ALGORITHMS FOR OPTIMAL PROCESS SCHEDULING USING NONLINEAR MODELS

by

ZHANG Xueya

March 1995

A thesis submitted for the degree of Doctor of Philosophy of the University of London
and for the Diploma of Membership of the Imperial College

Department of Chemical Engineering and Chemical Technology
Imperial College of Science, Technology and Medicine
London SW7 2BY, UK

BIBL.
LONDIN.
UNIV.

To my parents

ABSTRACT

There has been considerable progress in developing practical algorithms for the optimal scheduling of batch processes. But in order to maintain the problem as a mixed integer linear programming one, sufficient assumptions have to be made. This essentially limits the scope to the case of fixed recipes for the tasks, allowing only variation of batch sizes or throughput. However, further gains are possible by adjusting operating conditions to adapt to resource availability and timing restrictions.

The changing of operating conditions not only introduces significant nonlinearities but also requires a continuous time formulation rather than the discrete time approach used so successfully for fixed recipes.

In the increasingly common situation of customized products made to order, significant scheduling problems also arise in managing continuous multiproduct plants, and many processing sites have a mixture of batch and continuous plants producing a variety of products.

The objective of this work is therefore to develop a general mathematical formulation and solution approaches for dealing with these more general optimal operation problems. It would also be preferable to have a unified approach for different situations.

Different ways of formulating the scheduling problem based on continuous time are presented. Basically, these fall into two categories: those based on a State-Task Network (STN) process representation and those based on a Resource-Task Network (RTN) process representation. The flexibility of the proposed formulations in tackling a wide variety of situations is analyzed. The features of the different formulations are assessed to demonstrate the ensuing complications of the problem and the factors that affect the solution efficiency.

Finally, based on an extension of the RTN concept, a unified formulation for determining the optimal operating conditions for a mixed production facility, comprising multipurpose plant for both batch and continuous operations is generated. This yields a large mixed integer nonlinear programming problem. Exploiting the underlying problem structure reflected in this formulation, procedures for problem size reduction are discussed and alternative approaches for the efficient solution of the resulting optimization problem

are developed.

The novel features of the proposed technique are discussed. Areas needing further investigations are suggested.

ACKNOWLEDGEMENT

I would like to thank sincerely my supervisor Professor R.W.H. Sargent for his excellent supervision, invaluable support and continuous encouragement and interest throughout the course of this work.

I must also express my gratitude to Dr. C.C. Pantelides and Dr. N. Shah for their useful suggestions and various helps.

I would like to thank Prof. C.A. Floudas and Prof. I.E. Grossmann for their enthusiasm on this project and the time they spent in the interesting discussions with me during their visiting at Imperial College.

The financial support from the Centre for Process Systems Engineering at Imperial College is gratefully acknowledged.

Many friends and colleagues helped to make my stay in London an enjoyable one. In particular, thanks must go to Ana Barbosa-Póvoa, Eva Sørensen and Lazaros Papageorgiou.

I would like to thank my parents and my brother for their love, understanding and encouraging.

Finally, thanks Quanshi for everything.

Contents

Abstract 3

Acknowledgement 5

Contents 6

1 Introduction 11

1.1 Thesis Outline 12

2 Literature Review 14

2.1 General Process Scheduling Problem 14

2.2 Multiproduct Batch Plants 15

2.2.1 Effect of Storage Policies 15

2.2.2 Plants with Parallel Units 18

2.3 Multipurpose Batch Plants 19

2.3.1 Short-term Scheduling 19

2.3.2 Campaign Planning/Scheduling 21

2.4 Continuous Multiproduct Plants 22

2.5 General Literature Review 23

3 STN Representation and Discrete Time Formulation 24

3.1 The State-Task Network 24

3.2 Discrete-Time Formulation 26
3.2.1 Time Representation 28
3.2.2 Fundamental Constraints 28
3.2.3 Key Features 31
3.3 Requirements for a General Optimal Operation Algorithm 32

4 Continuous Time Representations – STN Framework 35

4.1 Basic Issues about the Formulation 35
4.2 Fixed Event Time Sequence (FET) 37
4.2.1 Time Representation 37
4.2.2 Variables 37
4.2.3 Constraints 39
4.2.4 Objective Function 43
4.2.5 Linearization Using Special Ordered Set Constraint Properties . . . 44
4.2.6 Problem Reformulation 46
4.2.7 Examples for Variable Processing Times 47
4.3 Variable Event Time Sequence (VET) 53
4.3.1 Time Representation 53
4.3.2 Variables 53
4.3.3 Constraints 55
4.3.4 Objective Function 61
4.3.5 An Illustrative Example 62
4.4 Operation Sequence 64
4.4.1 Units *vs* Tasks 64
4.4.2 Variables 66
4.4.3 Constraints 68
4.4.4 Ordering of System Events 72

4.4.4.1 Special Ordered-Set Constraints 73
4.4.4.2 Direct and Indirect Link to Times 73
4.4.5 Constraint Linearization 78
4.4.6 Objective Function 81
4.4.7 Tightening of the Formulation by Adding Integer Cuts 82
4.4.8 An Illustrative Example 84
4.5 Summary and Discussions 84

5 Extensions of the Basic Formulations **86**
5.1 Plant Connectivity 86
5.2 Operational Constraints 88
5.2.1 Task Precedence Concept 90
5.2.2 Generalized Operational Constraint 92
5.2.2.1 Frequency-dependent Cleaning 93
5.2.2.2 Time-dependent Cleaning 93
5.2.2.3 Precedence Constraints 94
5.3 Unavailability of Processing Equipment 96
5.3.1 Under FET 96
5.3.2 Under VET 96
5.3.3 Under Operation Sequence 97
5.4 Storage Tasks 97
5.4.1 General Storage Tasks 98
5.4.2 Material Held in Units After Processing 99
5.4.3 Storage of Feed Material 99
5.5 Continuous Tasks and Material Transfers 100
5.5.1 Under FET and VET 101
5.5.2 Under Operation Sequence 103

5.6 Critiques of the *STN* Framework 106

6 Optimal Operation for Mixed Production Facilities 108

6.1 A New View for Production Facilities 109

6.1.1 Processes 109

6.1.2 Resources and Resource Types 109

6.1.3 Tasks and Operations 111

6.1.3.1 Tasks and Resource Utilization 111

6.1.3.2 Task Operations 112

6.2 Resource Transfer Time Sequence 114

6.2.1 Time Representation 114

6.2.1.1 Separate Resource Transfer Time Sequences 114

6.2.1.2 Task Start-time Sequences 114

6.2.1.3 Plant Delivery Sequences 115

6.2.2 Variables 115

6.2.3 Task Operation Models 116

6.2.3.1 General Task Operation Models 116

6.2.3.2 Blending Tasks 117

6.2.3.3 Storage Tasks 118

6.2.4 Plant Delivery 120

6.2.5 Resource Balance 121

6.2.6 Time Assignment 121

6.2.7 Objective Function 122

6.3 Unified Formulation for Process Scheduling 123

6.3.1 Time Representation 123

6.3.2 Mathematical Model 124

6.3.3 Problem Simplifications 127

6.3.3.1 Fixed Event Time Sequence (FET) 129
6.3.3.2 Variable Event Time Sequence (VET) 129
6.3.4 Problem Solution 130
6.3.4.1 X-Reduction 130
6.3.4.2 ET-Reduction 132
6.3.5 Problem Reformulation 132
6.3.6 Integer Cuts and Valid Inequalities 133
6.4 Computational Results 134
6.4.1 Effect of X-Reduction for Original FET Algorithm 136
6.4.2 Effect of Reduction Techniques for Original VET Algorithm 136
6.4.3 Effect of the Reformulation 137
6.5 A Case Study – Scheduling of Fast Moving Consumer Goods (FMCG) Manufacturing 141
6.5.1 Description of the Process 141
6.5.2 Modelling 143
6.5.3 Solution 145
6.6 Conclusion 147

7 Conclusions and Future Work **148**

7.1 General Conclusions 148
7.2 Future Work 150

References **152**

A Comparing FET Procedure with Discrete Time Formulation **158**

B Derivation of Bounds for Original Time Constraints **161**

C Derivation of Bounds for Reformulated Time Constraints **163**

Chapter 1

Introduction

Three methods of processing are used in manufacturing on an industry scale: discrete, batch and continuous. In the chemical process industries, both batch and continuous processes are widely used.

Batch production has long been a common procedure for the manufacture of chemicals, particularly those of low volume and high value-added. The key factor favouring batch operation is the flexibility in utilizing the various available resources offered by batch operations. This is reflected in its easy reconfiguration and adaptation to allow for production modifications and to cover a wide range of operating conditions within the same plant configuration. Typical examples of batch processes are fine and specialty chemicals manufacture, pharmaceuticals production, polymerization, food and biochemical processing. The flexibility of batch processes usually implies a large number of degrees of freedom in the operating of batch plants. Therefore, an accelerating interest of research in production planning and scheduling methodology has been stimulated both in academia and in industry. Extensive reviews have highlighted the research activities and publications (Ku *et al.*, 1987; Reklaitis, 1991; Reklaitis, 1992; Rippin, 1992; Pantelides, 1993; Musier and Evans, 1990).

On the other hand, the common situation for quickly responding to demand changes and to the processing of a variety of products also results in a current trend in the chemical industry to move towards continuous flexible plant. Therefore, significant scheduling problems also arise in managing continuous multiproduct plants. Furthermore, many pro-

cessing sites have a mixture of batch and continuous plants producing a variety of products. Compared to the rapid developments in the exciting field of batch process scheduling, the scheduling of flexible continuous plant has received much less attention in the literature concerning the chemical processing industry.

Essentially, the problem of plant scheduling is defined as the allocation of available production resources over time to perform a set of processing tasks to best satisfy some set of criteria, subject to precedence constraints and resource constraints. It is required whenever there is competition among activities for limited resources which are available over a finite time period. Here the resources are considered as the arbitrary means which the tasks compete for.

Although there has been considerable progress in the development of practical algorithms for optimal scheduling, most of the published works have focussed on the solution of various specific problems with different restrictions. Most significantly, there has been ignorance of the consideration of adjusting operating conditions subject to resource availability and timing restrictions.

The objective of this thesis is to devise a unified framework for process scheduling to cover the broad range of real problems. General formulations and solution algorithms are posed using continuous time representations, which simultaneously determine the optimal schedules and operating conditions of mixed production facilities, comprising multipurpose plant for both batch and continuous operations with different resource constraints.

1.1 Thesis Outline

The rest of this thesis is organized as follows.

Chapter 2 reviews the existing techniques for the solution of process scheduling problem, the emphasis is given to the process system.

In *Chapter 3*, the concept of the State-Task Network (*STN*) is explained and the discrete-time formulation for scheduling under fixed product recipes is briefly summarised. Finally, the requirement of developing the continuous time representation to deal with more general optimal operation problems is discussed.

In *Chapter 4*, a variety of possible continuous time representation schemes are devel-

schemes
oped within the *STN* framework. The advantages and disadvantages of individual are examined through the development of continuous time formulations. Then in *Chapter 5*, the flexibility of various formulations given in *Chapter 4* is presented through the extension of them to consider more detailed situations arising in process scheduling. The limitations of the *STN* framework are analysed, giving rise to the need for a unified and equally general formulation covering individual features.

Chapter 6 describes the new view of mixed production facilities using a Resource-Task Network (*RTN*). Two types of time representations are developed. Formulations based on either time scheme are able to unify all problem features described in the previous chapters. The resulted problem is simplified to make the solution of the optimization problem more tractable.

Chapter 7 gives general conclusions and perspectives for future work.

Chapter 2

Literature Review

2.1 General Process Scheduling Problem

There are four components of scheduling lending themselves to mathematical description and computer based solution approaches (Reklaitis, 1992):

(i) *select* a number of activities to be executed within a given time horizon to fulfill given production requirements;

(ii) *assign* resources to these activities subject to the total resource constraints;

(iii) *sequence* the selected activities subject to the precedence constraints;

(iv) *determine* the precise timing of the utilization of resources by these activities.

Similar to the flowshop and the jobshop in manufacturing, batch chemical plants are classified into the multiproduct plant and the multipurpose plant. Various forms of general scheduling problems are commonly classified as those concerned with the "equipment dominant" case and those concerned with the "resource constrained" case (Reklaitis, 1992).

Since most scheduling techniques used in the chemical processing industry followed those from the discrete-part manufacturing industry, the so-called equipment dominant case has received the bulk of the attention in the scheduling literature, in which the key issues to be considered are processing units and products (instead of processing tasks), and sometimes shared intermediate storage. If, on the other hand, the amount of other

resource (*e.g.* utilities, manpower) is limited, this may impose severe restrictions on the processing equipment allocation and timing of processing tasks. Then the resource constrained scheduling is addressed. In this case, the presence of limited resource introduces the need to consider the impact of the availability of sufficient resource on the start and stop times or even execution of tasks which will require resource.

However, realistic industrial applications rarely fall neatly within any single type of scheduling problem. It could be argued that the complexity and diversity of process scheduling problems naturally leads to an equally diverse set of methodologies and techniques for their solution (Pantelides, 1993). Much of the literature followed the approach of making various degrees of simplifications and restrictions of the above four components to reduce the problem dimensionality and complexity and make it more tractable.

It is not the intention here to provide a complete reference to all published works. We will only focus on the recent progress in process scheduling relative to chemical process engineering. This review chapter will be divided into three sections, with each section focussing on different themes beginning with the scheduling in multiproduct batch plants, continuing with the scheduling in multipurpose batch plants and finally the scheduling for continuous multiproduct plant.

2.2 Multiproduct Batch Plants

The key characteristics of multiproduct plant are that the product recipes are highly similar in structure, equipment is customarily grouped into processing stages with or without parallel units and products always follow the same route through the plant.

Plants operating under the multiproduct strategy bear similarities to the "flowshop" problem in discrete part manufacturing. By far the greatest portion of the literature has focussed on various forms of multiproduct plants, so there are many different scheduling techniques available.

2.2.1 Effect of Storage Policies

From the scheduling point of view, the installation of intermediate storage will remove the need for perfect synchronization of the task sequences and increase the flexibility of

operation of batch plants.

No Intermediate Storage (NIS) is the case where the intermediate material is stable, but no storage vessels are available. However, the material may temporarily be stored in the processing vessels that produce it. Suhami and Mah (1981) give an implicit enumeration scheme using a probabilistic lower bound, which is based on a recurrence relation to determine the completion time. A node at the k-th level in the tree represents assignments of products to the first k positions in the sequence. A branch represents an assignment to the $(k+1)$-th position in the sequence. The branch and bound algorithm with backtracking is used, in which a probabilistic upper bound attempts to fathom partial assignment.

Zero-Wait (ZW) is the case where either the intermediate material is unstable and must immediately be processed by the next task in the processing sequence, or where no storage vessels are available and the processing units that produced are unsuitable for the storage.

Using linear constraints for scheduling in their earlier work (Birewar and Grossmann, 1989b) on the design of multiproduct plant, Birewar and Grossmann (1989a) reported a very compact LP formulation with plant consisting of a single unit per stage operating under ZW policy. It is shown that the makespan minimization can be replaced by overall cycle time minimization. By aggregating the batches in terms of products, it has also been shown that this formulation is especially efficient for problems involving a large number of batches of relatively few products. The solution of the proposed model provides a family of schedules, among which one or several schedules can be generated by a proposed graph representation. Later on, this work was extended to propose a multi-period LP formulation for simultaneous product planning and scheduling of multiproduct batch plant (Birewar and Grossmann, 1990). The formulation accommodated both single product campaign (SPC) with ZW policy and multiproduct campaign (MPC) with either ZW or UIS (unlimited intermediate storage).

Pekny and Miller (1990; 1991) take advantage of the fact that the ZW flowshop problem is equivalent to the asymmetric travelling salesman problem (ATSP), and present an exact parallel technique using effective Bounding and Branching procedures.

Taking account of the effects of limited availability of resources, Egli and Rippin (1986) develop an integrated heuristic-based package. Attention was focussed on multiproduct

plants, with UIS storage for stable intermediates and ZW for unstable ones. A batch sequence was timed by shifting the batches along the time axis to match as closely as possible cumulative demand curves expressing due dates for product requirements while satisfying additional constraints such as working practices, product priority sequence, raw material deliveries and limitation on shared resources. Alternative batch sequences are generated by enumeration, and an objective function is minimized including costs of storage, changeovers and utilities. Both the allocation of equipment to different tasks, and the associated batchsizes are fixed.

Finally, a most general situation is the *Finite Intermediate Storage (FIS)* case, in which certain amount of storage capacity is suitable to hold material of a certain state. The interstage storage capacity is often measured in terms of the number of units (each unit holds a batch) rather than the actual quantities.

Ku and Karimi (1988) considered the case that a finite number of storage units is available between stages, and storage units only receive finished products from the upstream batch unit. In this case, the completion times for a given product sequence are determined by recurrence relations that take into account the intermediate storage. The sequencing problem is solved using the Rapid Access with Extension Search (RAES), a two-phase heuristic with a recursive improvement strategy. The recurrence relation for completion time can be easily extended to include non-zero transfer and set-up times (Rajagopalan and Karimi, 1989).

Taking tardiness minimization as objective function, Ku and Karimi (1991) also considered the scheduling of an order-driven plant. A heuristic technique was proposed, in which an initial schedule was generated by ratio of due-date to tardiness penalty, and then iteratively improved by heuristic rules.

A more general case has been studied by Wiede and Reklaitis (1987), in which the available storage is not localized between two adjacent consecutive stages, but storage capacity is shared by all processing units. They proposed an approximate method to find the storage assignment for earliest completion time.

By measuring the intermediate storage in terms of actual quantities rather than number of batches, Kuriyan *et al.* (1987) also investigated a simple multiproduct plant consisting of two stages with FIS by discrete/continuous simulation.

2.2.2 Plants with Parallel Units

The presence of parallel processing units offers more flexibility in operation of batch plants. These units are run either in-phase or out-of-phase to increase the productivity either in capacity or in time.

Earlier works in this area have been confined to networks in which the parallel units in a stage are identical and are employed strictly out-of-phase. Kuriyan and Reklaitis (1989) have presented a series of approximate algorithms for general network flowshops operating under ZW or UIS, with the objective of minimizing makespan. A two-level approach was proposed which decomposes the problem into sequencing and evaluation steps. Heuristics (*e.g.* dispatching rules, bottleneck sequencing, local search sequencing and best-fit sequencing) are used for the determination of the sequence first, then the strategy of PAT - one product at a time, or SAT - one stage at a time are used for the sequence evaluation.

Musier and Evans (1989; 1991) investigate a more complicated case of single stage with non-identical parallel units and propose the HIM (Heuristic Improvement Method). Further extension is made to multistage flowshop with intermediate storage between each stage of parallel units. The procedure considered all single and pairwise interchanges of product batches, and repeated application of the HIM to improve an initial sequence (generated by a random problem). The objective function is restricted to the minimization of the total tardiness.

Wellons and Reklaitis (1989a) formulated the simple path sequence generation problem as a mixed integer nonlinear programming problem (MINLP). Serial processing networks with ZW and NIS policies are considered. Equipment is arranged into groups operating in parallel and strictly out-of-phase, where items belonging to the same group must operate in-phase. Since a large number of degenerate solutions exist due to rotational and operational equivalences, an explicit enumeration procedure was developed for identifying only unique simple path sequences. A reformulation of the MINLP was given by the same authors (Wellons and Reklaitis, 1989b) which restricts the feasible region of the model to distinct path sequences.

2.3 Multipurpose Batch Plants

Similar to a jobshop, multipurpose plant may be dynamically reconfigured in order to produce vastly different products. In this case, different products can follow different routes through the plant and therefore only utilize a subset of the stages of equipment. Even different batches of the same product may follow different routes through the plant.

The plant operation is also influenced by the business environment. Usually, plants are operated in one of the two distinctive modes: short-term mode or campaign mode. The former is often chosen when the demands for different products are subject to rapid change. The time horizon for scheduling is relatively short, ranging from one to several weeks. The latter are preferable for stable or long-term predictable product demands. In this operation mode, the planning horizon is usually divided into a number of long time periods (campaigns), each dedicated to a subset of products.

2.3.1 Short-term Scheduling

Rich and Prokopakis (1986) propose a MILP formulation for the short-term scheduling of multipurpose batch facilities. The scheduling problem is the determination of production quantities to satisfy the customer orders by assigning units to tasks, and sequencing problem is the determination of the actual timing of tasks on units. They assume that sufficient storage capacity is available to meet inventory requirements. The model is not very general in that each product is produced via a unique production route and each step in the process is associated with a pre-specified set of processors. But this results in very few binary variables for realistic problem and can provide solutions in reasonable time.

Based on the graphical representation of the scheduling problem for multipurpose plant, Jänicke (1984) proposes a heuristic algorithm, in which a critical path method is used to schedule the batch with a given starting time, and the overall scheduling problem is solved by heuristics to change the starting times of batches.

While the above works have various limitations on the scheduling problems tackled, a much more general algorithm was addressed by Kondili *et al.* (1988). The algorithm is based on the State-Task-Network (*STN*) representation of batch and semi-continuous plants. A uniform discrete time representation is used for the MILP formulation which

accommodates a wide range of scenarios such as the optimization of unit-task allocation, the splitting/merging of batches and the requirements for utilities *etc.* Following this work, several techniques were introduced for improving the solution efficiency (Sahinidis and Grossmann, 1991b; Shah *et al.*, 1993a). This work will be further discussed in the next chapter.

An attempt at an interval based approach was reported by Zentner and Reklaitis (1991a; 1991b). The key component of their work is the development of an interval processing framework which leads to a nonuniform discretization of time. The reasoning about coupled resource demands is handled by the interval processing through examination of the resource availability profiles. Resources are treated as nonrenewable (NRR) and renewable (RR) ones. Interval processing consists of three levels considering: the nonrenewable resources, the non-coupled renewable resources and the shared effects of renewable resources. Search in the NRR level establishes the earliest starting time for each task demanding such resources. The next level considers the resource demand by each task independently and chooses one interval for each task. Finally, the earliest starting and latest finishing times of each bounded interval established in the second level are refined by considering different modes of overlapping in time. A set of precedence constraints is also produced in the third searching level. The interval processor is a heuristic rule guided tree search. The exact task starting time and the assessment of the need for storage is finally solved using an LP formulation.

Recently, Pinto and Grossmann (1994a) addressed the short-term scheduling of batch plants of multiple stages with equipment in parallel. The total demand of a certain product is divided into several orders. They assume that each batch (order) can be processed by only one unit of every stage it goes through with fixed batch size. The problem is formulated as a mixed integer linear programming problem. Two solution strategies were proposed using pre-ordering constraints and decomposition scheme respectively. The key feature of this algorithm is the use of the continuous time representation with parallel time coordinates for units and tasks. However, resource constraints and intermediate storage have not been included.

2.3.2 Campaign Planning/Scheduling

Mauderli and Rippin (1979) employed an evolutionary enumerative technique, in which all possible groups (assigned to the batch) and routes are enumerated, and then inefficient routes are eliminated. Heuristic rules are used to avoid the potentially explosive number of routes. An LP screening procedure was used to obtain these dominant combinations of routes (campaigns). If change-over cost needs to be included, a MILP is used to allocate the available production time to a specific dominant campaign.

Lázaro *et al.* (1985; 1989) use the job-shop algorithm to handle the resource constraints for the UIS case. Firstly an exhaustive enumerative procedure is used to generate the route (in-phase operation only). Next, dominant routes are chosen on a greedy principle based on the value of the selection index, which incorporates the production rate, equipment utilization fraction, and production cost. The task sequences and scheduling for each period were accomplished with a heuristic jobshop scheduling algorithm, taking into account constraints on manpower and on the fixed or variable utility requirements.

Wellons and Reklaitis (1991a; 1991b) augmented their previous work on single product production-line scheduling to develop a rigorous MINLP formulation for single and multiple product campaign formation. The assignment of specific equipment items to specific groups in specific production routes is made automatic, and not only the route generation but also the most efficient route scheduling and selection are considered. Taking the number of groups available in each stage as complicating variables, Bender's decomposition is used to solve the resulting nonconvex and large scale MINLP. The latter work used the multiobjective approach based on a non-inferior set estimation method to obtain the dominant route combinations.

Shah and Pantelides (1991) proposed a single level MILP formulation for the simultaneous campaign formation and planning problem. They consider serial processing networks operating in a mixed ZW/UIS mode. Nonidentical parallel equipment items are restricted to operate in-phase. Limited availability levels of utilities and storage capacity for stable intermediates between campaigns may also be taken into account. By a relatively straightforward extension of the short-term scheduling formulation of Kondili *et al.* (1988), an approach suitable for general processing networks with general equipment utilization is reported by Shah *et al.* (1993b) to determine the cyclic operating schedules within a single

campaign.

Papageorgiou (1994) presented a general formulation for the multiple campaign planning and scheduling for multipurpose batch plants. This work removed some of the key limitations of earlier approaches (*e.g.* restrictions on the possible campaign structure and resource utilization). The problem has been formulated as a MILP model, encompassing both overall planning and detailed scheduling aspects simultaneously. An efficient mathematical decomposition approach is developed in order to solve the resulted large scale MILP problem efficiently.

Tsirukis *et al.* (1993) have posed the scheduling of a multipurpose batch plant with resource constraints as a MINLP model. Because of the high combinatorial complexity and nonconvexity, a formulation specific decomposition strategy is developed, which partitioned the solution of the original MINLP into two levels: the MINLP subproblem and the MINLP master problem. The subproblem determines the allocation of equipment and resources to specific orders and provides a lower bound, while the master problem determines the assignment of orders to campaigns providing an upper bound.

2.4 Continuous Multiproduct Plants

Multiproduct and multipurpose continuous plants are flexible manufacturing facilities used in a wide range of processes. They differ from the batch plants in that the processing stages of interest operate in a continuous fashion. Similarly, the scheduling problem also involves a combinatorial part (sequencing of products) and a continuous part (duration of product runs and inventory levels). There are a few studies reported in the chemical engineering literature considering the scheduling of continuous multiproduct plants.

Sahinidis and Grossmann (1991a) considered the cyclic scheduling of single stage continuous multiproduct plants with parallel lines operated independently with independent inventory. Each line uses a characteristic fixed cyclic schedule which involves a subset of the products. A cost based objective function is used, which accounts for productions, sequence dependent transition, and inventory holding costs. The problem is formulated as a large scale MINLP problem, and a solution method based on generalized Benders decomposition is developed.

Pinto and Grossmann (1994b) propose an optimization model and solution method for the cyclic scheduling of multistage continuous multiproduct plants. It is assumed that each product is processed only once at each stage. A large scale MINLP model is developed to handle intermediate storage between stages as well as sequence dependent changeovers. It is shown that the nondifferentiabilities in the inventory levels are best handled by the introduction of 0-1 variables rather than smooth approximations. A decomposition strategy is employed for the solution by combining the generalized Benders decomposition and the outer-approximation methods.

The short-term scheduling of continuous multiproduct plants under resource constraints was addressed by Kondili *et al.* (1993b). The formulation uses a continuous time representation and results in a MILP. It is applied to the short term scheduling of a cement plant.

Most recently, Schilling *et al.* (1994) reported the optimal scheduling of multipurpose continuous plants with processing networks of arbitrary generality. The plant considered comprises several multipurpose units, each of which may be suitable for more than one task. A task may be performed by different units, possibly at the same time. The use of other resource such as utilities and manpower are also taken into account. With approximation of inventory costs to avoid nonlinearities, the problem is formulated as a MILP.

2.5 General Literature Review

The above discussion presents the state of the art in scheduling of chemical batch and continuous plants. Overall, the research work reported to date falls into two broad categories: one is geared to the development of special-purpose algorithms for well-specified and frequently-occurring scheduling problems. The other is to develop general mathematical formulations. Due to the high combinatorial nature of the scheduling problem, approximate methods are usually applied to solve the scheduling problem which is thus confined to finding sub-optimal solutions.

Chapter 3

STN Representation and Discrete Time Formulation

In this *Chapter*, the original work of *gBSS* is reviewed. The concept of the State-Task network representation for batch process recipes is described followed by a summary of the discrete time formulation for scheduling with fixed product recipes. Then key features of the formulation as well as various possible extensions are surveyed. Finally limitations of the discrete time formulation are discussed, giving rise to the requirement of developing a continuous time formulation to deal with more general scheduling problems.

3.1 The State-Task Network

As pointed out by Kondili *et al.*(1988), the first stage of solving the scheduling problem is to develop a general representation of the process recipes. However, as argued by these authors, although the traditional flowsheet type representation is adequate for serial process structures, it will result in ambiguities for more complex process recipes involving, for example, shared intermediates, material recycles *etc.*

In order to establish a suitable and flexible representation of general batch process recipes, the novel concept of State-Task Network (*STN*) was proposed by Kondili *et al.* (1988) to represent process recipes in terms of the physico-chemical transformations required to convert raw materials into final products. The *STN* is a directed graph with two types of distinctive nodes: the *state* nodes denoted by a circle and the *task* nodes denoted by a rectangle box. By explicitly including both the individual batch operations (*tasks*) as well

as the feeds, intermediates and final products (*states*) as network nodes, *STN* provides an ambiguity-free representation of complex processes involving shared intermediates, multiple processing routes to the same intermediate or product, and material recycles. There are two rules followed in the construction of the State-Task Network:

- A task has as many input (output) states as different types of input (output) material.
- Two or more streams entering the same state are necessarily of the same quality. If mixing of different streams is involved, this operation should form a separate task.

The construction of a typical State-Task network from given product recipes can be illustrated through the following example.

Example: Consider the production of two products from five intermediate materials which are produced from two feedstocks according to the product recipe:

(i) Task1: Mix 50% raw material Feed1 and 50% Feed2 and let them react for 2 hours to form intermediate in state S1. S1 is not stable and must be used up immediately upon completion of processing .

(ii) Task2: React Feed2 for 1.5 hours to form two intermediates in states S2 (70%) and S3 (30%).

(iii) Task3: Process S1 for 1 hour to obtain intermediate S4.

(iv) Task4: Mix 50% S3 and 50% S4 and let them react for 2 hours to get impure intermediate S5.

(v) Task5: Distill impure S5 to separate pure product prod2 (80%) after 1 hour and pure intermediate S4 (20%) after 2 hours. Recycle intermediate S4.

(vi) Task6: Let S2 react for 3 hours to obtain product prod1.

The above process is represented by the *STN* in Figure 3.1

However, the *STN* only gives a description of the process recipe, and does not contain any details regarding plant resources. Plant resources are related to either tasks or states in the network structure. Each item of equipment is assumed to be suitable for a subset of

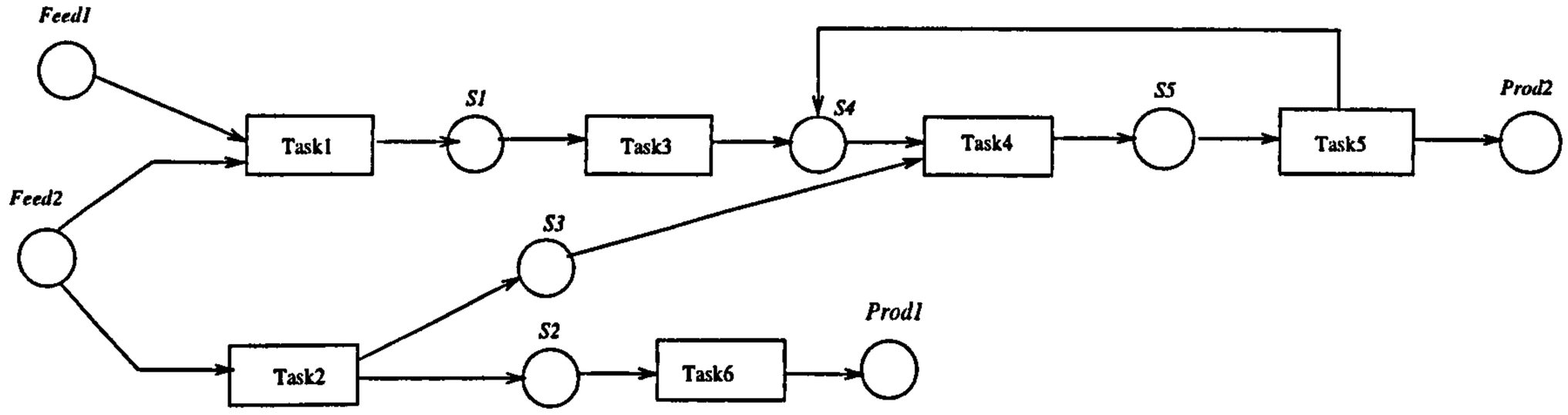


Figure 3.1: State-Task Network Representation

tasks described in the *STN*, with a given minimum and maximum capacity for each of such tasks. Each task may also employ a subset of the available utilities for their performance. These may be continuous (*e.g.* steam, cooling water) or discrete (*e.g.* manpower, auxiliary equipment). The amount of each utility consumed may vary over the duration of the task. On the other hand, the consumption of utilities during task processing may be either fixed or variable, depending on batch size, or a combination of the two. Given amounts of dedicated storage capacity are assumed to be associated with the various states of raw materials, intermediates and final products in the process.

Therefore, a number of parameters associated with the available resources must be attached separately. These parameters are summarised in Table 3.1 using a slightly different notation from Kondili *et al.* in order to be consistent throughout the thesis.

3.2 Discrete-Time Formulation

Given the representation of the processing structure and the information on associated plant resources, the next step is to formulate the scheduling problem mathematically in order to give a systematic and general way of dealing with the optimal process scheduling problem. Further assumptions are made in the mathematical formulation of Kondili *et al.*:

(i) no preemptive operation is allowed (*i.e.* no task may be interrupted, once started);

(ii) the task processing times are independent of operating conditions (*e.g.* batch-size), and must be determined *a priori*. Indeed, all data are deterministic and fixed over the time horizon of interest;

Table 3.1: Problem Parameters for Formulation of Kondili *et al.*

		A Task i is characterized by:
S_i	:	set of states used or produced by task i
ρ_{sik}	:	proportion of batch in state s used or produced by task i at time l_{ijk} (negative for feeds, positive for products)
l_{ijk}	:	transfer time for task i relative to the start of processing for $k \in \{K_i: 0, 1, \ldots, k_i^{max}\}$
$\bar{l}_{ij}$	:	processing time for task i in unit j
J_i	:	set of equipment items suitable for task i
α_{rijk}	:	fixed demand factor for utility r during interval k when task i is performed in unit j
β_{rijk}	:	variable demand factor for utility r during interval k when task i is performed in unit j
		A State s is characterized by:
I_s	:	set of tasks using or producing material in state s
C_{sl}^{max}	:	maximum storage capacity for state s
c_{sl}	:	unit cost or price of state s at beginning of time period l
C_{s0}	:	amount of state s available at beginning of time horizon
		A Unit j is characterized by:
I_j	:	set of tasks for which unit j is suitable
V_{ij}^{max}	:	maximum capacity of unit j when used for performing task i
V_{ij}^{min}	:	minimum capacity of unit j when used for performing task i
		A Utility r is characterized by:
I_r	:	set of tasks using or producing utility r
R_{rl}^{max}	:	maximum available level of utility r over interval l
c_{rl}	:	unit cost of utility r at time l

(iii) all material transfers are assumed to be instantaneous.

3.2.1 Time Representation

The fundamental issue of the formulation of any scheduling problem concerns the time representation. In the work of Kondili *et al.*(1988), the time is treated in a discrete manner *i.e.* the time horizon is divided into a number of time intervals of equal and fixed duration. Thus a single time grid for all system events can be provided. The discrete time representation is shown in Figure 3.2. The duration of time interval ΔT is chosen in a way that makes all processing times an integral multiple of ΔT. Events such as the starting and finishing of tasks, changes in resource availability and/or unit costs are all forced to occur only at the interval boundaries.

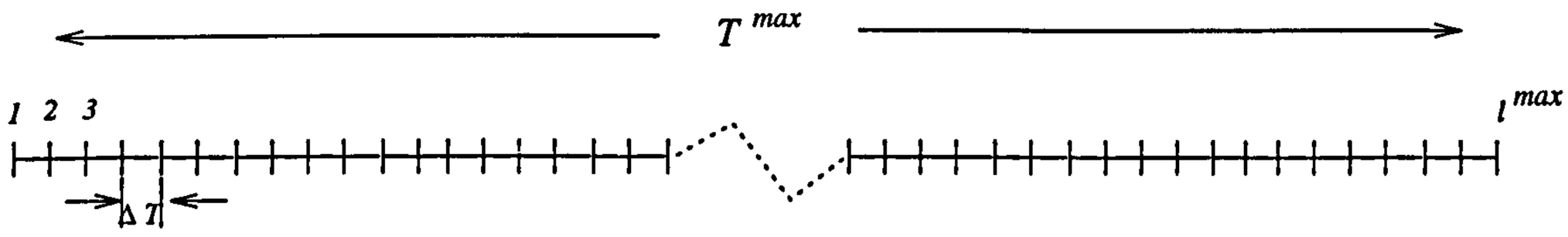


Figure 3.2: Discrete Time Representation

Based on the *STN* representation and discretization of the time domain, binary variables are introduced to control the allocation of units to tasks in a specific time period. Continuous variables are introduced to control the amount of material involved with a task, the amount of material stored and amount of utilities consumed over each interval. By assuming fixed product recipes (*e.g.* fixed task duration), the short-term scheduling problem is formulated as a mixed integer linear programming (MILP). Key problem variables used in this formulation are listed in Table 3.2 using the consistent notation for this thesis.

3.2.2 Fundamental Constraints

Given the information in Table 3.1 and Table 3.2, and a time horizon of interest, the scheduling problem is then to determine the timing of operation for each unit and the flow of material through the network, so as to optimise a given objective function. The fundamental constraints which must be satisfied are formulated by Kondili *et al.*(1988) as:

Table 3.2: Problem Variables for Formulation of Kondili *et al.*

1. Binary decision variables dealing with the allocation of units to tasks:
$$W_{ijl} = \begin{cases} 1 & \text{if task } i \text{ starts in unit } j \text{ at the beginning of time period } l \\ 0 & \text{otherwise} \end{cases}$$

2. Continuous variables dealing with the amount of material processed:
B_{ijl} = amount of material which starts undergoing task i in unit j at the beginning of time period l.

3. Continuous variables dealing with the amount of material stored:
C_{sl} = amount of material stored in state s over time period l.

4. Continuous variables dealing with the utility utilization:
R_{rl} = total demand for utility r over time period l.

5. Continuous variables dealing with amount of material delivered or received:
D_{sl} = amount of material in state s delivered or received at beginning of time period l (negative for receipts, positive for deliveries)

Unit Allocation Constraints

At any time l, each unit can at most start one task. Once a unit starts processing a task, it can not be allocated to process any other tasks before completing the current one. Originally this was ensured through:

$$\sum_{i' \in I_j} \sum_{l'=l}^{l+\bar{l}_{ij}-1} W_{i'jl'} - 1 \leq M(1 - W_{ijl}) \qquad \forall j, l, i \in I_j \tag{3.1}$$

However, the large value of M in this constraint causes inefficiency in the solution procedure. In order to address this difficulty, Sahinidis and Grossmann (1991b) first proposed a reformulated allocation constraint by using a disaggregated form of (3.1):

$$\sum_{i \in I_j} W_{ijl} \leq 1 \qquad \forall j, l \tag{3.2}$$

$$W_{ijl} + W_{i'jl'} \leq 1 \qquad \forall j, l, i, i' \in I_j, l' = l-1, \ldots, l - \bar{l}_{i'j} + 1 \tag{3.3}$$

This results in a smaller integrality gap but may also lead to a much larger number of constraints than that for the original form. Therefore, the later work of Shah *et al.* (1993a) addressed this problem by presenting a more aggregated form using the full "backward

aggregation" over time:

$$\sum_{i \in I_j} \sum_{l'=l}^{l-\bar{l}_{ij}+1} W_{ijl'} \leq 1 \qquad \forall j, l \tag{3.4}$$

This form of allocation constraint proved to be the tightest of all of the forms above and also gives the lowest number of constraints in the MILP problem.

Capacity Constraint

The amount of material processed at a time is bounded by the minimum and maximum capacities of the processing units to be used, and the amount of material stored in a state s is bounded by the maximum storage capacity to be occupied:

$$W_{ijl} V_{ij}^{min} \leq B_{ijl} \leq W_{ijl} V_{ij}^{max} \qquad \forall j, i \in I_j, l \tag{3.5}$$

$$0 \leq C_{sl} \leq C_{sl}^{max} \qquad \forall s, l \tag{3.6}$$

Material Balance Constraints

The net increase of the amount of material in state s, in storage at time l, is equal to the difference between the amount produced and that consumed. Any external deliveries or receipts of material to/from the state at the beginning of time period l must also be taken into account:

$$C_{sl} = C_{s,l-1} - D_{sl} + \sum_{i \in I_s} \sum_{j \in J_i} \sum_{k \in K_i} \rho_{sik} B_{ij,l-l_{ijk}} \qquad \forall s, l \tag{3.7}$$

Utility Constraints

The consumption of a specific utility must not, at any time, exceed its availability:

$$0 \leq R_{rl} \leq R_{rl}^{max} \qquad \forall r, l \tag{3.8}$$

$$\text{where} \qquad R_{rl} = \sum_{i \in I_r} \sum_{j \in J_i} \sum_{k \in K_i} (\alpha_{rijk} W_{ij,l-l_{ijk}} + \beta_{rijk} B_{ij,l-l_{ijk}}) \qquad \forall r, l \tag{3.9}$$

Objective Function

The objective function is an economic criterion of maximizing the total value added during the time horizon of interest.

$$\max \left\{ \sum_{s} c_{sl^{max}} C_{sl^{max}} + \sum_{s} \sum_{l=1}^{l^{max}-1} c_{sl} D_{sl} - \sum_{r} \sum_{l=1}^{l^{max}-1} c_{rl} R_{rl} \right\} \tag{3.10}$$

Therefore a multi-period mixed integer linear programming (MILP) model is developed which may be solved using the branch-and-bound procedure. However, for realistic

applications, the MILP resulting from this formulation may involve several thousands of binary decision variables. A number of measures were presented by later work of Shah *et al.* (1993a), aiming both at reducing the integrality gap of the mathematical formulation (*e.g.* the allocation constraint as described before), as well as accelerating the solution procedure using a modified branch-and-bound method. The combination of these techniques have greatly improved the efficiency of the solution procedure allowing relatively large problems to be solved with reasonable computational effort.

3.2.3 Key Features

The basic feature of the formulation is the mapping of all events onto a single uniformly spaced time sequence. By using a simple task model and assuming fixed processing times for tasks, the order of all events is defined simply by the binary decision variables W_{ijl}. The formulation can also accommodate various process characteristics, especially those listed below:

- tasks producing several outputs (intermediates, semi-products or final products) at different times (*e.g.* batch distillation);
- recycling, mixing and splitting of material between tasks;
- limited availability of utilities and/or manpower;
- the use of processing units for temporary storage of material;
- more than one product being manufactured simultaneously;
- successive batches following different routes through the plant;
- parallel processing of tasks;
- mixed intermediate storage policy;
- sequence/frequency-dependent cleaning of equipment items;
- due dates on deliveries of material;
- a variety of optimization objectives based on economic or performance measures.

Several works have since been reported in addressing a wide variety of problems. These are either achieved by the modification of the *STN* concept or extensions of the original discrete time formulation. The problems tackled include:

- periodic production scheduling (Shah *et al.*, 1993b);
- campaign planning and scheduling (Papageorgiou, 1994);
- the modification of the *STN* to consider limited plant connectivity (Crooks, 1992; Kondili *et al.*, 1993a);
- introduction of the Unit-State-Task Network concept in dealing with general cleaning requirements and/or pre-conditions for unit utilization (Crooks, 1992);
- maintenance scheduling (Dedopoulos and Shah, 1995), pipeless plant scheduling (Pantelides *et al.*, 1993) and energy integrated scheduling (Papageorgiou *et al.*, 1993).

Overall, the generality of the State-Task Network process description provides a flexible framework to accommodate various situations.

3.3 Requirements for a General Optimal Operation Algorithm

The discretization of the time domain uses a convenient time-unit, *i.e.* the highest common factor (HCF) of all processing time intervals. This provides a simple mechanism for positioning all operations in the presence of globally shared resources, the utilization of which is distributed over time. For fixed processing times, it has the key advantage of using a single integer variable to position all events, based on identifying task start-times.

However, the disadvantage resulting from this is that for a wide range of time intervals, or small differences between them, the HCF can be very small. Especially, when semi-continuous processes are taken into consideration, the continuous task (*e.g.* the continuous transfer of materials) can only be approximated as successive batch tasks with small processing time intervals. Since variables from both binary and continuous sets as well as related constraints must be defined at each discretization point on the time axis, the problem size will become extremely large because of the choice of a small time-unit, but

with no events at most times. Otherwise the problem would be how to strike a balance between optimality and problem size, in which case the interval size and rounding of processing times become important issues.

On the other hand, the idea of using a discrete time formulation to yield a mixed integer linear programme requires sufficient assumptions. Most important is that these assumptions make its suitability questionable for dealing with more general optimal operation problems. The key assumptions of the fixed problem data such as the fixed processing times has essentially limited the scope to fixed product recipes, allowing only the batch size or throughputs to be optimized, ignoring the coupling of operating conditions.

To tackle more general optimal operation problems, such assumptions must be relaxed, and also more realistic and detailed task models should be used. Obviously, this will preclude the use of the HCF and destroy the simplicity of relative positioning of all events using task start-time, since not only the task start-time but also its duration, and hence the transfers of material during task operation and even the amounts and times of resource usage become variable. They will be driven both by plant schedules as well as operating conditions and have to be explicitly handled in the problem formulation. Therefore, if the uniformly spaced discrete time is retained, the rounding problem and the problem of choosing an appropriate interval will always exist, and in addition a mechanism to decide relative positions of transfers is also necessary.

