Incorporating Responsiveness in Problem Formulation

In order to assess responsiveness, we need to calculate the expected deviation from the mix ratio of the real production to the ideal mix ratio. In Fig. 4.5, we demonstrate the deviation for distinctive α's. We use solid line (for $\alpha = 0.1$) and dotted line (for $\alpha = 0.9$) to denote the quantity of models produced, and use dashed line to denote the ideal mix ratio. Visually, it is obvious that $\alpha = 0.1$ generates larger deviation, since the 2-state Markov process dwells in State 1 with a small probability ($\alpha = 0.1$) for transitions into State 2, and observations emitted from State 1 has less uncertainty due to the distinct observation probability distribution ($b_1 : 1 - b_1 = 0.9 : 0.1$). Quantitatively, the deviation is defined as the squared vertical distance between the solid (or dotted) line and the dashed line. To compute the squared distance, we need to construct a backward recursion to calculate the expected deviation. The procedures of the calculation are discussed below.

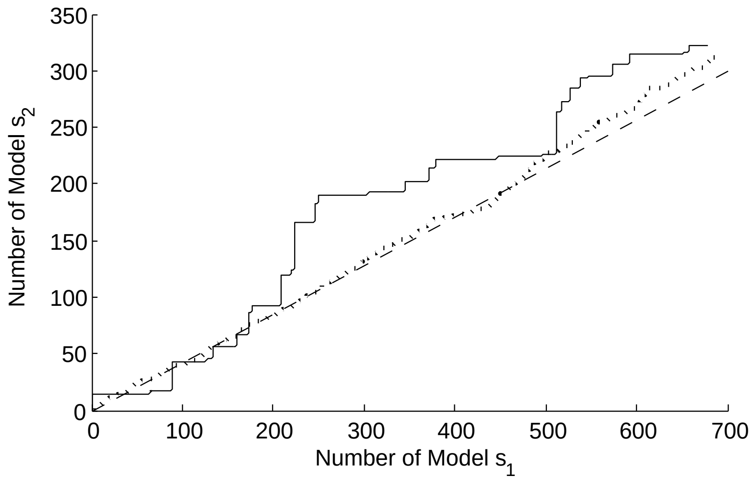

Figure 4.5: Deviation of Production from Ideal Mix Ratio

To simplify the explanation, we assume the Markov chain defined in Section 4.3 has the same number of states as that of the models, i.e., $N = N'$. Cases beyond this assumption can be tackled similarly by the same approach as follows.

Define (t, q, X) as the state for the recursion, where t denotes the index of time (production step) we are at, $t = 1, 2, \ldots, T$; q denotes the current state of the model, $q \in S$ and $S = \{s_1, s_2, \ldots, s_N\}$ is the set of states in the Markov chain; $X = [x_1, x_2, \ldots, x_N]^T$ and x_i denotes the cumulative number of the models corresponding to state s_i produced from time 1 through t, thus $x_i \in \{0, 1, \ldots, t\}$ and $t = \sum_{i=1}^{N} x_i$.

Let $g(X)$ be the deviation of production in state (t, q, X), hence,

$$g(t, X) = \sum_{i=1}^{N} (x_i - t\mu_i)^2$$

where μ_i is the mix ratio of the model corresponding to state i as defined before.

Let $\mathbf{e}_i$ be a unit vector with n entries, all of which are zero except for a single "1" in the i^{th} row, i.e., $\mathbf{e}_i = [0, \ldots, 0, 1, 0, \ldots, 0]$, where i refers to the event of scheduling a model corresponding to state s_i in the next production step. According to the transition probability distribution of the Markov chain, $P(\mathbf{e}_j | q = s_i) = \lambda_{ij}$, for $i, j \in \{1, 2, \ldots, N\}$.

Furthermore, we define $f_t(q, X)$ as the objective value function, which is the expected deviation from time t to T. In fact, $f_t(q, X)$ counts the average deviation for each production step from present to the end. Therefore, we obtain the following backward recursion.

$$f_t(q, X) = \begin{cases} g_t(X) + \sum_{i=1}^{N} f_{t+1}(i, X + \mathbf{e}_i) \cdot P(\mathbf{e}_j | q = s_i), \\ \quad \text{for } t = 1, 2, \ldots, T-1; q \in S; X = [x_1, x_2, \ldots, x_N] \\ \\ g_t(X), \text{ for } t = T; q \in S; X = [x_1, x_2, \ldots, x_N]; \end{cases}$$

The answer is $f_0(0, \mathbf{0}) = \sum_{i=1}^{N} \pi_i \cdot f_1(i, \mathbf{e}_i)$, which counts for the total expected deviation. Since the above procedures are developed based on a known α, we denote d_α as the value of $f_0(0, \mathbf{0})$ based on a specific α.

