SCHEDULING OF CRUDE OIL AND PRODUCT BLENDING AND DISTRIBUTION OPERATIONS IN A REFINERY

JIE LI

NATIONAL UNIVERSITY OF SINGAPORE

2009

SCHEDULING OF CRUDE OIL AND PRODUCT BLENDING AND DISTRIBUTION OPERATIONS IN A REFINERY

JIE LI

(M.Eng., Tianjin University)

A THESIS SUBMITTED

FOR THE DEGREE OF PHD OF ENGINEERING

DEPARTMENT OF CHEMICAL AND BIOMOLECULAR ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2009

ACKNOWLEDGEMENTS

I am very much thankful to my supervisors Professor I. A. Karimi and Professor Rajagopalan Srinivasan for their enthusiasm, constant encouragement, insight and invaluable suggestions， patience and understanding during my research at the National University of Singapore. Their recommendations and ideas have helped me very much in completing this research project successfully. I would like to express my heartfelt thanks to Professor I. A. Karimi and Professor Rajagopalan Srinivasan for their guidance on writing scientific papers including this thesis.

Special thanks go to all my lab mates, Mr. P. C. P. Reddy, Dr. Li Wenkai, Dr. Liu Yu, Mr. Ganesh Balla, Mr. Suresh Pitty, Mr. Arul Sundaramoorthy, Mr. B. Mohan Babu, Ms. Huang Cheng, Mr. Suresh Selvarasu, Dr. Mukta Bansal, Ms. Maryam Zargarzadeh, Mr. M. M. Faruque Hasan, Mr. Naresh Susarla, Ms. Sangeeta Balram and Dr. Hong-Choon Oh, for sharing their knowledge with me. I also wish to thank all my friends for their constant encouragement and appreciation.

I express my sincere and deepest gratitude to my parents, my younger sister, and my relatives (Mr. Yeo Kok Hong, Mrs. Wu Jian and Miss Wu Jinjin) in Singapore, for their boundless love, encouragement and moral support.

Finally, I would like to thank the National University of Singapore for providing a research scholarship to make this research project possible.

TABLE OF CONTENTS

SUMMARY

Ever-changing crude prices, deteriorating crude qualities, fluctuating demands for products, and growing environmental concerns are squeezing the profit margins of modern oil refineries like never before. Optimal scheduling of various operations in a refinery offers significant potential for saving costs and increasing profits. The overall refinery operations involve three main segments, namely crude oil storage and processing, intermediate processing, and product blending and distribution. This thesis addresses the first and third important components: scheduling of crude oil, and product blending and distribution.

First, a robust and efficient algorithm is developed to solve large, nonconvex, mixed integer nonlinear programming (MINLP) problems arising from crude blending during crude oil scheduling. The proposed algorithm solves all tested industrial-scale examples up to 20-day scheduling horizon. However, commercial solvers (DICOPT and BARON) and the existing algorithms in the literature fail to solve most of them. Moreover, the proposed algorithm gives profit within 6% of a conservative upper bound. In addition, the practical utility of Reddy et al. (*AIChE Journal, 2004b, 50(6), 1177-1197*)'s MINLP formulation is enhanced by adding appropriate linear blending correlations for fifteen crude properties that are critical to crude distillation and downstream processing, and controlling changes in feed rates of crude distillation unit (CDU).

Second, although the algorithm developed in the first part is intended for a marine-access refinery, the algorithmic strategy is successfully extended to in-land refineries involving both storage and charging tanks. A general discrete-time formulation for an in-land refinery is developed and several crude blending polices in storage and charging tanks are addressed. Four literature examples and eighteen other examples with varying structures, sizes, and complexities are used to illustrate the capability of the proposed formulation and algorithm. The results show that the proposed algorithm is superior to those in the literature.

Third, a general synchronous slot-based MINLP formulation is developed for an integrated treatment of recipe, specifications, blending, storage, and distribution. Many real-life features such as multi-purpose tanks, parallel non-identical blenders, constant rates during blending runs, minimum run lengths, changeovers, linear property indices, piecewise constant profiles for blend component qualities and feed rates, etc. are incorporated in the model. Since commercial MINLP solvers are unsatisfactory for solving this complex MINLP, a novel and efficient procedure that solves successive MILPs (mixed integer linear programming) instead of an MINLP, and gives excellent solutions is proposed.