Otherwise, since the changes of resource balance are restricted to a finite set of times when at least one system event is taking place, this should be the only times where verification for resource limitations is necessary. Therefore, one cure for the above problems would be to use a variable event-time sequence in which the number of event-times is limited to the number of actual events, which of course also needs a mechanism for the ordering of events.

The above statement clearly implies the demand for the development of a general continuous time algorithm for optimal process operation. However, both the use of a variable event-time sequence and the relaxation of fixed product recipes (*e.g.* fixed processing time) will result in significant nonlinearities and complications. As a first step for the development of the general algorithm, a sophisticated and efficient strategy for the ordering of individual events is required to keep the ensuing complications to a minimum level.

In the next *Chapter*, alternatives for formulating the basic scheduling problem within the *STN* framework using a continuous time representation will be presented.

Chapter 4

Continuous Time Representations – STN Framework

In this *Chapter*, we present mathematical formulations for the short term scheduling problem using continuous time representations. Various continuous time representations are developed and used as the time reference scheme. Their advantages and disadvantages are examined through the development of mathematical formulations based on these time schemes. The efficiencies of individual formulations are compared by analysing their underlying structures.

The time representations presented in this *Chapter* can be divided into two classes: the event-time sequence and the operation sequence. The former focusses on mapping all activities of a plant onto a common time sequence, either uniformly spaced and fixed (*FET*) or non-uniformly spaced and varying (*VET*). The latter is based on the view of process equipment as a kind of "critical" resource and concentrates on units performing tasks following different unit *Operation Sequences*.

4.1 Basic Issues about the Formulation

In the formulations presented in this chapter, *STN* is adopted for the representation of general batch plants. Information regarding the available resources associated with either tasks or states in the network are assumed to be given by the parameters described in *Chapter* 3.

Although based on the discrete time formulation the *STN* concept has been extended to accommodate various situations within the same framework, such as sequence and frequency dependent cleaning/maintenance, plant connectivity and storage tasks. For the sake of clarity to examine individual time representation schemes and the understanding of the underlying structure of different formulations, we shall limit our scope for the moment to the most basic issues. However, this will not limit the flexibility of the formulation, as we will show later in *Chapter* 5, the formulations presented in this chapter can be easily extended to accommodate these situations.

The key feature of our formulation is to relax the assumption of the discrete time formulation, where only fixed processing times are taken into account. We extend the formulation to deal with a more detailed task model where task processing time will depend on the operating conditions during its processing. Besides this, some straightforward extensions to the discrete time formulation are also included from the beginning of the formulation. These are listed below:

- processing times and resource (steam, manpower, *etc.*) utilization not only depend on tasks but also vary with different processing units;

- a sequence of plant deliveries of materials with time-dependent prices (either feeds or intermediate or final products) within the horizon;

- varying resource availabilities/price and storage capacities during the horizon;

- successive transfers of material, with transfer time also varying with operating conditions, and varying usage of resources during task processing. This will be referred to as a task subsequence later on in this chapter.

During the period of scheduling, different activities can take place either within the plant or between the plant and the outside environment. We regard all these activities as "*system events*". According to the issues identified above, the internal activities include the transfer of intermediates between processing units and/or between processing units and dedicated storage tanks, the starting up and shut down of processing units, the change-over of processing tasks in units, the consuming and generating of resources such as steam, manpower, changing of availability/price of resources and storage capacities. The activities

between plant and environment include purchasing of raw material or resources from outside suppliers, and the delivery of final products according to customer orders.

4.2 Fixed Event Time Sequence (FET)

4.2.1 Time Representation

We postulate a sequence of event times, T_l, along the time axis within the horizon. The event times are in strictly ascending order with fixed uniform spacing, ΔT. The time of occurrence of all system events should map onto one of the event times in the sequence. Because the processing time is no longer fixed, the possibility of using a pre-fixed highest common factor of all processing time intervals is precluded. Instead, the number of event times postulated will mainly depend on the requirement of accuracy of the solution.

This time representation is similar to that used for discrete time formulation. But from Figure 4.1 it can be seen that variation of processing time could change the order of individual events, and hence the profile of related resources.

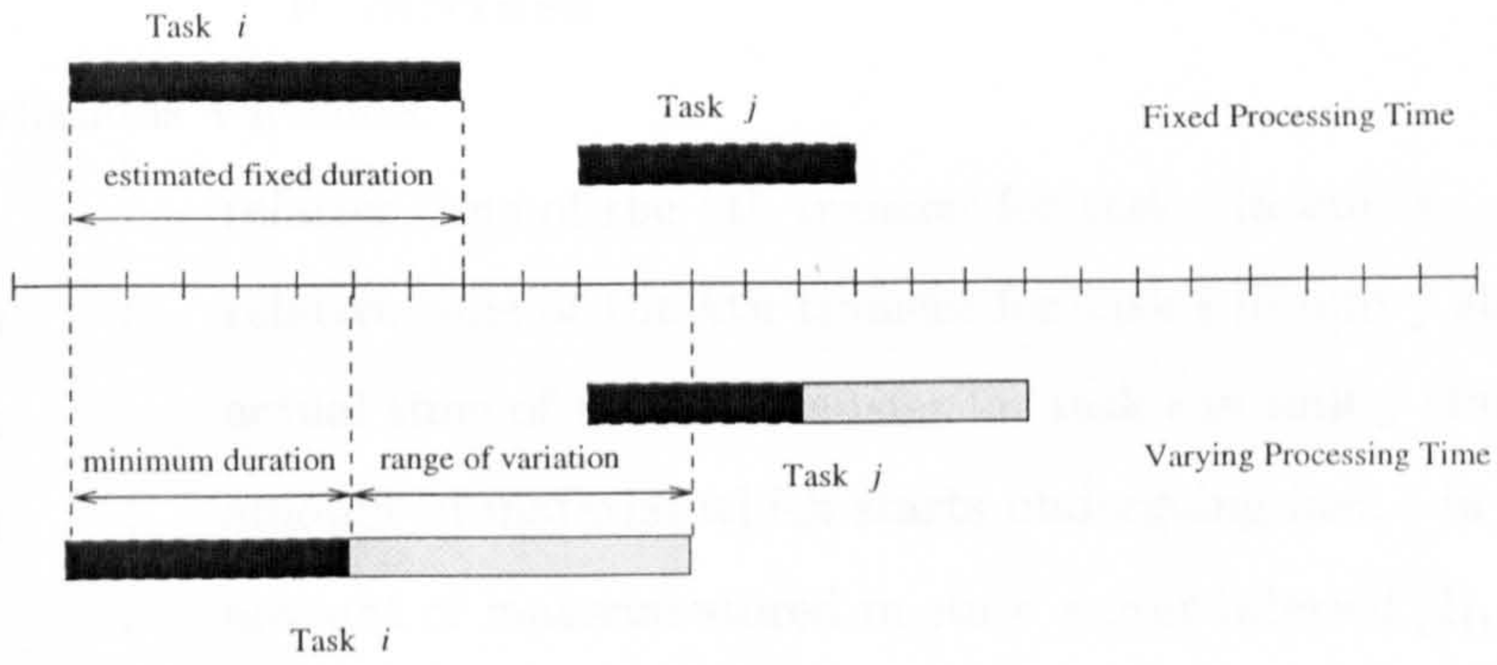


Figure 4.1: Fixed Event-time Representation

4.2.2 Variables

Since we also consider the possibility of resource generation during the execution of a task, the demand factors α_{rijk}, β_{rijk} used in the discrete time formulation (*see* Table 3.1) are redefined to give a unique representation for consumptions and generations. Also redefined are the limits for consumption/generation.

α_{rijk} : the fixed factor for generation/consumption of resource r in interval k for task i in unit j

β_{rijk} : the variable factor for generation/consumption of resource r in interval k for task i in unit j

$R_{rl}^{min}, R_{rl}^{max}$: the minimum and maximum limits for the consumption/generation of resource r over interval $[T_{rl},\ T_{r,l+1})$

The key optimization variables are identified as follows:

1. Binary variables indicating the activity of task i in unit j at event-time T_l:

$$W_{ijl} = \begin{cases} 1 & \text{if task } i \text{ starts in unit } j \text{ at } T_l \\ 0 & \text{otherwise} \end{cases}$$

2. Binary variables describing the order of events:

$$X_{ijkll'} = \begin{cases} 1 & \text{if task } i \text{ starts in unit } j \text{ at } T_l \text{ and has the } k\text{th transfer at } T_{l'} \\ 0 & \text{otherwise} \end{cases}$$

3. Binary variables indicating event-time within a task interval:

$$\tilde{X}_{ijkll'} = \begin{cases} 1 & \text{if } T_{l'} \text{ is in interval } k \text{ for task } i \text{ starting in unit } j \text{ at } T_l \\ 0 & \text{otherwise} \end{cases}$$

4. Continuous Variables:

τ_{ijk} : relative time of the kth transfer for task i in unit j

τ_{ijkl} : relative time of the kth transfer for task i in unit j starting at T_l

t_{ijkl} : actual time of the kth transfer for task i in unit j starting at T_l

B_{ijl} : amount of material which starts undergoing task i in unit j at T_l

C_{sl} : amount of material stored in state s over interval $[T_l, T_{l+1})$

D_{sl} : amount of material in state s delivered or received at T_l

R_{rl} : the level for the generation/consumption for resource r over $[T_l, T_{l+1})$

Keeping in mind that D_{sl} is a unique representation of delivery and receipt of materials, it can be either negative (for receipt) or positive (for delivery); it can also be specified, or a variable to be determined (within specified limits). Similarly, we also consider the situation that resource can both be consumed and generated using R_{rl} as a unique representation with different signs (negative for consumption, positive for generation). By convention, we assume that all other variables are non-negative. The sets and parameters used in our

formulation are consistent with those used in the discrete time formulation as described in the previous chapter.

4.2.3 Constraints

Having identified the key variables for optimization, we can then describe in detail all the appropriate constraints that need to be considered for the scheduling problem.

Time Constraints

No task should be preempted, therefore once a task starts at T_l, all of its associated sequence of transfers should take place at a time period of τ_{ijkl} from its starting time:

$$\tau_{ijkl} = W_{ijl}\tau_{ijk} \qquad \forall i, l, j \in J_i, k \in K_i \tag{4.1}$$

$$t_{ijkl} = T_l + \tau_{ijkl} \qquad \forall i, l, j \in J_i, k \in K_i \tag{4.2}$$

We assume that the task duration, and hence the transfer times of material/resources for successive subsequences, have a maximum and minimum limit, that is:

$$\tau_{ijk}^{min} \leq \tau_{ijk} \leq \tau_{ijk}^{max} \tag{4.3}$$

Note that by convention, $\tau_{ij0} = 0$, giving T_l the starting time of task i in unit j.

For each task starting at event time T_l, its succeeding transfers can take place at one and only one later time $T_{l'}$. Therefore, we have the "special-ordered set" constraint as following:

$$\sum_{l' \geq l} X_{ijkll'} = W_{ijl} \qquad \forall i, j \in J_i, k \in K_i, l \tag{4.4}$$

By convention, $X_{ijkll} = 0$, for $k > 0$, and $X_{ij0ll} = W_{ijl}$. This constraint also ensures that the successive transfers of a task can only happen if the task started at an earlier time.

In order to decide the sequence of events, we have to identify the actual transfer time with one and only one of the common event-times, $T_{l'}$. This can be achieved by the constraint below using (4.4):

$$\tau_{ijkl} = \sum_{l' \geq l} X_{ijkll'}(T_{l'} - T_l) \qquad \forall i, j \in J_i, k \in K_i, l \tag{4.5}$$

This constraint ensures that when $X_{ijkll'} = 1$, the actual transfer time t_{ijkl} will be mapped onto event-time $T_{l'}$.

However, since the task processing time and hence its successive transfer times vary with operating conditions, they may no longer be integral multiples of the fixed interval. In order to synchronise them with the common event-times T_l, τ_{ijkl} has to be rounded up to the next nearest event time. This can be achieved by replacing (4.5) with:

$$\sum_{l' \geq l} X_{ijkll'}(T_{l'-1} - T_l) < \tau_{ijkl} \leq \sum_{l' \geq l} X_{ijkll'}(T_{l'} - T_l) \;\; \forall i,j \in J_i, k \in K_i, l \tag{4.6}$$

Note that from the limit constraint (4.3), an upper and lower bound can be set on the range of the event-times which transfer k should be mapped onto or be rounded up to:

$$\underline{l}_{ijk} = \left\lceil \frac{\tau_{ijk}^{min}}{\Delta T} \right\rceil \qquad \bar{l}_{ijk} = \left\lceil \frac{\tau_{ijk}^{max}}{\Delta T} \right\rceil \tag{4.7}$$

Therefore the binary variable $X_{ijkll'}$ need only be introduced if $l + \underline{l}_{ijk} \leq l' \leq l + \bar{l}_{ijk}$. This is also true for all the constraints involving $X_{ijkll'}$.

Unit Allocation Constraints

To resolve the conflict of unit allocation, we must ensure that at each time a unit can at most be occupied by one step of one task, achieved by adding the following constraints:

$$\sum_{i \in I_j} \sum_{k < k_i^{max}} \sum_{l \leq l'} \tilde{X}_{ijkll'} \leq 1 \qquad \forall j, l' \tag{4.8}$$

In fact, the binary variable $\tilde{X}_{ijkll'}$ (which can be viewed as the unit occupation variable here) can be determined in terms of:

$$\begin{aligned} \tilde{X}_{ijkll'} &= \sum_{l'' \leq l'} X_{ijkll''} + \sum_{l'' > l'} X_{ij,k+1,ll''} - W_{ijl} \\ &= \sum_{l'' \leq l'} X_{ijkll''} - \sum_{l'' \leq l'} X_{ij,k+1,ll''} \qquad \forall i,j \in J_i, k < k_i^{max}, l, l' \geq l \end{aligned} \tag{4.9}$$

Note that $\tilde{X}_{ijkll'}$ need not be treated as 0-1 variables, enforcing the bound of $\tilde{X}_{ijkll'} \in [0,1]$ is sufficient.

Unit Capacity Constraints

The amount of material to be processed at any time T_l should be bounded by the minimum and maximum capacity of the unit in use. This leads to the following unit capacity constraints:

$$V_{ij}^{min} W_{ijl} \leq B_{ijl} \leq V_{ij}^{max} W_{ijl} \qquad \forall j, i \in I_j, l \tag{4.10}$$

Delivery Constraints

Generally, the delivery requirement may be given in two types:

(i) a specified amount, or an amount within specified limits $[\hat{D}_{sk}^{min},\ \hat{D}_{sk}^{max}]$ at a specified time t_{sk}^{D};

(ii) a specified amount, or an amount within specified limits $[\tilde{D}_{sk}^{min},\ \tilde{D}_{sk}^{max}]$ over a specified interval $[t_{sk}^{D},\ t_{s,k+1}^{D})$.

Since the event-times are all fixed with uniform spacing, the specific time, t_{sk}^{D}, or interval $[t_{sk}^{D},\ t_{s,k+1}^{D})$, can easily be allocated to the event-times. To meet the requirement of case(i) and (ii), the following constraints should be included:

$$\text{for case (i):} \quad \hat{D}_{sk}^{min} \leq \hat{D}_{sl} \leq \hat{D}_{sk}^{max} \qquad \text{where } T_l = t_{sk}^{D} \tag{4.11}$$

$$\text{for case (ii):} \quad \tilde{D}_{sk}^{min} \leq \sum_{l} \tilde{D}_{sl} \leq \tilde{D}_{sk}^{max} \qquad \text{where } t_{sk}^{D} \leq T_l < t_{s,k+1}^{D} \tag{4.12}$$

(4.12) states that the deliveries for case (ii) may be carried out several times with amount $\tilde{D}_{sl}$ within the specified interval. But it is not necessary to invoke the delivery at each event-time within this interval. The only requirement is that the total amount should be kept within the specified limits.

The total amount of delivery at T_l is made up of the amount to meet the requirement at specified time and the amount to meet the requirement over specified interval, which can be expressed as:

$$D_{sl} = \hat{D}_{sl} + \tilde{D}_{sl} \tag{4.13}$$

Limits may also be made on the total amount delivered at any given time T_l within a specified interval, so:

$$D_{sk}^{min} \leq D_{sl} \leq D_{sk}^{max} \qquad \text{where } t_{sk}^{D} \leq T_l < t_{s,k+1}^{D} \tag{4.14}$$

Material Balance/Storage Capacity Constraints

With an event-time sequence, the storage requirements can easily be checked by an incremental material balance over each event-time interval. The amount of material stored over interval $[T_{l'},\ T_{l'+1})$ (interval l') should be equal to the amount in storage at event-time $T_{l'-1}$, plus the net amount produced into state s at time $T_{l'}$ *(= the amount produced minus the amount consumed)*, minus the net amount of material in state s delivered at the same time. Mathematically, this can be written as:

$$C_{sl'} = C_{s,l'-1} - D_{sl'} + \sum_{i\in I_s}\sum_{j\in J_i}\sum_{k\in K_i}\rho_{sijk}\sum_{l\le l'}B_{ijl}X_{ijkll'} \qquad \forall s, l' \tag{4.15}$$

Note that due to the binary nature of $X_{ijkll'}$, the proportion of batch size B_{ijl} will only be taken into account in the material balance if the appropriate transfer time t_{ijkl} is identified to be $T_{l'}$ $(X_{ijkll'} = 1)$. C_{s0} gives the initial amount of material in state s and is assumed to be known for all states.

The amount of material to be stored should always be kept within the limits of the available storage capacity, C_{sl}^{max}. For the sake of generality, it is also desirable to include the situation where maximum available storage capacity varies over the time horizon. Since the event times are fixed, the times for such changes can be allocated to specific event times, hence there is no extra complexity for the capacity constraint.

$$0 \le C_{sl'} \le C_{sl'}^{max} \qquad \forall s, l' \tag{4.16}$$

Resource Usage Constraints

In addition to processing equipment, the processing tasks of a batch plant may also have to share some common resources, which might be either continuous (*e.g.* steam, cooling water) or discrete (*e.g.* manpower, auxiliary equipment). These resources usually have limited availability which may result in conflicts when they are demanded by several processing tasks simultaneously. It is also possible to generate resources during the execution of some tasks. Therefore, additional constraints must be embedded to account for such situations.

In general, the demand for resources by individual tasks may be time dependent through processing. With the division of tasks into subsequences, these variations can

be viewed as different subsequences of a task. Furthermore, the amount of resources required can be either fixed or variable, which is usually expressed as a fixed term dependent on the unit in use combined with a variable term proportional to the amount of material processed. The total amount of resource r used or generated over event-time interval l' is given by:

$$R_{rl'} = \sum_{i \in I_r} \sum_{j \in J_i} \sum_{k < k_i^{max}} \sum_{l \leq l'} \tilde{X}_{ijkll'} (\alpha_{rijk} + \beta_{rijk} B_{ijl}) \qquad \forall r, l' \tag{4.17}$$

As mentioned in section *4.1*, the maximum amount of resources available may also vary over the time horizon. Again, the intervals for the changes of resource availability can be allocated with the event-time intervals. Therefore, at any time, the total amount of resource r generated/consumed should be kept within the lower and upper limits, which results in the following bounds on $R_{rl'}$:

$$R_{rl'}^{min} \leq R_{rl'} \leq R_{rl'}^{max} \qquad \forall r, l' \tag{4.18}$$

Note that $R_{rl'}$ can be either positive or negative depending on the net amount of generation or consumption. The limits used in the above constraint must not restrict either of these two situations. Therefore, $R_{rl'}^{min}$ should take the negative value of the maximum availability over $T_{l'}$, while $R_{rl'}^{max}$ should take the maximum level of possible resource generation over $T_{l'}$.

4.2.4 Objective Function

A variety of objectives can be optimised to obtain an optimal schedule. The criterion we use here is a measure of the profit of the entire operation.

For simplicity, we now only consider three contributions to the profit. First is the cost of raw material, given by:

$$\texttt{Cost of raw material} = \sum_s (c_{s0} C_{s0} - \sum_l c_{sl} D_{sl}) \tag{4.19}$$

Note that the cost includes both the initial inventory and the receipt of raw materials during the processing. Second is the contribution of the value of selling final products, quantified as below:

$$\texttt{Value of products} = \sum_s (c_{slmax} C_{slmax} + \sum_l c_{sl} D_{sl}) \tag{4.20}$$

Again, both the value of products left in storage at the end of the horizon and the value of delivery of products to customers during the horizon are taken into account. The amount of both receipt and delivery of raw material or final products can vary during the horizon as well. With fixed event-time and fixed price at each time T_l, the above two contributions to the objective function still remain linear.

Third is the cost of resources (*e.g.* steam, manpower) which can be included via a term of the form:

$$\texttt{Cost of resources} = \sum_{r} \sum_{l<l^{max}} c_{rl}(-R_{rl})\Delta T \tag{4.21}$$

where c_{rl} is the unit cost of resource r over the time interval $[T_{rl},\ T_{r,l+1})$. The three terms combine to yield the profit.

4.2.5 Linearization Using Special Ordered Set Constraint Properties

In the *FET* procedure developed above, all the event times T_l are fixed with uniform space ΔT. Therefore the event time T_l is no longer a variable and the time constraints (4.5) and (4.6) become linear. However nonlinearities exist in the time constraints (4.1), material balance constraints (4.15) and resource usage constraints (4.17). Thus this still gives rise to a mixed integer nonlinear programming problem.

It can be seen that the nonlinearities in the formulation appear as bilinear terms, cross products between binary and continuous variables. A commonly used technique in the development of solution procedures for such problems is that of linearization, that is, the construction of an equivalent mixed integer linear representation of the problem.

However, a crucial concern in the construction of the mixed integer linear representation of such mixed integer nonlinear programming problems is how well the linear formulation lends itself to existing solution strategies. Usually, a mixed integer linear representation whose linear programming relaxation closely approximates the convex hull of feasible integer solutions in the vicinity of the maximum is computationally advantageous. Hence, a linearization technique that provides a tight linear programming relaxation, while keeping the problem computationally tractable is highly desired.

One choice of the linearization of the bilinear terms in the MINLP formulation is the use of the linearization techniques of Glover (1975). The main idea was the introduction

of a new continuous variable to represent the cross product. The equivalence between the bilinear term and the new variable was enforced by introducing a set of equivalence constraints. For example, in the material balance constraints (4.15), we define a new continuous variable, $B_{ijkll'}$ as:

$$B_{ijkll'} = B_{ijl} X_{ijkll'} \tag{4.22}$$

By introducing the following constraints, the cross products $B_{ijl} X_{ijkll'}$ in the original material balance constraints can then be equivalently represented by the new variable $B_{ijkll'}$.

$$V_{ij}^{min} X_{ijkll'} \leq B_{ijkll'} \leq V_{ij}^{max} X_{ijkll'} \tag{4.23}$$

$$V_{ij}^{min}(W_{ijl} - X_{ijkll'}) \leq B_{ijl} - B_{ijkll'} \leq V_{ij}^{max}(W_{ijl} - X_{ijkll'}) \tag{4.24}$$

However, we can actually produce a tighter linearization by using the "special ordered set" property in constraint (4.4) and introducing the following constraints instead.

$$\sum_{l'} B_{ijkll'} = B_{ijl} \qquad \forall i, j \in J_i, k \in K_i, l \tag{4.25}$$

$$V_{ij}^{min} X_{ijkll'} \leq B_{ijkll'} \leq V_{ij}^{max} X_{ijkll'} \tag{4.26}$$

As pointed out by Grossmann *et al.* (1992), this scheme needs fewer constraints than the first choice. Furthermore, while a point satisfying the set of constraints of the latter linearization scheme also satisfies the inequalities in the former one, the converse may not be true. Since both linearization techniques ensure the equivalence of the new introduced variable to the cross products, the first one may produce a weaker LP relaxation.

For the linearization of the resource usage constraints (4.17), since $\tilde{X}_{ijkll'}$ can be replaced by the equivalence in (4.9), the bilinear terms are transformed into the same form as those in the material balance constraints (4.15) and neither extra variables nor constraints are necessary for the linearization. Of course, this is at the price of making a set of denser resource usage constraints.

However, time constraints (4.1) can only be linearized through a set of constraints similar to (4.23) and (4.24).

Removing the bilinear terms in (4.15) and (4.17) results in the following linear material

balance and resource usage constraints:

$$C_{sl'} = C_{s,l'-1} - D_{sl'} + \sum_{i\in I_s}\sum_{j\in J_i}\sum_{k\in K_i}\rho_{sijk}\sum_{l\le l'}B_{ijkll'} \qquad \forall s,l' \qquad (4.27)$$

$$R_{rl'} = \sum_{i\in I_r}\sum_{j\in J_i}\sum_{k<k_i^{max}}\sum_{l\le l'}\sum_{l''\le l'}(\alpha_{rijk}(X_{ijkll''} - X_{ij,k+1,ll''}) + \beta_{rijk}(B_{ijkll''} - B_{ij,k+1,ll''})) \qquad \forall r,l' \qquad (4.28)$$

4.2.6 Problem Reformulation

One difficulty involved in the application of the mixed-integer programming techniques is that the problem formulation can have a very large impact on the computational efficiency for the solution. This makes the formulation a crucial step in the application of mixed integer programmes. Therefore, once constraints have been formulated, it is worth considering whether alternative formulations might be better suited for the computation. One emerging rational idea is to reformulate the problem by variable disaggregation.

To reformulate the above MILP, we disaggregate the variable τ_{ijkl} into $\tau_{ijkll'}$ and replace (4.1) and (4.5) as:

$$\tau_{ijkll'} = X_{ijkll'}\tau_{ijk} \qquad (4.29)$$

$$\tau_{ijkll'} = X_{ijkll'}(T_{l'} - T_l) \qquad (4.30)$$

If we further assume linear dependence of task processing time on the batch size, (4.29) can then be expressed as:

$$\tau_{ijkll'} = \alpha_{ijk}X_{ijkll'} + \beta_{ijk}X_{ijkll'}B_{ijl} \qquad (4.31)$$

The batch size variable B_{ijl} can be disaggregated as $B_{ijkll'} = B_{ijl}X_{ijkll'}$, which is in the same form as (4.22). Similar to the original formulation, (4.30) has to be replaced due to the rounding up of resource transfer times. Therefore:

$$X_{ijkll'}(T_{l'-1} - T_l) < \alpha_{ijk}X_{ijkll'} + \beta_{ijk}B_{ijkll'} \le X_{ijkll'}(T_{l'} - T_l) \qquad (4.32)$$

$$i.e. \quad X_{ijkll'}\frac{(T_{l'-1} - T_l - \alpha_{ijk})}{\beta_{ijk}} < B_{ijkll'} \le X_{ijkll'}\frac{(T_{l'} - T_l - \alpha_{ijk})}{\beta_{ijk}} \qquad (4.33)$$

Comparing the above constraint with (4.26), it is evident that (4.33) provides tighter bounds (a larger lower bound and a smaller upper bound) for $B_{ijkll'}$.

It is interesting to note that by simple transformation, the *FET* procedure is mathematically equivalent to the discrete time formulation of Kondili *et al.* (1988) and Shah *et al.*(1993a). The comparison is given in *Appendix A*.

4.2.7 Examples for Variable Processing Times

In this section, two examples are used to test the formulations given above, assuming linear dependence of processing time on batch size, by ±33% from the mean time $\overline{\tau}_{ijk}$ over the range of batch-sizes:

$$\tau_{ijkl} = \alpha_{ijk} W_{ijl} + \beta_{ijk} B_{ijl}$$

$$\text{where} \quad \alpha_{ijk} = \tau_{ijk}^{min} \qquad \beta_{ijk} = \frac{\tau_{ijk}^{min}}{V_{ij}^{max}} \qquad \tau_{ijk}^{min} = \frac{2}{3}\overline{\tau}_{ijk}$$

Example 1. In this simple example, the single product batch process is considered. The raw material is processed through three processing tasks with different processing times. All intermediates can be stored in a dedicated storage tank with finite storage capacity. Although the process itself is of simply serial processing structure, the wide range of the variation of individual processing time serves as a good example to show the effect of variable processing time on the process scheduling.

Example 2. This example describes a multipurpose batch process with two final products produced from three raw materials through heating, reaction and separation. In the separation task, the discharging of final Product2 and intermediate ImpureE are at different times, which provides an example of a task with two subsequences.

Figure 4.2 gives the State-Task Networks for the two examples above. The details of processing equipment, their suitability for task processing, the available storage capacity for each state of material, as well as the mean processing times are given in Table 4.1–4.2.

Both the original and reformulated model are implemented to compare the efficiency of the solution procedure. SCICONIC is used on a Sun Sparc10/41 workstation to solve the LPs and MILPs, with an optimality margin of 5%. Table 4.3 gives the results using the original formulation, while Table 4.4 shows the results using the reformulation. For both examples, it can be seen that the integrality gap and solution time are both greatly reduced by reformulating the problem using variable disaggregation. The optimal schedule and inventory level for intermediates are shown in Figure 4.3 – 4.4.

However, with the increasing requirement of time accuracy and/or the wide range of time variation, the problem size can be "exploded", which makes even the LP problem impossible to solve, let alone the MILP problem. This can be noticed also from the results where lower time accuracy gives smaller and easier problems, while sacrificing the time accuracy and hence the optimality.

Table 4.1: Specifications for Example1

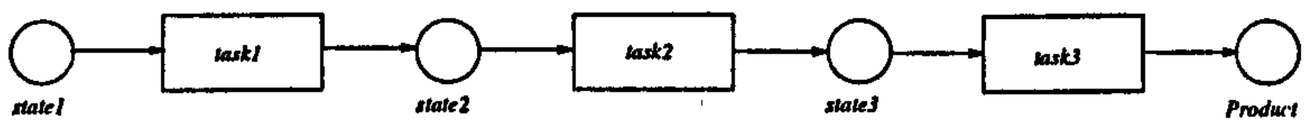
Time Horizon = 12 hrs

Process unit	Capacity (kg)	Suitability	Mean process time $\overline{\tau}_{ijk}$
unit1	100	task1	4.5
unit2	75	task2	3
unit3	50	task3	1.5
Process states	**Storage capacity(kg)**	**Initial amount(kg)**	**Material prices**
state1	Unlimited	Unlimited	0.0
state2	100	0.0	0.0
state3	100	0.0	0.0
Product	Unlimited	0.0	1.0

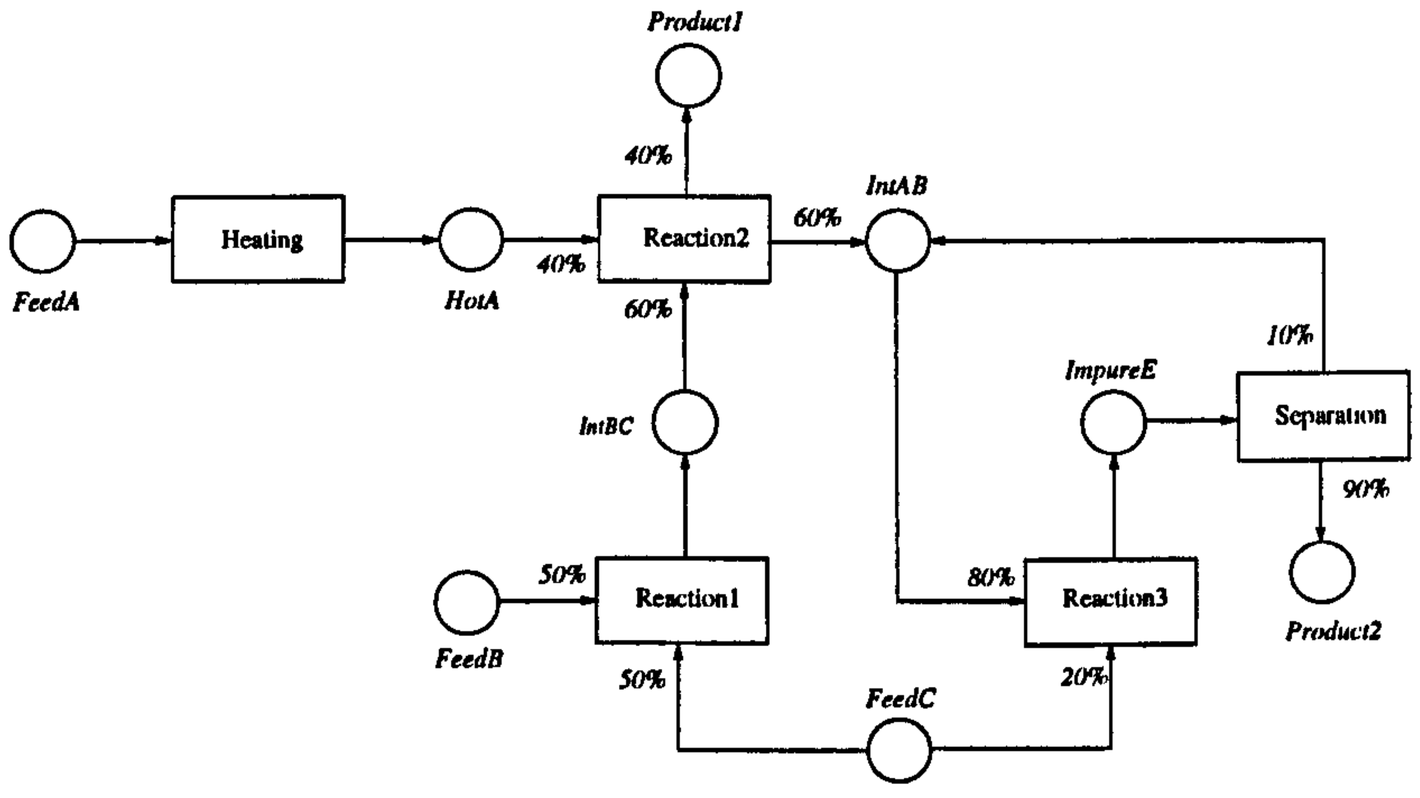
Table 4.2: Specifications for Example2

Time Horizon = 8 hrs

Process unit	Capacity (kg)	Suitability	Mean process time $\overline{\tau}_{ijk}$
Heater	100	Heating	1
Reactor1	50	Reaction1,2,3	2,2,1
Reactor2	80	Reaction 1,2,3	2,2,1
Still	200	Separation	(1 for Product2, 2 for ImpureE)
Process states	**Storage capacity(kg)**	**Initial amount(kg)**	**Material prices**
FeedA	Unlimited	Unlimited	0.0
FeedB	Unlimited	Unlimited	0.0
FeedC	Unlimited	Unlimited	0.0
HotA	100	0.0	0.0
IntAB	200	0.0	0.0
IntBC	150	0.0	0.0
ImpureE	200	0.0	0.0
Product1	Unlimited	0.0	10.0
Product2	Unlimited	0.0	10.0



State-Task-Network for Example1



State-Task-Network for Example2

Figure 4.2: State-Task Network Representation for Examples

BIBL.
LONDIN.
UNIV.

Table 4.3: Results of the Original Formulation(*FET*)

	Example 1		Example 2	
NEI	60	24	40	16
NC	3136	958	4960	1625
NV	3670	862	4945	1285
NIV	1623	345	2108	494
OBJ	66.66	66.66	1412.17	1274.35
*OBJ**	101.91	101.68	1537.19	1524.32
LPs	9018	326	289	220
CPU	12951.7	83.7	46828.2	143.5

Table 4.4: Results of the Reformulation(*FET*)

	Example 1		Example 2	
NEI	60	24	40	16
NC	5482	1288	7456	1925
NV	4933	1063	6373	1507
NIV	1623	345	2108	494
OBJ	68.33	66.66	1351.32	1272.91
*OBJ**	76.51	75.89	1489.17	1419.64
LPs	155	21	384	41
CPU	1967.9	20.0	6973.5	42.5

Symbols used:

NEI: Number of Event Intervals
NC: Number of Constraints
NV: Total Number of Variables
NIV: Number of Integer Variables
OBJ: Optimal Objective Function
*OBJ**: Objective Function for Fully Relaxed LP
LPs: Number of LPs Solved in MILP
CPU: CPU Time (in sec.) for MILP on SunSparc10

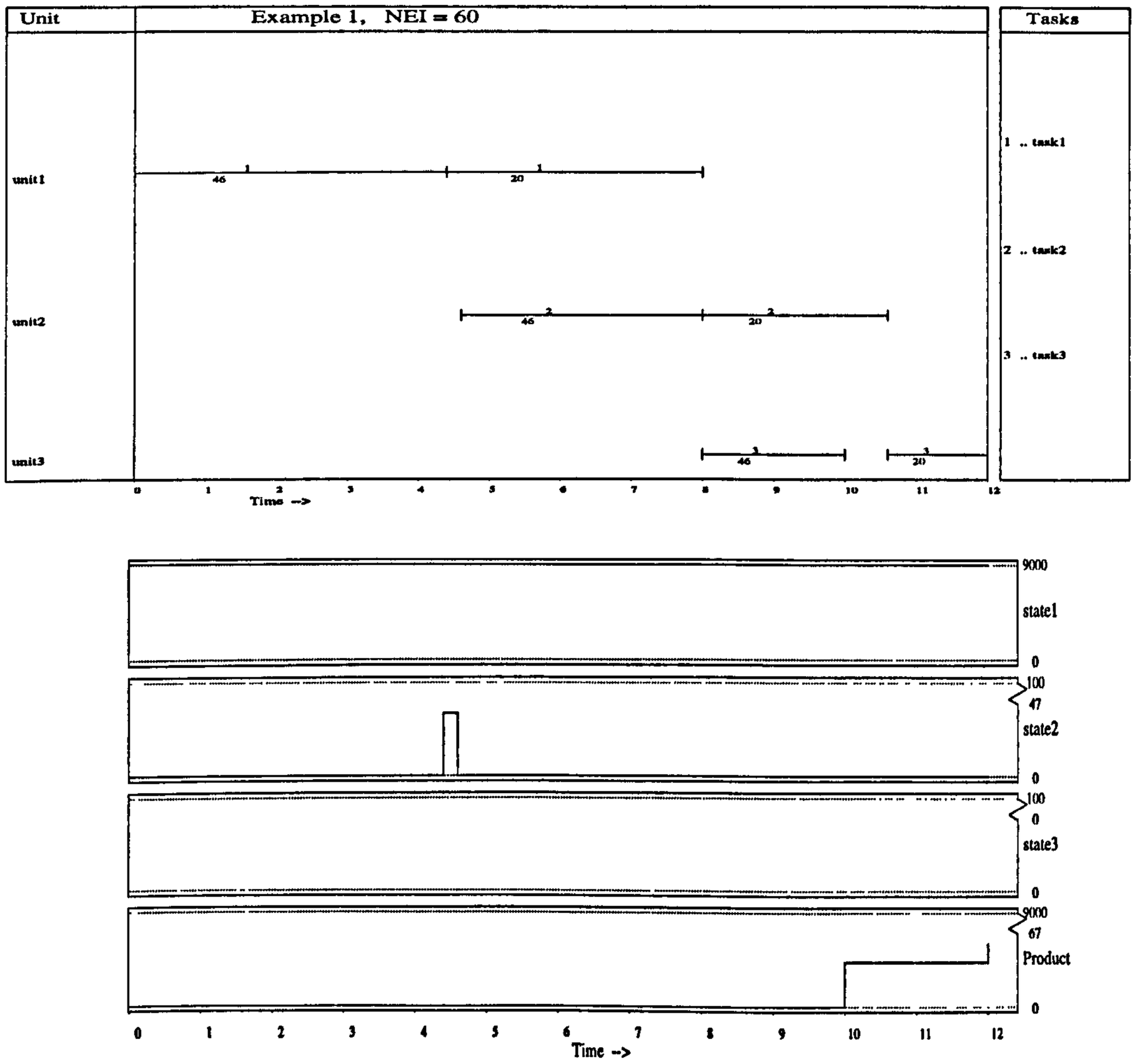


Figure 4.3: Optimal Schedule and Inventory Level for Example 1

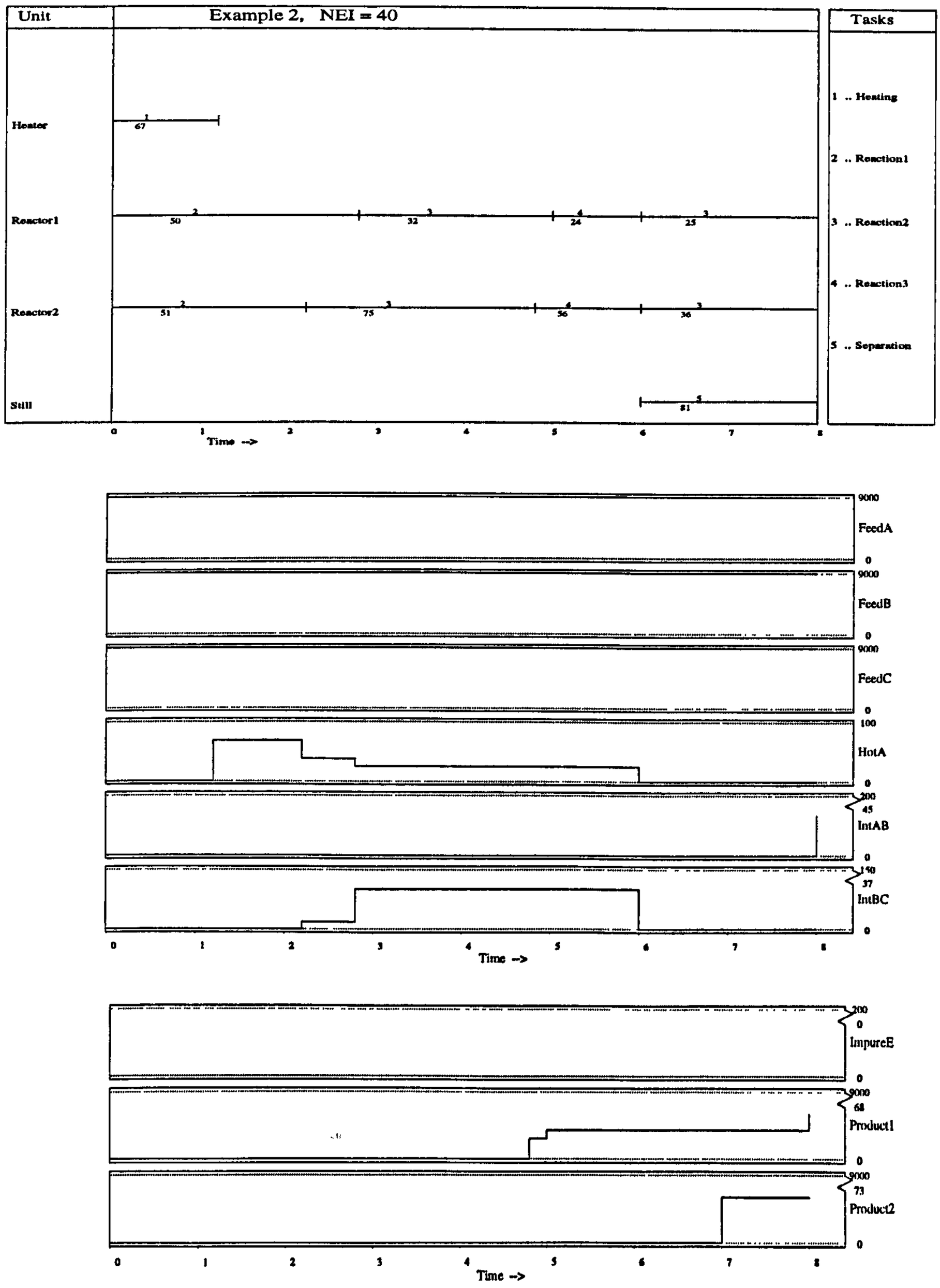


Figure 4.4: Optimal Schedule and Inventory Level for Example 2

4.3 Variable Event Time Sequence (VET)

4.3.1 Time Representation

Although fixing the event-time sequence gives simple linear time constraints for the formulation, the requirement of fine time resolution results in the necessity of using very small time intervals. Therefore, the same problem arises as for the discrete time formulation for fixed recipes – the higher the time accuracy, the larger the problem size. However, by examining the optimal schedule obtained through the *FET* procedure in the last section, we note that the number of real events (7 for *Example 1*) is much smaller than the number of those postulated (61 for *Example 1*). Therefore, most of the event-times are in fact idle with nothing occurring. Nevertheless, these inactive event-times have to be maintained in order to keep the event-time sequence fixed. In other words, as illustrated in Figure 4.5, these events are rather model related than problem related.

Ideally, we need only keep those event-times for which at least one of the system events occurs. This brings us to the idea of a variable event-time sequence. Conceptually, this is to keep the single event-time sequence in *FET* but to make the event-time variable. If we can know approximately the number of real events occurring during the horizon, this time scheme can be a cure for the conflicts between time accuracy and problem size.