We perform the calculation for the 2-model 2-part example and get the following result as shown in Fig. 4.6. In the figure, the expected deviation has been standardized by dividing the largest d_α, denoted as d_α^*. It can be seen that the expected deviation is monotonically decreasing with α at an exponential rate. In the example, since α and β (β is proportional to α) are both the probabilities of making transition to the other state, thus, intuitively, the larger α is, the more responsive the system is, and the less the deviation is. Therefore, the intuition is consistent with the result in Fig. 4.6.

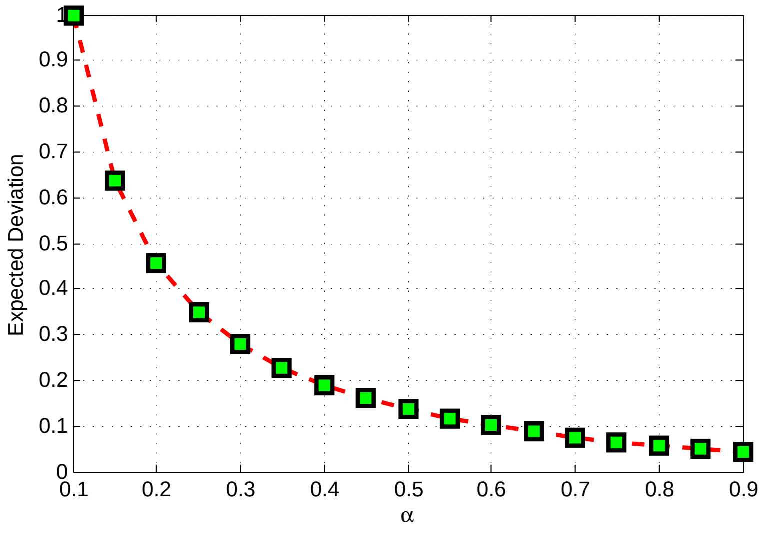

Figure 4.6: Expected Deviation Versus Values of α

By incorporating the expected deviation as the trading factor, we are able to trade-off complexity and responsiveness. The trade-off is formed by a multi-objective function as follows:

$$z(\alpha) = \frac{h(\mu_1, b_1, b_2, \alpha)}{h^*(\mu_1, b_1, b_2, \alpha^*)} + \frac{d_\alpha}{d_\alpha^*}$$

where the first term is the relative complexity compare to the maximum complexity when proportional production is adopted; the second term is the relative expected deviation under parameter α. Minimizing $z(\alpha)$ gives a balanced solution of α for both complexity and responsiveness.

From our example, we find α to be on the extreme point to the right of α^*. The solution suggests to promote responsiveness by choosing large α and at same time to keep away from the proportional production.

4.6 Summary

In this chapter, we have presented the procedure of minimizing complexity for assembly systems by build sequence scheduling. The complexity is defined as the operator choice complexity, which indirectly measures the human performance in making part choices in a multi-model, multi-stage, manual assembly environment.

Methodologies developed in this chapter extend the previous work on modeling complexity, and provide solution strategies for build sequence scheduling to minimize complexity. The chapter considers sequential dependencies in a random sequence, and develops a Hidden Markov Model for making decisions on build sequences. Such solution strategy is generally applicable to sequence scheduling problems with spacing rules. Through a numerical example, we demonstrate the use of the model and its solution. The results of the example suggest the optimal build sequence to minimize complexity. The maximum is attained in the proportional production situation, while the minimum is attained in the batch production. This conclusion is consistent with the intuition behind the example.

However, the minimum complexity is attained on the boundary, the solution is *not* satisfactorily interesting. Thus, responsiveness is introduced as a trading factor for the problem. The responsiveness is assessed by the expected deviation from the real production to the ideal mix ratio. Based on a multi-objective function, a balanced decision for complexity and responsiveness can be made.

CHAPTER 5

Conclusions and Future Work

In this chapter, the original contributions of the dissertation are summarized, and potential areas for future research are suggested.