Fourth, a general and efficient MINLP formulation using unit slots is developed for the above blending and distribution problem. This formulation incorporates all realistic features of the model proposed above. Furthermore, it relaxes an assumption to ensure sufficient supplies of components through the entire scheduling horizon. By solving fourteen examples, it shows that the proposed unit-slot based model obtains the same or better solutions than the process-slot model with fewer binary variables and less computational time.

Finally, a novel approach is first developed for reactive scheduling of crude oil operation. Then, a scenario-based MINLP model is developed to obtain robust schedule for demand uncertainty during crude oil scheduling. The obtained schedule is more robust than the initial schedule.

NOMENCLATURE

Chapter 3

Notation

Sets

IC	Set of pairs (tank *i*, crude *c*) that *i* can hold *c*
IU	Set of pairs (tank *i*, CDU *u*) that *i* can feed *u*
IIU	Set of pairs (tank *ii*, CDU *u*) that *ii* can feed *u*
$\boldsymbol{IF11}_{iuct}$	Dynamic set defined from the value of slack variable u^{-}_{iuct}
$\boldsymbol{IF12}_{i(i')uct}$	Dynamic set defined from the value of slack variable u^{-}_{iuct}
$\boldsymbol{IF21}_{iuct}$	Dynamic set defined from the value of slack variable u^{+}_{iuct}
$\boldsymbol{IF22}_{i(i')uct}$	Dynamic set defined from the value of slack variable u^{+}_{iuct}
$\boldsymbol{IE1}_{(ii)ukt}$	Dynamic set defined from the values of s^{-}_{ukt} and s^{+}_{ukt}
$\boldsymbol{\beta}$	Vessel-based blocks
$\boldsymbol{\alpha}$	Composition-based blocks
$\boldsymbol{\eta_{\beta}}$	First composition-based blocks in block β

Parameters

$\gamma_u^{L/U}$	Limits on period-period changes in crude feed flows to CDU *u*
$\theta_{ku}^{L/U}$	Limits on blending index for crude property *k* in the feed to CDU *u*
ρ_c	The density of crude *c*
θ_{kc}	Specification index for property *k* of crude *c*
NZ	The number of terms in the first summation
$xt_{ic}^{L/U}$	Limits on the composition of crude *c* in tank *i*

$FTU_{iu}^{L/U}$	Limits on the amount of crude charge per period from tank *i* to CDU *u*
$V_i^{L/U}$	Limits on crude inventory of tank *i*
$xc_{cu}^{L/U}$	Limits on the composition of crude *c* in feed to CDU *u*

Binary Variables

XP_{pt}	1 if parcel *p* is connected for transfer during period *t*
XT_{it}	1 if tank *i* is connected to receive crude during period *t*
Y_{iut}	1 if tank *i* feeds CDU *u* during period *t*

Continuous Variables

FTU_{iut}	Total amount of crude from tank *i* to CDU *u* during period *t*
$FCTU_{iuct}$	The amount of crude *c* from tank *i* to CDU *u* during period *t*
V_{it}	Total amount of crude in tank *i* at the end of period *t*
VCT_{ict}	The amount of crude *c* in tank *i* at the end of period *t*
f_{ict}	The fraction of crude *c* in tank *i* at the end of period *t*
FU_{ut}	Total amount of crude fed to CDU *u* during period *t*
u_{iuct}^{+}	Positive slack variables
u_{iuct}^{-}	Positive slack variables
s_{ukt}^{-}	Slack variables for property specification index constraints
s_{ukt}^{+}	Slack variables for property specification index constraints