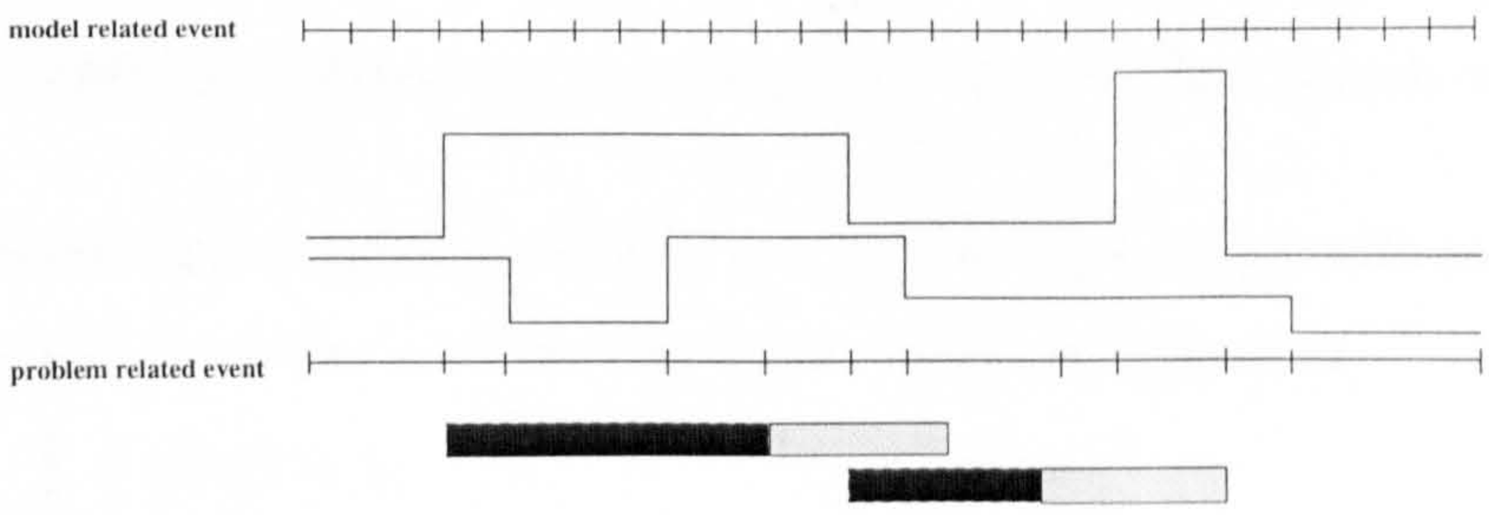


Figure 4.5: Variable Event-time Representation

4.3.2 Variables

The key binary variables, W_{ijl} and $X_{ijkll'}$, introduced in *FET* to decide the activity of tasks in units and the sequence of system events are also necessary for *VET*. Furthermore, it should be noted that, due to the variable event-times, the $X-$ variables still remain necessary even if the processing time is fixed, which is unlike the FET procedure (*see*

Appendix A).

On the other hand, the fact that event times are variable now makes it impossible to pre-identify the specified sequence of times for some system events with the common event-times. Such system events include plant delivery (delivery of material to or from the plant) during the time horizon, the changes in storage capacity and the changes of resource availabilities/price. A number of parameters and sets associated with them are listed below:

- Plant deliveries and receipts are characterised by:
  a sequence of specified times, t^D_{sk}, where $k \in K^D_s$;
  Limits for delivery/receipt, $(\hat{D}^{min}_{sk},\ \hat{D}^{max}_{sk})$ at t^D_{sk}, and $(\tilde{D}^{min}_{sk}, \tilde{D}^{max}_{sk})$ over $[t^D_{sk},\ t^D_{s,k+1})$;
  price for delivery or receipt over interval $[t^D_{sk},\ t^D_{s,k+1})$, c_{sk}.

- Storage capacity changes are characterised by:
  a sequence of specified times, t^C_{sk}, where $k \in K^C_s$;
  maximum available storage capacity over interval $[t^C_{sk},\ t^C_{s,k+1})$, C^{max}_{sk}.

- Resource availability/price changes are characterised by:
  a sequence of specified intervals, $[t_{rk},\ t_{r,k+1})$, where $k \in K_r$;
  Limits for consumption/generation over interval $[t_{rk},\ t_{r,k+1})$, R^{min}_{rk}, R^{max}_{rk};
  price for resource consumption/generation over interval $[t_{rk},\ t_{r,k+1})$, c_{rk}.

In order to identify individual times of sequences, $\{t^D_{sk}\}$, $\{t^C_{sk}\}$, $\{t_{rk}\}$, with common event-times, extra binary variables must be introduced. They are defined as:

$$Y^D_{skl'} = \begin{cases} 1 & \text{if } t^D_{sk} = T_{l'} \\ 0 & \text{otherwise} \end{cases}$$

$$Y^C_{skl'} = \begin{cases} 1 & \text{if } t^C_{sk} = T_{l'} \\ 0 & \text{otherwise} \end{cases}$$

$$Y_{rkl'} = \begin{cases} 1 & \text{if } t_{rk} = T_{l'} \\ 0 & \text{otherwise} \end{cases}$$

To ensure the proper delivery amount within a specified interval, the event-time within it should also be identified. On the other hand, the maximum storage capacity at each event-time, T_l, also becomes unknown. Therefore, another set of binary variables needs

to be introduced to identify the appropriate interval which contains the individual event-time hence giving the proper storage limits. Similarly, this is necessary for determining the limits on resource usage.

These binary variables are defined as:

$$\tilde{Y}^D_{skl'} = \begin{cases} 1 & \text{if } t^D_{sk} \leq T_{l'} < t^D_{s,k+1} \\ 0 & \text{otherwise} \end{cases}$$

$$\tilde{Y}^C_{skl'} = \begin{cases} 1 & \text{if } t^C_{sk} \leq T_{l'} < t^C_{s,k+1} \\ 0 & \text{otherwise} \end{cases}$$

$$\tilde{Y}_{rkl'} = \begin{cases} 1 & \text{if } t_{rk} \leq T_{l'} < t_{r,k+1} \\ 0 & \text{otherwise} \end{cases}$$

All other notation and the continuous variables dealing with the material transfer time during task processing, the amount of material and/or resource processed, delivered or required, are consistent with those described in *FET*.

4.3.3 Constraints

The fundamental constraints will still be those which resolve the conflicts of using resources, materials and processing equipment, and those ensuring the duration of individual processing subsequences under different operating conditions. However, making the event-time T_l variable results in many nonlinearities as will be described.

Time Constraints

First of all, in order to ensure the strictly ascending order of distinct event-times, we impose:

$$T_l - T_{l-1} \geq \delta T \qquad \forall l \qquad (4.34)$$

where δT is a small real number. Its value depends on the required time accuracy and is supposed to be known *a priori*.

Being a variable, the event-time can take any real value, hence there is no need for the round up of times. The time constraints for the transfer of material and/or use of

resources during task processing will be the same as those given in (4.2) and (4.5):

$$t_{ijkl} = T_l + \tau_{ijkl} \qquad \forall i, j \in J_i, k \in K_i, l \tag{4.35}$$

$$\tau_{ijkl} = \sum_{l' \geq l} X_{ijkll'}(T_{l'} - T_l) \qquad \forall i, j \in J_i, k \in K_i, l \tag{4.36}$$

where τ_{ijkl} is defined by (4.1). The special ordered-set constraint (4.4) will also hold.

Parallel to the FET procedure, disaggregation variable $\tau_{ijkll'}$ of (4.29) can also be introduced for VET procedure so that (4.36) can be replaced by:

$$\tau_{ijkll'} = X_{ijkll'}(T_{l'} - T_l) \qquad \forall i, j \in J_i, k \in K_i, l, l' > l \tag{4.37}$$

However, since T_l is a variable now, both constraints (4.36) and (4.37) become nonlinear with bilinear products. Therefore, the linearization technique of Glover (1975) is applied for the linearization.

First, if (4.36) is used for aligning transfer time to common event-time, it can be directly enforced by:

$$U^{min}(1 - X_{ijkll'}) \leq T_{l'} - t_{ijkl} \leq U^{max}(1 - X_{ijkll'}) \quad \forall i, j \in J_i, k \in K_i, l, l' \geq l \tag{4.38}$$

Alternatively, we can choose to use the reformulated time constraints (4.37) and apply the following linearization constraints:

$$\tau_{ijk}^{min} X_{ijkll'} \leq \tau_{ijkll'} \leq \tau_{ijk}^{max} X_{ijkll'} \qquad \forall i, j \in J_i, k \in K_i, l, l' \geq l \tag{4.39}$$

$$\Delta T_{ll'}^{min}(1 - X_{ijkll'}) \leq T_{l'} - T_l - \tau_{ijkll'} \leq \Delta T_{ll'}^{max}(1 - X_{ijkll'})$$

$$\forall i, j \in J_i, k \in K_i, l, l' \geq l \tag{4.40}$$

The $\Delta T_{ll'}^{min}$, $\Delta T_{ll'}^{max}$ and U^{min}, U^{max} are the lower and upper bounds on $T_{l'} - T_l$ and $T_{l'} - t_{ijkl}$, respectively. The tightness of these bounds significantly affect the solution efficiency of the problem. The tighter they are, the smaller the integrality gap between the integer optimal solution and the fully relaxed LP solution. We will leave the detailed investigation for such bounds to *Chapter 6.*

Furthermore, in order to check the material balance at each time when there is a change in the inventory level, we have to deal with the relation of plant delivery time to the event-time, and several integer constraints can be derived for binary variables $Y_{skl'}^{D}$, $\tilde{Y}_{skl'}^{D}$.

- The event-times are postulated in a strictly ascending order. Therefore, each delivery time can be mapped onto one and only one of the event-times;
- On the other hand, the delivery time sequence $\{t^D_{sk}\}$ is only a subset of the common event-time sequence $\{T_{l'}\}$. Therefore, only some of the event-times will be equal to a delivery time.
- It is also true that each event-time $T_{l'}$ must be within one and only one delivery interval but more than one event time can fall in the same interval.

Therefore we have the following integer constraints:

$$\sum_{l'} Y^D_{skl'} = 1 \qquad \forall s, k \in K^D_s \tag{4.41}$$

$$\sum_{k \in K^D_s} Y^D_{skl'} \leq 1 \qquad \forall s, l' \tag{4.42}$$

$$\sum_{k \in K^D_s} \tilde{Y}^D_{skl'} = 1 \qquad \forall s, l' \tag{4.43}$$

To map delivery time onto event-time, the following constraint could be embedded in the formulation:

$$t^D_{sk} = \sum_{l'} Y^D_{skl'} T_{l'} \qquad \forall s, k \in K^D_s \tag{4.44}$$

Note that this is again a nonlinear constraint with bilinear terms. However, knowing that t^D_{sk} is pre-specified, we can produce a set of linear constraints to replace (4.44). Also by using (4.42), tight bounds can be obtained for $T_{l'}$:

$$\sum_{k \in K^D_s} t^D_{sk} Y^D_{skl'} \leq T_{l'} \leq T^{max} - \sum_{k \in K^D_s} (T^{max} - t^D_{sk}) Y^D_{skl'} \qquad \forall s, l' \tag{4.45}$$

A similar procedure can be applied to the time of a storage capacity change and a resource availability/price change, leading to the following integer constraints:

$$\sum_{l'} Y^C_{skl'} = 1 \qquad \forall s, k \in K^C_s; \qquad \sum_{l'} Y_{rkl'} = 1 \qquad \forall r, k \in K_r \tag{4.46}$$

$$\sum_{k \in K^C_s} Y^C_{skl'} \leq 1 \qquad \forall s, l'; \qquad \sum_{k \in K_r} Y_{rkl'} \leq 1 \qquad \forall r, l' \tag{4.47}$$

$$\sum_{k \in K^C_s} \tilde{Y}^C_{skl'} = 1 \qquad \forall s, l'; \qquad \sum_{k \in K_r} \tilde{Y}_{rkl'} = 1 \qquad \forall r, l' \tag{4.48}$$

and the constraints ensuring the relations between t^C_{sk}, t_{rk} and $T_{l'}$:

$$\sum_{k \in K^C_s} t^C_{sk} Y^C_{skl'} \leq T_{l'} \leq T^{max} - \sum_{k \in K^C_s} (T^{max} - t^C_{sk}) Y^C_{skl'} \qquad \forall s, l' \tag{4.49}$$

$$\sum_{k \in K_r} t_{rk} Y_{rkl'} \leq T_{l'} \leq T^{max} - \sum_{k \in K_r} (T^{max} - t_{rk}) Y_{rkl'} \qquad \forall r, l' \tag{4.50}$$

As with the unit occupation variable $\tilde{X}_{ijkll'}$, the binary variables $\tilde{Y}^C_{skl'}$, $\tilde{Y}^D_{skl'}$ and $\tilde{Y}_{rkl'}$, indicating the location of event-time $T_{l'}$ within individual intervals, $[t^C_{sk}, t^C_{s,k+1})$, $[t^D_{sk}, t^D_{s,k+1})$ and $[t_{rk}, t_{r,k+1})$, can also be determined from $Y^C_{skl'}$, $Y^D_{skl'}$ and $Y_{rkl'}$, respectively:

$$\tilde{Y}^C_{skl'} = \sum_{l'' \leq l'} Y^C_{skl''} + \sum_{l'' > l'} Y^C_{s,k+1,l''} - 1 \qquad \forall s, k \in K^C_s, l' \tag{4.51}$$

$$\tilde{Y}^D_{skl'} = \sum_{l'' \leq l'} Y^D_{skl''} + \sum_{l'' > l'} Y^D_{s,k+1,l''} - 1 \qquad \forall s, k \in K^D_s, l' \tag{4.52}$$

$$\tilde{Y}_{rkl'} = \sum_{l'' \leq l'} Y_{rkl''} + \sum_{l'' > l'} Y_{r,k+1,l''} - 1 \qquad \forall r, k \in K_r, l' \tag{4.53}$$

Allocation Constraints

The unit allocation constraints remain unchanged as:

$$\sum_{i \in I_j} \sum_{k < k^{max}_i} \sum_{l \leq l'} \tilde{X}_{ijkll'} \leq 1 \qquad \forall j, l' \tag{4.54}$$

Delivery Constraints

According to the delivery requirements described in *4.2.3* (*see page 41*), two different situations have to be considered. These indicate the following limits:

$$\text{for case (i):} \qquad \hat{D}^{min}_{sk} \leq \hat{D}_{sk} \leq \hat{D}^{max}_{sk} \tag{4.55}$$

$$\text{where:} \qquad \hat{D}_{sk} = \sum_l \hat{D}_{sl} Y^D_{skl} \tag{4.56}$$

$$\text{for case (ii):} \qquad \tilde{D}^{min}_{sk} \leq \tilde{D}_{sk} \leq \tilde{D}^{max}_{sk} \tag{4.57}$$

$$\text{where:} \qquad \tilde{D}_{sk} = \sum_l \tilde{D}_{sl} \tilde{Y}^D_{skl} \tag{4.58}$$

where $\hat{D}_{sl}$ is the amount of delivery satisfying case (i), and $\tilde{D}_{sl}$ is the amount of delivery satisfying case (ii). The total amount of delivery at each event time is then made up of:

$$D_{sl} = \hat{D}_{sl} + \tilde{D}_{sl} \qquad \forall s, l \tag{4.59}$$

$\hat{D}_{sl}$ and D_{sl} may also be subject to specified limits as:

$$\sum_{k \in K_s^D} \hat{D}_{sk}^{min} Y_{skl}^D \leq \hat{D}_{sl} \leq \sum_{k \in K_s^D} \hat{D}_{sk}^{max} Y_{skl}^D \tag{4.60}$$

$$\sum_{k \in K_s^D} D_{sk}^{min} \tilde{Y}_{skl}^D \leq D_{sl} \leq \sum_{k \in K_s^D} D_{sk}^{max} \tilde{Y}_{skl}^D \tag{4.61}$$

The total amount delivered or received over interval $[t_{sk}^D,\ t_{s,k+1}^D)$, taking into account both delivery at specified times t_{sk}^D, and those carried out within this interval will be:

$$D_{sk} = \hat{D}_{sk} + \tilde{D}_{sk} \qquad \forall s, k \in K_s^D \tag{4.62}$$

However, (4.56) and (4.58) become nonlinear. Again, this has to be tackled by defining new continuous variables:

$$\hat{D}_{skl} = \hat{D}_{sl} Y_{skl}^D; \qquad \tilde{D}_{skl} = \tilde{D}_{sl} \tilde{Y}_{skl}^D$$

and imposing the constraints:

$$\hat{D}_{sl} = \sum_{k \in K_s^D} \hat{D}_{skl}; \qquad \hat{D}_{sk}^{min} Y_{skl}^D \leq \hat{D}_{skl} \leq \hat{D}_{sk}^{max} Y_{skl}^D \tag{4.63}$$

$$\tilde{D}_{sl} = \sum_{k} \tilde{D}_{skl}; \qquad D_{sk}^{min} \tilde{Y}_{skl}^D \leq \tilde{D}_{skl} \leq D_{sk}^{max} \tilde{Y}_{skl}^D \tag{4.64}$$

It should be noted that all the limits described above should be carefully chosen according to the net demands for deliveries and receipts, bearing in mind that we assume positive for delivery out of the plant and negative for delivery to the plant. If the interval considered is for delivery, $\tilde{D}_{skl}$ is positive and should be less than the maximum amount specified for this interval; but it can also take the value of zero since it is not necessary to carry out the delivery at each event-time within this interval. On the other hand, if the interval considered is for receipt of material, $\tilde{D}_{skl}$ is negative and its absolute value should be less than the maximum amount specified for this interval (*i.e* D_{sk}^{min} takes the negative value for the maximum limit for receipt). The inequality constraint in (4.64) can then be written in the form:

$$\min\{0,\ D_{sk}^{min} \tilde{Y}_{skl}^D\} \leq \tilde{D}_{skl} \leq \max\{0, D_{sk}^{max} \tilde{Y}_{skl}^D\} \tag{4.65}$$

Material Balance/Storage Capacity Constraints

Taking into account production, storage and deliveries, we get a similar material balance constraint as for *FET*:

$$C_{sl'} = C_{s,l'-1} - D_{sl'} + \sum_{i\in I_s}\sum_{j\in J_i}\sum_{k\in K_i}\rho_{sijk}\sum_{l\le l'}X_{ijkll'}B_{ijl} \quad \forall s, l' \tag{4.66}$$

Note that the bilinear terms can be tackled in exactly the same way as for the *FET* procedure.

At each event-time, the amount of material in storage must not exceed the maximum capacity available for storage at that time. Since the storage capacity may vary along the horizon, this may affect the inventory level of materials. Therefore, inventory has also to be checked at t^C_{sk} and this explains the necessity of introducing $Y^C_{skl'}$ and related constraints (4.46) and (4.49).

Using the binary variables $\tilde{Y}^C_{skl'}$, the storage capacity constraints can be written:

$$0 \le C_{sl'} \le \sum_{k\in K^C_s} C^{max}_{sk}\tilde{Y}^C_{skl'} \qquad \forall s, l' \tag{4.67}$$

Resource Usage Constraints

Imposing constraints (4.46) and (4.50) ensures that, if the resource generation or consumption is within the appropriate limits in each event-time interval, it will not then exceed the limits at the time when such availability changes.

$$\sum_{k\in K_r} R^{min}_{rk}\tilde{Y}_{rkl'} \le R_{rl'} \le \sum_{k\in K_r} R^{max}_{rk}\tilde{Y}_{rkl'} \qquad \forall r, l' \tag{4.68}$$

Following the same argument for *FET* formulation, the lower limit R^{min}_{rk} is again the negative value of the maximum availability of resource r during interval $[t_{rk},\ t_{r,k+1})$. This is also true for the later linearization in the objective function in (4.73).

The resource usage constraint remains unchanged in the form of (4.17):

$$R_{rl'} = \sum_{i\in I_r}\sum_{j\in J_i}\sum_{k<k^{max}_i}\sum_{l\le l'}\tilde{X}_{ijkll'}(\alpha_{rijk} + \beta_{rijk}B_{ijl}) \qquad \forall r, l' \tag{4.69}$$

4.3.4 Objective Function

Again, the maximization of profit is used as the criterion.

$$\texttt{Cost/Value of raw material/products} = \sum_{s} \sum_{k \in K_s^D} c_{sk} D_{sk} \tag{4.70}$$

Note that the cost of the initial amount of raw material and that of the delivery of products to customers at the end of the time horizon can all be included in the above function with the inclusion of $t_{s0}^D = 0$ and $t_{skmax}^D = T^{max}$, respectively.

Since the unit price of resource usage may vary for different time intervals, we can select the appropriate price by using the binary variable of $\tilde{Y}_{rkl'}$:

$$c_{rl'} = \sum_{k \in K_r} c_{rk} \tilde{Y}_{rkl'} \tag{4.71}$$

Then, the overall cost of all resources used is given by:

$$\begin{aligned} \texttt{Cost of resources} &= \sum_{r} \sum_{l' < l^{max}} (-R_{rl'})(T_{l'+1} - T_{l'}) c_{rl'} \\ &= \sum_{r} \sum_{l' < l^{max}} (-R_{rl'})(T_{l'+1} - T_{l'}) \sum_{k \in K_r} c_{rk} \tilde{Y}_{rkl'} \end{aligned} \tag{4.72}$$

Rearranging the summation, we get:

$$\texttt{Cost of resources} = \sum_{r} \sum_{k \in K_r} \sum_{l' < l^{max}} c_{rk} \tilde{Y}_{rkl'} (-R_{rl'})(T_{l'+1} - T_{l'}) \tag{4.73}$$

This gives a nonlinear function of products between binary variable and two continuous variables. To avoid the bilinear term, writing:

$$R_{rkl'} = R_{rl'} \tilde{Y}_{rkl'}$$

The above definition is then effected through the following constraints:

$$R_{rl'} = \sum_{k \in K_r} R_{rkl'} \tag{4.74}$$

$$R_{rk}^{min} \tilde{Y}_{rkl'} \leq R_{rkl'} \leq R_{rk}^{max} \tilde{Y}_{rkl'} \tag{4.75}$$

Substitute the new defined continuous variable into the cost function of resource usage:

$$\texttt{Cost of resources} = -\sum_{r} \sum_{k \in K_r} \sum_{l' < l^{max}} c_{rk} R_{rkl'} (T_{l'+1} - T_{l'}) \tag{4.76}$$

The nonlinearity caused by the cross products of two continuous variables still remains in this function. Unfortunately, nothing can be done with the exact linearization technique used for products between binary and continuous variables. This has to be solved as a MINLP problem with local linearization of (4.76) at each iteration.

4.3.5 An Illustrative Example

In order to illustrate the formulation presented in this section, we revisit the simple *Example 1.* of *4.2.7.* Again, we assume the linear dependence of processing time on batch size.

To solve the scheduling problem, two issues have to be dealt with. First is how many real events (l^{max}) there would be within the time horizon considered. However, this is difficult to decide and may be over-estimated. A simple way (worst case) is to assume that each unit repeatedly processes its suitable task with minimum processing time (with no idle time) and only a single transfer or delivery occurs at each event-time. For *Example 1* in *4.2.7*, this will give a l^{max} of 44. However, this would highly over-estimate the real number because most of the resource sharing effects could not be taken into account. Examining the results from the FET procedure, it can be seen that only seven event-times are active.

The second issue to be considered is how many other events may take place during the performing of one task in a unit, *i.e.* how to determine $\underline{l}_{ijk}$ and $\bar{l}_{ijk}$. Again, this is difficult to guess due to the fact that both the event-times and the processing times are variable. Simply, as a "worst case", we suppose that each subsequence can extend as long as the end of the time horizon from its starting time. Obviously, this could not be true for most of the scheduling problem. This is another factor affecting the efficiency of the solution.

Consider the maximum number of event-intervals as 7 from the examination of the *FET* results. Comparing the problem size of *NEI*= 7 in Table 4.5 with the problem size of using *NEI*= 60 in Table 4.3 and 4.4, it shows that using *VET* with the real number of event-times will greatly reduce the number of variables and constraints even by covering the "worst case" of choosing $\underline{l}_{ijk}$, $\bar{l}_{ijk}$.

The problem is solved using the time accuracy of $\delta T = 10^{-3}$. The possibility of satisfying fine time resolution by *VET* eliminates the sacrificing of optimality by round-

up, thus obtaining a better objective function than using *FET* with $\Delta T = 0.2$. Also, the more compact problem size due to considering only the real number of events reduces the effort of solving each LP during branch-and-bound. However, because of the linearization of timing constraints, the integrality gap between the optimal integer solution and the fully relaxed LP solution is quite big now. This makes the MILP more difficult to solve although the number of 0-1 variables is reduced. It is also shown in Table 4.5 that the reformulation of the time constraints does not prove to be as efficient as in FET for this problem.

The second row of each formulation in Table 4.5 gives the information about using the "worst case" estimation of the number of event-time intervals, (*i.e.* *NEI*=43). Again, the higher time accuracy of $\delta T = 10^{-3}$ is used. It can be seen that the "worst case" results in a larger problem. But if the same time accuracy is applied to the *FET* procedure in section *4.2*, the problem size will be explosively increased with *NEI*= 12,000.

Therefore in conclusion, the *VET* procedure will generally lead to a more compact problem compared with *FET*. The higher the time accuracy, the larger the difference. It does seem promising if more accurate estimation of l^{max} and $\underline{l}_{ijk}$, $\bar{l}_{ijk}$ can be made.

Table 4.5: Results of the *VET* formulation

	NEI	NC	NV	NIV	OBJ	OBJ^*	LPs	CPU
Original Time Constraint	7	428	271	105	71.46	199.99	14481	1245.8
	43	9519	6415	2967	–	199.99	–	917.93†
Reformulated Time Constraint	7	491	334	105	71.45	199.99	16599	1644.7
	43	12228	8992	2967	–	199.99	–	–

† this is the time for solving the fully relaxed LP only

4.4 Operation Sequence

In the last two sections, we presented two types of time representation schemes, and based on these formulations for scheduling under varying operating conditions were developed. The common feature of the two time representations is that a single event-time sequence is used as the reference for ordering of system events. Fixing the event-time intervals or making them variable, leads to the *FET* and *VET* procedures respectively. From the given illustrative examples, it demonstrates that the *VET* procedure gives a more compact problem in the sense that it involves fewer variables and constraints in the formulation. However, before solving the scheduling problem, the number of events usually has to be over-estimated in order to cover the "worst case".

However, by examining the system events, we note that all events are related to processing units and storage equipments. For example, the transfer of material will involve either two units (processing units or dedicated storage units) or a plant delivery and one unit, resources are also consumed/generated by units when they process different tasks, *etc.* Therefore, units can be identified as a kind of "critical" resource in the plant. Each unit (j) is used for a sequence of operations, $l \in L_j$, involving possibly different tasks, and we can thus use a sequence of event-times for each unit rather than the common event-time sequence described previously .

Subject to the suitability for different tasks and the availability of appropriate resources to perform such tasks, the maximum number of successive non-overlap operations, l_j^{max}, for each unit j can be decided *a priori*. Simply, this can be decided by the minimum processing time of all the tasks for which it is suitable:

$$l_j^{max} = \left\lfloor \frac{T^{max}}{\min\limits_{i \in I_j} \tau_{ijk^{max}}} \right\rfloor \qquad \forall j \tag{4.77}$$

where $\lfloor X \rfloor$ = the largest integer number $\leq X$.

4.4.1 Units *vs* Tasks

Tasks: Suppose materials $s \in S_i$ and resources $r \in R_i$ are transferred during the processing of task i. Instead of having a single sequence which synchronises all of these transfers, we define a separate sequence for each material and resource. Therefore, task i will be

characterized by:

K_{is} : a sequence of transfers of state s in task i;

K_{ir} : a sequence of intervals for resource r in task i;

τ_{sijk} : relative time of transfer $k \in K_{is}$ of state s for task i in unit j;

τ_{rijk} : relative start time of interval $k \in K_{ir}$ of resource r for task i in unit j;

ρ_{sijk} : proportion of batch in state s transferred at τ_{sijk} for task i in unit j (negative for feeds, positive for products);

α_{rijk} : fixed factor for the consumption/generation of resource r over interval $k \in K_{ir}$ for task i in unit j (positive for generation, negative for consumption)

β_{rijk} : variable factor for the consumption/generation of resource r over interval $k \in K_{ir}$ for task i in unit j (positive for generation, negative for consumption)

Units: Due to the flexibility of batch plant, usually a small number of units should be capable of performing a large number of different tasks. Therefore, each unit j will carry out a set of tasks I_j, with each of them having different sequences, K_{is} and K_{ir} for the related material and resource specified by the task recipe. In order to allow all of the possible transfers within the operation of a unit, transfer sequences are also defined for each material $s \in S_j$ and each resource $r \in R_j$:

$$k_{sj}^{max} = \max_{i \in I_j}(k_{is}^{max}); \qquad k_{rj}^{max} = \max_{i \in I_j}(k_{ir}^{max}) \qquad \forall j, s \in S_j, r \in R_j$$

The corresponding times for material transfer and resource consumption/generation, as well as the amounts of material transferred and resources consumed/generated, will then depend on the task being performed.

The sequence of possible transfers for each operation in a unit must be the union of transfers for each possible task, so for a particular task some transfers may not be required. We therefore assume that each task $i \in I_j$ has a number of null transfers, $k = k_{is}^{max} + 1, \ldots, k_{sj}^{max}$, with the corresponding transfer time and parameters as:

$$\tau_{sijk} = \tau_{sijk_{is}^{max}}; \qquad \rho_{sijk} = 0 \qquad \forall s, i \in I_s, j \in J_i, k_{is}^{max} < k \leq k_{sj}^{max} \tag{4.78}$$

$$\tau_{rijk} = \tau_{rijk_{ir}^{max}}; \qquad \forall r, i \in I_r, j \in J_i, k_{ir}^{max} < k \leq k_{rj}^{max} \tag{4.79}$$

$$\alpha_{rijk} = 0; \qquad \beta_{rijk} = 0 \qquad \forall r, i \in I_r, j \in J_i, k_{ir}^{max} < k \leq k_{rj}^{max} \tag{4.80}$$

Example: Consider a unit j suitable for Task1 and Task2. Task1 needs the transfer of each of the material A, B, C, D, E once during its processing ($k_{1s}^{max} = 1, s \in \{A, B, C, D, E\}$).

Task2 needs the transfer of each of the materials D, C, F once during its processing ($k_{2s}^{max} = 1, s \in \{D, C, F\}$). Therefore, in operation of unit j, the maximum number of transfer times for each material A, B, C, D, E, F will be $k_{sj}^{max} = 1$, $s \in \{A, B, C, D, E, F\}$.

The advantage of defining a separate transfer sequence for each material and each resource is that when deciding the relative order of interacting events, only transfers for the same state of material or resource need to be compared.

4.4.2 Variables

Based on the unit *Operation Sequence* representation and the transfer sequences for material/resources during task processing in units, a number of continuous and binary variables are defined to formulate the scheduling problem using this new time scheme. These are listed below as:

- Continuous variables describing unit operation sequence l:

  B_{ijl} : batch size for task i in operation l of unit j;

  t_{jl}^{0} : start time of operation l in unit j

  t_{jl} : start time of processing of operation l in unit j

  t_{jl}^{max} : end time of operation l in unit j

  τ_{sijk} : relative time of transfer $k \in K_{sj}$ of state $s \in S_j$ for task i in unit j

  τ_{rijk} : relative start time of interval $k \in K_{rj}$ of resource r for task i in unit j

  t_{sjkl} : actual time of transfer $k \in K_{sj}$ of state $s \in S_j$ in operation l of unit j

  t_{rjkl} : actual start time of interval $k \in K_{rj}$ of resource r in operation l of unit j

  R_{rijkl} : amount of resource consumed/generated over interval $[\tau_{rijk}, \tau_{rij,k+1})$ when unit j performing task i in operation l (positive for generation, negative for consumption)

- Binary variables describing the activity of tasks in operation of units:

$$W_{ijl} = \begin{cases} 1 & \text{if task } i \text{ is operation } l \text{ in unit } j \\ 0 & \text{otherwise} \end{cases}$$

We still assume that the sequences $\{t_{sk}^{D}\}$, $\{t_{sk}^{C}\}$, $\{t_{rk}\}$ are given as described in section *4.3.2.* Parameters and continuous variables describing the sequence of plant deliveries and changes of storage capacity and/or changes of resource availability/prices are also consistent with those described in the *VET* formulation.

With a continuous time representation, we still need binary variables to indicate the relative order of system events. However, abandoning the single time reference common to all system events, we now have to decide the order of each pair of events which share the same material and/or resource, including the comparison of the pre-known time sequence t^D_{sk}, t^C_{sk} and t_{rk} with transfer times t_{sjkl} and t_{rjkl}.

Here we propose two types of mechanism for the ordering of interacting events, the pair-wise ordering of each pair of events (direct link to times), and the locating of each event within the interval of another two (the indirect link to times). Binary variables are then introduced for each of the mechanisms with one set of variables denoted by X-, Y-, and the other as $\tilde{X}$-, $\tilde{Y}$-. Note that we need only consider the relative order of transfers separately for each material or resource, since balances affect only one material or resource.

- Binary variables indicating the order of material transfers during unit operations:

$$X_{sjkl,j'k'l'} = 1 \text{ iff } t_{sjkl} \le t_{sj'k'l'}; \qquad \tilde{X}_{sjkl,j'k'l'} = 1 \text{ iff } t_{sjkl} < t_{sj'k'l'} \le t_{sj,k+1,l}$$

- Binary variables indicating the order of plant delivery *vs* material transfers during unit operations:

$$Y^D_{sk,j'k'l'} = 1 \text{ iff } t^D_{sk} \le t_{sj'k'l'}; \qquad \tilde{Y}^D_{sk,j'k'l'} = 1 \text{ iff } t^D_{sk} \le t_{sj'k'l'} < t^D_{s,k+1}$$

$$Y^D_{sjkl,k'} = 1 \text{ iff } t_{sjkl} \le t^D_{sk'}; \qquad \tilde{Y}^D_{sjkl,k'} = 1 \text{ iff } t^D_{sk'} < t_{sjkl} \le t^D_{s,k'+1}$$

- Binary variables indicating the order of storage capacity changes *vs* material transfers during unit operations:

$$Y^C_{sk,j'k'l'} = 1 \text{ iff } t^C_{sk} \le t_{sj'k'l'}; \qquad \tilde{Y}^C_{sk,j'k'l'} = 1 \text{ iff } t^C_{sk} \le t_{sj'k'l'} < t^C_{s,k+1}$$

$$Y^C_{sjkl,k'} = 1 \text{ iff } t_{sjkl} \le t^C_{sk'}; \qquad \tilde{Y}^C_{sjkl,k'} = 1 \text{ iff } t^C_{sk'} < t_{sjkl} \le t^C_{s,k'+1}$$

- Binary variables indicating the order of resource interval start times in unit operations:

$$X_{rjkl,j'k'l'} = 1 \text{ iff } t_{rjkl} \le t_{rj'k'l'}; \qquad \tilde{X}_{rjkl,j'k'l'} = 1 \text{ iff } t_{rjkl} \le t_{rj'k'l'} < t_{rj,k+1,l}$$

- Binary variables indicating the order of resource availability/price changes *vs* the resource interval start times in unit operations

$$Y_{rk,j'k'l'} = 1 \text{ iff } t_{rk} \le t_{rj'k'l'}; \qquad \tilde{Y}_{rk,j'k'l'} = 1 \text{ iff } t_{rk} \le t_{rj'k'l'} < t_{r,k+1}$$

$$Y_{rjkl,k'} = 1 \text{ iff } t_{rjkl} \le t_{rk'}; \qquad \hat{Y}_{rjkl,k'} = 1 \text{ iff } t_{rjkl} \le t_{rk'} < t_{rj,k+1,l}$$

The usage and the relation between the two mechanisms and the two sets of variables will be described and become clear later in the formulation of constraints.

4.4.3 Constraints

The constraints necessary for the optimal scheduling of plants under variable operating conditions are described in detail below:

Time Assignment for Unit Operations

Two aspects should be considered for the timing during unit operations. First is how to avoid the overlap of any operations within the same unit. This can be ensured by imposing that no operation can start in a unit before the previous operation is finished in the same unit:

$$0 \leq t^{max}_{j,l-1} \leq t^0_{jl} \qquad \forall j, l \in L_j \tag{4.81}$$

The second aspect concerns the guarantee of individual material transfers and resource generations/consumptions taking place at the appropriate time depending on operating conditions. This is dealt with by imposing a set of constraints considering:

- Transfer of feed material can only start when the operation starts. Processing does not start before all feed materials are transferred into the unit. Once a unit operation starts, it can immediately consume/generate resources:

$$t^0_{jl} \leq t_{sj0l} \leq t_{jl}; \qquad t_{rj0l} = t^0_{jl} \qquad \forall j, s \in S_j, r \in R_j, l \in L_j \tag{4.82}$$

  Here we suppose that material initially available in the plant (at $t = 0$) is regarded as delivered to the plant at $t = 0$. If a specific feed should not be added until processing starts, then the right side inequality in (4.82) for this feed should be replaced by:

$$t_{sj0l} = t_{jl}$$

- The actual transfer time for feeds and products during the unit operation is decided by the task being processed and its operating conditions. The same applies to the

times defining resource consumption or generation:

$$t_{sjkl} = t_{jl} + \sum_{i \in I_j \cap I_s} \tau_{sijk} W_{ijl} \qquad \forall j, s \in S_j, k \in K_{sj}, l \in L_j \tag{4.83}$$

$$t_{rjkl} = t_{jl} + \sum_{i \in I_j \cap I_r} \tau_{rijk} W_{ijl} \qquad \forall j, r \in R_j, k \in K_{rj}, l \in L_j \tag{4.84}$$

- An operation can not be considered as finished unless all material/resource transfers have been completed. Furthermore, all operations should occur within the time horizon:

$$t_{sjkl} \leq t_{jl}^{max} \leq T^{max} \qquad \forall j, s \in S_j, k \in K_{sj}, l \in L_j \tag{4.85}$$

$$t_{rjkl} \leq t_{jl}^{max} \leq T^{max} \qquad \forall j, r \in R_j, k \in K_{rj}, l \in L_j \tag{4.86}$$

Allocation Constraints

Each unit carries out a sequence of operations, l ($l = 1, 2, \ldots, l_j^{max}$), each of which must be a specific task i. Therefore, we have the allocation constraint as:

$$\sum_{i \in I_j} W_{ijl} = 1 \qquad \forall j, l \in L_j \tag{4.87}$$

It must be noticed however that the upper limit l_j^{max} for the number of operations may not be attainable within the time horizon. Therefore it is assumed that each unit is also suitable for a null task $i = 0$ (contained in the set I_j) which takes no time and consumes no resources.

Clearly, the null tasks will not affect the objective function. To avoid degeneracy, we impose the following constraints to ensure that all null tasks occur at the end of the sequence:

$$W_{0jl} \geq W_{0j,l-1} \qquad \forall j, l \in L_j \tag{4.88}$$

Unit Capacity Constraints

The limits on the amount of materials processed in units can be kept by simply imposing the lower and upper bounds on individual operations:

$$V_{ij}^{min} W_{ijl} \leq B_{ijl} \leq V_{ij}^{max} W_{ijl} \qquad \forall j, i \in I_j, l \in L_j \tag{4.89}$$

Delivery Constraints

To be consistent with *FET* and *VET* formulation, again two kinds of deliveries are considered (*see page 41*): the amount delivered at t^D_{sk} to satisfy case (i) is denoted by $\hat{D}_{sk}$, and the amount delivered within interval $[t^D_{sk},\ t^D_{s,k+1})$ to satisfy case (ii) is denoted by $\tilde{D}_{sk}$. They should be kept within the specified limits:

$$\hat{D}^{min}_{sk} \leq \hat{D}_{sk} \leq \hat{D}^{max}_{sk} \tag{4.90}$$

$$\tilde{D}^{min}_{sk} \leq \tilde{D}_{sk} \leq \tilde{D}^{max}_{sk} \tag{4.91}$$

In general, all deliveries are carried out by the material transfer through processing units and storage units. To describe deliveries satisfying different requirements, we define:

- $\hat{D}_{sjkl}$: the amount delivered at time t_{sjkl} to satisfy case (i);
- $\tilde{D}_{sjkl}$: the amount delivered at time t_{sjkl} to satisfy case (ii);
- D_{sjkl} : the total delivery at any transfer time t_{sjkl}.

In order to ensure the delivery of required amount at a given time t^D_{sk}, it is necessary to detect corresponding material transfers occurring at this time. Binary variables $Y'^D_{sk,j'k'l'}$ are introduced for this purpose, where $Y'^D_{sk,j'k'l'} = 1$ indicates $t^D_{sk} = t_{sj'k'l'}$. However, $Y'^D_{sk,j'k'l'}$ does not need to be treated as an independent variable since it can be determined by other binary variables ordering plant delivery *vs* material transfers. This is explained in (4.156). We have:

$$\tilde{D}_{sk} = \sum_{j' \in J_s} \sum_{k' \in K_{sj'}} \sum_{l' \in L_{j'}} \tilde{D}_{sj'k'l'} \tilde{Y}^D_{sk,j'k'l'} \tag{4.92}$$

$$\hat{D}_{sk} = \sum_{j' \in J_s} \sum_{k' \in K_{sj'}} \sum_{l' \in L_{j'}} \hat{D}_{sj'k'l'} Y'^D_{sk,j'k'l'} \tag{4.93}$$

The total delivery D_{sjkl} at $t_{sj'k'l'}$ is:

$$D_{sj'k'l'} = \hat{D}_{sj'k'l'} + \tilde{D}_{sj'k'l'} \tag{4.94}$$

$$\sum_{k \in K^D_s} D^{min}_{sk} \tilde{Y}^D_{sk,j'k'l'} \leq D_{sj'k'l'} \leq \sum_{k \in K^D_s} D^{max}_{sk} \tilde{Y}^D_{sk,j'k'l'} \tag{4.95}$$

where D^{min}_{sk} and D^{max}_{sk} are again the additional limits on the amount delivered at any time within a specified interval.

The total delivery, D_{sk}, over each interval to satisfy both case (i) and (ii) is:

$$D_{sk} = \hat{D}_{sk} + \tilde{D}_{sk} \tag{4.96}$$

Material Balance/Storage Capacity Constraints

Without the single time reference to position all operations competing for shared materials, we can now obtain the storage requirement by using a cumulative material balance from initial time up to each transfer time or capacity changing time. The amount of material in state s stored at any time t is then generally expressed as:

$$C_s(t) = C_s(0) - \sum_{j\in J_s}\sum_{k\in K_{sj}}\sum_{l|t_{sjkl}\le t} D_{sjkl} + \sum_{j\in J_s}\sum_{k\in K_{sj}}\sum_{l|t_{sjkl}\le t}\sum_{i\in I_j} \rho_{sijk}B_{ijl} \tag{4.97}$$

The amount of material in storage over the time horizon should always be kept within the available storage capacity limits:

$$0 \le C_s(t) \le C_{sk}^{max} \qquad \text{where} \qquad t_{sk}^C \le t < t_{s,k+1}^C \tag{4.98}$$

In fact, the amount of material stored will only change when material is transferred. Therefore, the material balance and capacity constraints need only be verified at a finite set of times: t_{sjkl}, t_{sk}^D, t_{sk}^C. Since the times for plant delivery, t_{sk}^D, and capacity changes, t_{sk}^C, are known *a priori*, the appropriate capacity at these times is also known. Therefore, having defined the binary variables (X-,Y-) for the discontinuities in the above material balance constraints, (4.97)and (4.98) are reduced to:

$$C_s(t_{sj'k'l'}) = C_s(0) + \sum_{j\in J_s}\sum_{k\in K_{sj}}\sum_{l\in L_j}\left\{\sum_{i\in I_j}\rho_{sijk}B_{ijl} - D_{sjkl}\right\} X_{sjkl,j'k'l'} \tag{4.99}$$

$$0 \le C_s(t_{sj'k'l'}) \le \sum_{k\in K_s^C} C_{sk}^{max}\tilde{Y}_{sk,j'k'l'}^C \quad \forall s, j' \in J_s, k' \in K_{sj'}, l' \in L_{j'} \tag{4.100}$$

$$C_s(t_{sk'}^D) = C_s(0) + \sum_{j\in J_s}\sum_{k\in K_{sj}}\sum_{l\in L_j}\left\{\sum_{i\in I_j}\rho_{sijk}B_{ijl} - D_{sjkl}\right\} Y_{sjkl,k'}^D \tag{4.101}$$

$$0 \le C_s(t_{sk'}^D) \le C_{sk}^{max}, \qquad \text{where} \quad t_{sk}^C \le t_{sk'}^D < t_{s,k+1}^C \quad \forall s, k \in K_s^C, k' \in K_s^D \tag{4.102}$$

$$C_s(t_{sk'}^C) = C_s(0) + \sum_{j\in J_s}\sum_{k\in K_{sj}}\sum_{l\in L_j}\left\{\sum_{i\in I_j}\rho_{sijk}B_{ijl} - D_{sjkl}\right\} Y_{sjkl,k'}^C \tag{4.103}$$

$$0 \le C_s(t_{sk'}^C) \le C_{sk'}^{max} \qquad \forall s, k' \in K_s^C \tag{4.104}$$

Alternatively, the set of binary variables $\tilde{X}$-, $\tilde{Y}$- indicating the location of one transfer time within the interval of another two can be applied to obtain the incremental material balance over interval $[t_{sjkl},\ t_{sj,k+1,l})$, $[t^D_{sk},\ t^D_{s,k+1})$ and $[t^C_{sk},\ t^C_{s,k+1})$, respectively as:

$$C_s(t_{sj'k'l'}) = C_s(t_{sj',k'-1,l'}) + \sum_{j\in J_s}\sum_{k\in K_{sj}}\sum_{l\in L_j}\left\{\sum_{i\in I_j}\rho_{sijk}B_{ijl} - D_{sjkl}\right\}\tilde{X}_{sj',k'-1,l',jkl} \quad (4.105)$$

$$C_s(t^D_{sk'}) = C_s(t^D_{s,k'-1}) + \sum_{j\in J_s}\sum_{k\in K_{sj}}\sum_{l\in L_j}\left\{\sum_{i\in I_j}\rho_{sijk}B_{ijl} - D_{sjkl}\right\}\tilde{Y}^D_{sjkl,k'-1} \quad (4.106)$$

$$C_s(t^C_{sk'}) = C_s(t^C_{s,k'-1}) + \sum_{j\in J_s}\sum_{k\in K_{sj}}\sum_{l\in L_j}\left\{\sum_{i\in I_j}\rho_{sijk}B_{ijl} - D_{sjkl}\right\}\tilde{Y}^C_{sjkl,k'-1} \quad (4.107)$$

Resource Usage Constraints

The amount of resource used by unit j performing task i in operation l over the interval $[\tau_{rijk}, \tau_{rij,k+1})$ is determined by a variable and a fixed demand factor:

$$R_{rijkl} = \alpha_{rijk}W_{ijl} + \beta_{rijk}B_{ijl} \qquad \forall r, i\in I_r, j\in J_i, k\in K_{sj}, l\in L_j \quad (4.108)$$

Using the binary feature of $\tilde{X}_{rjkl,j'k'l'}$ and $\hat{Y}_{rjkl,k'}$, the total amounts of resource consumed/generated over the intervals starting at $t_{rj'k'l'}$ and $t_{rk'}$ are then calculated as:

$$R_r(t_{rj'k'l'}) = \sum_{i\in I_r}\sum_{j\in J_i}\sum_{k\in K_{rj}}\sum_{l\in L_j}\tilde{X}_{rjkl,j'k'l'}R_{rijkl} \qquad \forall r, j'\in J_r, k'\in K_{sj'}, l'\in L_j \quad (4.109)$$

$$R_r(t_{rk'}) = \sum_{i\in I_r}\sum_{j\in J_i}\sum_{k\in K_{rj}}\sum_{l\in L_j}\hat{Y}_{rjkl,k'}R_{rijkl} \qquad \forall r, k'\in K^D_r \quad (4.110)$$

The consuming/generating of resources should also be kept under the given limits at any time:

$$\sum_k R^{min}_{rk}\tilde{Y}_{rk,j'k'l'} \le R_r(t_{rj'k'l'}) \le \sum_k R^{max}_{rk}\tilde{Y}_{rk,j'k'l'} \qquad \forall r, j'\in J_r, k'\in K_{sj'}, l'\in L_{j'} \quad (4.111)$$

$$R^{min}_{rk'} \le R_r(t_{rk'}) \le R^{max}_{rk'} \qquad \forall r, k'\in K^D_r \quad (4.112)$$

4.4.4 Ordering of System Events

To detect the relative order of interacting events, binary variables have been defined in section *4.4.2*. However, enough constraints have to be introduced to link these variables

to the times. The link can be made either directly or indirectly which employs different sets of variables.