5.1 Conclusions and Original Contributions

This dissertation presents the original research work on modeling product variety induced manufacturing complexity and the application of the model for assembly system design. The presentation takes a multiple manuscript format, and conclusions have been drawn in each individual chapters. Thus, only the original contributions are summarized here.

1. Definition of a new complexity measure

   This dissertation defines a measure of complexity based on the choices that an operator has to make at the station level. This definition is based on a careful observation and understanding on the choice processes in the mixed-model assembly systems. The measure reflects the underlining "physics" of assembly process for product variety. It suggests that variety causes complexity by introducing uncertainty in the choice process.

2. Development of a multi-stage complexity model

   A "Stream of Complexity" model is proposed for multi-stage assembly systems. The model provides a detailed understanding on how variety interacts with assembly operations. Thus, applications can be formulated based on the model for system design. The model captures a unique complexity propagation as a result of modeling the interactions between variety and complexity. Based on the propagation, two categories of complexity are defined: feed complexity and transfer complexity. Feed complexity is static to the sequence of assembly operations, while transfer complexity changes with the sequence. The propagation and categorization of complexity facilitates the view of complexity in the system level, thus suggest potential applications of the complexity model in system design.

3. Enhancement of understanding on complexity

   The measure and model contribute towards a better understanding of manufacturing complexity and its impact on performances in mixed-model assembly systems. Based on the understanding, it becomes clear that product variety raises the level of uncertainty in making choices of parts, tools, fixtures, and assembly procedures, and thus the complexity in mixed-model assembly operations. Therefore, tracing down the variety in the assembly system and modeling its interactions with assembly operations are essential means to assess complexity and its impact on system performances.

4. Application of the complexity model for assembly sequence planning

   Based on the multi-stage complexity model and the fact that transfer complexity changes with the sequence, one can reduce complexity by proper assembly sequence planning. The planning is formulated as an optimization problem and a solution strategy based on model simplification and equivalent transformation is developed. The solution strategy overcomes the difficulties of handling the directions of complexity flows in the optimization, and effectively simplifies the original problem through an equivalent transformation. This makes the problem comparable to the traveling salesman problem with precedence constraints. By a careful construction of the state transition cost structure, the exact optimal solution could be obtained based on dynamic programming. Such solution strategy makes the application of the complexity model accessible for industries, and is also generally applicable to other similar problems in multi-stage systems where complex interactions between stages are concerned.

5. Application of the complexity model for build sequence scheduling

   When *non-i.i.d.* sequences of choices are studied, sequential dependency among the sequences presents an opportunity for minimizing complexity by build sequence scheduling. The scheduling is formulated by using a Hidden Markov Model to simulate the build sequence generation under spacing-rule constraints. Accordingly, a solution strategy is developed to find analytical solutions for minimum and maximum complexity. Such model and solution strategy are generally applicable to sequence scheduling problem with spacing rules. A numerical example is discussed to demonstrate the use of the model and its solution. The results of the example suggests the optimal points in terms of complexity. The maximum complexity is attained in the proportional production situation, while the minimum is in the batch production. Although this conclusion is consistent with the intuition behind the example, the solution is *not* satisfactorily interesting since the minimum complexity is attained on the boundary. Thus, responsiveness is introduced as a trade-off factor to the problem. Based on a multi-objective function, a balanced decision for complexity and responsiveness can be made.

6. General principles of design and operations for assembly systems

   How to reduce variety-induced complexity is a prevailing problem for today's mixed-model assembly system design. This dissertation proposes the new measure and models for complexity analysis, and also suggests some general principles of design and operations for the system design under variety. For example, the principles of "delayed differentiation" and "conditioning reduces entropy" are readily visible from the model. Moreover, some of the principles have already been proved and practiced in industry. The principles derived from the models can assist manufacturing system designers in managing complexity when designing assembly systems, which will result in improved operator and system performance. The results of the research will be highly applicable to all manufacturers who are interested in economically offering product variety without loss of quality and productivity.

5.2 Future Research Directions

Potential areas for future research are suggested below.

1. To investigate the causal relationship between complexity and performance

   Although this dissertation finds a measure of complexity which reflects the intrinsic characteristics of the system, the measure can not directly reflect the performance indices accepted by the industry. Thus it is important to investigate the causal relationship between complexity and performance. However, a typical mixed-model assembly system may involve various performance indices, and the indices may interconnect with each other. Due to the scale of complication and the randomness of manufacturing processes, simple mapping relationship between complexity and performance may not exist. Future research needs to address *how* rather than *whether* the performance indices and complexity are related.