Chapter 4

Indices

p	Parcel
v	Vessel
i	Storage tank
j	Charging tank

u	Crude Distillation Units (CDUs)
c	Crude
k	Key components

Sets

P	Parcel set
V	Vessel set
I	Storage tank set
J	Charging tank set
U	Crude Distillation Units (CDUs) set
C	Crude set
K	Key component set
JP	Set of jetty parcels
SP	Set of VLCC parcels
PT	Set of pairs (parcel *p*, period *t*) such that *p* can connect to SBM line during *t*
PI	Set of pairs (parcel *p*, storage tank *i*) such that *i* may receive crude from *p*
IJ	Set of pairs (storage tank *i*, charging tank *j*) that *j* may receive crude from *i*
JU	Set of pairs (parcel *p*, CDU *u*) such that *j* can feed crude to CDU *u*
IC	Set of pairs (storage tank *i*, crude type *c*) such that I can hold *c*
JC	Set of pairs (charging tank *j*, crude type *c*) such that *j* can hold *c*
PC	Set of pairs (pair *p*, crude type *c*) such that *p* is the last parcel of *v*

Parameters

ETA_p	Expected time of arrival of parcel *p*
$FPT_{pi}^{L/U}$	Limits on the amount of crude transfer per period from parcel *p* to tank *i*
$FTU_{iu}^{L/U}$	Limits on the amount of crude charge per period from tank *i* to CDU *u*
$FU_{u}^{L/U}$	Limits on the amount of crude processed per period by CDU *u*

$xcs_{ci}^{L/U}$	Limits on the composition of crude *c* in storage tank *i*
$xks_{ki}^{L/U}$	Limits on the composition of key component *k* in storage tank *i*
$xcb_{cj}^{L/U}$	Limits on the composition of crude *c* in charging tank *j*
$xkb_{kj}^{L/U}$	Limits on the composition of key component *k* in charging tank *i*
xcp_{cp}	The composition of crude *c* in parcel *p*
D_u	Total crude demand per CDU *u* in the scheduling horizon
D_{ut}	Crude demand per CDU *u* in each period *t*
CP_{cu}	Margin ($/unit volume) for crude *c* in CDU *u*
COC	Cost (k$) per changeover
SSP	Safety stock penalty ($ per unit volume below desired safety stock)
SS	Desired safety stock (kbbl) of crude inventory in any period
SWC_v	Demurrage or Sea waiting cost ($ per period)
ETD_v	Expected time of departure of vessel *v*
ETU_p	Earliest possible unloading period for parcel *p*
PS_p	Size of the parcel *p*
NJ	Number of identical Jetties

Binary Variables

XP_{pt}	1 if parcel *p* is connected to SBM/jetty discharge line during period *t*
XT_{it}	1 if a tank *i* is connected to SBM/jetty discharge line during period *t*
Y_{iut}	1 if a tank *i* feeds CDU *u* during period *t*

0-1 Continuous Variables

XF_{pt}	1 if a parcel *p* first connects to the SBM/jetty during period *t*
XL_{pt}	1 if a parcel *p* disconnects from the SBM/jetty during period *t*
X_{pit}	1 if a parcel *p* and tank *i* both connect to the SBM line at *t*
YY_{iut}	1 if a tank *i* is connected to CDU *u* during both period *t* and (*t*+1)

CO_{ut} 1 if a CDU *u* has a changeover during period *t*

XSB_{ijt} 0 if storage tank *i* does not feed charging tank *j*

Continuous Variables

TF_p Time at which parcel *p* first connects to SBM/jetty for unloading

TL_p Time at which parcel *p* disconnects from SBM/jetty after unloading

FPT_{pit} Amount of crude transferred from parcel *p* to storage tank *i* during period *t*

FSB_{ijt} Amount of crude transferred from storage tank *i* to charging tank *j*

FB_{jt} Total amount of crude fed to charging tank *j* during period *t*

$FCSB_{ijct}$ Amount of crude *c* delivered by tank *i* to tank *j* during period t

FTU_{iut} Amount of crude that tank *i* feeds to CDU *u* during period *t*

FU_{ut} Total amount of crude fed to CDU *u* during period *t*

$FCTU_{iuct}$ Amount of crude c delivered by tank *i* to CDU *u* during period *t*

$VCST_{ict}$ Amount of crude *c* in storage tank *i* at the end of period *t*

VST_{it} Crude level in storage tank *i* at the end of period *t*

$VCBT_{jct}$ Amount of crude *c* in charging tank *j* at the end of period *t*

VBT_{jt} Crude level in charging tank *j* at the end of period *t*

fs_{ict} Composition (volume fraction) of crude *c* in tank *i* at the end of period *t*

fb_{jct} Composition (volume fraction) of crude *c* in tank *j* at the end of period *t*