4.4.4.1 Special Ordered-Set Constraints

From the definition of the binary variables $\tilde{X}$-, $\tilde{Y}$-, it is obvious that each material transfer time or the start time of each interval, can always be located in one and only one interval. Therefore, we have the following special ordered-set constraints for all such variables as:

$$\sum_{k\in K_{sj}}\sum_{l\in L_j}\tilde{X}_{sjkl,j'k'l'}=1 \qquad \forall s,(j,j')\in J_s,k'\in K_{sj'},l'\in L_{j'} \tag{4.113}$$

$$\sum_{k\in K_s^D}\tilde{Y}^D_{sk,j'k'l'}=1 \qquad \forall s,j'\in J_s,k'\in K_{sj'},l'\in L_{j'} \tag{4.114}$$

$$\sum_{k'\in K_s^D}\tilde{Y}^D_{sjkl,k'}=1 \qquad \forall s,j\in J_s,k\in K_{sj},l\in L_j \tag{4.115}$$

$$\sum_{k\in K_s^C}\tilde{Y}^C_{sk,j'k'l'}=1 \qquad \forall s,j'\in J_s,k'\in K_{sj'},l'\in L_{j'} \tag{4.116}$$

$$\sum_{k'\in K_s^C}\tilde{Y}^C_{sjkl,k'}=1 \qquad \forall s,j\in J_s,k\in K_{sj},l\in L_j \tag{4.117}$$

$$\sum_{k\in K_{rj}}\sum_{l\in L_j}\tilde{X}_{rjkl,j'k'l'}=1 \qquad \forall r,(j,j')\in J_r,k'\in K_{rj'},l'\in L_{j'} \tag{4.118}$$

$$\sum_{k\in K_r}\tilde{Y}_{rk,j'k'l'}=1 \qquad \forall r,j'\in J_r,k'\in K_{rj'},l'\in L_{j'} \tag{4.119}$$

$$\sum_{k\in K_{rj}}\sum_{l\in L_j}\hat{Y}_{rjkl,k'}=1 \qquad \forall r,j\in J_r,k'\in K_r \tag{4.120}$$

In fact, (4.116) and (4.119) were already applied to the selection of storage capacity limits in (4.100) and resource limits in (4.111).

4.4.4.2 Direct and Indirect Link to Times

1. *Ordering of each pair of material transfers.* To ensure this we can either relate $X_{sjkl,j'k'l'}$ to transfer times t_{sjkl}, $t_{sj'k'l'}$ directly, or indirectly using the definition of $\tilde{X}_{sjkl,j'k'l'}$ as:

Direct:
$$\frac{t_{sj'k'l'}-t_{sjkl}}{T^{max}}<X_{sjkl,j'k'l'}\leq 1+\frac{t_{sj'k'l'}-t_{sjkl}}{T^{max}}$$
$$\forall s,(j,j')\in J_s,k'\in K_{sj'},k\in K_{sj},l'\in L_{j'},l\in L_j \tag{4.121}$$

$$\text{Indirect:} \quad \sum_{k\in K_{sj}}\sum_{l\in L_j} t_{sjkl}\tilde{X}_{sjkl,j'k'l'} < t_{sj'k'l'} \leq \sum_{k\in K_{sj}}\sum_{l\in L_j} t_{sj,k+1,l}\tilde{X}_{sjkl,j'k'l'}$$
$$\forall s,(j,j') \in J_s, k' \in K_{sj'}, l' \in L_{j'} \tag{4.122}$$

There are two exceptions to (4.122):

(i) when $t_{sjk_{sj}^{max}l} < t_{sj'k'l'} \leq t_{sj0,l+1}$, transfer at $t_{sj'k'l'}$ will then lie between two successive operations instead of two successive transfers of the same operation. To get (4.113) and (4.122) right, we assume by convention that:

$$t_{sj,k_{sj}^{max}+1,l} = t_{sj0,l+1} \qquad l > 0$$

(ii) transfers taking place at $t = 0$, we then need to assume that:

$$t_{sj,k_{sj}^{max}+1,0} = -\delta T \qquad \delta T \text{ is a small real value}$$

With these two assumptions, the above constraints will then always be valid.

If the direct link constraint is used, the $\tilde{X}_{sjkl,j'k'l'}$ can then be determined by the following equation:

$$\tilde{X}_{sjkl,j'k'l'} = X_{sj'k'l',j,k+1,l} - X_{sj'k'l',jkl} \tag{4.123}$$

This states that $\tilde{X}_{sjkl,j'k'l'} = 1$ if and only if $X_{sj'k'l',j,k+1,l} = 1$ and $X_{sj'k'l',jkl} = 0$, *i.e.* $t_{sjkl} < t_{sj'k'l'} \leq t_{sj,k+1,l}$, which just matches the definition. Note that the right hand side can never be negative since any transfers before t_{sjkl} must be before $t_{sj,k+1,l}$ as well.

Alternatively, when the indirect link is applied, the $X_{sjkl,j'k'l'}$ can then be determined by:

$$X_{sjkl,j'k'l'} = \sum_{k'',l''<k',l'} \tilde{X}_{sj'k''l'',jkl}$$
$$\forall s,(j,j') \in J_s, k \in K_{sj}, l \in L_j, k' \in K_{sj'}, l' \in L_{j'} \tag{4.124}$$

This simply states that if one transfer at t_{sjkl} takes place within any interval earlier than the current one under examination, it is definitely earlier than the present one as well, which again matches the definition of $X_{sjkl,j'k'l'}$.

2. *Ordering of material transfer and plant delivery.* Again, both direct and indirect link to times can be created. Taking advantage of the fixed delivery time sequence,

linear constraints can always be obtained:

$$\text{Direct:} \qquad \frac{t_{sj'k'l'} - t^D_{sk}}{T^{max}} < Y^D_{sk,j'k'l'} \leq 1 + \frac{t_{sj'k'l'} - t^D_{sk}}{T^{max}}$$

$$\forall s, k \in K^D_s, j' \in J_s, k' \in K_{sj'}, l' \in L_{j'} \quad (4.125)$$

$$\text{Then:} \qquad \tilde{Y}^D_{sk,j'k'l'} = Y^D_{sk,j'k'l'} - Y^D_{s,k+1,j'k'l'} \quad (4.126)$$

$$\text{Indirect:} \qquad \sum_{k \in K^D_s} t^D_{sk} \tilde{Y}^D_{sk,j'k'l'} \leq t_{sj'k'l'} < \sum_{k \in K^D_s} t^D_{s,k+1} \tilde{Y}^D_{sk,j'k'l'}$$

$$\forall s, j' \in J_s, k' \in K_{sj'}, l' \in L_{j'} \quad (4.127)$$

$$\text{Then:} \qquad Y^D_{sk,j'k'l'} = \sum_{k'' \geq k} \tilde{Y}^D_{sk'',j'k'l'} \quad (4.128)$$

While constraints (4.125) and (4.127) detect all deliveries made up to $t_{sj'k'l'}$, for pair-wise checking of the relative order, we also need another similar set of linear constraints to detect all transfers made up to plant delivery at t^D_{sk}:

$$\text{Direct:} \qquad \frac{t^D_{sk'} - t_{sjkl}}{T^{max}} < Y^D_{sjkl,k'} \leq 1 + \frac{t^D_{sk'} - t_{sjkl}}{T^{max}}$$

$$\forall s, j \in J_s, k \in K_{sj}, l \in L_j, k' \in K^D_s \quad (4.129)$$

$$\text{Then:} \qquad \tilde{Y}^D_{sjkl,k'} = Y^D_{sjkl,k'+1} - Y^D_{sjkl,k'} \quad (4.130)$$

$$\text{Indirect:} \qquad \sum_{k' \in K^D_s} t^D_{sk'} \tilde{Y}^D_{sjkl,k'} < t_{sjkl} \leq \sum_{k' \in K^D_s} t^D_{s,k'+1} \tilde{Y}^D_{sjkl,k'}$$

$$\forall s, j \in J_s, k \in K_{sj}, l \in L_j \quad (4.131)$$

$$\text{Then:} \qquad Y^D_{sjkl,k'} = \sum_{k'' < k'} \tilde{Y}^D_{sjkl,k''} \quad (4.132)$$

Assumptions needed are:

$$t^D_{s0} = -\delta T; \qquad t^D_{s,k^{max}+1} = T^{max} + \delta T$$

3. *Ordering of storage capacity changes and material transfer.* First, binary variables are introduced to select the right storage capacity for the checking at $t_{sj'k'l'}$:

$$\text{Direct:} \qquad \frac{t_{sj'k'l'} - t^C_{sk}}{T^{max}} < Y^C_{sk,j'k'l'} \leq 1 + \frac{t_{sj'k'l'} - t^C_{sk}}{T^{max}}$$

$$\forall s, k \in K^C_s, j' \in J_s, k' \in K_{sj'}, l' \in L_{j'} \quad (4.133)$$

$$\text{Then:} \qquad \tilde{Y}^C_{sk,j'k'l'} = Y^C_{sk,j'k'l'} - Y^C_{s,k+1,j'k'l'} \quad (4.134)$$

Indirect:
$$\sum_{k\in K_s^C} t_{sk}^C \tilde{Y}_{sk,j'k'l'}^C \leq t_{sj'k'l'} < \sum_{k\in K_s^C} t_{s,k+1}^C \tilde{Y}_{sk,j'k'l'}^C$$
$$\forall s, j' \in J_s, k' \in K_{sj'}, l' \in L_{j'} \quad (4.135)$$

Then:
$$Y_{sk,j'k'l'}^C = \sum_{k''\geq k} \tilde{Y}_{sk'',j'k'l'}^C \quad (4.136)$$

Second, variables are introduced for the checking of material balance at t_{sk}^C:

Direct:
$$\frac{t_{sk'}^C - t_{sjkl}}{T^{max}} < Y_{sjkl,k'}^C \leq 1 + \frac{t_{sk'}^C - t_{sjkl}}{T^{max}}$$
$$\forall s, j \in J_s, k \in K_{sj}, l \in L_j, k' \in K_s^C \quad (4.137)$$

Then:
$$\tilde{Y}_{sjkl,k'}^C = Y_{sjkl,k'+1}^C - Y_{sjkl,k'}^C \quad (4.138)$$

Indirect:
$$\sum_{k'\in K_s^C} t_{sk'}^C \tilde{Y}_{sjkl,k'}^C < t_{sjkl} \leq \sum_{k'\in K_s^C} t_{s,k'+1}^C \tilde{Y}_{sjkl,k'}^C$$
$$\forall s, j \in J_s, k \in K_{sj}, l \in L_j \quad (4.139)$$

Then:
$$Y_{sjkl,k'}^C = \sum_{k''<k'} \tilde{Y}_{sjkl,k''}^C \quad (4.140)$$

Assumptions needed are:

$$t_{s0}^C = -\delta T; \qquad t_{s,k^{max}+1}^C = T^{max} + \delta T$$

4. *Ordering of each pair of resource interval start-times:*

Direct:
$$\frac{t_{rj'k'l'} - t_{rjkl}}{T^{max}} < X_{rjkl,j'k'l'} \leq 1 + \frac{t_{rj'k'l'} - t_{rjkl}}{T^{max}}$$
$$\forall s, (j, j') \in J_r, k' \in K_{rj'}, k \in K_{rj}, l' \in L_{j'}, l \in L_j \quad (4.141)$$

Then:
$$\tilde{X}_{rjkl,j'k'l'} = X_{rjkl,j'k'l'} - X_{rj,k+1,l,j'k'l'} \quad (4.142)$$

Indirect:
$$\sum_{k,l} t_{rjkl} \tilde{X}_{rjkl,j'k'l'} \leq t_{rj'k'l'} < \sum_{k,l} t_{rj,k+1,l} \tilde{X}_{rjkl,j'k'l'}$$
$$\forall r, (j, j') \in J_r, k' \in K_{rj'}, l' \in L_{j'} \quad (4.143)$$

Then:
$$X_{rjkl,j'k'l'} = \sum_{k'',l''\geq k,l} \tilde{X}_{rjk''l'',j'k'l'} \quad (4.144)$$

Again, these will be valid with the assumptions:

$$t_{rj,k_{rj}^{max}+1,l} = t_{rj0,l+1}; \qquad t_{rj,k_{rj}^{max}+1,l_j^{max}} = T^{max} + \delta T$$

5. *Ordering of unit resource interval times and changes of resource availability/prices.* This can also be made similarly to **3**. First, for the selection of the proper availability interval of resource r:

$$\text{Direct:} \qquad \frac{t_{rj'k'l'} - t_{rk}}{T^{max}} < Y_{rk,j'k'l'} \leq 1 + \frac{t_{rj'k'l'} - t_{rk}}{T^{max}}$$

$$\forall r, k \in K_r, j' \in J_r, k' \in K_{rj'}, l' \in L_{j'} \quad (4.145)$$

$$\text{Then:} \qquad \tilde{Y}_{rk,j'k'l'} = Y_{rk,j'k'l'} - Y_{r,k+1,j'k'l'} \quad (4.146)$$

$$\text{Indirect:} \qquad \sum_{k \in K_r} t_{rk} \tilde{Y}_{rk,j'k'l'} \leq t_{rj'k'l'} < \sum_{k \in K_r} t_{r,k+1} \tilde{Y}_{rk,j'k'l'}$$

$$\forall r, j' \in J_r, k' \in K_{rj'}, l' \in L_{j'} \quad (4.147)$$

$$\text{Then:} \qquad Y_{rk,j'k'l'} = \sum_{k'' \geq k} \tilde{Y}_{rk'',j'k'l'} \quad (4.148)$$

For the checking of resource generation/consumption over t_{rk}:

$$\text{Direct:} \qquad \frac{t_{rk'} - t_{rjkl}}{T^{max}} < Y_{rjkl,k'} \leq 1 + \frac{t_{rk'} - t_{rjkl}}{T^{max}}$$

$$\forall r, k' \in K_r, j \in J_r, k \in K_{rj}, l \in L_j \quad (4.149)$$

$$\text{Then:} \qquad \hat{Y}_{rjkl,k'} = Y_{rjkl,k'} - Y_{rj,k+1,l,k'} \quad (4.150)$$

$$\text{Indirect:} \qquad \sum_{k \in K_{rj}} \sum_{l \in L_j} t_{rjkl} \hat{Y}_{rjkl,k'} \leq t_{rk'} < \sum_{k \in K_{rj}} \sum_{l \in L_j} t_{rj,k+1,l} \hat{Y}_{rjkl,k'}$$

$$\forall r, j \in J_r, k' \in K_r \quad (4.151)$$

$$\text{Then:} \qquad Y_{rjkl,k'} = \sum_{k'',l'' \geq k,l} \hat{Y}_{rjk''l'',k'} \quad (4.152)$$

Note that the indirect link (4.151) will lose the advantage of fixed times t_{rk} and result in more nonlinearities. Therefore, for this case it is better to choose (4.149), and either replacing $Y_{rjkl,k'}$ using (4.152) or replacing $\hat{Y}_{rjkl,k'}$ in (4.110) using (4.150). However, for other cases in **1-5**, when dealing with the fixed times such as t^D_{sk}, t^C_{sk}, t_{rk}, the indirect link to times may give a tighter constraint.

Finally, as stated in the definition of binary variables in section *4.4.2*, only one out of each pair of the integer variables (X, Y); ($\tilde{X}$, $\tilde{Y}$) are basic for the formulation. The choice of using (X, Y) usually leads to linear ordering constraints, while the choice of using ($\tilde{X}$, $\tilde{Y}$) normally requires fewer constraints for the linearization in the material balance equations.

4.4.5 Constraint Linearization

The formulation of constraints has now been completed. Note that nonlinearities in bilinear form still exist in the delivery constraints (4.92) and (4.93), the material balance constraints (4.99), (4.101) and (4.103) (or alternatively in (4.105), (4.106) and (4.107)), the resource usage constraints (4.109), (4.110) and ordering constraints (4.122), (4.143). To linearize them, we need again to define continuous variables to replace the bilinear terms and to effect such definition through a set of linear constraints.

Delivery Constraints Linearization

To linearize the bilinear term in the delivery constraints (4.92) and (4.93) , we define new continuous variables as:

$$\tilde{D}_{sk,j'k'l'} = \tilde{D}_{sj'k'l'}\tilde{Y}^{D}_{sk,j'k'l'} \qquad \hat{D}_{sk,j'k'l'} = \hat{D}_{sj'k'l'}Y'^{D}_{sk,j'k'l'}$$

Here the special ordered-set properties for $\tilde{Y}^{D}_{sk,j'k'l'}$ can be employed to derive the following linear constraints enforcing the above definition:

$$\tilde{D}_{sj'k'l'} = \sum_{k \in K^{D}_{s}} \tilde{D}_{sk,j'k'l'}; \qquad D^{min}_{sk}\tilde{Y}^{D}_{sk,j'k'l'} \leq \tilde{D}_{sk,j'k'l'} \leq D^{max}_{sk}\tilde{Y}^{D}_{sk,j'k'l'} \quad (4.153)$$

Straightforward linearization is applied to (4.93):

$$\hat{D}^{min}_{sk}Y'^{D}_{sk,j'k'l'} \leq \hat{D}_{sk,j'k'l'} \leq \hat{D}^{max}_{sk}Y'^{D}_{sk,j'k'l'} \quad (4.154)$$

$$\hat{D}^{min}_{sk}(1 - Y'^{D}_{sk,j'k'l'}) \leq \hat{D}_{sj'k'l'} - \hat{D}_{sk,j'k'l'} \leq \hat{D}^{max}_{sk}(1 - Y'^{D}_{sk,j'k'l'}) \quad (4.155)$$

From the definition of $Y^{D}_{sk,j'k'l'}$ and the relation between direct and indirect link variables, $Y'^{D}_{sk,j'k'l'}$ can be determined by:

$$Y'^{D}_{sk,j'k'l'} = Y^{D}_{sk,j'k'l'} - \sum_{k'' \geq k} \tilde{Y}^{D}_{sj'k'l',k''} \quad (4.156)$$

where the second term indicates $t^{D}_{sk} < t_{sj'k'l'}$.

Material Balance Constraints – Alternatives for Linearization

Alternative linearization can be derived for the material balance constraints by choosing different binary variables. First, the straightforward linearization of (4.99), (4.101) and

(4.103) is defined as:

$$B_{sijkl,j'k'l'} = B_{ijl}X_{sjkl,j'k'l'}; \qquad B^D_{sijkl,k'} = B_{ijl}Y^D_{sjkl,k'}; \qquad B^C_{sijkl,k'} = B_{ijl}Y^C_{sjkl,k'}$$

$$D_{sjkl,j'k'l'} = D_{sjkl}X_{sjkl,j'k'l'}; \qquad D^D_{sjkl,k'} = D_{sjkl}Y^D_{sjkl,k'}; \qquad D^C_{sjkl,k'} = D_{sjkl}Y^C_{sjkl,k'}$$

Each definition above is then effected through a set of linear inequality constraints:

$$0 \le B_{sijkl,j'k'l'} \le V^{max}_{ij} X_{sjkl,j'k'l'} \tag{4.157}$$

$$0 \le B_{ijl} - B_{sijkl,j'k'l'} \le V^{max}_{ij}(1 - X_{sjkl,j'k'l'}) \tag{4.158}$$

$$0 \le B^D_{sijkl,k'} \le V^{max}_{ij} Y^D_{sjkl,k'} \tag{4.159}$$

$$0 \le B_{ijl} - B^D_{sijkl,k'} \le V^{max}_{ij}(1 - Y^D_{sjkl,k'}) \tag{4.160}$$

$$0 \le B^C_{sijkl,k'} \le V^{max}_{ij} Y^C_{sjkl,k'} \tag{4.161}$$

$$0 \le B_{ijl} - B^C_{sijkl,k'} \le V^{max}_{ij}(1 - Y^C_{sjkl,k'}) \tag{4.162}$$

$$D^{min}_{sjkl} X_{sjkl,j'k'l'} \le D_{sjkl,j'k'l'} \le D^{max}_{sjkl} X_{sjkl,j'k'l'} \tag{4.163}$$

$$D^{min}_{sjkl}(1 - X_{sjkl,j'k'l'}) \le D_{sjkl} - D_{sjkl,j'k'l'} \le D^{max}_{sjkl}(1 - X_{sjkl,j'k'l'}) \tag{4.164}$$

$$D^{min}_{sjkl} Y^D_{sjkl,k'} \le D^D_{sjkl,k'} \le D^{max}_{sjkl} Y^D_{sjkl,k'} \tag{4.165}$$

$$D^{min}_{sjkl}(1 - Y^D_{sjkl,k'}) \le D_{sjkl} - D^D_{sjkl,k'} \le D^{max}_{sjkl}(1 - Y^D_{sjkl,k'}) \tag{4.166}$$

$$D^{min}_{sjkl} Y^C_{sjkl,k'} \le D^C_{sjkl,k'} \le D^{max}_{sjkl} Y^C_{sjkl,k'} \tag{4.167}$$

$$D^{min}_{sjkl}(1 - Y^C_{sjkl,k'}) \le D_{sjkl} - D^C_{sjkl,k'} \le D^{max}_{sjkl}(1 - Y^C_{sjkl,k'}) \tag{4.168}$$

Alternatively, integer variables with special-ordered set properties can be applied for the linearization, the advantage of which was already mentioned in section *4.2.5*. The material balance constraints are modified into the form of (4.105), (4.106) and (4.107). The introduced variables are defined as:

$$\tilde{B}_{sijkl,j'k'l'} = B_{ijl}\tilde{X}_{sj'k'l',jkl}; \qquad \tilde{B}^D_{sijkl,k'} = B_{ijl}\tilde{Y}^D_{sjkl,k'}; \qquad \tilde{B}^C_{sijkl,k'} = B_{ijl}\tilde{Y}^C_{sjkl,k'}$$

$$\tilde{D}_{sjkl,j'k'l'} = D_{sjkl}\tilde{X}_{sj'k'l',jkl}; \qquad \tilde{D}^D_{sjkl,k'} = D_{sjkl}\tilde{Y}^D_{sjkl,k'}; \qquad \tilde{D}^C_{sjkl,k'} = D_{sjkl}\tilde{Y}^C_{sjkl,k'}$$

Then the special ordered set properties can be employed to produce a tight linearization form.

$$B_{ijl} = \sum_{k'l'} \tilde{B}_{sijkl,j'k'l'}; \qquad 0 \le \tilde{B}_{sijkl,j'k'l'} \le V^{max}_{ij} \tilde{X}_{sj'k'l',jkl} \tag{4.169a}$$

$$B_{ijl} = \sum_{k'} \tilde{B}^{D}_{sijkl,k'}; \qquad 0 \leq \tilde{B}^{D}_{sijkl,k'} \leq V^{max}_{ij} \tilde{Y}^{D}_{sjkl,k'} \tag{4.169b}$$

$$B_{ijl} = \sum_{k'} \tilde{B}^{C}_{sijkl,k'}; \qquad 0 \leq \tilde{B}^{C}_{sijkl,k'} \leq V^{max}_{ij} \tilde{Y}^{C}_{sjkl,k'} \tag{4.169c}$$

$$D_{sjkl} = \sum_{k'l'} \tilde{D}_{sjkl,j'k'l'}; \qquad D^{min}_{sjkl} \tilde{X}_{sj'k'l',jkl} \leq \tilde{D}_{sjkl,j'k'l'} \leq D^{max}_{sjkl} \tilde{X}_{sj'k'l',jkl} \tag{4.169d}$$

$$D_{sjkl} = \sum_{k'} \tilde{D}^{D}_{sjkl,k'}; \qquad D^{min}_{sk'} \tilde{Y}^{D}_{sjkl,k'} \leq \tilde{D}^{D}_{sjkl,k'} \leq D^{max}_{sk'} \tilde{Y}^{D}_{sjkl,k'} \tag{4.169e}$$

$$D_{sjkl} = \sum_{k'} \tilde{D}^{C}_{sjkl,k'}; \qquad D^{min}_{sjkl} \tilde{Y}^{C}_{sjkl,k'} \leq \tilde{D}^{C}_{sjkl,k'} \leq D^{max}_{sjkl} \tilde{Y}^{C}_{sjkl,k'} \tag{4.169f}$$

Resource Usage Constraints Linearization

For the resource usage constraints, (4.109), (4.110), the special ordered set property, although it exists, can not help. Only straightforward linearizations are applied to these constraints by defining:

$$R_{rijkl,j'k'l'} = R_{rijkl} \tilde{X}_{rjkl,j'k'l'}; \qquad R_{rijkl,k'} = R_{rijkl} \hat{Y}_{rjkl,k'}$$

then imposing:

$$R^{min}_{rijk}(1 - \tilde{X}_{rjkl,j'k'l'}) \leq R_{rijkl} - R_{rijkl,j'k'l'} \leq R^{max}_{rijk}(1 - \tilde{X}_{rjkl,j'k'l'}) \tag{4.170a}$$

$$R^{min}_{rijk} \tilde{X}_{rjkl,j'k'l'} \leq R_{rijkl,j'k'l'} \leq R^{max}_{rijk} \tilde{X}_{rjkl,j'k'l'} \tag{4.170b}$$

$$\text{where} \qquad R^{max}_{rijk} = \alpha_{rijk} + \beta_{rijk} V^{max}_{ij} \tag{4.170c}$$

$$R^{min}_{rijk}(1 - \hat{Y}_{rjkl,k'}) \leq R_{rijkl} - R_{rijkl,k'} \leq R^{max}_{rijk}(1 - \hat{Y}_{rjkl,k'}) \tag{4.170d}$$

$$R^{min}_{rijk} \hat{Y}_{rjkl,k'} \leq R_{rijkl,k'} \leq R^{max}_{rijk} \hat{Y}_{rjkl,k'} \tag{4.170e}$$

Ordering Constraints Linearization

Again, for ordering constraints, (4.122), (4.143), only straightforward linearizations can be applied:

$$\underline{t}_{sjkl,j'k'l'} = t_{sjkl} \tilde{X}_{sjkl,j'k'l'} \qquad \overline{t}_{sjkl,j'k'l'} = t_{sj,k+1,l} \tilde{X}_{sjkl,j'k'l'}$$

$$\underline{t}_{rjkl,j'k'l'} = t_{rjkl} \tilde{X}_{rjkl,j'k'l'} \qquad \overline{t}_{rjkl,j'k'l'} = t_{rj,k+1,l} \tilde{X}_{rjkl,j'k'l'}$$

For $\underline{t}_{sjkl,j'k'l'}$ and $\bar{t}_{sjkl,j'k'l'}$:

$$0 \leq \underline{t}_{sjkl,j'k'l'} \leq T^{max}\tilde{X}_{sjkl,j'k'l'};\;\; 0 \leq t_{sjkl} - \underline{t}_{sjkl,j'k'l'} \leq T^{max}(1-\tilde{X}_{sjkl,j'k'l'}) \quad (4.171a)$$

$$0 \leq \bar{t}_{sjkl,j'k'l'} \leq T^{max}\tilde{X}_{sjkl,j'k'l'};\;\; 0 \leq t_{sj,k+1,l} - \bar{t}_{sjkl,j'k'l'} \leq T^{max}(1-\tilde{X}_{sjkl,j'k'l'}) \quad (4.171b)$$

For $\underline{t}_{rjkl,j'k'l'}$ and $\bar{t}_{rjkl,j'k'l'}$:

$$0 \leq \underline{t}_{rjkl,j'k'l'} \leq T^{max}\tilde{X}_{rjkl,j'k'l'};\;\; 0 \leq t_{rjkl} - \underline{t}_{rjkl,j'k'l'} \leq T^{max}(1-\tilde{X}_{rjkl,j'k'l'}) \quad (4.172a)$$

$$0 \leq \bar{t}_{rjkl,j'k'l'} \leq T^{max}\tilde{X}_{rjkl,j'k'l'};\;\; 0 \leq t_{rj,k+1,l} - \bar{t}_{rjkl,j'k'l'} \leq T^{max}(1-\tilde{X}_{rjkl,j'k'l'}) \quad (4.172b)$$

When using $\tilde{X}_{sjkl,j'k'l'}$ and $\tilde{X}_{rjkl,j'k'l'}$ for ordering, in order to avoid the introduction of nonlinearities, we can dispense with (4.122), (4.143). Then the linear time constraints (4.121) and (4.141) can be applied for the ordering by replacing $X_{sjkl,j'k'l'}$ and $X_{rjkl,j'k'l'}$ using (4.124) and (4.144) respectively.

4.4.6 Objective Function

To maximize profit, two types of cost/values should be taken into account: the cost/value for delivery of raw material and products, and the cost of resource usage. As we are now only considering instantaneous material transfer and deliveries, the cost can simply be obtained by:

$$\texttt{Cost/Value of raw material/products} = \sum_{s}\sum_{k\in K_s^D} c_{sk}D_{sk} \quad (4.173)$$

However, for the cost of resource usage, we have to allow for different costs in each interval $[t_{rjkl},\ t_{rj,k+1,l})$. By the definition of the binary variables $Y_{rjkl,k'}$, we can introduce: $\tilde{Y}_{rjkl,k'-1} = Y_{rjkl,k'} - Y_{rjkl,k'-1}$, indicating: $t_{r,k'-1} < t_{rjkl} \leq t_{rk'}$. Hence:

$$\texttt{Resource usage cost} = \sum_{r}\sum_{i\in I_r}\sum_{j\in J_i}\sum_{k\in K_{rj}}\sum_{l\in L_j}(-R_{rijkl})\left\{\sum_{k'\in K_r}\hat{Y}_{rjkl,k'}(t_{rk'} - t_{r,k'-1})c_{r,k'-1}\right.$$

$$\left. - \sum_{k'\in K_r}\tilde{Y}_{rjkl,k'-1}(t_{rjkl} - t_{r,k'-1})c_{r,k'-1} + \sum_{k'\in K_r}\tilde{Y}_{rj,k+1,l,k'}(t_{rj,k+1,l} - t_{rk'})c_{rk'}\right\} \quad \forall r \quad (4.174)$$

The first inner summation states cost within the interval $[t_{rjkl},\ t_{rj,k+1,l})$ according to the cost within $[t_{r,k'-1},\ t_{rk'})$, as shown in Figure 4.6. However, since there may be no usage of resource in $[t_{r,k'-1},\ t_{rjkl})$, this over-charged cost is deleted by the second inner summation term. Furthermore, nothing has been taken into account for the usage in

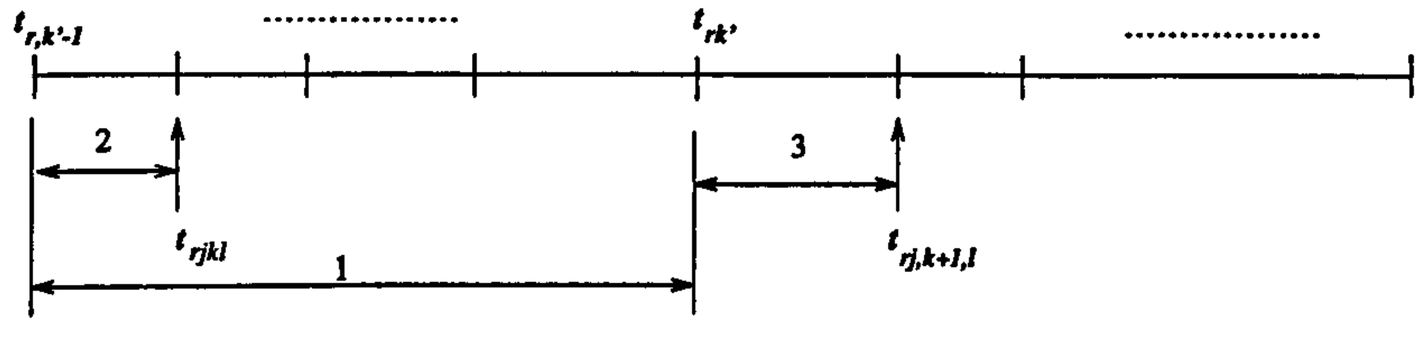


Figure 4.6: Resource price changing during unit operation

$[t_{rk'},\ t_{rj,k+1,l})$, and we add the third inner summation for this reason. There may also be the possibility that no change of prices takes place within the interval $[t_{rjkl}, t_{rj,k+1,l})$. In such a case, the first term becomes zero. The second and third terms together will give the cost of resource usage during this interval of operation. For example, assume $[t_{rjkl},\ t_{rj,k+1,l}] \subset (t_{r,k'-1},\ t_{rk'})$, then $\tilde{Y}_{rjkl,k'-1} = \tilde{Y}_{rj,k+1,l,k'-1} = 1$ and the cost function becomes:

$$\sum_{r}\sum_{i\in I_r}\sum_{j\in J_i}\sum_{k\in K_{sj}}\sum_{l\in L_j}(-R_{rijkl})\cdot(t_{rj,k+1,l}-t_{rjkl})c_{r,k'-1}$$

However, bilinear forms arise in the cost function of resource usage. Therefore, we define new continuous variables to linearize them:

$$R_{rijkl,k'} = R_{rijkl}\hat{Y}_{rjkl,k'};\qquad R_{rk',ijkl} = R_{rijkl}\tilde{Y}_{rjkl,k'};\qquad R_{rk',ij,k+1,l} = R_{rijkl}\tilde{Y}_{rj,k+1,l,k'}$$

The linearization of $R_{rijkl,k'}$ is already given in (4.170d, e). By using the special ordered set constraints property, we can produce a tighter linearization for the other two variables:

$$\sum_{k'\in K_r} R_{rk',ijkl} = R_{rijkl};\qquad R^{min}_{rijk}\tilde{Y}_{rjkl,k'} \le R_{rk',ijkl} \le R^{max}_{rijk}\tilde{Y}_{rjkl,k'} \tag{4.175}$$

$$\sum_{k'\in K_r} R_{rk',ij,k+1,l} = R_{rijkl};\qquad R^{min}_{rijk}\tilde{Y}_{rj,k+1,l,k'} \le R_{rk',ij,k+1,l} \le R^{max}_{rijk}\tilde{Y}_{rj,k+1,l,k'} \tag{4.176}$$

Again, nonlinearities exist in the objective function. This has to be solved as a MINLP problem with local linearization of (4.174) at each iteration.

4.4.7 Tightening of the Formulation by Adding Integer Cuts

In order to reduce the search space for the optimization problem, some logical constraints can be added as integer cuts by exploring the underlying structure of the operation sequences.

Considering the time constraints describing the details of the *Operation Sequence*, it is always true for the operations in the same unit that:

$$t_{sjkl} \leq t_{sjk''l''} \qquad \forall s, j \in J_s, (k, k'') \in K_{sj}, (l'' > l) \in L_j \tag{4.177a}$$

$$\text{or} \quad t_{sjkl} \leq t_{sjk''l} \qquad \forall s, j \in J_s, l \in L_j, (k'' > k) \in K_{sj} \tag{4.177b}$$

This also indicates that the binary variables defining event orders need only be introduced for different units, *i.e.* for $j \neq j'$. Furthermore, logic between the integer variables can be inferred as:

$$X_{sjkl,j'k'l'} \geq X_{sjk''l'',j'k'l'} \quad \forall s, (j, j') \in J_s, (k, k'') \in K_{sj}, (l'' > l) \in L_j, k' \in K_{sj'}, l' \in L_{j'} \tag{4.178a}$$

$$X_{sjkl,j'k'l'} \geq X_{sjk''l,j'k'l'} \quad \forall s, (j, j') \in J_s, (k'' > k) \in K_{sj}, l \in L_j, k' \in K_{sj'}, l' \in L_{j'} \tag{4.178b}$$

(4.178a) and (4.178b) can also be implemented in an alternative way, which will reduce the number of constraints without changing the efficiency:

$$X_{sjkl,j'k'l'} \geq X_{sjk'',l+1,j'k'l'} \quad \forall s, (j, j') \in J_s, (k, k'') \in K_{sj}, l \in L_j, k' \in K_{sj'}, l' \in L_{j'} \tag{4.179a}$$

$$X_{sjkl,j'k'l'} \geq X_{sj,k+1,l,j'k'l'} \quad \forall s, (j, j') \in J_s, k \in K_{sj}, l \in L_j, k' \in K_{sj'}, l' \in L_{j'} \tag{4.179b}$$

While (4.179a), (4.179b) are obtained from the ordering of material transfers for operations in unit j, symmetric constraints can be inferred from the ordering of material transfers for operations in unit j' as well, written as:

$$X_{sjkl,j'k'l'} \geq X_{sjkl,j'k'',l'-1} \quad \forall s, (j, j') \in J_s, k \in K_{sj}, l \in L_j, (k', k'') \in K_{sj'}, l' \in L_{j'} \tag{4.180a}$$

$$X_{sjkl,j'k'l'} \geq X_{sjkl,j',k'-1,l'} \quad \forall s, (j, j') \in J_s, k \in K_{sj}, l \in L_j, k' \in K_{sj'}, l' \in L_{j'} \tag{4.180b}$$

Furthermore, from the definition of $X_{sjkl,j'k'l'}$, at least one of the binary variables $X_{sjkl,j'k'l'}$, $X_{sj'k'l',jkl}$ must take the upper bound value. Therefore the following additional cuts can also be introduced:

$$X_{sjkl,j'k'l'} + X_{sj'k'l',jkl} \geq 1 \quad \forall s, (j, j') \in J_s, k \in K_{sj}, l \in L_j, k' \in K_{sj'}, l' \in L_j \tag{4.181}$$

These integer cuts can also be applied to $\tilde{X}_{sjkl,j'k'l'}$ using the relation in (4.124). However, in this case, (4.180b) will prove to be redundant, while (4.180a) will only be valid (necessary) for $k'' > k'$.

The same procedure also applies to $\tilde{X}_{rjkl,j'k'l'}$, $Y^D_{sk,j'k'l'}$, $\tilde{Y}^D_{sk,j'k'l'}$, $Y^D_{sjkl,k'}$, $\tilde{Y}^D_{sjkl,k'}$, $\tilde{Y}^C_{sk,j'k'l'}$, $Y^C_{sjkl,k'}$, $\tilde{Y}^C_{sjkl,k'}$, $\tilde{Y}_{rk,j'k'l'}$, $\hat{Y}_{rjkl,k'}$, to get similar integer cuts.

4.4.8 An Illustrative Example

Example 1. in *4.2.7* for scheduling with variable processing times is tested again using the proposed formulation based on unit *Operation Sequences* presented in this section. The maximum number of operations for each unit are estimated as: 4, 6, 12, respectively, using the minimum duration of tasks 1, 2, 3.

To get the minimum number of constraints and avoid nonlinearities in the ordering constraints, we employed the direct link constraints (4.121), but with the $X_{sjkl,j'k'l'}$ replaced by $\tilde{X}_{sjkl,j'k'l'}$ using (4.124). Comparing with the *FET* in section *4.2.7* and the "worst case" of *VET* algorithm in section *4.3.5*, this formulation gives a smaller problem with fewer binary variables ($NC = 1084$, $NV = 582$, $NIV = 236$). But the maximum number of non-overlap operations for each unit is over-estimated and the ordering mechanism still results in a large integrality gap as in *VET*. The objective function of the fully relaxed problem is 199.99. The computational result shows that it is still very time consuming to solve the problem to optimality. Using SCICONIC as the MILP solver, the best integer solution obtained is 71.46 but the search was not completed after 150,000 LPs with 45,000 seconds CPU time.

4.5 Summary and Discussions

In this *Chapter*, several continuous time representation schemes were proposed and different formulations based on these time schemes were developed. The key feature of the formulations presented is that a detailed task model is used in the formulation of scheduling problems. More importantly, by using the continuous time representation, it is possible to deal with the scheduling problem with variable operating conditions, such as varying processing times depending on operating variables.

The essential difference between these time schemes is in the way of ordering individual system events. First, a single time reference is postulated to reflect all system events, giving the *FET* and *VET* formulations by choosing the event-time interval differently. If the event-time interval is chosen in a uniformly fixed manner, we have the *FET* procedure, and the advantage of doing so is that it provides a fixed reference against the variation of individual transfers, keeping most constraints, especially the time constraint linear.

The integrality gap is relatively small hence making the MILP easier to solve. This advantage is even more obvious by reformulating the problem using variable and constraint disaggregation. However, when fine time resolution is required, the problem will become very large. Therefore, there is a balance between sacrificing time accuracy and problem size.

To cope with the problem arising with *FET*, we can alternatively use variable event times. The basic idea is to eliminate most of the postulated event-times in *FET* by mapping only the real system events onto the common time sequence. This will reduce the problem size greatly if successful. One difficulty is that the use of variable event times makes the identification of individual transfer times on the common time sequence more complicated compared with *FET*. Furthermore, significant nonlinearities will arise concerning the description of event orders in the formulation. Although most of the nonlinearities can be reformulated into a linear form since they appear as the cross product between binary and continuous variables, the linearization leads to a larger integrality gap. Other issues concerned with the efficiency of *VET* is how to decide the real number of system events and the reflection of their variation range as accurately as possible. This will directly affect the final problem size.

Finally, to address the latter issues mentioned in *VET*, the single time sequence was abandoned. Recognising that the individual events are related to units performing tasks, the unit *Operation Sequence* is introduced as a new time reference. In this case, the relative order of each pair of interacting events has to be detected. Two types of ordering mechanism, direct and indirect link to times, are proposed and their relation is explored. Using this as a new way of labelling system events, the problem size will be smaller compared with the "worst case" used in *VET* even with the most conservative estimation of the number of operations in each unit. However, with the still over-estimated number of non-overlap operations, this scheme does not reduce the integrality gap of the problem nor make it easier to solve.

Chapter 5

Extensions of the Basic Formulations

As mentioned in *Chapter 3*, the original *STN* has since been modified and the discrete time algorithm based on it has been extended to address some detailed yet practical aspects in the scheduling problem. Aspects such as limited plant connectivity, the requirement of cleaning of processing equipment which can be either sequence, frequency or time dependent, temporary unavailability of equipment, general storage consideration of materials in storage tanks as well as in processing units, the storage of feeding and product material in units before and after processing, the approximation of continuous transfers and so on. Thanks to the enthusiasm on the basic *STN* concept, some of the aspects have been under improvement by successive works.

In this *Chapter*, we are going to show that it is also possible, and in some cases even easier, to address these issues within our general context. The basic ideas of dealing with these issues in the discrete time algorithm will be reviewed first and the accommodation of them in the various continuous time representations (*FET, VET, Operation Sequence*) will then be discussed.

5.1 Plant Connectivity

In the original discrete time formulation, processing equipment was simply characterised by its suitability for performing various tasks. This actually ignored the plant topology altogether by implicitly assuming that full plant connectivity exists between all available

processing units and to/from the storage. However, as posed by Crooks (1992), there are usually well defined and restricted connections between units in multipurpose plants to reduce costs. This obviously raises the issue of plant connectivity in the scheduling formulation.