2. To apply the complexity models for product family design

   The complete, system level complexity model presented in Eqs. (2.25) and (2.26) of Chapter 3 suggest the application of complexity analysis for product family design. The complete model is capable of incorporating the detailed information of process flexibility and commonality, and evaluating their impact on complexity. Such detailed information may be reviewed and revised during the early product design phase in order to reduce complexity later in manufacturing. Therefore, an effective handler could be provided from the model to design product family with the simultaneous consideration of manufacturing complexity. Future research needs to address how this concurrent design process should be conducted.

3. To apply the complexity models for factory design

   The complexity models are capable of evaluating complexity at each station, and can be used for factory design. One of the tasks in factory design is to determine the deployment of error-proof devices. The error-proof devices are used to assist operators to make choices, thus reduce complexity. However, on the plant floor, the resources for installing error-proof devices are limited. For example, one of the resources is the Programmable Logic Controller (PLC) capacity. In a line zone, the PLC capacity refers to the number of control nodes available from the PLC to the error-proof devices, and this number is limited. Additionally, the error-proof devices are expensive to purchase, install, and use. Thus, an optimal deployment plan of the error-proof devices is needed to minimize complexity with resource constraints. Another task in factory design is to determine the system configurations or layout of stations. Since different configurations have profound impact on the performance of the system [28], selecting an assembly system configuration other than a pure serial line may help reduce complexity. For instance, according to the complexity "influence index" analysis in Chapter 2, using parallel workstations at the later stages of a mixed-model assembly process reduces the number of choices on these stations if we can properly route the variants at the joint of the ramified paths, see Fig. 2.9. However, balancing such types of manufacturing systems will be a challenge since the system configuration is no longer serial [26]. A novel method for task-machine assignment and system balancing needs to be developed to minimize complexity while maintaining manufacturing system efficiency.

4. To extend the approach of complexity analysis for other processes in mixed-model assembly

   Complexity analysis performed in this dissertation is based on an intrinsic complexity measure derived from the "physics" of the mixed-model assembly process. The "physics" is limited to the choices that operators have to make. Such approach is generic and could be extended and applied to other processes in the mixed-model assembly process. For each of the processes, the complexity measure could be defined differently, since the mechanism through which variety causes complexity are different. For example, a part supply processes with product variety can have high complexity because of large number of variants and fluctuating demands. Thus a complexity measure may be established based on the product rate variation (PRV) [35, 29]. The PRV measures the variation in the rate at which different part variants are consumed at a station. Thus a lower PRV means a steady (rather than varying) part supply. The steady supply requires a simple material replenishment schedule, which leads to low complexity for the process.

5. To apply the complexity models for the "whole" supply chain

   From a supply chain perspective, this dissertation deals with the part delivery process from the line side storage to the assemblage by operator choices. Other processes are also essential to the performance of the "whole" supply chain. For example, the part supply process discussed in the pervious item deals with the part of supply chain from the supermarket in the plant to the line side storages. Similar complexity analysis could also be pursued for the part of supply chain from the outside suppliers to the supermarket. The author has made preliminary attempts in collaboration with other researchers [22, 50] in this area to conduct a complete complexity analysis for the "whole" supply chain with product variety.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] C. Becker and A. Scholl. A survey on problems and methods in generalized assembly line balancing. *European Journal of Operational Research*, 168:694–715, 2006.

[2] J. Benders and M. Morita. Changes in toyota motors' operations management. *International Journal of Production Research*, 42(3):433–444, 2004.

[3] L. Bianco, A. Mingozzi, S. Ricciardelli, and M. Spadoni. Exact and heuristic procedures for the traveling salesman problem with precedence constraints based on dynamic programming. *INFOR*, 32(1):19–31, 1994.

[4] R. R. Bishu and C. G. Drury. Information processing in assembly tasks – a case study. *Applied Ergonomics*, 19(2):90–98, 2003.

[5] A. Bourjault. *Contribution a une approche mthodologique de assemblage automatis: Elaboration automatique des squences opratoires.* PhD thesis, Universit de Franche-Comt, Besanon, France, 1984.

[6] S. K. Card, T. P. Moran, and A. Newell. *The Psychology of Human-Computer Interaction.* Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1983.