DC_v Demurrage cost for vessel *v*

SC_t Safety stock penalty for period *t*

Chapter 5

Notation

Sets

BP Set of (*b*, *p*) pairs such that blender *b* can process product *p*

BJ Set of pairs (blender *b*, product tank *j*) such that blender *b* can feed product tank *j*

PJ Set of pairs (product *p*, product tank *j*) such that tank *j* can hold product *p*

JO Set of pair (product tank *j*, order *o*) such that tank *j* can deliver order *o*

OP Set of pair (order *o*, product *p*) such that order *o* is for product *p*

TK Set of pair (slot *k*, period *t*) such that slot *k* is in period *t*

Subscripts

i Blend component and its dedicated tank

b Blender

p Product

o Order

j Product tank

s Gasoline property specification

k Slot

t Period

Superscripts

U Upper limit

L Lower limit

Parameters

T_0 Time zero or the time at which slot 1 starts

VP_j^U Capacity of product tank *j*

H Scheduling horizon

RL_{bp}^L Minimum run length of blender *b* for product *p*

RL_b Maximum of the minimum blend run lengths for blender *b*

M_b Most volume that blender *b* can process during a slot

$F_b^{L/U}$	Limits on the processing rate of blender *b*
$\theta_{ps}^{L/U}$	Limits on the index for property *s* of product *p*
θ_{is}	Blend index of property *s* of component *i*
ρ_i	Density of component *i*
ρ_{max}	Maximum density among all products
$r_{pi}^{L/U}$	Limits on the fraction of component *i* in product *p*
DR_{jo}	Delivery rate of product tank *j* to order *o*
DR_j^U	Maximum cumulative delivery rate of product tank *j*
TQ_o	Amount of order *o*
DD_o^L	Earliest delivery time of order *o*
DD_o^U	Due date of order *o*
F_i	Constant feed rate of component *i* into its tank
$V_i^{L/U}$	Limits on the holdup in component tank *i*
c_i	Price ($ per unit volume) of component *i*
CB_b	Cost ($ per occurrence) of a transition on blender *b*
CT_j	Cost ($ per occurrence) of a transition in product tank *j*
DM_o	Demurrage ($ per unit time) for order *o*
R_{bk}	Rate of blender *b* in slot *k*
TCQ_{bk}	Volume processed by blender *b* during slot *k*
F_{it}	Constant feed rate of component *i* to its tank during period *t*
CRL_{bk}	Corrected run length of blender *b* at the end of slot *k*
CCQ_{bk}	Corrected volume processed by blender *b* during slot *k*
TCQ_{bk}	Total volume processed by blender *b* during the current run at the end of slot *k*

θ_{ist} Blend index for a property *s* of component *i* during period *t*

ρ_{it} Density of component *i* during period *t*

$\rho_i^{\max}$ Maximum possible density among all products during period *t*

N The number of products that are needed to process in blenders

N_b The number of blenders

Binary Variables

v_{bjk} 1, if blender *b* feeds product tank *j* ($0 < j \leq \boldsymbol{J}$) during slot *k*

u_{jpk} 1, if product *p* is stored in product tank *p* during slot *k*

z_{jok} 1, if product tank *j* is delivering order *o* during slot *k*

0-1 Continuous Variables

v_{b0k} 1, if blender *b* is idle during slot *k*

ue_{jk} 1, if product tank *j* switches products at the end of slot *k*

x_{bpk} 1, if blender *b* produces product *p* during slot *k*

xe_{bk} 1, if blender *b* ends its current run for a product during slot *k*

Continuous Variables

T_k Time at which slot *k* ends

SL_k Length of slot *k*

G_{bjk} Volume that blender *b* feeds product tank *j* during slot *k*

VP_{jk} Inventory in product tank *j* at the end of slot *k*

RL_{bk} Length of the current run of blender *b* at the end of slot *k*

Q_{bk} Volume processed in blender *b* during slot *k*

q_{ibk} Volume of component *i* used by blender *b* during slot *k*

CQ_{bk} Volume processed by blender *b* during the current run, if the run does not end during slot *k*