To accommodate such complications, Kondili *et al.* (1993a) also proposed a simple modification of the *STN* by task "splitting", as shown in Figure 5.1. The original *STN* segment corresponding to the full equipment connectivity is shown in Figure 5.1(a). The suitability of units for tasks is given by:

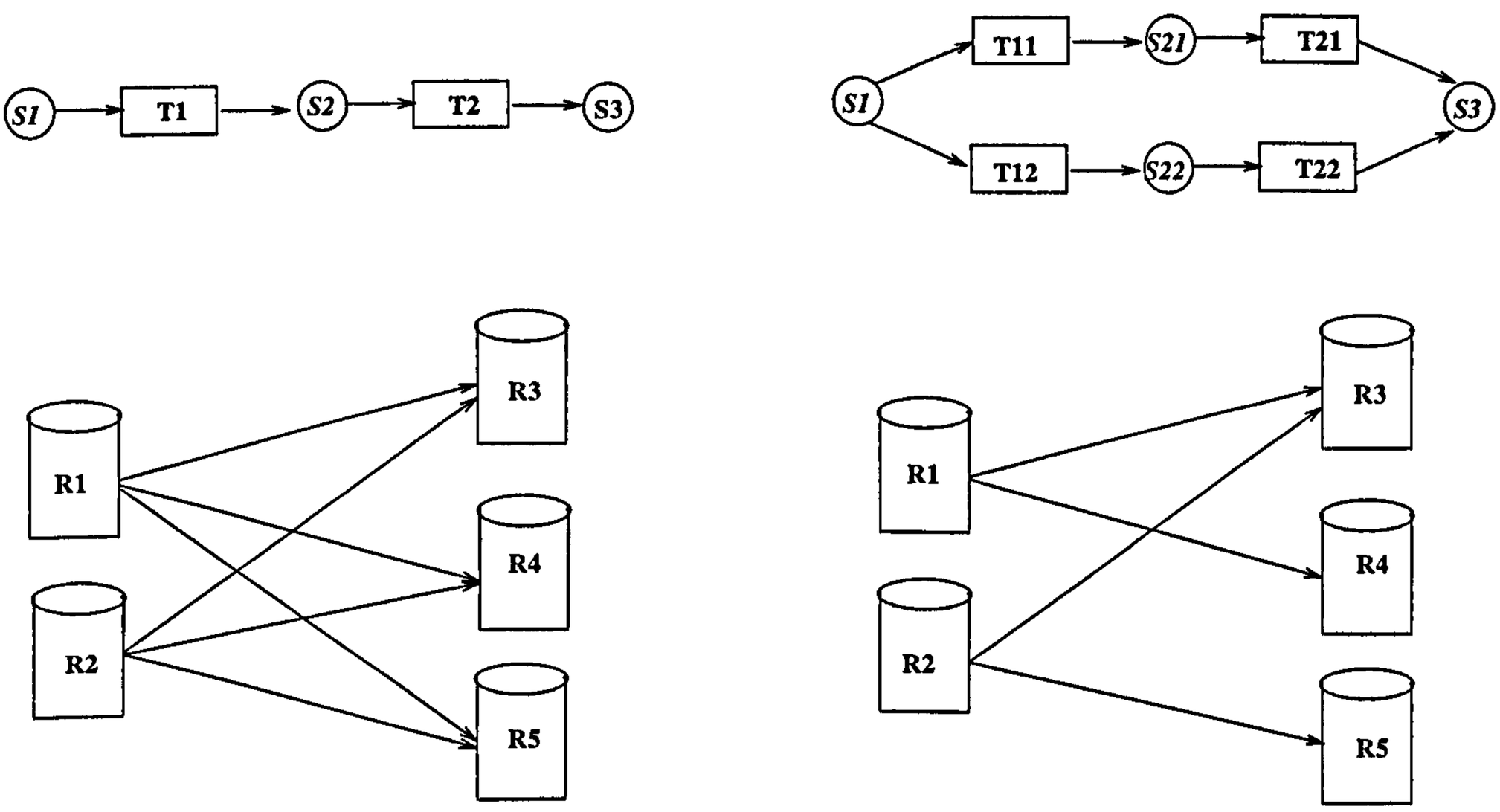
Task	Suitable equipment
T1	R1, R2
T2	R3, R4, R5

The table above gives no information about the connections between the available equipment R1 – R5. However, if connectivity is limited as shown in Figure 5.1(b), the original *STN* is modified into the form of Figure 5.1(b) by splitting task T1 into T11, T12 and T2 into T21, T22, respectively. The effect of limited connectivity complications can then be accommodated by defining the new unit-task suitabilities as following:

Task	Suitable equipment
T11	R1
T12	R2
T21	R3, R4
T22	R3, R5

By doing this, the decision variables W_{ijl} are then defined according to the new suitabilities. No fundamental change in the original formulation of the scheduling problem is needed no matter which time representation scheme, *FET, VET or Operation Sequence*, is used. This indicates that using the continuous time representation does not complicate the way connectivity is treated.

To describe the connectivity and modified unit-task suitability automatically, Crooks (1992) proposed the *maximal State-Task Network (mSTN)* representation. The *mSTN* will expand the original *STN* according to equipment connectivities and then apply the *mSTN* reduction scheme ("shrink" *mSTN*) to give a more efficient representation by reducing the number of variables and constraints. Since no explicit timings are involved in the



(a)Original STN -- Full Equipment Connectivity

(b) Modified STN -- Limited Equipment Connectivity

Figure 5.1: *STN* modifications for limited plant connectivity

consideration of such connectivity, this procedure can also be applied to the formulation using continuous time representation.

5.2 Operational Constraints

To guarantee the product quality (*e.g.* food, paint) or to prevent formation of a hazardous mixture (*e.g.* explosive residuals), there is often an important requirement of cleaning or some other preparation of processing equipment between operations. Such preparation may also incur the use of resources (*e.g.* manpower, steam *etc.*). To deal with this, we can then treat them as new tasks.

As one type of operational constraints, the general cleaning requirement was originally included in the formulation of Kondili *et al.* Two types of cleaning were considered: sequence and frequency dependent cleaning. In the first type, the cleaning requirement depends on the identity and relative order of the two tasks between which the cleaning is taking place. In the second type, the need for cleaning is expressed in terms of the frequency of unit utilization. These two kinds of cleaning were both expressed mathematically.

However, as pointed out by Pantelides (1993), one of the deficiencies of the sequence-dependent cleaning requirements formulated originally by Kondili *et al.* was that the number of constraints was of $O((l^{max})^2)$, where l^{max} is the number of discretization intervals over the entire time horizon of interest. This problem was then addressed by the later work of Crooks (1992) who introduced the concept of *unit-state* to describe the different states (*Ustates*), $k \in \theta_j$, in which an item of processing equipment can exist (*e.g.* "clean", "dirty", "cold", "hot"). A network – *Unit State-Task Network (USTN)* is defined to represent the transition of unit state (*e.g.* from k to k') by carrying out a set of tasks (*USTN* tasks), $i \in I_{jkk'}$. Only allowable state transitions are included in the network.

By introducing a new set of variables characterizing the state of a vessel over the time horizon, $\hat{W}_{jkl}$, a set of constraints to account for the change of unit states and task pre-conditions are formulated according to four rules:

(i) at any time, unit j can only exist in exactly one of a finite set of *Ustates*;

$$\sum_{k \in \theta_j} \hat{W}_{jkl} = 1 \tag{5.1}$$

(ii) a *USTN* task i can only use unit j if it is in a defined subset of its allowable *Ustates*

$$W_{ijl} \leq \sum_{k | i \in \bigcup_{k'} I_{jkk'}} \hat{W}_{jkl} \tag{5.2}$$

(iii) if no *USTN* task starts in a unit at time l, the unit state remains unchanged at $l+1$;

$$-\sum_{i \in \bigcup_{k'} I_{jkk'}} W_{ijl} \leq \hat{W}_{jk,l+1} - \hat{W}_{jkl} \leq \sum_{i \in \bigcup_{k'} I_{jkk'}} W_{ijl} \tag{5.3}$$

(iv) a *USTN* task i starting in unit j at time l switches the unit at $l+1$ to a unique defined *Ustate* (one of its allowable *Ustate* set).

$$\hat{W}_{jkl} - \hat{W}_{jk',l+1} + \sum_{i \in I_{jkk'}} W_{ijl} \leq 1 \tag{5.4}$$

The initial and final required states of a unit can be met by fixing the value of $\hat{W}_{jk_o1}$ and $\hat{W}_{jk_f,l^{max}}$, respectively. Not only the sequence-dependent cleaning requirements, but also more general task pre-conditions for unit utilization can be formulated.

A third category of cleaning requirement is the time-dependent cleaning (Barbosa-Póvoa, 1994). This is similar to the frequency dependent requirement but with the frequency limits expressed in terms of time length instead of the number of unit utilizations.

All of the three types of cleaning have been treated individually by developing a different mathematical formulation for each of them. However, it is possible to describe them in a generalized way. The basic idea is to introduce a measure of unit *usability* for each required operational condition. For the frequency and time dependent cleaning, the definition of usability is straightforward, while for the sequence dependent cleaning, which is a special situation of the general precedence constraint, we introduced the new concept of task-precedence as a basis for the definition of usability.

5.2.1 Task Precedence Concept

The basic idea for the *task precedence* is that we consider all the suitable pre-conditions for a certain task i being performed in a specific unit as conditions left by the performing of other tasks (of course, the initial condition should be known *a priori*). Therefore, a limited set of tasks may precede (or follow) a given task in a given unit. The following sets are then defined to characterise the task precedence order in each unit.

$\overline{\Omega}_{ij}$: Set of tasks which can precede task i in unit j

Ω_{ij} : Set of tasks which can follow task i in unit j

Example: Suppose unit j is suitable for producing both "dark" (Do-dark) and "light" (Do-light) dyes. Cleaning is required (Do-clean) if Do-dark is going to be followed by Do-light but not vice versa. No cleaning is required between successive Do-dark or Do-light operations.

From this information, the precedence sets can be created as:

$\overline{\Omega}_{Do-idle,j}$	=	{any task};	$\Omega_{Do-idle,j}$	=	{?}
$\overline{\Omega}_{Do-light,j}$	=	{Do-light, Do-clean};	$\Omega_{Do-light,j}$	=	{any task}
$\overline{\Omega}_{Do-dark,j}$	=	{Do-light, Do-dark, Do-clean};	$\Omega_{Do-dark,j}$	=	{Do-idle, Do-dark, Do-clean}
$\overline{\Omega}_{Do-clean,j}$	=	{(Do-light), Do-dark};	$\Omega_{Do-clean,j}$	=	{Do-idle, Do-light, Do-dark}

To compare the unit-state concept of *USTN* with the task precedence, the former can be formalized by defining the following two sets of unit states:

$\overline{\Sigma}_{ij}$: the set of initial unit-states of unit j suitable for task i

Σ_{ij} : the final state of unit j after task i

Then, for the example above we have:

$\overline{\Sigma}_{Do-idle,j}$	=	{Light-Soiled, Dark-Soiled, Clean};	$\Sigma_{Do-idle,j}$	=	{?}
$\overline{\Sigma}_{Do-light,j}$	=	{Light-Soiled, Clean};	$\Sigma_{Do-light,j}$	=	{Light-Soiled}
$\overline{\Sigma}_{Do-dark,j}$	=	{Light-Soiled, Dark-Soiled, Clean};	$\Sigma_{Do-dark,j}$	=	{Dark-Soiled}
$\overline{\Sigma}_{Do-clean,j}$	=	{(Light-Soiled), Dark-Soiled};	$\Sigma_{Do-clean,j}$	=	{Clean}

Since there is a one-to-one correspondence between task i and the final state Σ_{ij} resulting from it (note that Do-idle is in fact a non-USTN task which makes no transition of unit-state), the sets $\overline{\Sigma}_{ij}$ and $\overline{\Omega}_{ij}$ are identical. It is also true that the concept of unit-state is implicit in writing the set $\overline{\Omega}_{ij}$. Figure 5.2 and 5.3 gives the task-precedence diagram and the *USTN* representation for this example, respectively.

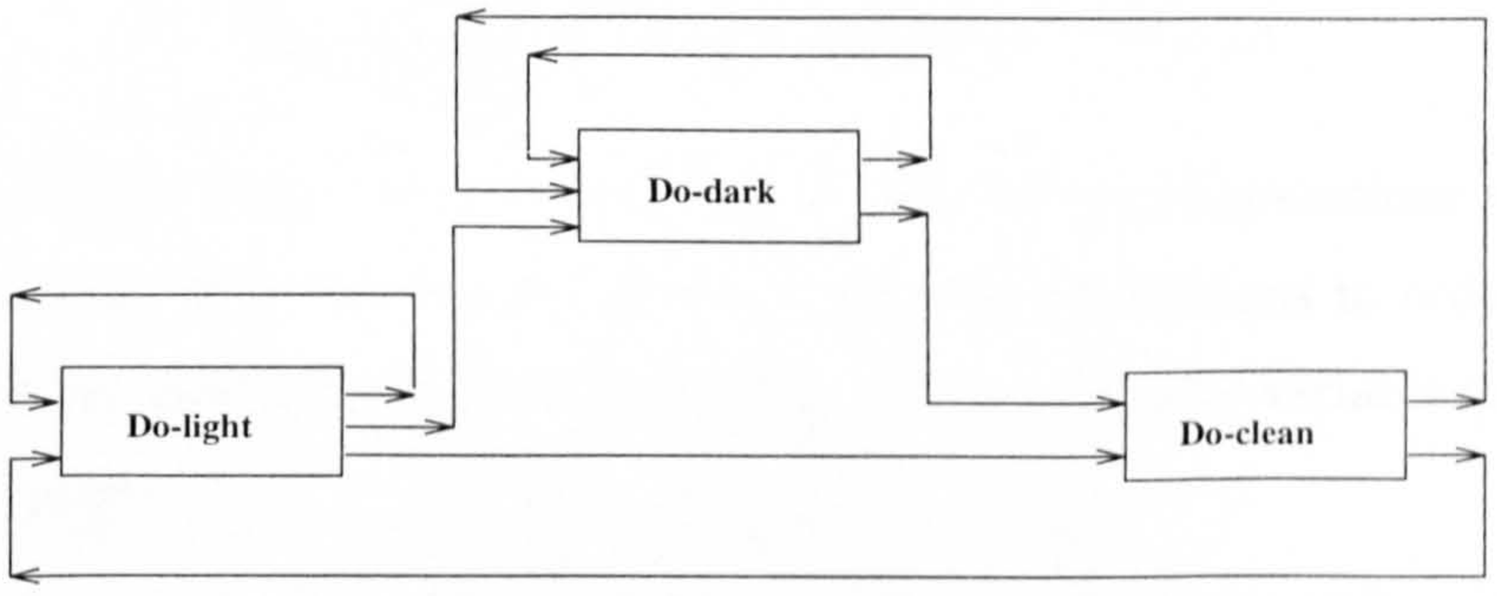


Figure 5.2: Task precedence diagram

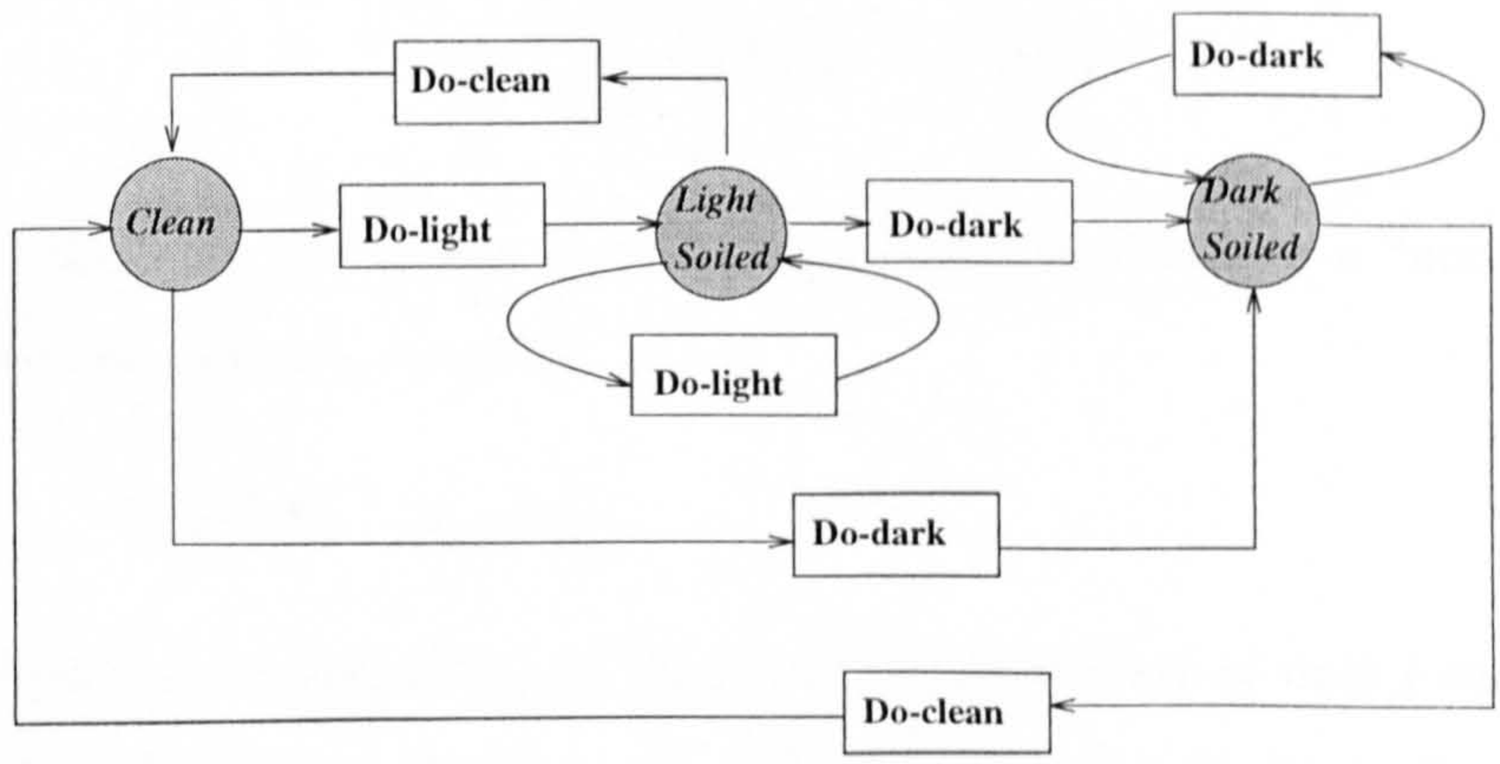


Figure 5.3: *USTN* representation

5.2.2 Generalized Operational Constraint

We assume the existence of null tasks, and hence have the following constraint:

$$\sum_{i \in I_j} W_{ijl} = 1 \qquad \forall j, l \tag{5.5}$$

where l is for an event-time T_l in *FET, VET* or an operation l in the *Operation Sequence.* For *Operation Sequence*, (5.5) is exactly the same as the allocation constraint and the null tasks are forced to the end of the sequence, while for *FET* and *VET*, $W_{0jl} = 1$ indicates the unit is either idle at T_l or occupied by a task started earlier. In either case, no change of usability would occur in this unit.

For each unit j at a given time (event-time T_l or start-time of operation l), we assign measures of usability U_{ujl}. The usability $u = 1, 2, 3, \ldots$ is created by set-up tasks (*e.g.* cleaning, descaling, maintenance, inspection/testing, *etc.*). Each task executed in the unit changes its usabilities, so that:

$$U_{uj,l+1} = \sum_{i \in I_j} (\alpha_{uij} U_{ujl} + \beta_{uij}) W_{ijl} \qquad \forall u, j, l \tag{5.6}$$

where α_{uij} is zero or one, depending on whether the history of operations will affect the usability of a unit. But β_{uij} could depend on operating conditions in order to give the information about the task previously processed. U_{ujl} is a positive variable and we assume that U_{uj0} are given.

Each task has a threshold usability value γ_{uij} for execution in the unit, so the feasibility condition is:

$$U_{ujl} \geq \sum_{i \in I_j} \gamma_{uij} W_{ijl} \qquad \forall u, j, l \tag{5.7}$$

If a task $\bar{i}$ does not change the usabilities, such as a null task or a "non-USTN" task as in the unit-state concept, we have:

$$\alpha_{u\bar{i}j} = 1; \quad \beta_{u\bar{i}j} = 0; \quad \gamma_{u\bar{i}j} = 0$$

Also, by fixing the value of $U_{ujl^{max}}$, the specified final state of unit j can be ensured. Therefore, defining the α_{uij}, β_{uij}, γ_{uij} in different ways, the above constraints can then be equally applied to the three identified cleaning requirements as well as to the general precedence constraints.

The binary term in (5.6) can again be linearized by introducing:

$$U_{uijl} = U_{ujl} W_{ijl}$$

and using the special ordered set constraints property of (5.5):

$$\sum_{i \in I_j} U_{uijl} = U_{ujl}; \qquad 0 \leq U_{uijl} \leq U_{uj}^{max} W_{ijl} \tag{5.8}$$

5.2.2.1 Frequency-dependent Cleaning

The unit must be cleaned after a given number of successive operations since previous cleaning. However, cleaning does not necessarily have to take place only when the unit is used up to its frequency limit. Instead, it can be performed before this limit is reached. For instance, the otherwise idle period can be scheduled to fulfill the cleaning requirement in order to get the unit ready for a busy operation period later on.

This can also be treated by the generalized operational constraints given in (5.6) and (5.7). To measure the unit usability, both the history of operations and the status of the task just processed should be taken into account. The history records the number of operations carried out and should always be reset after a cleaning task is executed.

Assume at most m_j successive operations can be performed in unit j after the previous cleaning. The parameters in (5.6) are defined as:

$$\text{For cleaning task } i_c: \qquad \alpha_{ui_cj} = 0; \quad \beta_{ui_cj} = m_j; \quad \gamma_{ui_cj} = 0 \tag{5.9}$$

$$\text{For other tasks } i: \qquad \alpha_{uij} = 1; \quad \beta_{uij} = -1; \quad \gamma_{uij} = 0 \tag{5.10}$$

This also applies to the case where units need maintenance after a limited number of operations.

5.2.2.2 Time-dependent Cleaning

The usability of units depends on time in use. This is a similar case to frequency-dependent cleaning, except that the operation history should record the total time, instead of the number, of previous operations. Suppose the maximum time for which unit j may be used without cleaning is t_j^{max}, then it can also be catered for through (5.6) and (5.7) by

defining:

$$\text{For cleaning task } i_c\text{:} \quad \alpha_{ui_cj} = 0; \quad \beta_{ui_cj} = t_j^{max}; \quad \gamma_{ui_cj} = 0 \tag{5.11}$$

$$\text{For other tasks } i\text{:} \quad \alpha_{uij} = 1; \quad \beta_{uij} = -\tau_{ij}; \quad \gamma_{uij} = 0 \tag{5.12}$$

where τ_{ij} is the processing time of task i in unit j. The bilinear term $\tau_{ij}W_{ijl}$ in (5.6) can then be replaced by using the time constraints in *Chapter 4*, *i.e.*

$$\text{For Event-time Sequence:} \quad \tau_{ij}W_{ijl} = t_{ijk_i^{max}l} - T_l \tag{5.13}$$

$$\text{For Operation Sequence:} \quad \sum_{i \in I_j} \tau_{ij}W_{ijl} = t_{jl}^{max} - t_{jl}^{0} \tag{5.14}$$

5.2.2.3 Precedence Constraints

When task precedence needs to be considered, only information on the previously processed task is necessary to determine the unit usability. Therefore, for all tasks which change the unit usability after processing, $\alpha_{uij} = 0$.

To conserve the required task-precedence by the generalized operational constraints, it suffices to define β_{uij}, γ_{uij} so that:

$$\beta_{ui'j} \geq \gamma_{uij} \quad \forall i' \in \overline{\Omega}_{ij} \tag{5.15}$$

$$\beta_{ui'j} < \gamma_{uij} \quad \forall i' \notin \overline{\Omega}_{ij} \tag{5.16}$$

Any sequence dependent tasks such as sequence dependent cleaning and maintenance can all simply be treated just as a special case of precedence constraints. Here, (5.7) needs only be applied to those tasks which will change the usability of units after processing in it.

As a simple example, suppose we have a cleaning task i_c, and a set of tasks, $i = 1, 2, \cdots, i^{max}$, such that cleaning between tasks is necessary only if the preceding task has a smaller task number, *i.e.*

$$\overline{\Omega}_{ij} = \{i' | i' \geq i\} \cup \{i' | i' = i_c\}$$

$$\text{For cleaning task } i_c\text{:} \quad \alpha_{ui_cj} = 0; \quad \beta_{ui_cj} = i^{max}; \quad \gamma_{ui_cj} = 0 \tag{5.17}$$

$$\text{For other tasks } i\text{:} \quad \alpha_{uij} = 0; \quad \beta_{uij} = i; \quad \gamma_{uij} = i \tag{5.18}$$

A more complicated sequence-dependent situation arises when all suitable tasks for a unit are grouped in families. In the most general case, cleaning or change-over may be required between tasks from different families as well as between different tasks within the same family. Furthermore, different cleaning effort may be required within and between different families. The example below is used to illustrate the introduction of usabilities and the choice of parameters β_{uij}, γ_{uij}.

Example: Considering a unit j suitable for tasks from two families: Family1={A, B}; Family2={C, D}. Cleaning between tasks in Family1 and Family2 are denoted by "CH-F1" and "CH-F2" respectively. Cleaning between families is described as "CH-ALL". Using the task precedence concept, set $\overline{\Omega}_{ij}$ for each task can be obtained as:

$$\begin{aligned}
\overline{\Omega}_{A,j} &= \{\text{A, CH-F1, CH-ALL}\}; & \overline{\Omega}_{C,j} &= \{\text{C, CH-F2, CH-ALL}\}\\
\overline{\Omega}_{B,j} &= \{\text{B, CH-F1, CH-ALL}\}; & \overline{\Omega}_{D,j} &= \{\text{D, CH-F2, CH-ALL}\}\\
\overline{\Omega}_{CH-F1,j} &= \{\text{A, B}\}; & \overline{\Omega}_{CH-F2,j} &= \{\text{C, D}\}\\
\overline{\Omega}_{CH-ALL,j} &= \{\text{A, B, C, D}\}
\end{aligned}$$

For this example, the introduced usabilities are: $u \in \{U_A,\ U_B,\ U_C,\ U_D,\ U_{CH-F1},\ U_{CH-F2},\ U_{CH-ALL}\}$. According to (5.15) and (5.16), β_{uij}, γ_{uij} can be chosen as given in the following table:

i	$\beta_{U_A ij}$	$\beta_{U_B ij}$	$\beta_{U_C ij}$	$\beta_{U_D ij}$	$\beta_{U_{CH-F1} ij}$	$\beta_{U_{CH-F2} ij}$	$\beta_{U_{CH-ALL} ij}$
A	1	0	0	0	1	0	1
B	0	1	0	0	1	0	1
C	0	0	1	0	0	1	1
D	0	0	0	1	0	1	1
CH-F1	1	1	0	0	0	0	0
CH-F2	0	0	1	1	0	0	0
CH-ALL	1	1	1	1	0	0	0

i	$\gamma_{U_A ij}$	$\gamma_{U_B ij}$	$\gamma_{U_C ij}$	$\gamma_{U_D ij}$	$\gamma_{U_{CH-F1} ij}$	$\gamma_{U_{CH-F2} ij}$	$\gamma_{U_{CH-ALL} ij}$
A	1	0	0	0	0	0	0
B	0	1	0	0	0	0	0
C	0	0	1	0	0	0	0
D	0	0	0	1	0	0	0
CH-F1	0	0	0	0	1	0	0
CH-F2	0	0	0	0	0	1	0
CH-ALL	0	0	0	0	0	0	1

5.3 Unavailability of Processing Equipment

Items of processing equipment may not be available over specified periods due to the need for maintenance or breakdown, or because of use elsewhere. The general maintenance task has already been treated uniformly as the cleaning and other precedence tasks in section *5.2*. The only remaining problem is when equipment is only available for specified periods. To deal with this case, we could ignore the resource usage during unavailability since this is outside the control of the scheduler.

Suppose unit j is unavailable over the time intervals $[t_{jk}^{Us},\ t_{jk}^{Uf}),\ k \in K_j^U$. We assume then a dummy maintenance task must be allocated to unit j during this period to prevent other "normal" tasks from taking place.

5.3.1 Under FET

Since all event-times T_l are fixed in *FET*, the fixed unavailable interval can then be easily identified. Therefore, we need only ensure that no task will be performed during this period as:

$$\sum_{i \in I_j} \sum_{k < k_i^{max}} \sum_{l \leq l'} \tilde{X}_{ijkll'} = 0 \qquad \forall j, k' \in K_j^U, T_{l'} \in [t_{jk'}^{Us},\ t_{jk'}^{Uf}) \tag{5.19}$$

5.3.2 Under VET

The unavailability period has to be mapped onto the event-time sequence. Define:

$$\tilde{Y}_{jkl}^U = 1 \qquad \text{if } T_l \in [t_{jk}^{Us},\ t_{jk}^{Uf}); \quad 0 \text{ otherwise}$$

$$\tilde{Y}_{jkl}^A = 1 \qquad \text{if } T_l \in [t_{jk}^{Uf},\ t_{j,k+1}^{Us}); \quad 0 \text{ otherwise}$$

The allocation constraint is then modified to include the "execution" of the dummy maintenance task so as to ensure no other tasks will be executed during the unavailability period:

$$\sum_{i \in I_j} \sum_{k < k_i^{max}} \sum_{l \leq l'} \tilde{X}_{ijkll'} + \sum_{k \in K_j^U} \tilde{Y}_{jkl'}^U \leq 1 \qquad \forall j, l' \tag{5.20}$$

Each event-time T_l must be within either the available or unavailable interval. The timing and ordering constraints can then be treated in an analogous way to other events

with fixed intervals, such as the plant delivery and capacity changes. Therefore, the following constraints are included:

$$\sum_{k\in K_j^U} \tilde{Y}_{jkl}^U + \sum_{k\in K_j^U} \tilde{Y}_{jkl}^A = 1 \qquad \forall j,l \quad (5.21)$$

$$\sum_{k\in K_j^U} t_{jk}^{Us}\tilde{Y}_{jkl}^U + \sum_{k\in K_j^U} t_{jk}^{Uf}\tilde{Y}_{jkl}^A \le T_l < \sum_{k\in K_j^U} t_{jk}^{Uf}\tilde{Y}_{jkl}^U + \sum_{k\in K_j^U} t_{j,k+1}^{Us}\tilde{Y}_{jkl}^A \qquad \forall j,l \quad (5.22)$$

5.3.3 Under Operation Sequence

Again a binary variable is defined to indicate the operation of a dummy maintenance task in a unit as:

$\tilde{Y}_{jkl}^U = 1$ if operation l covers unavailable period k for unit j

It follows that each unavailable period must be covered by an operation for the dummy maintenance task:

$$\sum_{l\in L_j} \tilde{Y}_{jkl}^U = 1 \qquad \forall j,k \in K_j^U \quad (5.23)$$

$$\sum_{k\in K_j^U} t_{jk}^{Us}\tilde{Y}_{jkl}^U \le t_{jl}^0 \le T^{max} - \sum_{k\in K_j^U} (T^{max} - t_{jk}^{Us})\tilde{Y}_{jkl}^U \qquad \forall j,l \in L_j \quad (5.24)$$

$$\sum_{k\in K_j^U} t_{jk}^{Uf}\tilde{Y}_{jkl}^U \le t_{jl}^{max} \le T^{max} - \sum_{k\in K_j^U} (T^{max} - t_{jk}^{Uf})\tilde{Y}_{jkl}^U \qquad \forall j,l \in L_j \quad (5.25)$$

The allocation constraint also needs to be modified to take account of the operation of the dummy maintenance task:

$$\sum_{i\in I_j} W_{ijl} + \sum_{k\in K_j^U} \tilde{Y}_{jkl}^U = 1, \qquad \forall j,l \quad (5.26)$$

5.4 Storage Tasks

Up to now, we have only considered the storage of material by assuming that the available storage capacity is dedicated to individual states. However, in general, it is often the case that a large number of states of material share a relatively small number of storage units. They have to be used in a multipurpose mode, being capable of storing different material states and even performing different tasks. Therefore material transfer is allowed to/from such units either simply for storage purposes or for processing.

On the other hand, it may also be desirable for units to temporarily hold feed or product immediately preceding or following the processing of tasks. In such cases, the unit may not be suitable for general storage purposes as described above, and no transfers from other units or dedicated storage tanks just for storage purposes should be permitted. Of course, this can be easily covered using the task-precedence concept by regarding them as new tasks. However, with the continuous time representation, this can be regarded as two special cases separately because, as will be shown later on, they can be simply addressed without introducing any additional tasks.

5.4.1 General Storage Tasks

To allow for successive additions and withdrawals, the period during which a unit is allocated to storage of a specific state ($i = i_s$) is subdivided into a sequence of *Storage Tasks*. Each *Storage Task* has a single transfer of the whole quantity into the unit at the beginning, and a single transfer of the same material out of the unit (emptied) at the end.

Addition of such tasks makes no difference to the formulation of the problem. For event-time sequences (*FET, VET*), one *Storage Task* is defined for each event-time interval $[T_l,\ T_{l+1})$, with feeds at the beginning and products at the end and uniform resource usage over the period, *i.e.*:

$$k_{i_s}^{max} = 1; \quad t_{i_s j 0 l} = T_l; \quad t_{i_s j 1 l} = T_{l+1}^{-} \quad \forall s \in S_{i_s}, j \in J_{i_s}, l$$

$$\rho_{s i_s j 0} = -1; \quad \rho_{s i_s j 1} = 1 \quad \forall s \in S_{i_s}$$

For *Operation Sequence*, the introduction of a *Storage Task* can be limited in the same way:

$$k_{i_s s}^{max} = 1; \quad k_{i_s r}^{max} = 1 \quad \forall s \in S_{i_s}, r \in R_{i_s}$$

$$\rho_{s i_s j 0} = -1; \quad \rho_{s i_s j 1} = 1 \quad \forall s \in S_{i_s}, j \in J_{i_s}$$

The processing time for such tasks, $\tau_{s i_s j 1}$ is simply left as a free variable. Also resource is assumed to be consumed over the interval of storage and only one interval is required.

The concept of *Storage Task* makes it possible for the storage to be treated uniformly in the same way as other tasks which usually transform material from one state to another.

In particular, the use of resources to maintain storage conditions (*e.g.* keep the material at a certain temperature) can be naturally catered for within the general framework.

5.4.2 Material Held in Units After Processing

When there is no unit available acting as a buffer storage, the well known storage policy for batch processing, No Intermediate Storage (*NIS*) policy, can be implemented provided that the material concerned is stable. In this case, it is not necessary to remove the product from the processing unit immediately upon completion of the task. Instead of introducing an explicit *Storage Task*, we can simply modify the time constraint.

In the case of *FET, VET*, we can take the advantage of the general task model described in *Chapter 4*, where each task consists of a subsequence of material/resource transfers according to operating conditions. This has been restricted so that once a task starts at time T_l, its successive transfers should then take place at the exact required time later on, as given in (4.35). To adopt the *NIS* policy however, this restriction can simply be relaxed: for example the final transfer of material out of units can occur at any time later than the finishing of processing as stated by the following constraint, which can be a substitution of (4.35):

$$t_{ijk_i^{max}l} \geq T_l + \tau_{ijk_i^{max}l} \qquad \forall i \in I_s, j \in J_i, k \in K_i, l \tag{5.27}$$

Similarly, for *Operation Sequence*, the time constraint describing material transfer during unit operations, (4.83), can be modified for the appropriate states:

$$t_{sjk_{sj}^{max}l} \geq t_{jl} + \sum_{i \in I_j \cap I_s} \tau_{sijk_{sj}^{max}} W_{ijl} \qquad \forall s, j \in J_s, k \in K_{sj}, l \in L_j \tag{5.28}$$

5.4.3 Storage of Feed Material

Feed material can be transferred to units as available and held until they are required (the duration would depend on various factors such as the condition of "upstream" task operation, the availability of other related material/resources *etc.*)

Generally speaking, it should be possible to allow partial transfers of feed to a suitable unit, summing to the recipe requirement, before a task using such feeds takes place in the same or another unit (For the latter, this feed should then be transferred to the unit

performing such a task). However, at this stage only the simple case is considered, that is the feed material is simply held in the unit which uses it in a later operation. The general "blending" of feed material is left for later description in *Chapter 6.*

This is again readily addressed by using the general task model in the continuous time formulations. For example, if feed material needs to be held in the unit before operation starts, the duration between the time of feed transfers and the starting time of task operation can be a variable. In the case of *FET* and *VET*, this is handled by substituting (4.35) with (5.27) for $k = 1$. For the *Operation Sequence*, on the other hand, this is already included in the description of the unit operation sequence in (4.82), so no further extension is necessary.

5.5 Continuous Tasks and Material Transfers

In many processing sites, there is a mixture of batch and continuous production facilities producing various products. Therefore continuous material transfers in the process would form an important class of continuous operations. On the other hand, the increasingly common situation in moving towards flexible multiproduct continuous plants in order to respond to customer orders quickly also gives rise to significant scheduling problems in continuous plants. To be general, there is a need to describe continuous operations in the scheduling formulation.

Although this issue was already identified, the discrete time representation gives much less power to accommodate this satisfactorily. As a compromise, approximation rules were proposed to address this problem (Crooks, 1992) by modelling the continuous task as a batch task of one time step duration which can then be repeatedly allocated to a unit to approximate continuous behaviour. Obviously, discretizing error will occur as a result of such approximations. From the material flow point of view, this is to approximate the continuous profile by a piecewise constant function, as shown in Figure 5.4.

A continuous time formulation will be more powerful to tackle such a situation naturally. With constant transfer rates, the amount in storage changes linearly with time. Normally, a multiproduct/multipurpose continuous plant and the continuous tasks within a mixed batch/continuous plant runs at a constant rate with occasional changes, so stor-

age levels need only be checked at the finite set of points at which rate changes (*e.g.* point B) or instantaneous material transfers (*e.g.* point E) occur. However, the need for small time-intervals to keep the discretizing error at a satisfactory level in the discrete time algorithm will dramatically increase the problem size. This can also be seen in Figure 5.4.

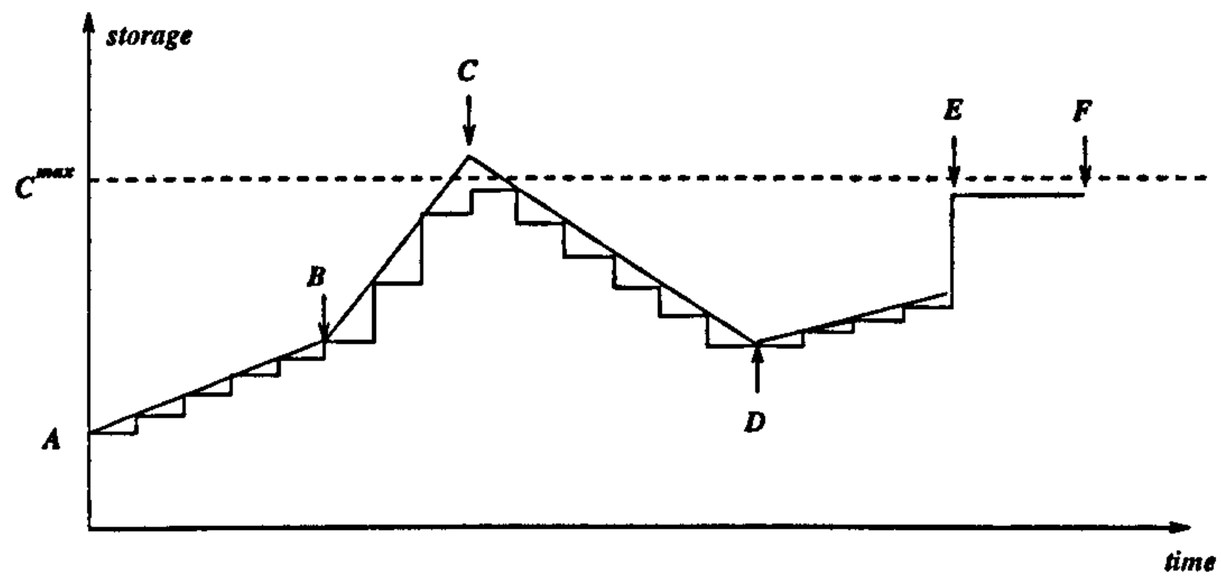


Figure 5.4: Continuous material flow approximated by a piecewise constant function

With the continuous time formulation, the following continuous material transfers are considered:

(i) continuous plant deliveries within the transfer rate limits, $[\overline{D}_{sk}^{min},\ \overline{D}_{sk}^{max}]$ over fixed period $[t_{sk}^{D},\ t_{s,k+1}^{D})$;

(ii) continuous material transfer during task processing within the transfer rate limits $[\overline{R}_{ij}^{min},\ \overline{R}_{ij}^{max}]$.

5.5.1 Under FET and VET

Define:

$\overline{D}_{sl}$: delivery rate over event-time interval $[T_l,\ T_{l+1})$;

$\overline{R}_{ijl}$: transfer rate for task i in unit j over event-time interval $[T_l,\ T_{l+1})$;

C_{sl}^{-} : amount of material s in storage at T_l^{-};

C_{sl}^{+} : amount of material s in storage at T_l^{+}.

Both the delivery and transfer rate should be subject to the appropriate limits:

$$\sum_{k \in K_s^D} \overline{D}_{sk}^{min} \tilde{Y}_{skl}^{D} \leq \overline{D}_{sl} \leq \sum_{k \in K_s^D} \overline{D}_{sk}^{max} \tilde{Y}_{skl}^{D} \tag{5.29}$$

$$\overline{R}_{ij}^{min} W_{ijl} \leq \overline{R}_{ijl} \leq \overline{R}_{ij}^{max} W_{ijl} \tag{5.30}$$

Since continuous delivery is possible over a specified interval, the total amount delivered over this interval should then include both the amount delivered instantaneously and the amount delivered continuously. Therefore, the delivery constraints in (4.58) should be modified to:

$$\tilde{D}_{sk} = \sum_{l} \tilde{Y}^{D}_{skl}(\tilde{D}_{sl} + \overline{D}_{sl}(T_{l+1} - T_l)) \tag{5.31}$$

The total amount of material in storage at $T^{-}_{l'+1}$ is then made up of the amount left in storage at $T^{+}_{l'}$ plus the amount transferred continuously over the interval $[T_{l'},\ T_{l'+1})$:

$$C^{-}_{s,l'+1} = C^{+}_{sl'} - \overline{D}_{sl'}(T_{l'+1} - T_{l'}) + \sum_{i\in I_s}\sum_{j\in J_i}\sum_{l\le l'}\sum_{k\in K_i} \rho_{sijk}\overline{R}_{ijl}\tilde{X}_{ijkll'}(T_{l'+1} - T_{l'}) \tag{5.32}$$

The instantaneous transfer at $T_{l'}$ would then alter the storage amount from $C^{-}_{sl'}$ to $C^{+}_{sl'}$:

$$C^{+}_{s,l'} = C^{-}_{sl'} - D_{sl'} + \sum_{i\in I_s}\sum_{j\in J_i}\sum_{k\in K_i}\sum_{l\le l'} \rho_{sijk}B_{ijl}X_{ijkll'} \tag{5.33}$$

For *FET* again $\tilde{Y}^{D}_{skl}$ can be dispensed with since t^{D}_{sk} can be pre-identified with the fixed event-times T_l. Therefore, all constraints remain linear. However, all terms involving event-times become nonlinear in *VET*. If the continuous delivery and transfer are at specified constant rate, these terms will reduced to bilinear products between integer and continuous variables, and can be linearized in the same spirit as *Chapter 4*. If otherwise, the transfer rate can vary within given bounds, (5.32) should be changed accordingly in order to make the exact linearization. First, we notice that a continuous task usually has one transfer subsequence, *i.e.* $k^{max}_i = 1$. By assuming that each operation of a continuous task lasts one time interval, *i.e.* $\tilde{l}_{ijll} = l+1$, the following relation will then hold according to (4.9):

$$\tilde{X}_{ij0ll} = W_{ijl}$$

This assumption is based on the fact that the duration of a continuous task is a free variable. Any task taking more than one time interval to finish can be deemed as successive operation of the same task over several intervals. (5.32) is then replaced by:

$$C^{-}_{s,l'+1} = C^{+}_{sl'} - \overline{D}_{sl'}(T_{l'+1} - T_{l'}) + \sum_{i\in I_s}\sum_{j\in J_i} \rho_{sij0}\overline{R}_{ijl'}W_{ijl'}(T_{l'+1} - T_{l'}) \tag{5.34}$$

The nonlinear term in the above constraint can now be linearized by defining the total amount of material transferred over interval $[T_l,\ T_{l+1})$ as:

$$R_{ijl} = \overline{R}_{ijl}\tau_{ijl} \qquad \text{where} \quad \tau_{ijl} = W_{ijl}(T_{l+1} - T_l)$$

The above nonlinear definition can be linearized through the following constraints, which is actually a replacement of (5.30):

$$\overline{R}_{ij}^{min}\tau_{ijl} \le R_{ijl} \le \overline{R}_{ij}^{max}\tau_{ijl} \tag{5.35}$$

$$0 \le T_{l+1} - T_l - \tau_{ijl} \le (1 - W_{ijl})\Delta\overline{T} \tag{5.36}$$

The nonlinearity concerning plant delivery in (5.31) and (5.34) can be linearized in the similar way.

5.5.2 Under Operation Sequence

Continuous variables are introduced to account for the continuous plant delivery and continuous transfer during unit operations:

$\overline{D}_{sjkl}$	:	delivery rate of material s over interval $[t_{sjkl}\ t_{sj,k+1,l})$ for operation l in unit j;
$\overline{R}_{ijl}$	:	transfer rate for task i in operation l of unit j;
$C_s(t^-)$	:	amount of materials in storage at t^-;
$C_s(t^+)$	:	amount of materials in storage at t^+.