[7] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory.* John Wiley & Sons, 2 edition, 1991.

[8] T. L. De Fazio and D. E. Whitney. Simplified generation of all mechanical assembly sequences. *IEEE Journal of Robotics and Automation*, RA-3(6):640–658, December 1987.

[9] A. V. Deshmukh, J. J. Talavage, and M. M. Barash. Characteristics of part mix complexity measure for manufacturing systems. In *IEEE International Conference on Systems, Man and Cybernetics*, volume **2**, pages 1384–1389, 1992.

[10] A. V. Deshmukh, J. J. Talavage, and M. M. Barash. Complexity in manufacturing systems, part 1: Analysis of static complexity. *IIE Transactions*, 30(7):645–655, 1998.

[11] M. Dorigo and L. M. Gambardella. Ant colonies for the travelling salesman problem. *BioSystems*, 43:73–81, 1997.

[12] A. Drexl and A. Kimms. Sequencing JIT mixed-model assembly lines under station-load and part-usage constraints. *Management Science*, 47(3):480–491, 2001.

[13] H. A. ElMaraghy, O. Kuzgunkaya, and R. J. Urbanic. Manufacturing systems configuration complexity. In *Annals of the CIRP*, 2005.

[14] M. L. Fisher and C. D. Ittner. The impact of product variety on automobile assembly operations: Empirical evidence and simulation. *Management Science*, 45(6):771–786, 1999.

[15] M. L. Fisher, A. Jain, and J. P. Macduffie. Strategies for product variety: Lessons from the auto industry. In E. H. Bowman and B. M. Kogut, editors, *Redesigning the Firm*, chapter 6. Oxford University Press, 1995.

[16] H. Fujimoto and A. Ahmed. Entropic evaluation of assemblability in concurrent approach to assembly planning. In *Proceedings of the IEEE International Symposium on Assembly and Task Planning*, pages 306–311, 2001.

[17] H. Fujimoto, A. Ahmed, Y. Iida, and M. Hanai. Assembly process design for managing manufacturing complexities because of product varieties. *International Journal of Flexible Manufacturing Systems*, 15(4):283–307, 2003.

[18] S. M. Gatchell. The effect of part proliferation on assembly line operators' decision making capabilities. In *Proceedings of the Human Factors Society, 23rd Annual Meeting.* Santa Monica: The Human Factors Society, 1979.

[19] S. Gupta and V. Krishnan. Product family-based assembly sequence design methodology. *IIE Transactions*, 30:933–945, 1998.

[20] M. Held and R. M. Karp. A dynamic programming approach to sequencing problems. *Journal of the Society for Industrial and Applied Mathematics*, 10(1):196–210, 1962.

[21] W. E. Hick. On the rate of gain of information. *Journal of Experimental Psychology*, 4:11–26, 1952.

[22] S. J. Hu, X. Zhu, H. Wang, and Y. Koren. Product variety and manufacturing complexity in assembly systems and supply chains. *Annuals of CIRP*, 57:45–48, 2008.

[23] R. Hyman. Stimulus information as a determinant of reaction time. *Journal of Experimental Psychology*, 45:188–196, 1953.

[24] R. R. Inman and D. M. Schmeling. Algorithm for agile assembling-to-order in the automotive industry. *International Journal of Production Research*, 41(16):3831–3848, November 2003.

[25] A. Jessop. *Informed Assessments: An Introduction to Information, Entropy, and Statistics.* Ellis Horwood, New York, 1995.

[26] J. Ko and S. J. Hu. Balancing of manufacturing systems with asymmetric configurations for delayed product differentiation. *International Journal of Production Research*, 45, 2007.

[27] Y. Koren. *The Global Manufacturing Revolution: Product-Process-Business Integration and Reconfigurable Systems.* University of Michigan, 2006. ME/MFG 587 course pack.

[28] Y. Koren, S. J. Hu, and T. W. Weber. Impact of manufacturing system configurations on performance. *Annals of the CIRP*, 47(1):369–372, 1998.

[29] W. Kubiak. Minimizing variation of production rates in Just-In-Time systems - a survey. *European Journal of Operational Research*, 66:259–271, 1993.

[30] H. L. Lee and C. S. Tang. Modeling the costs and benefits of delayed product differentiation. *Management Science*, 43(1):40–53, 1997.