DQ_{jok} Volume of order *o* delivered by product tank *j* during slot *k*

d_o Demurrage ($) for order o

V_{ik} Inventory in component tank i at the end of slot k

TC Total operating cost ($)

F_{bk} Rate of blender b during slot k

Chapter 6

Notation

Sets

JP Set of pairs (product tank j, product p) such that tank j can hold product p

Parameters

NP The number of distinct products that must be processed by blenders during the scheduling horizon

B The number of blenders

Binary Variables

y_{ibk} 1, if component tank i feeds blender b during slot k

0-1 Continuous Variables

ze_{jok} 1, if product tank j ends current delivery run of order o in slot k

Continuous Variables

T_{qk} Time at which slot k on unit q ends

T_{ik} Time at which slot k on component tank i ends

T_{bk} Time at which slot k on blender b ends

T_{jk} Time at which slot k on product tank j ends

BL_{bk} Time for which blender b processes real products ($p > 0$) in slot k

ts_{jok} the time at which the delivery of order o by product tank j begins in slot k.

t_{ik} An intermediate point between $T_{i(k-1)}$ and T_{ik}

VC_{ik} Inventory in component tank i at the end of slot k

Chapter 7

Notation

Subscript

s	Scenario

Parameters

dt	The period at the beginning of which a disruption is informed
M	Big number

0-1 Continuous Variables

$PPXP_{pt}$	1, if the parcel-to-SBM/jetty connection changes
$PPXT_{it}$	1, if the SBM/jetty-to-tank connection changes
PPY_{iut}	1, if the tank-to-CDU connection changes

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

During the last century, the petroleum industry has risen from being relatively small to a position where whole economies are profoundly influenced by the need for and prices of petroleum products. The petroleum business involves many independent operations, beginning with the exploration for oil and gas and extending to the delivery of finished products, with complex refining processes in the middle. These processes turn crudes into a wide range of products including gasoline, diesel, heating oil, residual fuel, coke, lubricants, asphalt, and waxes. Unlike batch manufacturing industry such as food and pharmaceutical industries, petroleum refinery is typically a continuous process plant that has a continuous flow of materials going in and coming out. In recent years, globalization has made the refining industry an extremely competitive business characterized by fluctuating demands for products, ever-changing raw material prices, and incessant push towards cleaner fuels. Facing these stringent situations, refineries seek efficient managerial tools and apply new technology to maximize profit margins and minimize wastes simultaneously to improve their operations. The following sections briefly introduce refinery operations, the entire supply chain of petroleum industry, its managerial activities, etc.

1.1 Refinery Operations

Crude oil as the basic raw material of the petroleum industry is explored at different fields that are located in different countries all over the world such as Brazil and Middle East, and transported from these fields to refineries by vessels, trains, or oil pipelines for refining. After its arrival, crude is stored or mixed in tanks, then charged to the crude distillation unit (CDU) and is separated into several component streams (distillation cuts) such as light gases, propane, butanes, light naphtha, heavy naphtha, kerosene, light gas oil, heavy gas oil, vacuum gas oils and residue, whose boiling points lie within certain ranges e.g. 30℃-130℃, 130℃-270℃, 270℃-370℃, etc. Some of these streams are desirable, while others are undesirable. The undesirable fractions are either sent to the downstream units for further treatment and undergo specific unit operations and processes in separate units such as Fluid Catalytic Cracker (FCC) unit, hydrocracking (crackers), hydrotreating, reformers, alkylation and isomerization units to yield desirable products by chemically altering the hydrocarbon molecules, splitting them or removing sulfur for instance. The desirable products have a wide range of physical properties such as density, viscosity, sulfur content, pour point, flash point, Reid vapor pressure (RVP), vanadium and nickel content. On their own, these desirable products may not be suitable for commercial use, but when blended together or with those desirable streams in various ways, they form final products, which are known as Liquefied Petroleum Gas (LPG), gasoline, diesel, kerosene, etc. These final products are stored in the corresponding product tanks and then delivered to the customers by trains, trucks, pipelines or ships.