Again, both the delivery and transfer rate should be subject to the appropriate limits:

$$\sum_{k\in K_s^D} \overline{D}_{sk}^{min}\tilde{Y}^D_{sk,j'k'l'} \le \overline{D}_{sj'k'l'} \le \sum_{k\in K_s^D} \overline{D}_{sk}^{max}\tilde{Y}^D_{sk,j'k'l'} \tag{5.37}$$

$$\overline{R}_{ij}^{min}W_{ijl} \le \overline{R}_{ijl} \le \overline{R}_{ij}^{max}W_{ijl} \tag{5.38}$$

The total amount delivered over each interval $[t^D_{sk},\ t^D_{s,k+1})$ including both the contribution from instantaneous deliveries and continuous deliveries is then:

$$\tilde{D}_{sk} = \sum_{j'\in J_s}\sum_{k'\in K_{sj'}}\sum_{l'\in L_{j'}} \left\{\tilde{Y}^D_{sk,j'k'l'}\tilde{D}_{sj'k'l'} + \tilde{Y}^D_{sj'k'l',k}\overline{D}_{sj'k'l'}\cdot(t_{sj',k'+1,l'} - t_{sj'k'l'})\right\}$$
$$- \sum_{j'\in J_s}\sum_{k'\in K_{sj'}}\sum_{l'\in L_{j'}} \hat{Y}^D_{sj'k'l',k+1}\overline{D}_{sj'k'l'}\cdot(t_{sj',k'+1,l'} - t^D_{s,k+1})$$

$$+\sum_{j'\in J_s}\sum_{k'\in K_{sj'}}\sum_{l'\in L_{j'}}\hat{Y}^{D}_{sj'k'l',k}\overline{D}_{sj'k'l'}\cdot(t_{sj',k'+1,l'}-t^{D}_{sk}) \tag{5.39}$$

where $\hat{Y}^{D}_{sj'k'l',k}=Y^{D}_{sj'k'l',k}-Y^{D}_{sj',k'+1,l',k}$ indicate: $t_{sj'k'l'}\le t^{D}_{sk}<t_{sj',k'+1,l'}$

The above constraint states that except the instantaneous deliveries carried out within the considered interval $[t^{D}_{sk},\ t^{D}_{s,k+1})$, we have to consider three different situations of continuous delivery(*see* Figure 5.5):

(i) continuous deliveries carried out within the current interval;

(ii) continuous deliveries starting within the current interval but finishing after it;

(iii) continuous deliveries finishing within the current interval but starting before it.

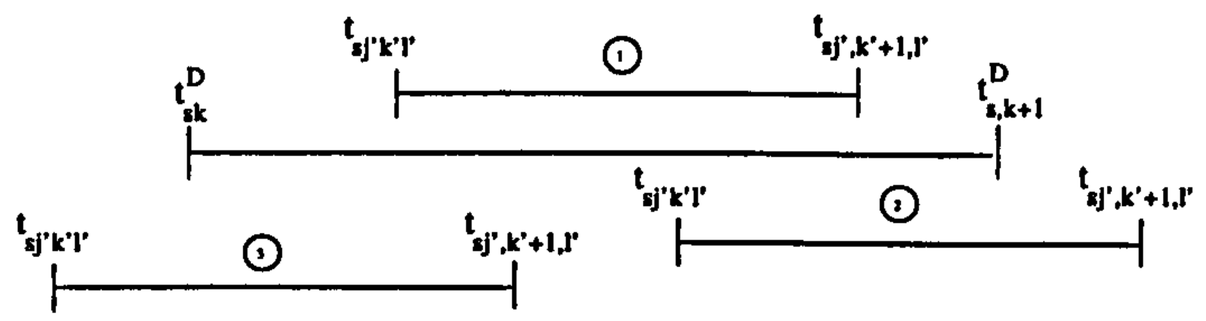


Figure 5.5: Continuous delivery *vs* current delivery interval $[t^{D}_{sk},\ t^{D}_{s,k+1})$

The check of storage demand against capacity can again be carried out at a finite set of times $t_{sj'k'l'}$, $t^{D}_{sk'}$ and $t^{C}_{sk'}$ either by a cumulative material balance or an incremental one. Also, the amount of material in storage must be checked immediately before and after each time. Taking the balance at transfer time $t_{sj'k'l'}$ as an example:

Cumulative balance

The amount of material in storage up to the current time is:

$$C_s(t^{+}_{sj'k'l'})=C_s(0)+\sum_{j\in J_s}\sum_{k\in K_{sj}}\sum_{l\in L_j}\left\{\sum_{i\in I_j}\rho_{sijk}B_{ijl}-D_{sjkl}\right\}X_{sjkl,j'k'l'}$$
$$+\sum_{j\in J_s}\sum_{k\in K_{sj}}\sum_{l\in L_j}\left\{\sum_{i\in I_j}\rho_{sijk}\overline{R}_{ijl}-\overline{D}_{sjkl}\right\}X_{sjkl,j'k'l'}\cdot(t_{sj,k+1,l}-t_{sjkl})$$
$$-\sum_{j\in J_s}\sum_{k\in K_{sj}}\sum_{l\in L_j}\left\{\sum_{i\in I_j}\rho_{sijk}\overline{R}_{ijl}-\overline{D}_{sjkl}\right\}\hat{X}_{sjkl,j'k'l'}\cdot(t_{sj,k+1,l}-t_{sj'k'l'}) \tag{5.40}$$

where $\hat{X}_{sjkl,j'k'l'}=X_{sjkl,j'k'l'}-X_{sj,k+1,l,j'k'l'}$ indicate: $t_{sjkl}\le t_{sj'k'l'}<t_{sj,k+1,l}$

The first sum gives the net amount accumulated up to $t_{sj'k'l'}$ by instantaneous transfers and deliveries. The second term includes all continuous transfers and deliveries started before $t_{sj'k'l'}$. Since these may last over $t_{sj'k'l'}$, this is accounted for by subtracting the amount delivered after $t_{sj'k'l'}$. Different situations of continuous transfer and delivery are given in Figure 5.6.

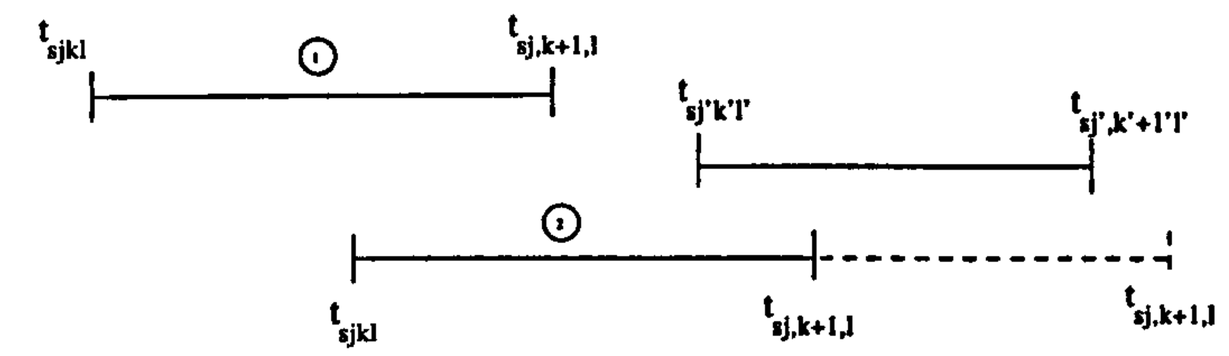


Figure 5.6: Continuous transfer and delivery *vs* current transfer time $t_{sj'k'l'}$

To check the amount in storage before the instantaneous transfer taking place at the current time, we have:

$$C_s(t^-_{sj'k'l'}) = C_s(0) + \sum_{j\in J_s}\sum_{k\in K_{sj}}\sum_{l\in L_j}\left\{\sum_{i\in I_j}\rho_{sijk}B_{ijl} - D_{sjkl}\right\}X'_{sjkl,j'k'l'}$$

$$+\sum_{j\in J_s}\sum_{k\in K_{sj}}\sum_{l\in L_j}\left\{\sum_{i\in I_j}\rho_{sijk}\overline{R}_{ijl} - \overline{D}_{sjkl}\right\}X_{sjkl,j'k'l'}\cdot(t_{sj,k+1,l} - t_{sjkl})$$

$$-\sum_{j\in J_s}\sum_{k\in K_{sj}}\sum_{l\in L_j}\left\{\sum_{i\in I_j}\rho_{sijk}\overline{R}_{ijl} - \overline{D}_{sjkl}\right\}\hat{X}_{sjkl,j'k'l'}\cdot(t_{sj,k+1,l} - t_{sj'k'l'}) \quad (5.41)$$

where $X'_{sjkl,j'k'l'} = \sum_{k'',l''\geq k,l}\tilde{X}_{sjk''l'',j'k'l'}$ indicates: $t_{sjkl} < t_{sj'k'l'}$

Incremental balance

The incremental balance over each transfer interval can be obtained by subtracting the cumulative balance written at $t^-_{sj'k'l'}$, $t^+_{sj'k'l'}$ from its counterpart written at the previous times. The relation between the two sets of integer variables (X-, Y-) and ($\tilde{X}$-, $\tilde{Y}$-) in *Chapter 4* are then applied to obtain:

$$C_s(t^+_{sj'k'l'}) = C_s(t^+_{sj',k'-1,l'}) + \sum_{j\in J_s}\sum_{k\in K_{sj}}\sum_{l\in L_j}\left\{\sum_{i\in I_j}\rho_{sijk}B_{ijl} - D_{sjkl}\right\}\tilde{X}_{sj',k'-1,l',jkl}$$

$$+\sum_{j\in J_s}\sum_{k\in K_{sj}}\sum_{l\in L_j}\left\{\sum_{i\in I_j}\rho_{sijk}\overline{R}_{ijl} - \overline{D}_{sjkl}\right\}\tilde{X}_{sj',k'-1,l',jkl}\cdot(t_{sj,k+1,l} - t_{sjkl})$$

$$-\sum_{j\in J_s}\sum_{k\in K_{sj}}\sum_{l\in L_j}\left\{\sum_{i\in I_j}\rho_{sijk}\overline{R}_{ijl}-\overline{D}_{sjkl}\right\}\hat{X}_{sjkl,j'k'l'}\cdot(t_{sj,k+1,l}-t_{sj'k'l'})$$

$$+\sum_{j\in J_s}\sum_{k\in K_{sj}}\sum_{l\in L_j}\left\{\sum_{i\in I_j}\rho_{sijk}\overline{R}_{ijl}-\overline{D}_{sjkl}\right\}\hat{X}_{sjkl,j',k'-1,l'}\cdot(t_{sj,k+1,l}-t_{sj',k'-1,l'}) \quad (5.42)$$

$$C_s(t^-_{sj'k'l'})=C_s(t^-_{sj',k'-1,l'})+\sum_{j\in J_s}\sum_{k\in K_{sj}}\sum_{l\in L_j}\left\{\sum_{i\in I_j}\rho_{sijk}B_{ijl}-D_{sjkl}\right\}\hat{X}_{sj',k'-1,l',jkl}$$

$$+\sum_{j\in J_s}\sum_{k\in K_{sj}}\sum_{l\in L_j}\left\{\sum_{i\in I_j}\rho_{sijk}\overline{R}_{ijl}-\overline{D}_{sjkl}\right\}\tilde{X}_{sj',k'-1,l',jkl}\cdot(t_{sj,k+1,l}-t_{sjkl})$$

$$-\sum_{j\in J_s}\sum_{k\in K_{sj}}\sum_{l\in L_j}\left\{\sum_{i\in I_j}\rho_{sijk}\overline{R}_{ijl}-\overline{D}_{sjkl}\right\}\hat{X}_{sjkl,j'k'l'}\cdot(t_{sj,k+1,l}-t_{sj'k'l'})$$

$$+\sum_{j\in J_s}\sum_{k\in K_{sj}}\sum_{l\in L_j}\left\{\sum_{i\in I_j}\rho_{sijk}\overline{R}_{ijl}-\overline{D}_{sjkl}\right\}\hat{X}_{sjkl,j',k'-1,l'}\cdot(t_{sj,k+1,l}-t_{sj',k'-1,l'}) \quad (5.43)$$

Similarly, we can write the balance at other times. All terms involving transfer times t_{sjkl} become nonlinear with variable transfer rates. The linearization can be done analogously to the description for event-time sequence.

5.6 Critiques of the *STN* Framework

Since Kondili (1988), work based on the *STN* has covered a very wide variety of process scheduling problems, leading to the development of a set of conceptually related mathematical formulations which are able to take detailed account of the features of industrial applications. Areas such as short term production scheduling, periodic scheduling, campaign planning and scheduling, maintenance scheduling, scheduling of pipeless plant and multisite scheduling can all be accommodated in the same *STN* framework.

However, as reviewed by Pantelides (1993), the *STN* model of operation is rather restrictive or inefficient in some cases:

- A task always involves a transformation of material from one set of states to another. However, this is not always true especially when considering cleaning operations where states of equipment rather than material are involved.

- A task always takes place in a single item of processing equipment, while in practice some tasks may not be associated with any processing units (such as moving of transferable vessels in pipeless plants), or they may involve more than one item of processing equipment such as in heat-integrated operations.

- Each item of processing equipment is treated distinctly. However, it would be more desirable to describe processing equipment similar to other resources in an aggregated way in cases involving many identical items of equipment.

Overall, the *STN* framework treats the various resources (*e.g.* material, processing equipment, storage capacity, utilities, material transfer and transport devices and utilities) in a non-uniform manner. As a consequence of such non-uniformity, a very large number of different classes of constraints must be introduced in order to describe resource utilization (*e.g.* material balance, resource usage, cleaning tasks, storage tasks *etc.*). On the other hand, the handling of novel types of process scheduling problem will often raise the necessities of formulation of specialised constraints.

The resulting tediousness concerning discrete time formulation has already been reviewed by Pantelides (1993). From our experience of developing the general continuous time representations, even more complication will result, especially in developing the appropriate timing constraints for each new situation.

Therefore, it does give rise to the necessity of addressing the scheduling problem efficiently and uniformly. To achieve this, a new unified framework viewing all production facilities in a uniform way is necessary to establish a new representation of plant so that new scheduling problems can be handled with minimum effort.

Chapter 6

Optimal Operation for Mixed Production Facilities

In *Chapter 4*, several continuous time representations are presented, which provides the basis for various continuous time formulations within the *STN* framework. Their strengths and weaknesses have been analysed. Then, in *Chapter 5*, the flexibility of extending these formulations to accommodate various special yet practical situations related to process scheduling is examined.

However, as noted in *Chapter 5*, the restriction of the *STN* concept on tasks transforming one state of material into another through a single item of equipment will result in complexity in dealing with various special situations arising from process scheduling problems. In this *Chapter*, the concept of Resource-Task Network (*RTN*) (Pantelides, 1993) will be extended to develop new continuous time formulations unifying individual situations described in *Chapter 4* and *Chapter 5*, aiming at tackling general optimal operation of mixed production facilities, comprising multipurpose plant for both batch and continuous operations.

First, the new view of a process will be given using the *RTN* concept with the extension to detailed task models considering general operating conditions. Then the ideas of previous formulations are summarised by giving a uniform characterization of various resource transfers, yielding a similar yet somewhat simpler description. Finally, the unified approach will be presented and alternative procedures for solving the general large mixed integer nonlinear programming problem are discussed.

6.1 A New View for Production Facilities

6.1.1 Processes

The main characteristic of the *RTN* framework is the entirely uniform description and characterization of the available resources, with no distinction between equipment, material and utility. From this point of view, a mixed production facility is then considered as a collection of facilities performing a variety of tasks which consume some resources and generate others. The task operation may be carried out either continuously or batch-wise.

The interaction between the two fundamental concepts in *RTN*, *Resources* and *Tasks*, leads to the *Resource-Task Network*, a bipartite directed graph illustrated in Figure 6.1.

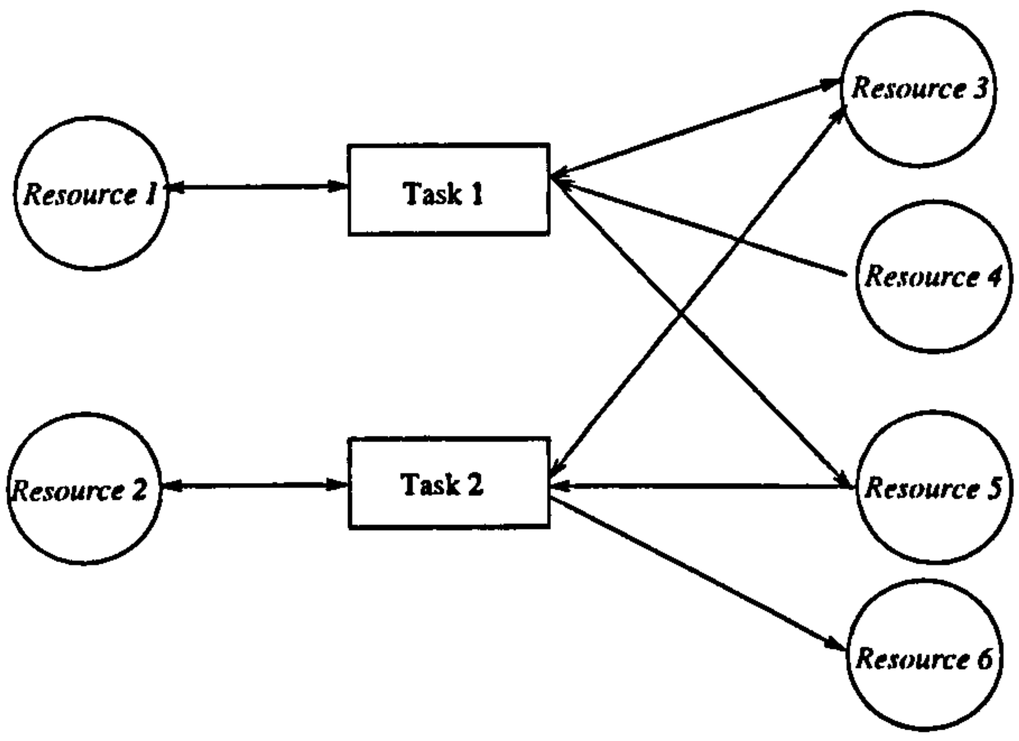


Figure 6.1: Resource Task Network Representation

6.1.2 Resources and Resource Types

All resources are allowed to be generated as well as consumed by tasks at any time during their execution, either as main outputs or as by-products of the operation. The types of resources and their consumption/generation manner can be characterised as:

- Discrete resources – Resources come in discrete units, such as manpower, items of storage facilities, transportation facilities, auxiliary equipment, processing equipment, *etc.* The common feature of these resources is that they are usually temporarily engaged or released, *e.g.* only required during the operation of the task and can be replenished after use. Using the *RTN* concept, this is simply considered as if they

were "consumed" by process tasks at the time of engagement and "generated" at the time of release.

- Continuous resources – Resources come in a continuous stream such as amounts of raw materials, intermediates, products, steam, electricity, cooling water, *etc.* This type of resource is usually irrevocably consumed or generated. Note that although utilities like steam can be replenished, a certain amount is still consumed permanently.

In addition, some resources may be used or generated continuously over certain periods, as in the case of feeds and products, while others involve allocation or generation of finite amounts at certain times, as in the case of equipment or manpower. Also a given resource may be consumed or generated *both* continuously and intermittently.

For a given process, the available resources and the way in which they are used for carrying out processing tasks can be easily identified. The key question is how to choose resources for a distinct *resource type.* Generally, subdivision of a resource of given nominal type (*e.g.* a given intermediate or item of processing equipment) may be necessary to allow for differences in availability or batch identity. The set of attributes which do or do not characterise a resource type is mainly context dependent.

One illustration can be made through the division of items of processing equipment. Generally, items of processing equipment can be classified into different resource types according to their capacity, suitability, condition (*e.g.* "fresh", "used", "cold", "hot"), or connectivity. For instance, if sequence-dependent cleaning or maintenance is required between production tasks, this is dealt with by treating the "fresh" equipment as a different resource type from "used" equipment. The production task is deemed to consume the fresh unit from one resource type and generate a used one in another resource type. A cleaning or maintenance task then carries out the inverse operation. The condition of equipment items can even be divided into different levels of cleanliness with each level as a distinct resource type. The cleaning task will then convert the unit from one level to another. This will further facilitate the requirement of frequency-dependent cleaning or maintenance.

Similarly, material in the same state can be considered as different resource types according to their location (*e.g.* for multisite plant scheduling), age (*e.g.* in food processing), or batch identity (*e.g.* in pharmaceuticals production, where batch identity is dictated by

regulatory considerations).

6.1.3 Tasks and Operations

In order to cover more general cases such as variable operating conditions, we present here a more detailed and general description of the conversion of resources through task operations. This provides the basis for the development of unified continuous time formulations in the *RTN* framework.

6.1.3.1 Tasks and Resource Utilization

Each task is characterised by:

1. a set of resources $\Re_i$ consumed or generated;

2. a set of *operating variables* (ξ_i), which must satisfy a set of operating constraints:

$$\Phi_i^{min} \leq \Phi_i(\xi_i) \leq \Phi_i^{max} \qquad \forall i \tag{6.1}$$

The operating variables (ξ_i) can have integer values denoting the use of "discrete resources" (*e.g.* the number of identical units of type S used to execute the task i, ξ_{si}), or real values denoting the use of continuous resources (*e.g.* the batch size) or the operating conditions for performing the task. For a given task, the operating variables determine:

(2.1) a sequence of times $\tau_{ik}(\xi_i)$, $k \in \{K_i : 0, 1, \ldots, k_i^{max}\}$, relative to the task starting time, $\tau_{i0}(\xi_i) \equiv 0$, at which resource transfer takes place[1];

(2.2) instantaneous resource transfers at times τ_{ik} with amounts $R_{rik}(\xi_i)$, $r \in \Re_i$;

(2.3) there may also be continuous transfer of resource r over the interval $[\tau_{ik}(\xi_i),\ \tau_{i,k+1}(\xi_i))$, $k < k_i^{max}$, at the rate $\overline{R}_{rik}(\xi_i)$.

Here, $R_{rik}(\xi_i)$, $\overline{R}_{rik}(\xi_i)$ are negative for consumptions. We can determine *a priori* limits R_{rik}^{min}, R_{rik}^{max}, $\overline{R}_{rik}^{min}$, $\overline{R}_{rik}^{max}$, τ_{ik}^{min}, τ_{ik}^{max} by minimization or maximization over ξ_i.

Resource transfers along the time sequence $\tau_{ik}(\xi_i)$ during task processing are shown in Figure 6.2 for the example given below.

[1] the term "resource transfer" can be interpreted either as "consuming" or "generating" as appropriate.

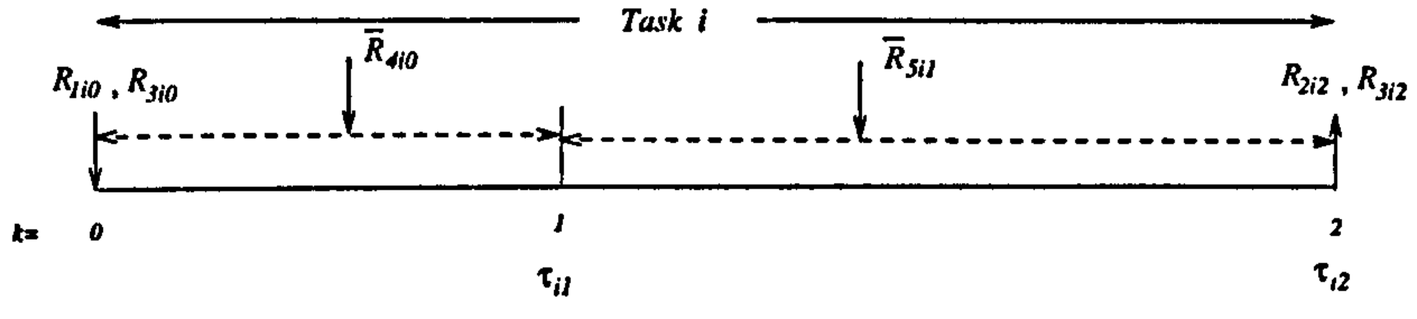


Figure 6.2: Characterization of Tasks in *RTN*

Example: As an illustrative example, we consider the exothermic batch reaction task i converting material A to material B. The reaction can be carried out in a $10te$ glass-lined reactor vessel. It uses steam for the initial period of processing. After this, the reaction will reach the normal operating temperature and is kept there for the rest of the processing by external cooling using cooling water.

From the description of task characterization, it is clear that this reaction process involves the transfer of five distinct resource types, namely, materials A $(r = 1)$ and B $(r = 2)$, glass-lined reactor vessel $(r = 3)$, steam $(r = 4)$ and cooling water $(r = 5)$. The resource transfers constitute task subsequences listed below.

(i) at $\tau_{i0}(\xi_i) \equiv 0$: the reaction task "consumes" an amount $R_{1i0}(\xi_i)$ of material A, and a number $R_{3i0}(\xi_i)$ of reactor vessels;

(ii) over $[\tau_{i0}(\xi_i),\ \tau_{i1}(\xi_i))$: the reaction task consumes the steam utility continuously with the rate of $\overline{R}_{4i0}(\xi_i)$;

(iii) over $[\tau_{i1}(\xi_i),\ \tau_{i2}(\xi_i))$: the reaction task consumes cooling water continuously with the rate of $\overline{R}_{5i1}(\xi_i)$;

(iv) Finally, at $\tau_{i2}(\xi_i)$: the reaction task "generates" an amount $R_{2i2}(\xi_i)$ of material B, and a number $R_{3i2}(\xi_i) = -R_{3i0}(\xi_i)$ of reactor vessels.

6.1.3.2 Task Operations

Within the *STN* framework, a specific operation is completely identified by the task i, the unit j in which it is carried out and its start-time T_l or operation l. However, with *RTN*, units are simply regarded as a resource, so a given "operation" may use several units, operating in-phase all with identical transfer-times for the different resources involved. On the other hand, several tasks of the same kind (i) may start simultaneously,

but employ different resources or use different operating conditions, so that the resource transfers are not necessarily in-phase. We refer to these as different *Task Operations*, $j \in \{J_i : 1, 2, \ldots, j_i^{max}\}$.

Since transfer-times $\tau_{ik}(\xi_i)$ can vary continuously with operating conditions ξ_i, there is a continuous spectrum of different possible operations, but in general there will be one or more critical discrete resources (such as suitable equipment items) which limit the number of possible different simultaneous operations, so the bounds j_i^{max} can be established *a priori*.

Example: In the reaction example described above, suppose that there is another glass-lined vessel with capacity of $20te$, the single reaction task can then have two different simultaneous operations, *i.e.* $j_i^{max} = 2$, with the set of resources employed $\Re_{ij} \subset \Re_i$, $j = 1, 2$ as:

- $\Re_{i1} \equiv$ {material A, material B, $10te$ glass-lined reactor, steam, cooling water};
- $\Re_{i2} \equiv$ {material A, material B, $20te$ glass-lined reactor, steam, cooling water};

The introduction of the *Task Operation* concept implies that any items of equipment in the same resource type operating in parallel within a given operation must operate in-phase (*i.e.* with the same starting time and duration) and have identical type, size and loading. This of course does not preclude the possibility of simultaneous operation of the same task with different loadings. It is simply that each item of equipment must then be treated as a different resource type, carrying out different task operations.

In conclusion, the *Resource Task Network* concept provides a more general description of tasks than that in the *STN* framework. There is nothing in the definition that requires a task to be carried out in a single item of equipment. Instead, simultaneous use of several items may be required. Furthermore, several alternative items of different types or sizes may be suitable.

A careful choice of tasks and resource types enables a very wide range of systems and situations to be covered in the *RTN* framework. It also permits the development of general yet simple mathematical formulations describing a range of process scheduling problems. As an example, Pantelides (1993) presents a simple formulation based on a uniform discrete time representation considering fixed product recipes. In the rest of this

Chapter, we will present formulations using the task models described here within the *RTN* framework. Again, with different time representations, different formulations can be obtained.

6.2 Resource Transfer Time Sequence

6.2.1 Time Representation

6.2.1.1 Separate Resource Transfer Time Sequences

By focussing on tasks converting resources rather than on units executing task operations, the only interactions between the activities are transfers of resources. If a sequence of times, $T_{rl}, l \in \{L_r : 0, 1, \ldots, l_r^{max}\}$ is defined at which transfer of resource r takes place, individual events can then be ordered by mapping the corresponding time to this sequence. These time sequences will be referred to as *Resource Transfer Time Sequences*. It should also be noted that the unit *Operation Sequence* of *Chapter 4* becomes a special type of *Resource Transfer Time Sequence* in the new scheme.

To distinguish transfers of resources in time, T_{rl} is postulated in an ascending order:

$$0 = T_{r0} < T_{r1} < \ldots < T_{rl_r^{max}} = T^{max}$$

These times should include all of the instantaneous resource transfer times and interval limits during the task operation, and the times or interval-limits at/over which instantaneous or continuous deliveries are required. Therefore, the maximum number of transfers l_r^{max} must be sufficiently large to include all possibilities. On the other hand, there may be times at which more than one transfer takes place simultaneously; hence in general there will be some redundant T_{rl} at which no transfers take place.

6.2.1.2 Task Start-time Sequences

Each task may be executed several times over the time horizon, [0, T^{max}]. Successive executions starting at different times are denoted by $q \in \{Q_i : 1, 2, \ldots, q_i^{max}\}$. Therefore, each execution is characterised by its starting time t_{iq} and the values of its operating variables (ξ_{ijq}) depending on the type of operation (j) executed. The execution of a

task i must start at one of the t_{iq}, but several *Task Operations* j with different operating conditions ξ_{ijq} may start at the same time. We label the starting time of all operations in chronological order:

$$0 \leq t_{i0} < t_{i1} < \ldots < t_{iq_i^{max}} \leq T^{max}$$

The maximum possible number of different starting times q_i^{max} for each task during the time horizon [0, T^{max}] can be estimated *a priori*. Suppose each task operation starting at different times, we have q_i^{max} as:

$$q_i^{max} = \sum_{j=1}^{j=j_i^{max}} \left\lfloor \frac{T^{max}}{\tau_{ijk^{max}}^{min}} \right\rfloor \tag{6.2}$$

6.2.1.3 Plant Delivery Sequences

Some resources may be on site at the start of operation, and it may also be possible to obtain deliveries of further resources during operations. On the other hand, it may be possible to deliver some resources as they are produced or there may be required deliveries at specific times or over specific periods. These transfers taking place between the plant and the environment are considered as plant deliveries. In general, resources can either be delivered intermittently with specified or limited amount at specified times/over specified intervals, or continuously at specified or limited rates over specified periods.

We define the delivery times or interval limits as:

$$0 = t_{r0} < t_{r1} < \ldots < t_{rk_r^{max}} = T^{max}$$

Note that within the *RTN* framework, the unavailability of processing equipments can also be simply treated as the "delivery" of the unit from the plant at the beginning of each period of unavailability, and a "delivery" to the plant at the end.

6.2.2 Variables

The formulation uses the following key binary and continuous variables:

T_{rl} : transfer time sequence for resource r, $l \in L_r$

t_{iq} : task operation start times

τ_{rijkq} : relative transfer time of resource $r \in \Re_i$ during operation j of task i starting at t_{iq}

t_{rijkq} : actual transfer time of resource $r \in \Re_i$ during operation j of task i starting at t_{iq}

ξ_{ijq} : operating variables for operation j of task i starting at t_{iq}

R_{rijkq} : amount of resource $r \in \Re_i$ transferred during operation j of task i at t_{rijkq}

$\overline{R}_{rijkq}$: transfer rate of resource $r \in \Re_i$ during operation j of task i over time period $[t_{rijkq},\ t_{rij,k+1,q})$

$$W_{ijq} = \begin{cases} 1 & \text{if operation } j \text{ , of task } i \text{ starts at } t_{iq} \\ 0 & \text{otherwise} \end{cases}$$

$$X_{rijkql} = \begin{cases} 1 & \text{if } T_{rl} = t_{iq} + \tau_{rijkq} \\ 0 & \text{otherwise} \end{cases}$$

$$\tilde{X}_{rijkql} = \begin{cases} 1 & \text{if } T_{rl} \in [t_{rijkq},\ t_{rij,k+1,q}) \\ 0 & \text{otherwise} \end{cases}$$

$$Y_{rkl} = \begin{cases} 1 & \text{if } T_{rl} = t_{rk} \\ 0 & \text{otherwise} \end{cases}$$

$$\tilde{Y}_{rkl} = \begin{cases} 1 & \text{if } T_{rl} \in [t_{rk},\ t_{r,k+1}) \\ 0 & \text{otherwise} \end{cases}$$

6.2.3 Task Operation Models

6.2.3.1 General Task Operation Models

During a specific operation, the amounts and times of the resource transfers can be determined by operating variables. If the task-operation does not take place (*i.e.* $W_{ijq} = 0$), there will of course be no resource transfers. Therefore, we have:

$$R_{rijkq} = W_{ijq} R_{rik}(\xi_{ijq}) \qquad \forall i, r \in \Re_{ij}, k \in K_i, j \in J_i, q \in Q_i \tag{6.3}$$

$$\overline{R}_{rijkq} = W_{ijq} \overline{R}_{rik}(\xi_{ijq}) \qquad \forall i, r \in \Re_{ij}, k \in K_i, j \in J_i, q \in Q_i \tag{6.4}$$

$$\tau_{rijkq} = W_{ijq} \tau_{ik}(\xi_{ijq}) \qquad \forall i, r \in \Re_{ij}, k \in K_i, j \in J_i, q \in Q_i \tag{6.5}$$

Note that since sequence K_i is common to all resources, (6.5) need only be introduced for those elements of subsequence $k = k_r \in K_i$ at which resource r is transferred. The corresponding actual resource transfer times are then obtained by:

$$t_{rijkq} = t_{iq} + \tau_{rijkq} \qquad \forall i, r \in \Re_{ij}, k \in K_i, j \in J_i, q \in Q_i \tag{6.6}$$

Again, each transfer time during task operation may be subject to a minimum and a maximum limit:

$$W_{ijl}\tau_{ijk}^{min} \leq \tau_{rijkq} \leq \tau_{ijk}^{max} W_{ijl} \tag{6.7}$$

Of course in some cases, particularly for continuous tasks, the transfer times τ_{rijkq} can be free decision variables, but this is merely a special case of $\tau_{ik}(\xi_{ijq})$, in which τ_{ik} is identified with one of the operating variables ξ_{ijq}.

6.2.3.2 Blending Tasks

Up to now, we have only been able to consider the situations where all the required resources are loaded as the required mixture with a single transfer. We have also restricted our scope to the simple situation of coping with only the holding of feed mixture before processing taking place in the same unit. However, as mentioned briefly in section *5.4.3*, it will be more general to allow some or all of the ingredients to be transferred as available to constitute the required mixture. This is a more complicated case which can not be simply solved as a storage task as given in *Chapter 5*. To deal with this, we define a special task – the *blending task*.

A *blending task i* may also require various types of resources (*e.g.* manpower, conveyors or heating/cooling to maintain the prescribed temperatures). With the concept of *RTN*, this can be readily taken into account as for other processing tasks. However, special features of the blending task makes it different from other processing tasks in the following respects:

(i) ingredients $r \in \overline{\Re}_i$ can only be added;

(ii) the final charge should meet the requirements for the initial charge of the subsequent processing task;

(iii) the final mixture can only be withdrawn after completion of the task.

Suppose the required mixture $\bar{r} \in \Re_i$ has given composition ξ_i^r, $r \in \overline{\Re}_i \subset \Re_i$. The condition on feed ingredients to meet this composition is then:

$$\xi_i^r R_{\bar{r}ijk_i^{max}q} + \sum_{k \in K_i} \{R_{rijkq} + \overline{R}_{rijkq}(\tau_{rij,k+1,q} - \tau_{rijkq})\} = 0 \qquad \forall q, j \in J_i, r \in \overline{\Re}_i \quad (6.8)$$

To meet the requirements on additions and the final withdrawal, the following constraints should be satisfied:

$$\left.\begin{array}{llll} R_{rijkq} \leq 0, & \overline{R}_{rijkq} \leq 0, & \forall k, q, & j \in J_i, r \in \overline{\Re}_i \\ R_{\bar{r}ijkq} = & \overline{R}_{\bar{r}ijkq} = 0, & k < k_i^{max}, q, & j \in J_i \end{array}\right\} \quad (6.9)$$

We note that processing need not immediately follow the blending, nor need the two operations be carried out in the same item of equipment, though additional conditions on this could be imposed if desired. For example, setting $k_i^{max} = 0$ imposes instantaneous blending, but otherwise τ_{rijkq} are free variables, satisfying:

$$0 < \tau_{rij1q} < \tau_{rij2q} < \ldots < \tau_{rijkq} < \ldots < \tau_{rijk_i^{max}q} \quad (6.10)$$

here k_i^{max} is the pre-specified maximum number of transfers.

6.2.3.3 Storage Tasks

The demand for any resource must not exceed the amount available. Any unused resource must either be removed from the plant or stored in the plant. To meet these storage requirements, we define a *storage task* i for storing each resource r over each interval $[T_{rl},\ T_{r,l+1})$. This implies that we can use the resource transfer time sequence $\{T_{rl}\}$ as the start time sequence $\{t_{iq}\}$ for the storage task.

Each storage task has only one interval of resource transfer, with $t_{rij0l} = T_l$ and $t_{rij1l} = T_{l+1}^-$. Resource transfer can be either instantaneous at the starting and ending time with amount of $(-R_{rij0l})$ and R_{rij1l} respectively, or continuous over the whole interval at a rate of $\overline{R}_{rij0l}$. We also assume that all units used for storage are emptied at the end, t_{rij1l}. The quantity to be stored is then:

$$\text{At } T_{rl}: \qquad -\sum_{j \in J_i} R_{rij0l} \geq 0 \qquad \forall r, l \in L_r \quad (6.11a)$$

$$\text{At } T_{r,l+1}^-: \qquad \sum_{j \in J_i} R_{rij1l} = -\sum_{j \in J_i} \{R_{rij0l} + \overline{R}_{rij0l}(T_{r,l+1} - T_{rl})\} \geq 0 \qquad \forall r, l \in L_r \quad (6.11b)$$

In order to guarantee the excess resource in storage be always kept within the capacity limits, it has to be ensured that neither the amount of storage after the instantaneous transfer, nor the amount accumulated by continuous transfer over the interval can exceed the available capacity limits. Following the fact that the requirement for storage varies linearly over the interval (*i.e.* continuous transfers are made through constant transfer rate within each interval), we need only check the amounts R_{rij0l} and R_{rij1l} at the boundary of each interval.

Once a unit is engaged (consumed) in the storage of a type of resource at the beginning of an interval, it then will be occupied over the whole interval and released (generated) at the end of the storage task. The storage equipment can then be described in terms of:

$-R_{sij0l}$ = number of units of type s allocated to the storage of resource r in operation j over interval $[T_{rl},\ T_{r,l+1})$, with capacity limits of $\xi_{si}^{min},\ \xi_{si}^{max}$

The storage capacity constraints can then be expressed as a special case of the operating constraints (6.1):

$$\sum_{s\in S} \xi_{si}^{min} \cdot (-R_{sij0l}) \leq -R_{rij0l} \leq \sum_{s\in S} \xi_{si}^{max} \cdot (-R_{sij0l}) \qquad \forall r, j \in J_i, l \in L_r \quad (6.12a)$$

$$\sum_{s\in S} \xi_{si}^{min} \cdot (-R_{sij0l}) \leq R_{rij1l} \leq \sum_{s\in S} \xi_{si}^{max} \cdot (-R_{sij0l}) \qquad \forall r, j \in J_i, l \in L_r \quad (6.12b)$$

Storage units may be specialised units, or units also used for processing. Condition (6.11a, b) does not prevent use of a unit over several intervals, which merely requires allocation of the unit for each of these intervals. Thus for example a unit could be allocated first to a blending task to prepare an appropriate feed, then to a processing task to produce one or more products, and finally to a sequence of storage tasks for the final product until this is required.

Except for the requirement of storage tanks or suitable equipment, these storage tasks may also need other resources such as compression requirements or heating/cooling to maintain the appropriate storage conditions. All resource requirements for storage and transfers of resource r between units are associated with the storage task for the relevant interval. Of course, some resources, such as idle equipment or manpower, can be "stored" (left unused) without invoking the use of other resources. Storage tasks must still be defined for these resources, in order to "store" the excess.

6.2.4 Plant Delivery

Summarising the deliveries considered in *Chapter 4* and *5*, they can be of three types:

(i) Delivery of a specified amount, or an amount within specified limits, $[\hat{D}_{rk}^{min},\ \hat{D}_{rk}^{max}]$ at a specified time t_{rk};

(ii) Deliveries of a specified amount, or an amount within specified limits $[\tilde{D}_{rk}^{min},\ \tilde{D}_{rk}^{max}]$ over a specified interval $[t_{rk},\ t_{r,k+1})$;

(iii) Continuous delivery at a specified rate, or within specified limits $[\overline{D}_{rk}^{min},\ \overline{D}_{rk}^{max}]$ over a specified interval, $[t_{rk},\ t_{r,k+1})$.

The features of individual delivery types have already been described in *Chapter 4* and *5*, and the use of the *RTN* representation does not affect them. However, the method to meet these requirements can be more general in that:

- deliveries are not limited to the delivery of materials;
- incoming and outgoing deliveries can be treated as separate resources, with different unit costs, if desired.

Again, we can have instantaneous deliveries of amount D_{rl} at time T_{rl}, or continuous deliveries at the rate $\overline{D}_{rl}$ over the period $[T_{rl},\ T_{r,l+1})$.

Then:

$$\sum_{k\in K_r^D} \hat{D}_{rk}^{min} Y_{rkl} \le \hat{D}_{rl} \le \sum_{k\in K_r^D} \hat{D}_{rk}^{max} Y_{rkl} \qquad \forall r, l \in L_r \qquad (6.13a)$$

$$\sum_{k\in K_r^D} D_{rk}^{min} \tilde{Y}_{rkl} \le D_{rl} \le \sum_{k\in K_r^D} D_{rk}^{max} \tilde{Y}_{rkl} \qquad \forall r, l \in L_r \qquad (6.13b)$$

$$\sum_{k\in K_r^D} \overline{D}_{rk}^{min} \tilde{Y}_{rkl} \le \overline{D}_{rl} \le \sum_{k\in K_r^D} \overline{D}_{rk}^{max} \tilde{Y}_{rkl} \qquad \forall r, l \in L_r \qquad (6.13c)$$

$$\tilde{D}_{rk}^{min} \le \tilde{D}_{rk} \le \tilde{D}_{rk}^{max} \qquad \forall r, k \in K_r^D \qquad (6.13d)$$

Note that the total delivery D_{rl} at event-time T_{rl} includes the quantity $\hat{D}_{rl}$ required to satisfy case (i), as given by (6.13a), and there may be an additional contribution $\tilde{D}_{rl}$ towards satisfying case (ii), so in general D_{rl} is given as (6.14c). There may also be separate limits on the total D_{rl} at any given event-time within this interval as given by (6.13b).

In general, the limits in (6.13a, b, c, d) are on the net demands for deliveries to or from the plant, remembering that deliveries to the plant are negative quantities, so care is needed in setting these limits.

The total delivery over a specified interval to satisfy case (ii) is given in (6.14a), while the total delivery satisfying case (i) – (iii) over this interval is given in (6.14d).

$$\tilde{D}_{rk} = \sum_{l \in L_r} \tilde{Y}_{rkl}\{\tilde{D}_{rl} + \overline{D}_{rl}(T_{r,l+1} - T_{rl})\} \qquad \forall r, k \in K_r^D \qquad (6.14a)$$

$$\hat{D}_{rk} = \sum_{l \in L_r} \hat{D}_{rl} Y_{rkl} \qquad \forall r, k \in K_r^D \qquad (6.14b)$$

$$D_{rl} = \hat{D}_{rl} + \tilde{D}_{rl} \qquad \forall r, l \in L_r \qquad (6.14c)$$

$$D_{rk} = \hat{D}_{rk} + \tilde{D}_{rk} \qquad \forall r, k \in K_r^D \qquad (6.14d)$$

6.2.5 Resource Balance

The amount of resource transferred at each time T_{rl} must balance, and so must the rate of transfer over each interval $[T_{rl},\ T_{r,l+1})$. Therefore we have the uniform resource balance constraints:

$$\sum_{j \in J_i} R_{rij1,l-1} + \sum_{j \in J_i} R_{rij0l} - D_{rl} + \sum_{i' \in I_r} \sum_{j' \in J_{i'}} \sum_{k' \in K_{i'}} \sum_{q \in Q_{i'}} R_{ri'j'k'q} X_{ri'j'k'ql} = 0 \quad \forall r, l \in L_r \qquad (6.15)$$

$$\sum_{j \in J_i} \overline{R}_{rij0l} - \overline{D}_{rl} + \sum_{i' \in I_r} \sum_{j' \in J_{i'}} \sum_{k' \in K_{i'}} \sum_{q \in Q_{i'}} \overline{R}_{ri'j'k'q} \tilde{X}_{ri'j'k'ql} = 0 \quad \forall r, l \in L_r \qquad (6.16)$$

The important feature of the resource balance constraints presented here is that they encompass all of the constraints expressing the unit allocation, material balance and resource utilization in the *STN*-based formulation presented in *Chapter 4* and *5*. Also, as illustrated in section *6.1*, with careful choice of resource types, it can also easily deal with plant connectivity and general operational constraints.

6.2.6 Time Assignment

To relate the individual time sequences for tasks and resource transfers, we impose the following constraints:

$$\sum_{l' \in L_r} X_{rijkql'} = W_{ijq} \qquad \forall i, r \in \Re_{ij}, k \in K_i, j \in J_i, q \in Q_i \qquad (6.17)$$

$$\tau_{rijkq} = \sum_{l' \in L_r} X_{rijkql'}(T_{rl'} - t_{iq}) \quad \forall i, r \in \Re_{ij}, k \in K_i, j \in J_i, q \in Q_i \tag{6.18}$$

The delivery sequence must also be included in the sequence $\{T_{rl}\}$. To relate the plant delivery times t_{rk}, to the resource transfer times $T_{rl'}$, we have:

$$\sum_{l' \in L_r} Y_{rkl'} = 1 \qquad \forall r, k \in K_r^D \tag{6.19a}$$

$$\sum_{k \in K_r^D} Y_{rkl'} \leq 1 \qquad \forall r, l' \in L_r \tag{6.19b}$$

$$\sum_{k \in K_r^D} \tilde{Y}_{rkl'} = 1 \qquad \forall r, l' \in L_r \tag{6.19c}$$

and, taking advantage of using the fixed times t_{rk}, tighter linear constraints can be generated:

$$\sum_{k \in K_r^D} t_{rk} Y_{rkl'} \leq T_{rl'} \leq T^{max} - \sum_{k \in K_r^D} (T^{max} - t_{rk}) Y_{rkl'} \qquad \forall r, l' \in L_r \tag{6.20}$$

The definitions of $\tilde{X}_{rijkql'}$ and $\tilde{Y}_{rkl'}$ are governed by:

$$\tilde{X}_{rijkql'} = \sum_{l'' \leq l'} X_{rijkql''} + \sum_{l'' > l'} X_{rij,k+1,ql''} - W_{ijq} \qquad \forall i, r \in \Re_{ij}, k \in K_i, j \in J_i, q \in Q_i \tag{6.21}$$

$$\tilde{Y}_{rkl'} = \sum_{l'' \leq l'} Y_{rkl''} + \sum_{l'' > l'} Y_{r,k+1,l''} - 1 \qquad \forall r, k \in K_r^D, l' \in L_r \tag{6.22}$$

We can also get a direct relation between $\tilde{Y}_{rkl'}$ and event-time $T_{l'}$, which is again a linear constraint due to the specified t_{rk}:

$$\sum_{k \in K_r^D} t_{rk} \tilde{Y}_{rkl'} \leq T_{rl'} < \sum_{k \in K_r^D} t_{r,k+1} \tilde{Y}_{rkl'} \qquad \forall r, l' \in L_r \tag{6.23}$$

However, the value of $Y_{rkl'}$ can not be determined through $\tilde{Y}_{rkl'}$. Therefore, (6.20) is still necessary for this purpose.