[31] P. De Lit, A. Delchambre, and J. Henrioud. An integrated approach for product family and assembly system design. *IEEE Transactions on Robotics and Automation*, 19(2):324–334, April 2003.

[32] Y. Liu. Queueing network modeling of elementary mental processes. *Psychological Review*, 103(1):116–136, 1996.

[33] J. P. MacDuffie, K. Sethuraman, and M. L. Fisher. Product variety and manufacturing performance: Evidence from the international automotive assembly plant study. *Management Science*, 42(3):350–369, 1996.

[34] C. Merengo, F. Nava, and A. Pozzetti. Balancing and sequencing manual mixed-model assembly lines. *International Journal of Production Research*, 37(12):2835–2860, 1999.

[35] J. Miltenburg. Level schedules for mixed-model assembly lines in Just-In-Time production systems. *Management Science*, 35(2):192–207, 1989.

[36] J. Miltenburg, G. Steiner, and S. Yeomans. A dynamic-programming algorithm for scheduling mixed-model, Just-In-Time production systems. *Mathematical and Computer Modelling*, 13(3):57–66, 1990.

[37] K. G. Murty. *Network Programming.* Englewood Cliffs, N.J.: Prentice Hall, 1992.

[38] H. Nakazawa and N. P. Suh. Process planning based on information concept. *Robotics & Computer-Integrated Manufacturing*, 1(1):115–123, 1984.

[39] A. Niimi and Y. Matsudaira. Development of a vehicle assembly line at toyota: Worker-orientated, autonomous, new assembly system. In K. Shimokawa, U. Jurgens, and T. Fujimoto, editors, *Transforming Automobile Assembly: Experience in Automation and Work Organization.* Springer Verlag, Berlin, 1997.

[40] B. J. Pine. *Mass Customization: The New Frontier in Business Competition.* Harvard Business School Press, Boston, 1993.

[41] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[42] B. Rekiek, P. De Lit, and A. Delchambre. Designing mixed-product assembly lines. *IEEE Transactions on Robotics and Automation*, 16(3):268–280, 2000.

[43] A. Scholl. *Balancing and Sequencing of Assembly Lines.* Heidelberg; New York: Physica-Verlag, 1999.

[44] A. Scholl and C. Becker. State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. *European Journal of Operational Research*, 168:666–693, 2006.

[45] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 1948.

[46] S. Sivadasan, J. Efstathiou, A. Calinescu, and L. Huaccho Huatuco. Advances on measuring the operational complexity of suppliercustomer systems. *European Journal of Operational Research*, 171:208–226, 2006.

[47] N. P. Suh. *The Principles of Design.* Oxford University Press, New York, 1990.

[48] M. M. Tseng and J. Jiao. Design for mass customization. *Annals of the CIRP*, 45(1):153–156, 1996.

[49] P. M. Vilarinho and A. S. Simaria. A two-stage heuristic method for balancing mixed-model assembly lines with parallel workstations. *International Journal of Production Research*, 40(6):1405–1420, 2002.

[50] H. Wang, X. Zhu, S. J. Hu, and Y. Koren. Complexity analysis of assembly supply chain configuraions. In *Proceedings of the 9th Biennial ASME Conference on Engineering Systems Design and Analysis*, 2008.

[51] A. T. Welford. *Fundamentals of Skill.* Methuen, London, 1968.

[52] R. S. Woodworth. *Experimental Psychology.* Holt, New York, 1938.

[53] C. A. Yano and R. Rachamadugu. Sequencing to minimize work overload in assembly lines with product options. *Management Science*, 37(5):572–586, May 1991.

[54] X. Zhu, S. J. Hu, Y. Koren, and S. P. Marin. Modeling of manufacturing complexity in mixed-model assembly lines. *Journal of Manufacturing Science and Engineering*, 130(5):051013–10, 2008. Also appears in the Proceedings of 2006 ASME International Conference on Manufacturing Science and Engineering.

[55] X. Zhu, S. J. Hu, Y. Koren, S. P. Marin, and N. Huang. Sequence planning to minimize complexity in assembly mixed-model lines. In *Proceedings of the 2007 IEEE International Symposium on Assembly and Manufacturing*, 2007.

[56] X. Zhu, H. Wang, S. J. Hu, and Y. Koren. Build sequence scheduling to minimize complexity in mixed-model assembly lines. In *Proceedings of the 9th Biennial ASME Conference on Engineering Systems Design and Analysis*, 2008.