6.2.7 Objective Function

The objective function is the maximization of the net profit of the whole operation which can be expressed simply as:

$$\texttt{Profit}(C) = \sum_{r} \sum_{k \in K_r^D} c_{rk} D_{rk} \tag{6.24}$$

since all costs are associated with delivery of resources to or from the plant. c_{rk} is the unit value of resource r over interval $[t_{rk},\ t_{r,k+1})$.

The exact linearization of the bilinear terms in the model can be achieved using the special-ordered set constraint properties in a similar way to those presented previously. However, if continuous resource transfer or plant delivery happens, nonlinearities will also exist in the form of real variable products in constraints (6.8), (6.11b) and (6.14a). These can again be linearized exactly as described in section *5.5.1.*

6.3 Unified Formulation for Process Scheduling

6.3.1 Time Representation

In the previous section, we defined three kinds of time sequence to describe the time varying activities within the time horizon:

(1) task start-time sequence t_{iq}:

$$0 \leq t_{i0} < t_{i1} < \ldots < t_{iq_i^{max}} \leq T^{max}$$

(2) resource transfer time sequence during the operation of a task t_{rijkq}:

$$t_{rijkq} = t_{iq} + \tau_{rijk}$$

(3) common transfer times T_{rl} for each resource, to map all transfer times t_{rijkq}:

$$0 = T_{r0} < T_{r1} < \ldots < T_{rl_r^{max}} = T^{max}$$

In this section, an alternative formulation is examined in which we refer all events to a single common event-time sequence:

$$0 = T_0 < T_1 < T_2 < \ldots < T_l < \ldots < T_{l^{max}} \leq T^{max}$$

This event-time sequence will synchronise all of the activities within the plant and activities between plant and environment. Therefore:

- The start of operations of a task must always be one of the T_l;
- All succeeding resource transfers during the operation of a task must occur at one of the $T_{l'}$ $(l' > l)$;

- All plant delivery times t_{rk} must also correspond to one of the T_l.

Based on this single event-time sequence, we can modify the definition of variables used in the formulation:

T_l	:	common event-time sequence for all resource transfers
τ_{ijkl}	:	relative transfer time of resources during operation j of task i starting at T_l
t_{ijkl}	:	actual transfer time of resources during operation j of task i starting at T_l
ξ_{ijl}	:	operating variables for task i in operation j starting at T_l
R_{rijkl}	:	amount of resource $r \in \Re_i$ transferred during operation j for task i at time t_{ijkl}
$\overline{R}_{rijkl}$	:	transfer rate of resource $r \in \Re_i$ during operation j for task i over time period $[t_{ijkl},\ t_{ij,k+1,l})$

$$W_{ijl} = \begin{cases} 1 & \text{if operation } j \text{ of task } i \text{ starts at } T_l \\ 0 & \text{otherwise} \end{cases}$$

$$X_{ijkll'} = \begin{cases} 1 & \text{if } T_{l'} = T_l + \tau_{ijkl} \\ 0 & \text{otherwise} \end{cases}$$

$$\tilde{X}_{ijkll'} = \begin{cases} 1 & \text{if } T_{l'} \in [t_{ijkl},\ t_{ij,k+1,l}) \\ 0 & \text{otherwise} \end{cases}$$

$$Y_{rkl'} = \begin{cases} 1 & \text{if } T_{l'} = t_{rk} \\ 0 & \text{otherwise} \end{cases}$$

$$\tilde{Y}_{rkl'} = \begin{cases} 1 & \text{if } T_{l'} \in [t_{rk},\ t_{r,k+1}) \\ 0 & \text{otherwise} \end{cases}$$

6.3.2 Mathematical Model

Again, a task is characterized by a set of resources $\Re_i$ and operating variables ξ_i. The operating constraints for operating variables remains the same as in the previous section:

$$\Phi_i^{min} \leq \Phi_i(\xi_i) \leq \Phi_i^{max} \tag{6.25}$$

Several different *Task Operations* of a given task i, corresponding to different ξ_{ijl} and employ different resources, may start at the same time T_l.

For the resource transfers during a specific task-operation, we can use a single common sequence of transfer times t_{ijkl} instead of the previous t_{rijkq}. Then we have the following model of task operation at each event-time T_l:

$$R_{rijkl} = W_{ijl} R_{rik}(\xi_{ijl}) \qquad \forall i, r \in \Re_{ij}, k \in K_i, j \in J_i, l \tag{6.26a}$$

$$\overline{R}_{rijkl} = W_{ijl} \overline{R}_{rik}(\xi_{ijl}) \qquad \forall i, r \in \Re_{ij}, k \in K_i, j \in J_i, l \tag{6.26b}$$

$$\tau_{ijkl} = W_{ijl} \tau_{ik}(\xi_{ijl}) \qquad \forall i, k \in K_i, j \in J_i, l \tag{6.27}$$

$$t_{ijkl} = T_l + \tau_{ijkl} \qquad \forall i, k \in K_i, j \in J_i, l \tag{6.28}$$

and it follows that:

$$W_{ijl} = \sum_{l' \geq l} X_{ijkll'} \qquad \forall i, k \in K_i, j \in J_i, l \tag{6.29}$$

$$\tau_{ijkl} = \sum_{l' \geq l} X_{ijkll'} (T_{l'} - T_l) \qquad \forall i, k \in K_i, j \in J_i, l \tag{6.30}$$

$$\tilde{X}_{ijkll'} = \sum_{l'' \leq l'} X_{ijkll''} + \sum_{l'' > l'} X_{ij,k+1,ll''} - W_{ijl} \qquad \forall i, k \in K_i, j \in J_i, l \leq l' \tag{6.31}$$

Note that we need only to define the X- variables for $l' > l$, since successive resource transfer (*i.e.* with $k > 0$) must be later than the operation start time, and $X_{ij0ll'} = W_{ijl}$.

The conditions for the special blending and storage tasks now take the following form:

Blending Task

Suppose the required mixture $\bar{r} \in \Re_i$ has given composition ξ_i^r, $r \in \overline{\Re}_i \subset \Re_i$. Then the condition of feed ingredients to meet this composition is:

$$\xi_i^r R_{\bar{r}ijk_i^{max}l} + \sum_k \{R_{rijkl} + \overline{R}_{rijkl}(\tau_{ij,k+1,l} - \tau_{ijkl})\} = 0 \qquad \forall l, j \in J_i, r \in \overline{\Re}_i \tag{6.32}$$

Again, ingredients $r \in \overline{\Re}_i$ can only be added, and the final mixture can only be withdrawn after completion of the task:

$$\left.\begin{array}{llllll} R_{rijkl} \leq & 0, & \overline{R}_{rijkl} \leq & 0, & \forall k, l, & j \in J_i, r \in \overline{\Re}_i \\ R_{\bar{r}ijkl} = & \overline{R}_{\bar{r}ijkl} = & 0, & & k < k_i^{max}, l, & j \in J_i \end{array}\right\} \tag{6.33}$$

Storage Task

Based on the common event-time sequence, we define a *storage task* (i) over each interval $[T_l, T_{l+1})$ for each resource (r). Hence by definition $k_i^{max} = 1$, $t_{ij0l} = T_l$, $t_{ij1l} = T_{l+1}^-$. The resource transfer for storage task operations and its capacity constraints will then take a similar form as (6.11) and (6.12):

$$-\sum_{j\in J_i} R_{rij0l} = \sum_{j\in J_i} \{\overline{R}_{rij0l}(T_{l+1} - T_l) + R_{rij1l}\} \geq 0 \qquad \forall r, l \tag{6.34}$$

$$\sum_{s\in \mathbf{S}} \xi_{si}^{min}(-R_{sij0l}) \leq -R_{rij0l} \leq \sum_{s\in \mathbf{S}} \xi_{si}^{max}(-R_{sij0l}) \qquad \forall r, l, j \in J_i \tag{6.35a}$$

$$\sum_{s\in \mathbf{S}} \xi_{si}^{min}(-R_{sij0l}) \leq R_{rij1l} \leq \sum_{s\in \mathbf{S}} \xi_{si}^{max}(-R_{sij0l}) \qquad \forall r, l, j \in J_i \tag{6.35b}$$

Plant Delivery

The delivery constraints (6.14a, b) now take the form of:

$$\tilde{D}_{rk} = \sum_l \tilde{Y}_{rkl}\{\tilde{D}_{rl} + \overline{D}_{rl}(T_{l+1} - T_l)\} \qquad \forall r, k \in K_r^D \tag{6.36a}$$

$$\hat{D}_{rk} = \sum_l \hat{D}_{rl} Y_{rkl} \qquad \forall r, k \in K_r^D \tag{6.36b}$$

All variables denoting the deliveries under different circumstances should be subject to the same limits as given in (6.13a, b, c, d). Also, the total delivery D_{rl} at each event-time T_l and D_{rk} over each interval $[t_{rk},\ t_{r,k+1})$ are given in the same form as (6.14c, d).

The delivery times should then be identified with the common event-times, so the corresponding timing constraints are changed into:

$$\sum_{k\in K_r^D} t_{rk} Y_{rkl} \leq T_l \leq T^{max} - \sum_{k\in K_r^D} (T^{max} - t_{rk}) Y_{rkl} \qquad \forall r, l \tag{6.37}$$

and/or:
$$\sum_{k\in K_r^D} t_{rk} \tilde{Y}_{rkl} \leq T_l < \sum_{k\in K_r^D} t_{r,k+1} \tilde{Y}_{rkl} \qquad \forall r, l \tag{6.38}$$

where integer variables Y_{rkl}, $\tilde{Y}_{rkl}$ are subject to the special-ordered set constraints in (6.19).

Resource Balance

Finally, amounts of each resource r transferred at each T_l, and rates of transfer over each interval $[T_l,\ T_{l+1})$, must balance:

$$\sum_j R_{rij1,l-1} + \sum_j R_{rij0l} - D_{rl} + \sum_{i' \in I_r} \sum_{j' \in J_{i'}} \sum_{k' \in K_{i'}} \sum_{l' \leq l} R_{ri'j'k'l'} X_{i'j'k'l'l} = 0 \qquad \forall r, l \quad (6.39a)$$

$$\sum_j \overline{R}_{rij0l} - \overline{D}_{rl} + \sum_{i' \in I_r} \sum_{j' \in J_{i'}} \sum_{k' \in K_{i'}} \sum_{l' \leq l} \overline{R}_{ri'j'k'l'} \tilde{X}_{i'j'k'l'l} = 0 \qquad \forall r, l \quad (6.39b)$$

where i is the storage task for resource r, and $i' \in \mathbf{I_r}$ are tasks consuming or generating resource r, $\mathbf{I_r} = \{i | r \in \Re_i\}$.

As pointed out in section *6.2.5*, this resource balance constraint can easily cover the feature of general operational constraints with careful choice of resources. This is based on the division of an original resource into different types. Alternatively, the idea of introducing the unit usability to measure the unit conditions (*cf.* section *5.2.2*) can also be applied to this unified formulation. The application of it is demonstrated through a case study in section *6.5*.

Objective Function

The net profit of the whole operation, (C), which must be maximized subject to the above constraints, is given by:

$$C = \sum_{r,k} c_{rk} D_{rk} \quad (6.40)$$

6.3.3 Problem Simplifications

Both of the formulations in this section and in section *6.2* yield a large mixed integer nonlinear programming problem (MINLP), which is beyond the capabilities of existing packages for the solution of such problems. In this section, we describe techniques for reducing the size of the problem and removing many of the nonlinearities, which makes the problem more tractable.

The two formulations are very similar, differing only in the treatment of timing constraints. We would expect the use of separate resource transfer time sequences to be more

efficient when most resource transfers are distinct in time, but for most tasks resource transfers occur mainly at the beginning and end of the task and when many transfers occur simultaneously, the single unified time sequence will yield a smaller problem.

Because of the close similarity of the two representations, we consider below only the unified formulation.

We propose to solve the problem as an iterative sequence of mixed integer linear programmes (MILPs), based on local linearization of the functions involved, noting that non-linearities may occur in equations (6.25), (6.26a, b), (6.27), (6.30), (6.32), (6.34), (6.36a, b) and (6.39a, b). Unfortunately there is as yet no algorithm which finds a global optimum for a MINLP involving non-convex functions. Pending development of a satisfactory technique for the solution of this problem (see for example Viswanathan and Grossmann (1990), Floudas and Visweswaren (1993)), we assume that in the nonlinear subproblems, a local optimization with respect to real variables, for fixed values of the integer variables, will yield a satisfactory answer.

In the local MILP, (6.26a, b), (6.27) will be replaced by:

$$R_{rijkl} = W_{ijl}(\alpha_{rijkl} + \beta^T_{rijkl}\xi_{ijl}) \tag{6.41}$$

$$\overline{R}_{rijkl} = W_{ijl}(\overline{\alpha}_{rijkl} + \overline{\beta}^T_{rijkl}\xi_{ijl}) \tag{6.42}$$

$$\tau_{ijkl} = W_{ijl}(\alpha_{ijkl} + \beta^T_{ijkl}\xi_{ijl}) \tag{6.43}$$

where α_{ijkl}, β_{ijkl}, α_{rijkl}, β_{rijkl}, $\overline{\alpha}_{rijkl}$, $\overline{\beta}_{rijkl}$ are fixed constants.

As shown in *Chapter 4*, products like $W_{ijl}\xi_{ijl}$ in the above equations can then be linearized, at the cost of introducing an integrality gap. When the integer in an integer-real variable product belongs to a "special-ordered set", such as $\tilde{Y}_{rkl}\tilde{D}_{rl}$ in (6.36a), the constraint can be simplified and the integrality gap is smaller. For the products of real-real variables as in (6.32), (6.34), (6.36a), the linearization can be done analogously to section *5.5.1.*

The main problem arises in dealing with the timing constraints (6.30), and again we have investigated the two alternative approaches, using either a fixed or variable sequence of event-times:

6.3.3.1 Fixed Event Time Sequence (FET)

First we can use a sequence of fixed event-times.

If the processing times τ_{ik} are fixed, choosing the highest common factor (HCF) of the intervals $(\tau_{ik} - \tau_{i,k-1})$ as a uniform event-time spacing ΔT enables us to dispense with the $X_{ijkll'}$ (*see Appendix A*). The resulting MILP is of the same form as that treated by Shah *et al.* (1993a), so the problem can be solved efficiently.

However, if some intervals are small, or if there are small differences between intervals, the HCF can be very small, leading to a very large number of event-times and hence a large problem. In this situation, it is better to use a larger, arbitrary ΔT, and tolerate rounding.

For variable processing times, we must choose an arbitrary ΔT, and the $X_{ijkll'}$ are essential for identifying transfer-times with event-times. We must now round τ_{ijkl}, as given by (6.27) or (6.43), to an integral number of intervals, which is achieved by replacing (6.30) by:

$$\sum_{l' \geq l} X_{ijkll'} \cdot (T_{l'-1} - T_l) < \tau_{ijkl} \leq \sum_{l' \geq l} X_{ijkll'} \cdot (T_{l'} - T_l) \tag{6.30b}$$

Of course we must choose ΔT small enough for the rounding to be insignificant, and we must also have:

$$\Delta T \leq \min_{i,k} \min_{\xi_i} (\tau_{i,k+1}(\xi_i) - \tau_{ik}(\xi_i)) \tag{6.44}$$

This again leads to a large problem if ΔT is small.

6.3.3.2 Variable Event Time Sequence (VET)

As before, by allowing the T_l to be variable, we can hope to use a much smaller set of event-times, close to the actual number of distinct events. However, we now need Y_{rkl}, $\tilde{Y}_{rkl}$ and $X_{ijkll'}$, and hence have nonlinear terms in (6.30), (6.34) and (6.36a, b).

The integer-real variable products can be linearized as described earlier, but instead of using (6.30), we have a more direct way of effecting the definition of $X_{ijkll'}$:

First, to ensure that the event-times remain distinct, we set a fixed minimum spacing

δT, which is chosen according to the required time accuracy:

$$T_l - T_{l-1} \geq \delta T, \qquad l = 1, 2, \ldots, l^{max} \tag{6.45}$$

We can also derive an upper bound on the spacing between event-times:

$$\Delta \overline{T} = \max_{i,k} \{ \max_{\xi_i} (\tau_{i,k+1}(\xi_i) - \tau_{ik}(\xi_i)) \} \tag{6.46}$$

Then as shown in *Appendix B*, we have the bounds:

$$T_{l'} - t_{ijkl} \geq (1 - W_{ijl}) \cdot (l' - l)\delta T + \sum_{l'' \leq l'} X_{ijkll''} \cdot (l' - l'')\delta T$$
$$- \sum_{l'' > l'} X_{ijkll''} \cdot \min\{(l'' - l')\Delta \overline{T},\ \tau_{ik}^{max} - (l' - l)\delta T\} \tag{6.47a}$$

$$T_{l'} - t_{ijkl} \leq (1 - W_{ijl}) \cdot (l' - l) \cdot \Delta \overline{T} + \sum_{l'' \leq l'} X_{ijkll''} \cdot (l' - l'')\Delta \overline{T}$$
$$- \sum_{l'' > l'} X_{ijkll''} \cdot \max\{(l'' - l')\delta T,\ \tau_{ik}^{min} - (l' - l) \cdot \Delta \overline{T}\} \tag{6.47b}$$

6.3.4 Problem Solution

By choosing the event-times differently, we proposed two alternative procedures, *FET* and *VET*. However, in the case of variable processing times, *a priori* bounds on l'_{ijkl}, $\overline{l}'_{ijkl}$ would have to be rather large, since it is difficult to estimate the spacing of the event-times. Therefore this results in a very large problem for both *FET* and *VET*. Furthermore, in the case of *VET*, it is also difficult to estimate the largest number of event-times l^{max}, which further enlarges the problem size. Therefore, we propose iterative procedures to reduce the problem size for both *FET* and *VET*.

6.3.4.1 X-Reduction

It is clear that for each task operation, there must be a maximum possible range between its starting time T_l and the appropriate resource transfer times $T_{l'}$. If $T_{l'}$ is outside the range, there is no need to introduce the integer variable $X_{ijkll'}$ since they will surely take the value of zero.

For the FET algorithm, tighter bounds on the range of l' can be defined:

$$\underline{l}'_{ijk} \leq l' \leq \bar{l}'_{ijk} \tag{6.48a}$$

where $$\underline{l}'_{ijk} = l + \max\{k, \lceil \tau^{min}_{ijk}/\Delta T \rceil\} \tag{6.48b}$$

$$\bar{l}'_{ijk} = l + \lceil \tau^{max}_{ijk}/\Delta T \rceil \tag{6.48c}$$

However, these bounds may still be rather large, and we can improve things further by adopting an iterative approach to decide $\underline{l}'_{ijkl}$, $\bar{l}'_{ijkl}$:

(i) Give an initial estimate of l'_{ijkl} and set bounds:

$$\underline{l}'_{ijkl} = \max(\underline{l}'_{ijk}, l'_{ijkl} - 1); \quad \bar{l}'_{ijkl} = \min\{\bar{l}'_{ijk},\ l'_{ijkl} + 1\} \tag{6.49}$$

(ii) Set fixed values:

$$\text{If} \quad \underline{l}'_{ijkl} > \underline{l}'_{ijk}, \qquad X_{ijkl\underline{l}'_{ijkl}} = 0; \tag{6.50a}$$

$$X_{ijkl\bar{l}'_{ijkl}} = 0 \tag{6.50b}$$

and solve the relaxed LP problem.

(iii) A: If $\underline{l}'_{ijkl} = \underline{l}'_{ijk}$, and $X_{ijkl\underline{l}'_{ijkl}} = 0$ at the solution, reset $\underline{l}'_{ijkl} = \underline{l}'_{ijkl} + 1$;

B: If $\underline{l}'_{ijkl} > \underline{l}'_{ijk}$ and (6.50a) is inactive at the solution, set $\underline{l}'_{ijkl} = \underline{l}'_{ijkl} + 1$;
Otherwise set $\underline{l}'_{ijkl} = \max(\underline{l}'_{ijk}, \underline{l}'_{ijkl} - 1)$.

C: Similar adjustment strategy for $\bar{l}'_{ijkl}$.

(iv) If no bound changes are indicated, terminate iteration; otherwise goto (ii).

However, since cycling can occur, we do not increase $\underline{l}'_{ijkl}$ if this implies more than one return to a given previous value.

For *FET*, ΔT has been chosen *a priori*, which imposes bounds on l' as given in (6.48b, c). It is also possible to make an initial estimate of τ_{ik}, and hence set the initial $l'_{ijkl} = l + \lceil \tau_{ijk}/\Delta T \rceil$.

For *VET* however, we have no way of estimating the spacing between event-times to obtain reasonable estimates of the range of l' for which the $X_{ijkll'}$ variables need to be defined, so we can only take the "worst case" to have the bounds $\underline{l}'_{ijkl} = l + k$, $\bar{l}'_{ijkl} = l^{max}$, and have the initial estimate as $l'_{ijkl} = l + k + 1$. Again we proceed iteratively, exactly as described for *FET*.

6.3.4.2 ET-Reduction

In the case of *FET*, once ΔT is chosen, the number of event-times l^{max} is fixed as $\lceil \frac{T^{max}}{\Delta T} \rceil$ whether processing time varies or not. However, for the *VET* procedure, this will only give a reasonable estimate of l^{max} in the case of fixed processing times. When considering variable processing-times, we also need a way of estimating and adjusting l^{max}.

We can obtain the worst case value by computing the maximum possible number of times for the performing of each *task-operation*, $\lfloor T^{max}/\tau^{min}_{ijk^{max}_i} \rfloor$, then assuming that all the resulting resource transfers occur at different event-times. However this in general leads to a very large over-estimate. So again we adopt an iterative procedure.

The basic idea is that if two successive times T_{l-1} and T_l have a spacing equal to the required accuracy δT, any event which occurs at these times could be considered as simultaneous. Therefore, we can actually merge these two times into one. Starting with a more reasonable estimate of l^{max}, the following procedure can be therefore applied:

Following solution of the relaxed LP, we merge any pair of event-times for which constraint (6.45) is active (indicated by equality). Conversely, if none are active, we increase l^{max} by a small number, and if this occurs on two successive iterations, we terminate the procedure. Clearly a serious under-estimate of l^{max} will lead to a large number of iterations, and it is better to start with an over-estimate. However each solution is an excellent starting point for the following iteration.

6.3.5 Problem Reformulation

For the local MILP in solving the MINLP, we have been using the linear approximations (6.41) – (6.43). The variables can again be disaggregated by defining:

$$\tau_{ijkll'} = X_{ijkll'}\tau_{ik}(\xi_{ijl}); \qquad \xi_{ijkll'} = X_{ijkll'}\xi_{ijl}$$

$$R_{rijkll'} = X_{ijkll'}R_{rik}(\xi_{ijl}); \qquad \overline{R}_{rijkll'} = X_{ijkll'}\overline{R}_{rik}(\xi_{ijl})$$

Then the original constraints denoting the amounts, rates and times of resource transfers during task-operations are also disaggregated to the following forms:

$$\tau_{ijkll'} = \alpha_{ijkl}X_{ijkll'} + \beta^T_{ijkl}\xi_{ijkll'} \tag{6.51}$$

$$R_{rijkll'} = \alpha_{rijkl}X_{ijkll'} + \beta^T_{rijkl}\xi_{ijkll'} \tag{6.52}$$

$$\overline{R}_{rijkll'} = \overline{\alpha}_{rijkl} X_{ijkll'} + \overline{\beta}^{T}_{rijkl} \xi_{ijkll'} \tag{6.53}$$

where: $$\xi_{ij}^{max} X_{ijkll'} \leq \xi_{ijkll'} \leq \xi_{ij}^{max} X_{ijkll'} \tag{6.54}$$

To relate $\tau_{ijkll'}$ to $T_{l'}$, (6.30) is replaced by:

$$\tau_{ijkll'} = X_{ijkll'} \cdot (T_{l'} - T_l) \qquad \forall i,j \in J_i, k \in K_i, l \leq l' \tag{6.30'}$$

Using the disaggregated variables and (6.31), the resource balance constraints (6.39a, b) are disaggregated to the linear form:

$$\sum_j R_{rij1,l-1} + \sum_j R_{rij0l} - D_{rl} + \sum_{i',j',k',l'} R_{ri'j'k'l'l} = 0 \tag{6.55a}$$

$$\sum_j \overline{R}_{rij0l} - \overline{D}_{rl} + \sum_{i',j',k',l'} \sum_{l'' \leq l} (\overline{R}_{i'j'k'l'l''} - \overline{R}_{i'j',k'+1,l'l''}) = 0 \tag{6.55b}$$

For *FET*, to round up $\tau_{ijkll'}$ to the next nearest event-time $T_{l'}$, the disaggregated constraint (6.30′) has to be replaced by:

$$X_{ijkll'}(T_{l'-1} - T_l) < \tau_{ijkll'} \leq X_{ijkll'}(T_{l'} - T_l) \tag{6.30'a}$$

which is a linear constraint.

For *VET*, the linearized form of (6.30′) can be produced similarly to those given in (6.47a, b):

$$T_{l'} - T_l - \tau_{ijkll'} \geq (1 - W_{ijl}) \cdot (l' - l)\delta T + \sum_{l''<l'} X_{ijkll''} \cdot \{\tau_{ik}^{min} + (l' - l'')\delta T\}$$
$$+ \sum_{l''>l'} X_{ijkll''} \cdot \max\{(l' - l)\delta T,\ \tau_{ik}^{min} - (l'' - l')\Delta\overline{T}\} \tag{6.56a}$$

$$T_{l'} - T_l - \tau_{ijkll'} \leq (1 - W_{ijl}) \cdot (l' - l)\Delta\overline{T} + \sum_{l''<l'} X_{ijkll''} \cdot \{\tau_{ik}^{max} + (l' - l'')\Delta\overline{T}\}$$
$$+ \sum_{l''>l'} X_{ijkll''} \cdot \min\{(l' - l)\Delta\overline{T},\ \tau_{ik}^{max} - (l'' - l')\delta T\} \tag{6.56b}$$

The details of the derivation is given in *Appendix C.*

6.3.6 Integer Cuts and Valid Inequalities

To enhance the solution of the MILP, some cut constraints can often be imposed from the analysis of the problem feature.

The first set of integer cuts is generated from the fact that there is a limit on the number of executions of each type of operation of a task, which can be decided by its minimum processing time:

$$\sum_{l} W_{ijl} \leq \left\lfloor \frac{T^{max}}{\tau^{min}_{ijk^{max}}} \right\rfloor \qquad \forall i,j \in J_i \tag{6.57}$$

The second set of integer cuts is created from the underlying relations between successive resource transfers during the task operation. Obviously, the X- variables indicating the ordering of events on the common event-time sequence must satisfy:

$$\sum_{l'} l' (X_{ijkll'} - X_{ij,k+1,ll'}) \leq 0 \qquad \forall i,j \in J_i, k \in K_i, l \tag{6.58}$$

using the special ordered set constraint properties (Wilson, 1990), we get:

$$\sum_{l'=\underline{l}'_{ijkl}}^{k_1-1} X_{ijkll'} - \sum_{l'=\underline{l}'_{ij,k+1,l}}^{k_1} X_{ij,k+1,ll'} \geq 0 \qquad \forall i,j \in J_i, k \in K_i, l, \quad k_1 = \underline{l}'_{ijkl} \ldots \bar{l}'_{ijkl} \tag{6.59}$$

Also from the special ordered set, we can have the following integer cuts:

$$\sum_{l \leq l'} X_{ijkll'} \leq 1 \qquad \forall i,j \in J_i, k \in K_i, l' \tag{6.60}$$

This simply states that each event-time $T_{l'}$ can be identified with one and only one resource transfer time for a specific task-operation started earlier.

Finally, a set of constraints for the continuous variables have also proved to be useful. This stems from the fact that the total duration of all executions for a specific task-operation must not exceed the time horizon.

$$\sum_{l} \tau_{ijk^{max}_i l} \leq T^{max} \qquad \forall i,j \in J_i, k \in K_i \tag{6.61}$$

6.4 Computational Results

The unified formulation in section *6.3* has been tested on several examples with processing times varying linearly with the batch-size, by ±33% from the mean time over the range of batch-sizes (*cf.* section *4.2.7*). SCICONIC is used on a Sun Sparc10/41 workstation to solve the LPs and MILPs, with an optimality margin of 5%. The details of *Examples* 1 and 2 are the same as those given in section *4.2.7*, details for *Examples* 3 and 4 are given in Table 6.1 and 6.2. The *STN* representations for *Examples* 3 and 4 are shown in Figure 6.3, 6.4.

Table 6.1: Specifications for Example 3

Time Horizon = 60 hrs

Process units	Capacity (kg)	Suitability	Mean process time
RA1,RA2,RA3,RA4	100	R-INTA	10
SEPA1,SEPA2	400	SEP-A	15
RB1	300	R-INTB	10
WB1	400	Wash-B	5
RC1,RC2,RC3,RC4	100	R-INTC	10
WC1	400	Wash-C	5
Process states	**Storage capacity(kg)**	**Initial amount(kg)**	**Material prices**
Feeds	Unlimited	Unlimited	0.0
IMP-A,IMP-B,IMP-C	0.0	0.0	0.0
INT-A	1000	0.0	0.0
INT-B	1000	0.0	0.0
Prod-C	Unlimited	0.0	1.0

Table 6.2: Specifications for Example 4

Time Horizon = 76 hrs

Process units	Capacity (kg)	Suitability	Mean process time
Reactor	20	Reaction	26
Mixtank1,Mixtank2	20	Mixing	4
Filter	20	Filtering	6
Strtank1,Strtank2	20	Stripping	8
Process states	**Storage capacity(kg)**	**Initial amount(kg)**	**Material prices**
Feed,Add1	Unlimited	Unlimited	0.0
R-Prod,Blend,Filt	100.0	0.0	0.0
Prod	Unlimited	0.0	2.0

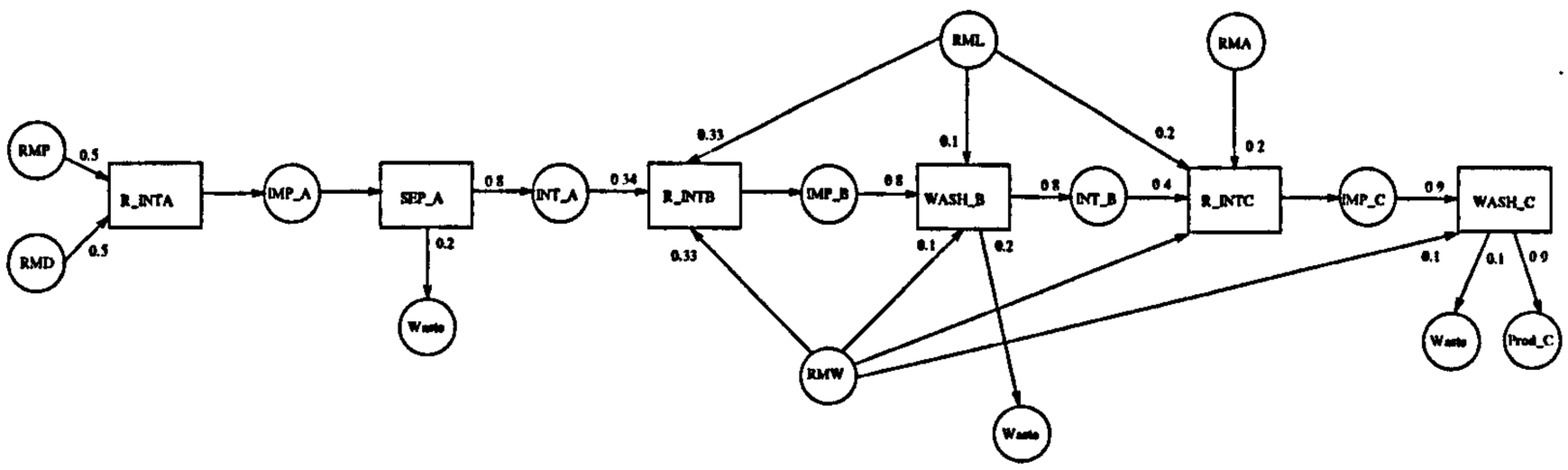


Figure 6.3: *STN* representation for Example 3

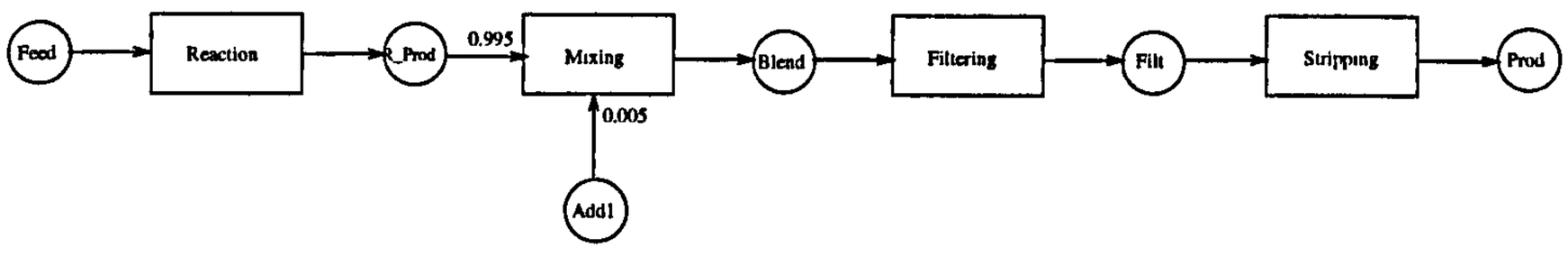


Figure 6.4: *STN* representation for Example 4

6.4.1 Effect of X-Reduction for Original FET Algorithm

To test the *FET* algorithm with the original formulation, *Examples* 1 and 2 are first solved using the full range of l' given by (6.48). Then the X- reduction technique is applied to generate the reduced range iteratively. Comparing the results given in Table 6.3 and Table 6.4, it is clear that the reduced range usually makes a substantial savings on CPU time to solve the MILP, and the reduction is larger for the larger number of event intervals.

However, the total CPU times need further explanation. We did not alter the SCICONIC package to make the reductions on successive LPs as the MILP is solved, but simply iterated on the initial fully relaxed LP, then solved the MILP for the resulting fixed range of l'. In all cases, the bounds were in fact inactive at the solution of the MILP. The number of LPs and the CPU time for the preprocessing is shown separately, and we would hope to make a significant saving on the overall time in a proper implementation.

6.4.2 Effect of Reduction Techniques for Original VET Algorithm

Using the *VET* algorithm, the problems are solved first with ET-reduction only, then additionally with X-reduction. Table 6.5 gives the results comparing the problem size for the worst-case of choosing l^{max} with the reduced version, for event-times only. The initial

estimates of l^{max} were 16, 16, 16, 18 for *Examples* 1, 2, 3, 4, respectively. It can be seen that the ET-reduction procedure gives an estimate of a smaller number of event-times close to the actual number of distinct events hence the problem size is greatly reduced. Table 6.6 gives the results by applying additionally the X-variable range-reduction. The problem size is further reduced and the problem is also easier to solve compared with using only ET-reductions.

Table 6.4 and Table 6.6 show the loss of optimality due to the time discretization in the *FET* algorithm, and if the interval ΔT for *FET* were reduced to the tolerance δT (10^{-3}) used in *VET*, the problem would obviously be impracticably large. On the other hand, for *Examples* 3 and 4, even with a very low time resolution, say ΔT= 0.5, the problem resulted by using the *FET* algorithm are still very large. However, when applying the reduction techniques to *VET*, again, a much smaller problem can be obtained. For *Example* 3 in particular, it can be solved easily without the additional X-reduction.

6.4.3 Effect of the Reformulation

The reformulation is also tested using both *FET* and *VET* algorithms. As shown in Table 6.7, for *FET*, the reformulated model leads to a tighter formulation with smaller integrality gap between optimal integer solution and the fully relaxed LP solution. Both the number of LPs and CPU time required by the solution are greatly reduced compared to the original formulation. Comparing Table 6.4 with Table 6.7, it can be seen that the time used for X-reduction for the original formulation is even longer than that for solving the MILP problem of the reformulation, therefore, the X-reduction procedure has not been applied to the reformulation.

However, it is disappointing to find, as shown in Table 6.8, that the reformulation did not lead to a consistent improvement for the *VET* algorithm.

Table 6.3: Results of the Original *FET* Algorithm Without X-reduction

	Example 1		Example 2	
NEI	60	24	40	16
NC	3136	958	4960	1625
NV	3853	937	5109	1353
NIV	1623	345	2108	494
OBJ for MILP	66.66	66.66	1396.09	1274.35
OBJ for Relaxed LP	101.91	101.68	1537.19	1524.32
LPs	13068	348	296	168
CPU	17827.4	91.7	14179.4	122.3

Table 6.4: Effect of X-reduction for Original *FET* Algorithm

	Example 1		Example 2	
NEI	60	24	40	16
NC	2567	842	4326	1524
NV	2715	705	3969	1151
NIV	1054	229	1538	393
OBJ for MILP	68.23	66.66	1395.46	1274.35
OBJ for Relaxed LP	101.70	96.67	1534.26	1524.32
LPs for MILP	2683	112	249	49
CPU for MILP	2627.2	26.6	1176.3	47.1
X-Reduction:				
LPs	41	14	27	6
CPU	3257.8	87.3	7134.9	106.0

Symbols used:

NEI: Number of Event Intervals
NV: Total Number of Variables
NC: Number of Constraints
NIV: Number of Integer Variables

Table 6.5: Effect of ET-reduction for Original *VET* Algorithm

	Example 1		Example 2		Example 3		Example 4	
NEI	43	7	113	7	101	6	27	8
NC	9519	428	–	1204	–	1416	7993	1017
NV	6547	295	–	777	–	941	5386	693
NIV	2967	105	> 10,000	301	–	351	2430	264
OBJ for MILP	–	71.47	–	1497.96	–	360.0	–	57.34†
OBJ for Relaxed LP	199.99	199.99	–	2910.36	–	360.0	–	88.12
LPs for MILP	–	9617	–	102766	–	37	–	> 10,5000
CPU for MILP	–	812.4	–	28423.6	–	17.4	–	> 27923.7
ET-Reduction:								
Initial l^{max}	–	16	–	16	–	16	–	18
LPs	–	5	–	5	–	5	–	5
CPU	–	49.5	–	666.6	–	783.8	–	215.9

†Best integer solution so far

Table 6.6: Effect of Reduction Techniques for Original *VET* Algorithm

	Example 1	Example 2	Example 3†	Example 4
NEI	7	7	6	8
NC	260	725	1416	576
NV	183	471	941	399
NIV	49	148	351	117
OBJ for MILP	71.47	1497.96	360.0	57.72
OBJ for Relaxed LP	180.65	2258.73	360.0	88.12
LPs for MILP	1002	1685	37	2643
CPU for MILP	35.2	211.0	17.4	221.6
X-Reduction:				
LPs	1	2	–	4
CPU	1.6	12.5	–	14.6

†Results for Example 3 do not use X-reductions

Table 6.7: Results of the Reformulated *FET* Algorithm Without X-reduction

	Example 1		Example 2	
NEI	60	24	40	16
NC	5482	1288	7456	1925
NV	5116	1138	6537	1575
NIV	1623	345	2108	494
OBJ for MILP	67.5	66.66	1401.32	1272.91
OBJ for Relaxed LP	76.51	75.89	1489.17	1419.64
LPs	81	19	69	71
CPU	1615.7	16.7	1885.5	78.4

Table 6.8: Effect of Reduction Techniques for Reformulated *VET* Algorithm

	Example 1	Example 2	Example 3†	Example 4
NEI	7	7	6	8
NC	263	741	1611	605
NV	187	497	1136	426
NIV	48	147	351	119
OBJ for MILP	71.45	1497.69	360.0	57.72
OBJ for Relaxed LP	149.99	2258.71	360.0	88.12
LPs for MILP	528	9575	37	1836
CPU for MILP	21.9	1027.5	18.5	176.2
X-Reduction:				
LPs	1	1	–	6
CPU	1.6	7.2	–	25.6

†Results for Example 3 do not use X-reductions

6.5 A Case Study – Scheduling of Fast Moving Consumer Goods (FMCG) Manufacturing

In this section, the features of the unified scheduling formulation are demonstrated through a case study. The case study considered is based on an industrial fast moving consumer goods manufacturing plant.

The objective is to schedule the manufacture of a set of products in a factory using all the resources on site. The schematic manufacturing route is given as: Making, Intermediate Storage, Packing and Warehousing.

6.5.1 Description of the Process

Recipes:

There are three bases (available as required) used to produce fifteen products. These comprise seven intermediate variants each with different packaging sizes. Five of the products are manufactured in two variants from the first base, and are packed in three different sizes. Another three products are produced in the form of two variants from another base each with different pack sizes. The remaining seven products are made in three variants from the third base, again with different pack sizes. All intermediate variants are produced by mixing from one of the three bases. The making systems are operated in a continuous fashion with variable production rates. Packaging takes place in five different packing lines which also run continuously. Products are packed either in bottles or carton boxes. Three multipurpose storage tanks are available for the storage of all seven intermediates.

All except one of the packing lines are subject to sequence-dependent change-over. The product packed in each of these lines belongs to different "families" defined by products from different bases, of different concentrations or of different pack sizes. The change-over requirements are: 1 hour when switching between products from different bases; 2 hours when switching between products of different packing sizes; 4 hours when switching between dilute and concentrate products.

Equipment :

Two mixing systems are available, one with a single mixer producing two of the variants (intermediates) and another with two identical mixers producing the remaining five intermediates. Both mixers are operated in a continuous fashion and run within given rate limits.

Each final product is packed in a specified packing line which also operates at a variable rate within given limits.

The available storage facilities include three identical storage tanks, each of capacity 60 tonnes, suitable for the temporary storage of all intermediates. Unlimited storage is available for the final products.

The equipment units are available at all times and no constraints on run lengths are imposed. The rate limits may be different on the same mixer or packing line for producing different variants/products. All continuous operations can be started and stopped as required. The suitability of available equipment and the detailed change-over requirements on individual packing lines are given in Table 6.9 below.

Table 6.9: Characteristics for Equipment

Time Horizon = 120 hrs

Process units	throughput (te/hr)/ capacity (te)	Suitability	Change-over Requirements
Mixer1	17	Make-INT1, Make-INT2	–
Mixer2,3	12.24	Make-INT5, Make-INT6	–
		Make-INT7	–
	17	Make-INT3, Make-INT4	–
Tank 1,2,3	60	Store All Intermediates	–
LINE 1	5.8333	Pack-D12, Pack-D21, Pack-D42	between {Pack-D12, Pack-D21} and {Pack-D42} – 1 hour
LINE 2	2.7083	Pack-D13, Pack-D22 Pack-C11, Pack-C31	between {Pack-D13, Pack-D22} and {Pack-C11, Pack-C31} – 4 hour
LINE 3	5.5714	Pack-D11, Pack-D31	between {Pack-D11} and {Pack-D31} – 1 hour
LINE 4	2.2410	Pack-C23, Pack-D41	between {Pack-C23, Pack-D41} and {Pack-C22, Pack-C32} – 2 hour
	3.3333	Pack-C22, Pack-C32	
LINE 5	5.3571	Pack-C12, Pack-C21	–

Production Scheduling Problem:

The factory works for 120 hours a week (5 working days), fifteen shifts per week with 8 hours per shift. It is required to create an optimal schedule to meet minimum production requirements. However, it would be desirable to maximize the total production. The imposed production requirements are shown in Table 6.10. The product families are also given in this table.

Table 6.10: The production requirements and product families

Product	PD11	PD12	PD21	PD13	PD22	PD31	PD42	
Family	D1	D2	D2	D3	D3	D4	D5	
Requirements (te)	220	251	15	116	7	47	144	
Product	PC11	PC31	PC12	PC21	PC23	PD41	PC22	PC32
Family	C1	C1	C2	C2	S1	S1	S2	S2
Requirements	42.5	13.5	114.5	53	16.5	8.5	2.5	17.5

6.5.2 Modelling

Process

The product recipe is depicted by the STN representation shown in Figure 6.5. All mixing and packing operations are modelled as continuous tasks operating at variable production rates within given limits. The change-overs are modelled as batch tasks of fixed processing times. Therefore, this results in a scheduling problem of mixed plant with both continuous and batch operations. The problem is formulated within the RTN framework using the unified formulation in section *6.3*.

In order to tackle the sequence dependent changovers, the unit usability idea described in the STN framework in section *5.2* is combined here with the unified formulation. Since no division of original resource is necessary, this will give a more compact formulation for this problem. The generalized operational constraints measuring the usabilities and the feasibility of tasks in units (*cf.* (5.6), (5.7)) are now defined as:

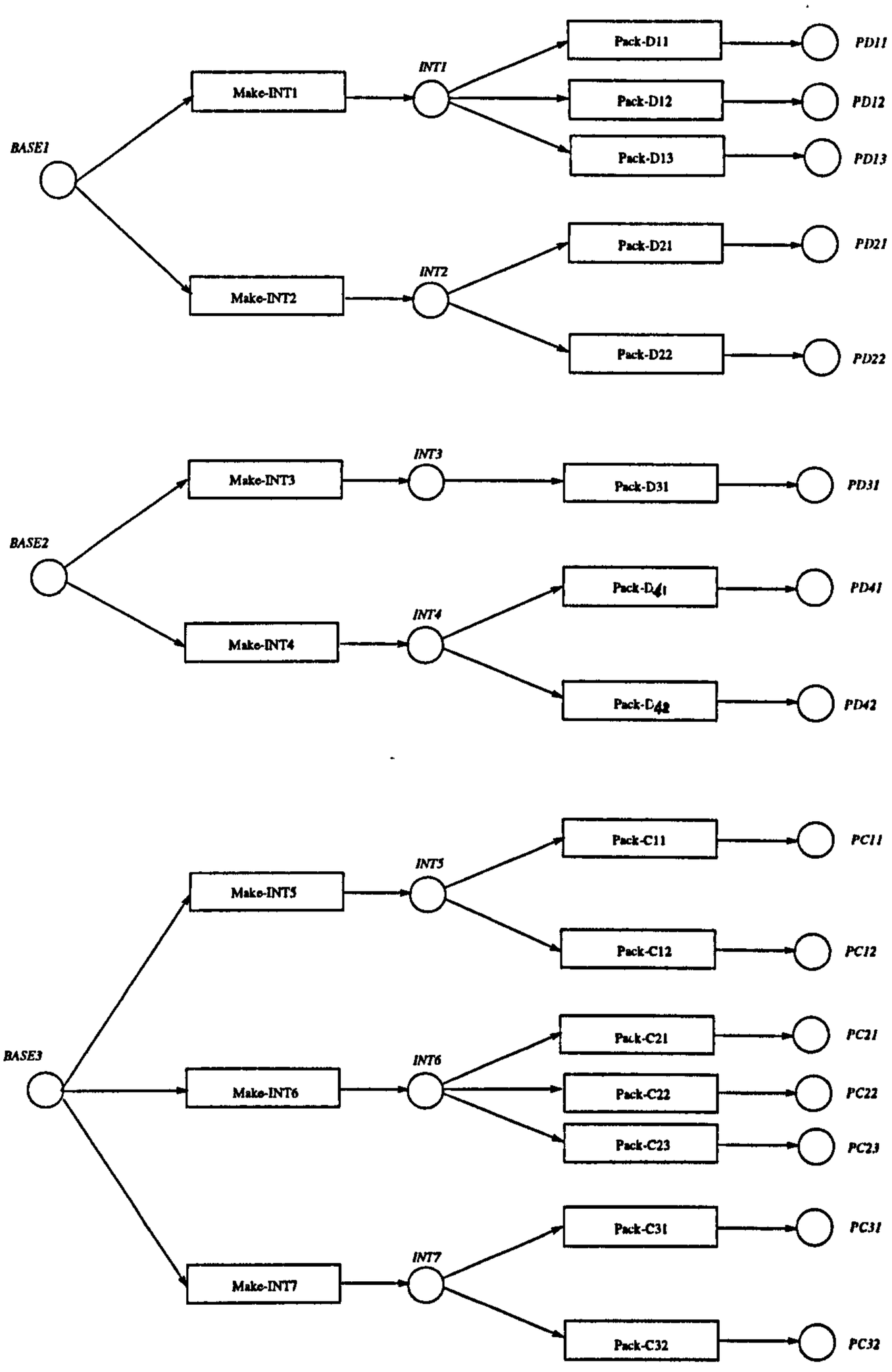


Figure 6.5: STN representation of the FMCG manufacturing plant

$$U_{ur,l+1} = \sum_{i \in I_r} \sum_{j \in J_i} (\alpha_{uir} U_{url} + \beta_{uir}) W_{ijl} \qquad \forall u, r, l \tag{6.62}$$

$$U_{url} \geq \sum_{i \in I_r} \sum_{j \in J_i} \gamma_{uir} W_{ijl} \qquad \forall u, r, l \tag{6.63}$$

The parameters α_{uir}, β_{uir} and γ_{uir} can be chosen analogously to the example given in section *5.2.2.3.* However, since no change-over is required within the products of the same family, it is only necessary to introduce one usability for each family.

Choosing l^{max} and limits of Event-interval

The VET algorithm of the unified formulation is used to solve the problem. The minimum spacing between event-times is determined by the minimum change-over time - 1 hour, while the maximum duration of each event interval is estimated as one work shift, 8 hour. We choose an initial estimate of l^{max} as 31.

6.5.3 Solution

The objective function of the scheduling problem is the maximization of the total production subject to the resource availability. The overall scheduling horizon is 120 hours. The problem is solved using the CPLEX optimization package on a SunSparc10/41 workstation. The computational details are outlined in Table 6.11. A solution with an objective function within about 7% of that of the fully relaxed problem is obtained after 298 nodes.

Table 6.11: Computational details for the scheduling of FMCG manufacturing

NEI	NC	NV	NIV	LPs	CPU (secs)	Value of the Objective Function	
						MILP	Fully relaxed LP
30	4801	4555	1318	298	1085	*2556*	*2724*

The optimal schedule is given in Figure 6.6. The Gantt chart shows the production, storage and change-over operations. The change-overs are indicated by the hatched lines. The "ST" string in the Tasks section describes the storage tasks, and the "CH" string denotes the change-over tasks. The number above each horizontal segment denotes the task being performed while the number below is the total amount of material processed or stored. Note that the amount of storage varies in a constant rate over each interval, hence the storage tank can only be released at the end of the interval in which the denoted storage amount is zero. The actual storage profile of the intermediates in each storage tank is shown in Figure 6.7.

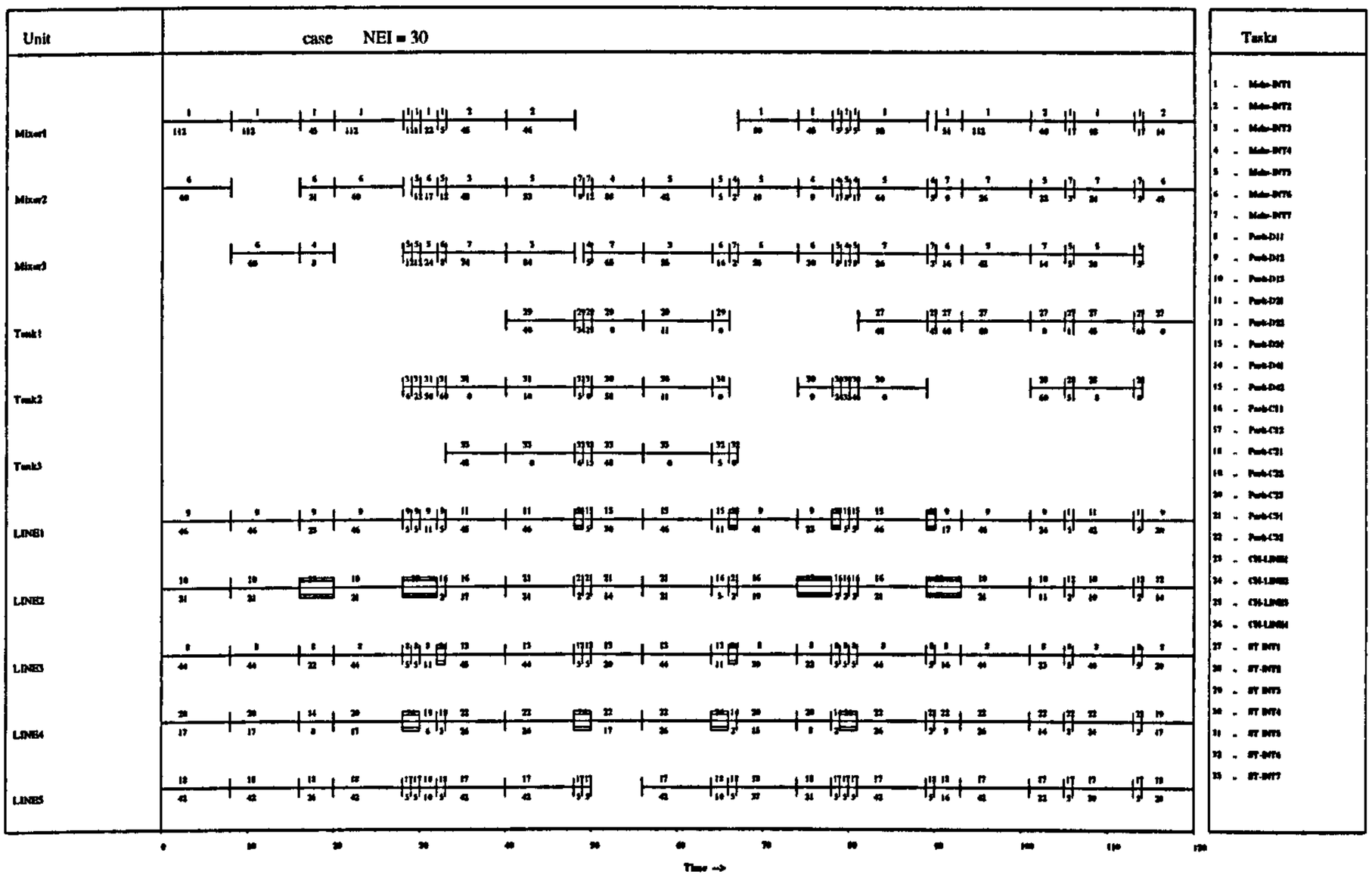


Figure 6.6: Optimal schedule for FMCG manufacturing

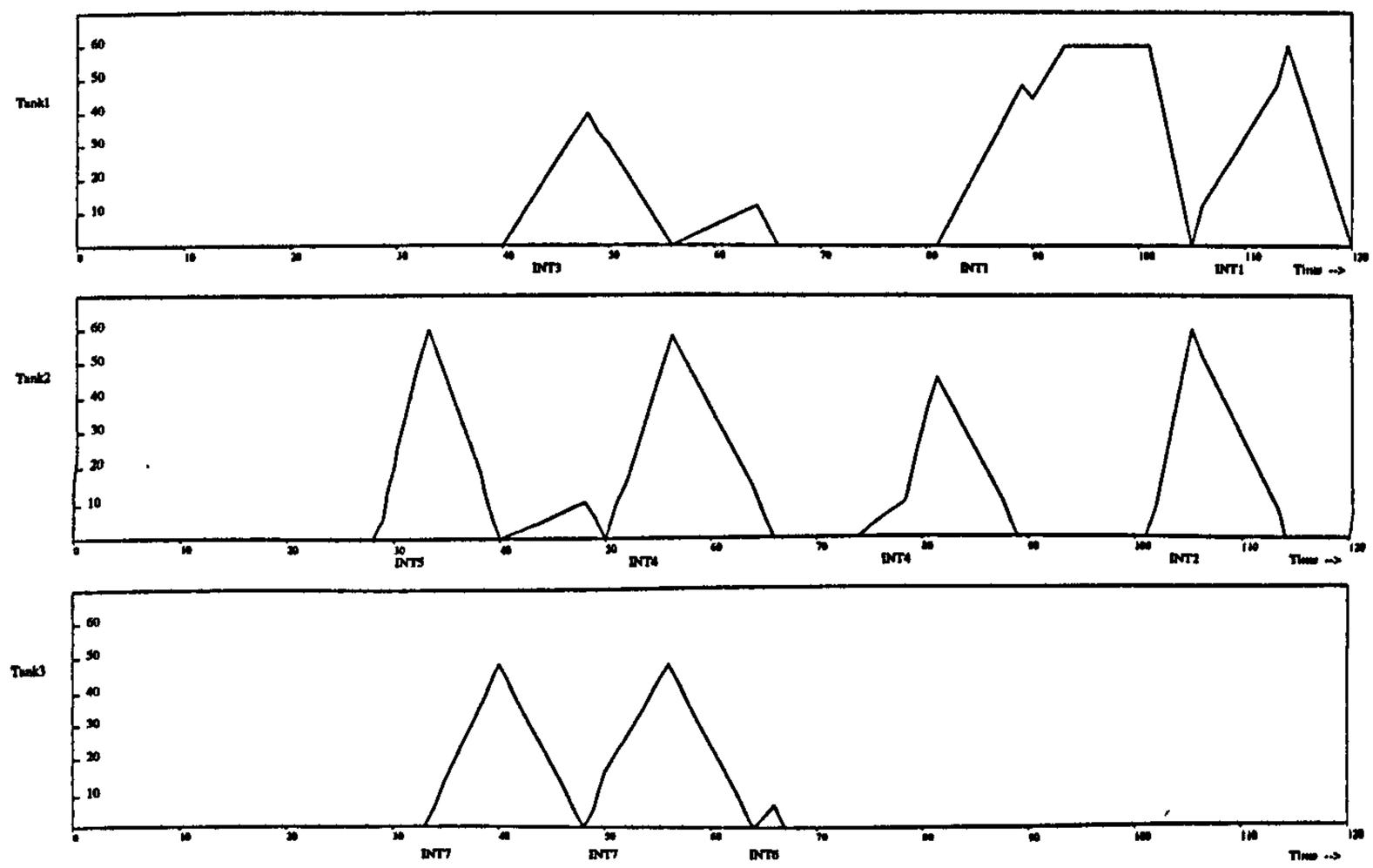


Figure 6.7: Storage profile of intermediates

6.6 Conclusion

In this *Chapter*, the *RTN* concept is extended to consider the general optimal operational scheduling problem. Two types of time representations are developed, the *Resource Transfer Time Sequence* and the *Event Time-Sequence*. Formulations based on either time scheme is able to unify all problem features described in the previous chapters.

The two formulations are very similar, differing only in the treatment of timing constraints. Although we have not given computational results comparing the two time representations, we would expect the use of separate resource transfer time sequences to be more efficient when most resource transfers are distinct in time, but for most tasks resource transfers occur mainly at the beginning and end of the task and when many transfers occur simultaneously, the single unified time sequence will yield a smaller problem.

The formulations yield a large MINLP under the variable operating conditions. In order to make the problem more tractable, ideas of simplification have been presented. First, with different choices of the event-time variables, the unified formulation results in two alternative procedures, *VET* and *FET*. Then, in order to reduce the large problem size, two reduction schemes are proposed for the solution. The X-reduction applies to both *FET* and *VET*. For *VET*, the actual number of distinctive event-times can be estimated through the ET-reduction procedure. On the relative merits of the *FET* and *VET* algorithms, the latter appears more promising, in the sense that it yields much smaller problems without sacrificing accuracy. The algorithm reduces to an existing efficient algorithm for batch processes with fixed recipes.

Finally, the idea of variable and constraint disaggregation is applied to reformulate the problem. From the results it can be seen that for *FET*, the reformulation reduces the integrality gap as well as the effort to solve the problem, however, with the *VET* algorithm, the effect of reformulation has not been evident.

Chapter 7

Conclusions and Future Work

The scheduling problem has been given extensive attention in the chemical processing industrial and academic community. However, in spite of the wide range covered by considering together the previous work, only specific features of the scheduling problem are formulated and solved by earlier algorithms. Even for the most general algorithm reported so far (Kondili *et al.*, 1988), significant assumptions have been made to formulate the problem on a discrete-time representation and to avoid nonlinearities. As a result, only batch process operation with fixed recipes has been tackled satisfactorily.

In this thesis, main emphasis is paid to developing a unified framework with continuous time formulation to simultaneously determine the optimal operating conditions and schedules of a mixed production facility, comprising multipurpose plant for both batch and continuous operations with different resource constraints. The key contributions of the thesis are summarised in the next section and section *7.2* suggests possible directions for future work.

7.1 General Conclusions

In *Chapter 4*, several continuous time representation schemes were proposed and different formulations based on them were developed within the *STN* framework. The essential difference between these time schemes is the ordering of individual system events. They can be divided into two classes: the event-time sequence and the operation sequence. The

former focusses on mapping all activities of a plant onto a common *time sequence*, either uniformly spaced and fixed (*FET*) or non-uniformly spaced and varying (*VET*). The latter is based on the view of process equipment as a kind of "critical" resource and concentrates on units performing tasks following different unit *Operation Sequences.*

By comparing the *FET* and *VET*, the advantage of using *FET* is that it provides a fixed reference against the variation of individual transfers, keeping most constraints, especially the time constraints, linear. Under fixed product recipes, the mathematical equivalence of the *FET* formulation to its discrete time counterpart is proved. Thus, the latter becomes a special case for the general formulation. However, when fine time resolution is required, the problem will become very large. Therefore, there is a balance between sacrificing time accuracy and problem size.

To cope with the problem arising with *FET*, only the real system events should be mapped onto the common time sequence, which requires the event time to be variable, giving the *VET* representation. Nonlinearities will arise concerning the description of event orders in the formulation and it is shown that most of them can be reformulated into a linear form. Other issues concerned with the efficiency of *VET* is how to decide the real number of system events and the reflection of their variation range as accurately as possible. This will directly affect the final problem size.

Abandoning the single time sequence and recognising that the individual events are related to units performing tasks, the unit *Operation Sequence* is introduced as a new time reference. Two types of ordering mechanism are posed for detecting the relative order of each pair of interacting events. Using this as a new way of labelling system events, the problem size will be smaller compared with the "worst case" used in *VET* even with the most conservative estimation of the number of operations in each unit. However, with the still over-estimated number of non-overlap operations, this scheme does not reduce the integrality gap of the problem nor makes it easier to solve.

The aim of *Chapter 4* is to compare and evaluate these time representations considering the basic problem features. In *Chapter 5*, more general and practical aspects are embedded in the formulations presented before. These include limited plant connectivity, the requirement of cleaning of processing equipment which can be either sequence, frequency or time dependent, temporary unavailability of equipment, general storage consideration

of materials in storage tanks as well as in processing units, the storage of feeding and product material in units before and after processing, and continuous transfers.

In *Chapter 6*, the *RTN* concept is extended to consider the general optimal process scheduling problem. Two types of time representations are developed, the *Resource Transfer Time Sequence* and the *Event-Time Sequence.* Formulations based on either time scheme is able to unify all problem features described in the *STN* framework.

In order to reduce the large problem size for variable product recipes such as varying processing times, we then developed two reduction schemes. The X-reduction is used to estimate the number of events covered by one subsequence of a task and is applied to both *FET* and *VET*. Furthermore, ET-reduction scheme has been proposed for *VET* to decide the maximum possible number of event-times.

On the relative merits of the *FET* and *VET* algorithms, the latter appears more promising, in the sense that it yields much smaller problems without sacrificing accuracy.

In summary, the main contributions of this work are on the following aspects:

- A variety of continuous time representations with different event-ordering schemes are devised. The advantages and weaknesses of them are analysed;
- The simultaneous optimization for operating conditions and production scheduling by employing a detailed task model are tackled;
- the optimal operation of batch, continuous plants and mixed plants with both operations are embedded in the unified formulations;
- General frameworks based on both *STN* and *RTN* process representations are developed, which give a unification of previous formulation.

7.2 Future Work

This thesis has developed algorithms within the STN and RTN frameworks for optimal process operation using continuous time representations. Some aspects remain to be addressed or improved in the future.

As far as the computational issues are concerned, further effort is needed to improve the efficiency of the MILP solution. Disaggregating does seem to be promising but more tests

need to be carried out with the VET procedure. On the other hand, a more sophisticated rule is required for the determination and adjustment of the number of event-times.

To account for the interaction between operating conditions and scheduling, the ensuing optimization problem will normally be a MINLP. Having solved the MILP problem based on local linearization of the nonlinear functions, future work should envisage the solution of the full MINLP problem based on the improvement of local MILP solution efficiencies.

Short-term scheduling problem has been the focus of this thesis. Obviously, applying the same ideas to the long term planning problem can be another area for further investigation.

The establishment of a continuous time representation has opened the way for tackling problems such as synthesis with more sophisticated models. A differential algebraic model rather than a relatively simple algebraic model will be required for the description of the dynamic behaviour of processes. This emerges as a mixed integer optimal control (MIOC) problem where planning and control are interacting with each other.

References

W.P. Adams and H.D. Sherali. Linearization strategies for a class of zero-one mixed integer programming problems. *Operations Research*, 38(2):217–226, 1990.

A.P. Barbosa-Póvoa. *Detailed Design and Retrofit of Multipurpose Batch Plants.* PhD thesis, Imperial College of Science, Technology and Medicine, 1994.

C. Barnhart, E.L. Johnson, G.L. Nemhauser, G. Sigismondi, and P. Vance. Formulating a mixed integer programming problem to improve solvability. *Operations Research*, 41(6):1013–1019, 1993.

D.B. Birewar and I.E. Grossmann. Efficient optimization algorithms for zero-wait scheduling of multiproduct batch plants. *Ind. Eng. Chem. Res.*, 28(9):1333–1345, 1989.

D.B. Birewar and I.E. Grossmann. Incorporating schedule in the optimal design of multiproduct batch plants. *Comput. Chem. Engng.*, 13:141–161, 1989.

D.B. Birewar and I.E. Grossmann. Simultaneous production planning and scheduling in multiproduct batch plants. *Ind. Eng. Chem. Res.*, 29:570–580, 1990.

C.A. Crooks. *Synthesis of Operating Procedures for Chemical Plants.* PhD thesis, Imperial College of Science, Technology and Medicine, 1992.

H. Crowder, E.L. Johnson, and M. Padberg. Solving large-scale zero-one linear programming problems. *Operations Research*, 31(5):803–834, 1983.

I. Dedopoulos and N. Shah. Optimal short-term scheduling of maintenance and production for multipurpose plants. *Ind. Eng. Chem. Res.*, 34(1):192–201, 1995.

U.M. Egli and D.W.T. Rippin. Short-term scheduling for multipurpose batch chemical plants. *Comput. Chem. Engng.*, 10(4):303–325, 1986.

G.A. Floudas and V. Visweswaren. A primal-relaxed dual global optimization approach. *JOTA*, 78(2):187–225, 1993.

F. Glover. Improved linear integer programming formulations of nonlinear integer problems. *Management Science*, 22(4):445–460, 1975.

I.E. Grossmann, I. Quesada, R. Raman , and T.V. Voudouris. Mixed-integer optimization techniques for the design and scheduling of batch processes. In *NATO Conference on Batch Processes*, Antalya, Turkey, 1992.

Jänicke. On the solution of scheduling problems for mutipurpose batch chemical plants. *Comput. Chem. Engng.*, 8(6):339–343, 1984.

E.L. Johnson. Modelling and strong linear programs for mixed integer programming. In *NATO ASI Series, Vol. F51*, Mathematical Programming, 1989.

E. Kondili, C. C. Pantelides, and R. W. H. Sargent. A general algorithm for scheduling of batch operations. In *Intl. Symp. on Process Systems Engineering*, pages 62–75, Sydney, Australia, 1988.

E. Kondili, C.C. Pantelides, and R.W.H. Sargent. A general algorithm for scheduling of batch operations. *Comput. Chem. Engng.*, 23:221–227, 1993.

E. Kondili, N. Shah, and C.C. Pantelides. Production planning for the rational use of energy in multiproduct continuous plants. *Comput. Chem. Engng.*, 17:S123–S128, 1993.

E. Kondili. *Optimal Scheduling of Batch Chemical Processes.* PhD thesis, Imperial College of Science, Technology and Medicine, 1988.

H. Ku and I.A. Karimi. Scheduling in serial multiproduct processes with finite intermediate storage: A mixed integer linear program formulation. *Ind. Eng. Chem. Res.*, 27:1840–1848, 1988.

H. Ku and I.A. Karimi. Scheduling algorithms for serial multiproduct processes with tardiness penalties. *Comput. Chem. Engng.*, 15:283–286, 1991.

H. Ku, D. Rajagopalan, and I.A. Karimi. Scheduling in batch processes. *Chem. Eng. Prog.*, pages April, 35–45, 1987.

K. Kuriyan and G.V. Reklaitis. Scheduling network flowshops so as to minimise makespan. *Comput. Chem. Engng.*, 13:187–200, 1989.

K. Kuriyan, G. Joglekar, and G.V. Reklaitis. Multiproduct plant scheduling studies using BOSS. *Ind. Eng. Chem. Res.*, 26(8):1551–1558, 1987.

A. Lázaro and L. Puigjaner. Simulation and optimization of multiproduct plants for batch and semi-batch operations. In *In. Chem. Eng. Symp. Ser.*, volume 92, pages 209–222, 1985.

M. Lázaro, A. Espunã, and L. Puigjaner. A comprehensive approach to production planning in multipurpose batch plants. *Comput. Chem. Engng.*, 13:1031–1047, 1989.

R.S.H. Mah. *Chemical Process Structures and Information Flows.* Butterworths, Boston, 1989.

R.K. Martin and D.J. Sweeney. An ideal column algorithm for integer programs with special ordered sets of variables. *Mathematical Programming*, 26:48–63, 1983.

A. Mauderli and D.W.T. Rippin. Production planning and scheduling for multipurpose batch chemical plants. *Comput. Chem. Engng.*, 3:199–206, 1979.

B.A. Murtagh. *Advanced Linear Programming Computation and Practice.* McGraw – Hill, New York, 1981.

R.F.H. Musier and L.B. Evans. An approximate method for the production scheduling of industrial batch process with parallel units. *Comput. Chem. Engng.*, 13:229–238, 1989.

R.F.H. Musier and L.B. Evans. Batch process management. *Chem. Eng. Prog.*, 86(6):66–77, 1990.

R.F.H. Musier and L.B. Evans. Schedule optimization with simultaneous lot-sizing in chemical process plants. *AIChE J.*, 37:886–896, 1991.

G.L. Nemhauser. The age of optimization: solving large-scale real-world problems. *Operations Research*, 42(1):5–13, 1994.

C.C. Pantelides, Realff M.J., and Shah N. Short-term scheduling of pipeless batch plants. In *AICHE National Meeting*, Miami Beach, Miami, November, 1993.

C.C. Pantelides. Unified framework for optimal process planning and scheduling. In *Second Conf. on Foundations of Computer Aided Process Operations (FOCAPO II)*, Mt. Crested Butte, Colorado, 1993.

L.G. Papageorgiou, C.C. Pantelides, and Shah N. Energy integration in multipurpose plant operation. In *CHISA'93*, Prague, Czech Republic, August, 1993.

L.G. Papageorgiou. *Optimal Scheduling of Multipurpose Batch/Semicontinuous Plants.* PhD thesis, Imperial College of Science, Technology and Medicine, 1994.

J.F. Pekny and D.L. Miller. Exact solution of the no-wait flowshop scheduling problem with a comparison to heuristic methods. *Comput. Chem. Engng.*, 15:741–748, 1991.

J.F. Pekny D.L. Miller, and G.J. McRae. An exact parallel algorithm for scheduling when production costs depend on consecutive system states. *Comput. Chem. Engng.*, 14:1009–1023, 1990.

J.M. Pinto and I.E. Grossmann. An MILP model for short term scheduling of batch plants. In *paper 235b, AIChE Annual Meeting*, San Francisco, Nov., 1994.

J.M. Pinto and I.E. Grossmann. Optimal cyclic scheduling of multistage continous multiproduct plants. *Comput. Chem. Engng.*, 18(9):797–816, 1994.

D. Rajagopalan and I. A. Karimi. Completion times in serial mixed storage multiproduct processes with transfer and set-up times. *Comput. Chem. Engng.*, 13:175–186, 1989.

G.V. Reklaitis. Perspectives on scheduling and planning of process operations. In *Fourth Intl. Symp. on Process Systems Engineering*, Montebello, Canada, 1991.

G.V. Reklaitis. Overview of scheduling and planning of batch process operations. In *NATO Conference on Batch Processing Systems Engineering*, Antalya, Turkey, 1992.

S.A. Rich and G. J. Prokopakis. Scheduling and sequencing of batch operations in a multipurpose plants. *Ind. Eng. Chem. Process Des. Dev.*, 25(4):979–988, 1986.

D.W.T. Rippin. Batch process systems engineering: a retrospective and prospective review. *Comput. Chem. Engng.*, 32(11):S1–S32, 1992.

N.V. Sahinidis and I. E. Grossmann. MINLP models for cyclic multiproduct scheduling on continuous parallel lines. *Comput. Chem. Engng.*, 15(2):85–103, 1991.

N.V. Sahinidis and I. E. Grossmann. Reformulation of multiperiod MILP models for planning and scheduling of chemical processes. *Comput. Chem. Engng.*, 15(4):255–272, 1991.

G. Schilling, Y.-E. Pineau, C.C. Pantelides, and N. Shah. Optimal scheduling of multipurpose continuous plants. In *AIChE Annual Meeting*, San Francisco, Nov., 1994.

SD-SCICON. *SCICONIC vs 2.11 -User Manual.* 1990.

N. Shah and C.C. Pantelides. Optimal long term campaign planning and design of batch operation. *Ind. Eng. Chem. Res.*, 30(10):2309–2321, 1991.

N. Shah, C.C. Pantelides, and R.W.H. Sargent. Efficient scheduling techniques for optimal scheduling of batch plants. *Comput. Chem. Engng.*, 30:229–244, 1993.

N. Shah, C.C. Pantelides, and R.W.H. Sargent. Optimal periodic scheduling of multipurpose batch plants. *Annals of Operations Research*, 42:193–228, 1993.

N. Shah. *Efficient Scheduling, Planning and Design of Multipurpose Batch Plants.* PhD thesis, Imperial College of Science, Technology and Medicine, 1992.

I. Suhami and R.S.H. Mah. An implicit enumeration scheme for the flowshop problem with no intermediate storage. *Comput. Chem. Engng.*, 5:83–91, 1981.

F.E. Torres. Linearization of mixed-integer products. *Math. Prog.*, 49:427–428, 1991.

A.G. Tsirukis, S. Papageorgaki, and G.V. Reklaitis. Scheduling of multipurpose batch chemical plants with resource constraints. *Ind. Eng. Chem. Res.*, 32:3037–3050, 1993.

J. Viswanathan and I.E. Grossmann. A combined penalty function and outer approximation method for MINLP optimization. *Comput. Chem. Engng.*, 14(7):769–782, 1990.

M.C. Wellons and G.V. Reklaitis. Optimal schedule generation for a single production line – I. problem formulation. *Comput. Chem. Engng.*, 13:201–212, 1989.

M.C. Wellons and G.V. Reklaitis. Optimal schedule generation for a single production line – II. identification of dominant unique path sequences. *Comput. Chem. Engng.*, 13:213–227, 1989.

M.C. Wellons and G.V. Reklaitis. Scheduling of multipurpose batch chemical plants. 1. formation of single-product campaigns. *Ind. Eng. Chem. Res.*, 30(4):671–688, 1991.

M.C. Wellons and G.V. Reklaitis. Scheduling of multipurpose batch chemical plants. 2. multiple-product campaign formulation and production planning. *Ind. Eng. Chem. Res.*, 30(4):688–705, 1991.

W. Wiede and G.V. Reklaitis. Determination of completion times for serial multiproduct processes. *Comput. Chem. Engng.*, 11(4):337–368, 1987.

H.P. Williams. *Model Building in Mathematical Programming.* John Wiley & Son Ltd., 1990.

J.M. Wilson. Generating cuts in integer programming with families of special ordered sets. *European J. of Operational Research*, 46:101–108, 1990.

M.G. Zentner and G. V. Reklaitis. An interval based approach for resource constrained batch processing scheduling, Part I. interval processing framework. In *Proceedings of COPE-91*, Barcelona, Spain Oct., 1991.

M.G. Zentner and G. V. Reklaitis. An interval based approach for resource constrained batch processing scheduling, Part II. assignment and adaptive storage retrofitting. In *paper 140d, AIChE Annual Meeting*, Los Angeles Nov., 1991.

Appendix A

Comparing FET Procedure with Discrete Time Formulation

The comparison of the formulation in *Chapter 4* based on *Fixed Event Time* representation and the discrete time formulation in *Chapter 3* is presented below.

When task processing times are fixed, *i.e.* independent of the operating variables (such as the batch size), the spacing between event times can easily be chosen as the highest common factor of all the transfer time intervals. The value of the binary variables, $X_{ijkll'}$, introduced to identify the actual transfer time with the fixed event times can be determined *a priori* according to the value of the W_{ijl} variables, *i.e.* in (4.4), only one $X_{ijkll'}$ variable exists:

$$X_{ijkll'} = W_{ijl} \qquad \text{where: } l' = l + \frac{\tau_{ijk}}{\Delta T} = l + l_{ijk} \qquad (\underline{l}_{ijk} = \bar{l}_{ijk} = l_{ijk}) \qquad \text{(A.1)}$$

Allocation Constraint

Substituting the r.h.s. of (4.9) into the allocation constraint, the l.h.s. of (4.8) becomes:

$$\sum_{i\in I_j}\sum_{k<k_i^{max}}\sum_{l\le l'}\sum_{l''\le l'}(X_{ijkll''} - X_{ij,k+1,ll''}) = \sum_{i\in I_j}\sum_{l\le l'}\sum_{l''\le l'}(X_{ij0ll''} - X_{ijk_i^{max}ll''})$$

From (A.1):

$$X_{ij0ll''} = \begin{cases} W_{ijl} & l'' = l \\ 0 & \text{otherwise} \end{cases}$$

$$X_{ijk_i^{max}ll''} = \begin{cases} W_{ijl} & l'' = l + \frac{\tau_{ijk_i^{max}}}{\Delta T} = l + \bar{l}_{ij} \leq l' \\ 0 & \text{otherwise} \end{cases}$$

Therefore, the l.h.s. of (4.8) becomes:

$$\sum_{i \in I_j} \sum_{l=1}^{l'} W_{ijl} - \sum_{i \in I_j} \sum_{l=1}^{l'-\bar{l}_{ij}} X_{ijk_i^{max}l,l+\bar{l}_{ij}}$$

$$= \sum_{i \in I_j} \sum_{l=1}^{l'} W_{ijl} - \sum_{i \in I_j} \sum_{l=1}^{l'-\bar{l}_{ij}} W_{ijl} = \sum_{i \in I_j} \sum_{l=l'-\bar{l}_{ij}+1}^{l'} W_{ijl} \qquad \forall j, l' \qquad \text{(A.2)}$$

Thus the l.h.s. of the allocation constraint is exactly the same as that presented in Shah *et al.* (1993a).

Material Balance Constraint

Similarly, in the last summation of the last term in the r.h.s. of linearized material balance constraint (4.27), only one $B_{ijkll'}$, $l = l' - l_{ijk}$ is nonzero. Therefore, the material balance constraint reduces to:

$$C_{sl'} = C_{s,l'-1} - D_{sl'} + \sum_{i \in I_s} \sum_{j \in J_i} \sum_{k \in K_i} \rho_{sik} B_{ijk,l'-l_{ijk},l'}$$

$$= C_{s,l'-1} - D_{sl'} + \sum_{i \in I_s} \sum_{j \in J_i} \sum_{k \in K_i} \rho_{sik} B_{ij,l'-l_{ijk}} \qquad \forall s, l' \qquad \text{(A.3)}$$

Again, this is in the same form as presented in Kondili *et al.* (1988).

Resource Usage Constraint

A similar procedure can be used to transform the resource usage constraint (4.28) to the following form:

$$R_{rl'} = \sum_{i \in I_r} \sum_{j \in J_i} \sum_{k < K_i^{max}} (\alpha_{rijk} (\sum_{l=1}^{l'-l_{ijk}} X_{ijkl,l+l_{ijk}} - \sum_{l=1}^{l'-l_{ij,k+1}} X_{ij,k+1,l,l+l_{ij,k+1}})$$

$$+ \beta_{rijk} (\sum_{l=1}^{l'-l_{ijk}} B_{ijkl,l+l_{ijk}} - \sum_{l=1}^{l'-l_{ij,k+1}} B_{ij,k+1,l,l+l_{ij,k+1}}))$$

$$= \sum_{i\in I_r}\sum_{j\in J_i}\sum_{k<K_i^{max}} (\alpha_{rijk}(\sum_{l=1}^{l'-l_{ijk}} W_{ijl} - \sum_{l=1}^{l'-l_{ij,k+1}} W_{ijl}) + \beta_{rijk}(\sum_{l=1}^{l'-l_{ijk}} B_{ijl} - \sum_{l=1}^{l'-l_{ij,k+1}} B_{ijl}))$$

$$= \sum_{i\in I_r}\sum_{j\in J_i}\sum_{k<K_i^{max}}\sum_{l=l'-l_{ij,k+1}+1}^{l'-l_{ijk}} (\alpha_{rijk}W_{ijl} + \beta_{rijk}B_{ijl}) \qquad \forall r, l' \tag{A.4}$$

It should be noted that this reduced resource usage constraint is slightly different from that of Kondili *et al.* (1988). This is because of the difference in the definition of task subsequence. In the discrete time formulation, the task subsequence k (or as originally called by the authors "the successive feeding or producing steps") is just the number of time units covered by each task, while our definition here is the real transfer steps according to the task recipe. However, this doesn't make any difference in the solution of the problem. Using the definition of Kondili *et al.* (1988), it will be simply true that:

$$l_{ij,k+1} - l_{ijk} = 1, \qquad \text{and} \quad l_{ijk} = k, \qquad k = 0, \ldots, \bar{l}_{ij} \tag{A.5}$$

Therefore, the resource usage constraints have the same form as those in Kondili *et al.* (1988):

$$R_{rl'} = \sum_{i\in I_r}\sum_{j\in J_i}\sum_{k=0}^{\bar{l}_{ij}-1} (W_{ij,l'-l_{ijk}}\alpha_{rijk} + \beta_{rijk}B_{ij,l'-l_{ijk}}) \qquad \forall r, l' \tag{A.6}$$

Hence the discrete time formulation dealing with fixed product recipes then becomes a special case of the *FET* procedure described.

Appendix B

Derivation of Bounds for Original Time Constraints

Bounds for linearized time constraints (6.47) are derived from the direct comparison between time t_{ijkl} and $T_{l'}$:

$$\begin{aligned} T_{l'} - t_{ijkl} = & \quad (T_{l'} - T_l) - (t_{ijkl} - T_l) \\ = & (T_{l'} - T_l) - \sum_{l'' \geq l} X_{ijkll''}(T_{l''} - T_l) \end{aligned} \tag{B.1}$$

$$\text{Define:} \qquad \Delta\overline{T} = \max_{i,k}\{\tau_{i,k+1} - \tau_{ik}\} \tag{B.2}$$

The determination of the upper and lower bounds of $T_{l'} - t_{ijkl}$ will depend on whether the corresponding resource transfer takes place before or after time $T_{l'}$, as well as whether or not the corresponding task-operation is active at time T_l. Therefore, three situations exist:

(A) Resource transfer takes place before $T_{l'}$:

Suppose $X_{ijkll''} = 1$ for $l'' \leq l'$, then $X_{ijkll''} = 0 \quad \forall l'' > l'$. So:

$$\begin{aligned} T_{l'} - t_{ijkl} = & (T_{l'} - T_l) - \sum_{l'' \leq l'} X_{ijkll''}(T_{l''} - T_l) \\ = & T_{l'} - T_{l''} \end{aligned} \tag{B.3}$$

Using (B.2), the above constraint has simply to satisfy the following bounds:

$$(l' - l'') \cdot \delta T \leq T_{l'} - t_{ijkl} \leq (l' - l'') \cdot \Delta\overline{T} \tag{B.4}$$

(B) Resource transfer takes place after time $T_{l'}$:

Suppose $X_{ijkll''} = 1$ for $l'' > l'$, then $X_{ijkll''} = 0 \quad \forall l'' \leq l'$. So:

$$t_{ijkl} - T_{l'} = \sum_{l''>l'} X_{ijkll''}(T_{l''} - T_l) - (T_{l'} - T_l)$$
$$= (T_{l''} - T_l) - (T_{l'} - T_l) = T_{l''} - T_{l'} \tag{B.5}$$

Therefore:

(I) $(l'' - l') \cdot \delta T \leq t_{ijkl} - T_{l'} \leq (l'' - l') \cdot \Delta\overline{T}$

(II) $\tau_{ik}^{min} - (l' - l) \cdot \Delta\overline{T} \leq t_{ijkl} - T_{l'} \leq \tau_{ik}^{max} - (l' - l) \cdot \delta T$

The upper bounds (UB) and lower bounds (LB) are undecidable for the above two cases.

(C) The corresponding operation does not start at T_l, then $W_{ijl} = 0$, *i.e.* $t_{ijkl} = T_l$.
No restriction should be placed on $(T_{l'} - t_{ijkl})$, hence use the bounds:

$$(l' - l) \cdot \delta T \leq T_{l'} - t_{ijkl} \leq (l' - l) \cdot \Delta\overline{T} \tag{B.6}$$

Combining the above three cases (A),(B),(C), we have the upper and lower bounds as:

$$T_{l'} - t_{ijkl} \geq (1 - W_{ijl}) \cdot (l' - l)\delta T + \sum_{l'' \leq l'} X_{ijkll''} \cdot (l' - l'')\delta T$$
$$- \sum_{l''>l'} X_{ijkll''} \cdot \min\{(l'' - l')\Delta\overline{T},\ \tau_{ik}^{max} - (l' - l)\delta T\} \tag{B.7}$$
$$T_{l'} - t_{ijkl} \leq (1 - W_{ijl}) \cdot (l' - l) \cdot \Delta\overline{T} + \sum_{l'' \leq l'} X_{ijkll''} \cdot (l' - l'')\Delta\overline{T}$$
$$- \sum_{l''>l'} X_{ijkll''} \cdot \max\{(l'' - l')\delta T,\ \tau_{ik}^{min} - (l' - l)\Delta\overline{T}\} \tag{B.8}$$

Appendix C

Derivation of Bounds for Reformulated Time Constraints

The determination of the upper and lower bounds of $T_{l'} - T_l - \tau_{ijkll'}$ in (6.56a, b) will depend on whether the corresponding resource transfer takes place before or after time $T_{l'}$, as well as whether or not the corresponding task-operation is active at time T_l. Therefore, three situations exist:

(A) Resource transfer takes place before $T_{l'}$:

Suppose $X_{ijkll''} = 1$ for $l'' < l'$, then $X_{ijkll''} = 0 \quad \forall l'' > l'$. $\tau_{ijkll'} = 0$. So:

$$T_{l'} - T_l - \tau_{ijkll'} = T_{l'} - T_l = (T_{l'} - T_{l''}) + (T_{l''} - T_l) \tag{C.1}$$

Therefore:

(I) $(l' - l)\delta T \leq T_{l'} - T_l - \tau_{ijkll'} \leq (l' - l)\Delta\overline{T}$

(II) $\tau_{ik}^{min} + (l' - l'')\delta T \leq T_{l'} - T_l - \tau_{ijkll'} \leq \tau_{ik}^{max} + (l' - l'')\Delta\overline{T}$

The upper bounds (UB) and lower bounds (LB) will be:

- UBI $= (l' - l)\Delta\overline{T} = (l'' - l)\Delta\overline{T} + (l' - l'')\Delta\overline{T}$
- UBII $= \tau_{ik}^{max} + (l' - l'')\Delta\overline{T} \leq$UBI
- LBI $= (l' - l)\delta T = (l'' - l)\delta T + (l' - l'')\delta T$
- LBII $= \tau_{ik}^{min} + (l' - l'')\delta T \geq$ LBI

(B) Resource transfer takes place after time $T_{l'}$:

Suppose $X_{ijkll''} = 1$ for $l'' > l'$, then $X_{ijkll''} = 0 \quad \forall l'' \leq l'$. Again, $\tau_{ijkll'} = 0$. So:

$$T_{l'} - T_l - \tau_{ijkll'} = T_{l'} - T_l = (T_{l''} - T_l) - (T_{l''} - T_{l'}) \tag{C.2}$$

Again, two possibilities exist:

(I) $(l' - l)\delta T \leq T_{l'} - T_l - \tau_{ijkll'} \leq (l' - l)\Delta\overline{T}$

(II) $\tau_{ik}^{min} - (l'' - l')\Delta\overline{T} \leq T_{l'} - T_l - \tau_{ijkll'} \leq \tau_{ik}^{max} - (l'' - l')\delta T$

The upper bounds (UB) and lower bounds (LB) are undecidable for the above two cases.

(C) The corresponding operation does not start at T_l, then $W_{ijl} = 0$, *i.e.* $\tau_{ijkll'} = 0, \quad \forall l'$.

No restriction should be placed on $(T_{l'} - T_l - \tau_{ijkll'})$, hence use the bounds:

$$(l' - l) \cdot \delta T \leq T_{l'} - T_l - \tau_{ijkll'} \leq (l' - l) \cdot \Delta\overline{T} \tag{C.3}$$

Combining the above three cases (A),(B),(C), we have the upper and lower bounds as:

$$T_{l'} - T_l - \tau_{ijkll'} \geq (1 - W_{ijl}) \cdot (l' - l)\delta T + \sum_{l''<l'} X_{ijkll''} \cdot \{\tau_{ik}^{min} + (l' - l'')\delta T\}$$
$$+ \sum_{l''>l'} X_{ijkll''} \cdot \max\{(l' - l)\delta T,\ \tau_{ik}^{min} - (l'' - l')\Delta\overline{T}\} \tag{C.4}$$

$$T_{l'} - T_l - \tau_{ijkll'} \leq (1 - W_{ijl}) \cdot (l' - l)\Delta\overline{T} + \sum_{l''<l'} X_{ijkll''} \cdot \{\tau_{ik}^{max} + (l' - l'')\Delta\overline{T}\}$$
$$+ \sum_{l''>l'} X_{ijkll''} \cdot \min\{(l' - l)\Delta\overline{T},\ \tau_{ik}^{max} - (l'' - l')\delta T\} \tag{C.5}$$

BIBL. LORRIN. CNAM.