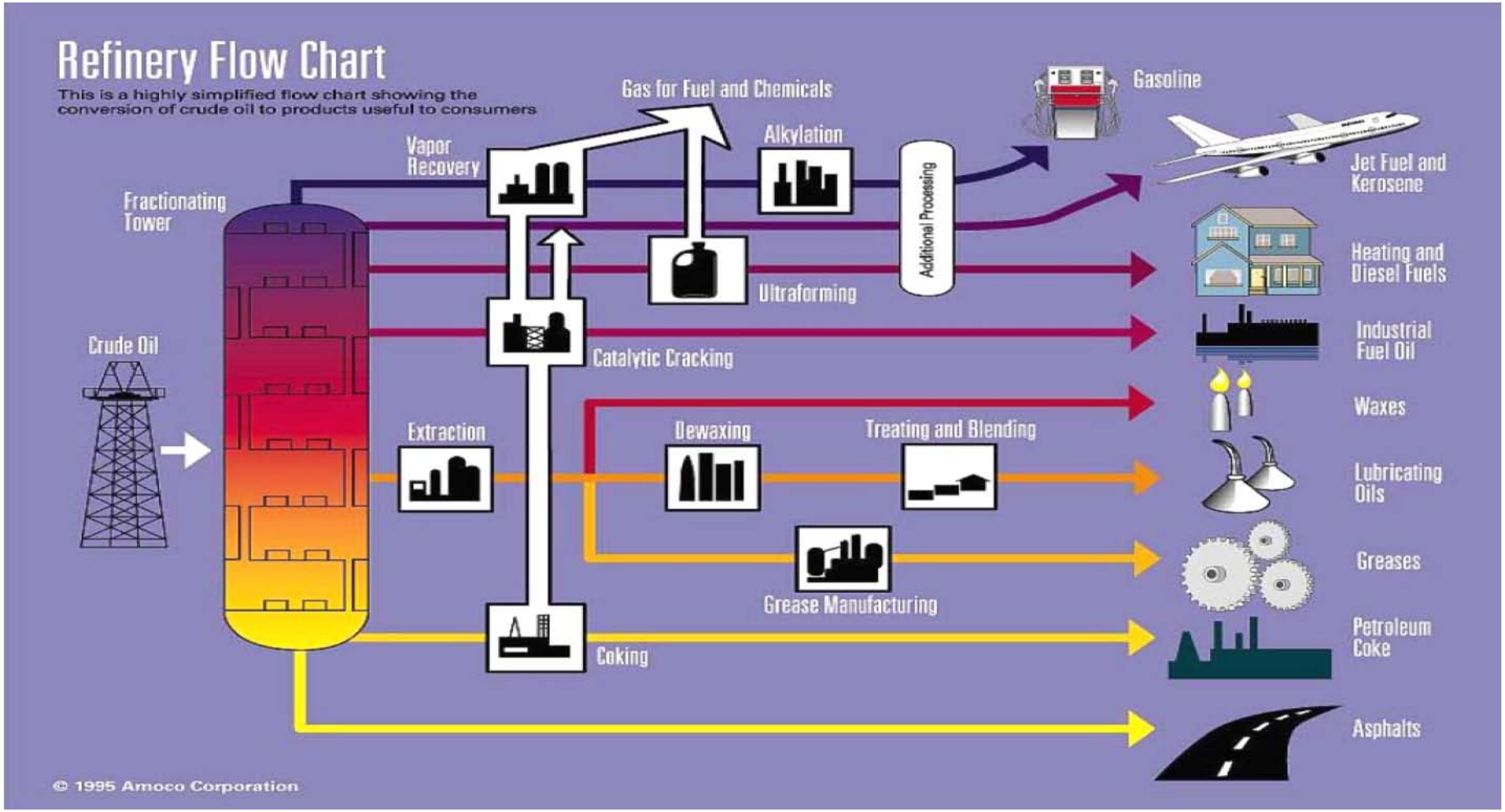

Figure 1.1 A simplified configuration of the petroleum industry
(Http://www.energy.ca.gov/oil/refinery_flow.html)

Additionally, undesirable streams may be sold off or used as low-cost fuels. Figure 1.1 shows a general configuration of the petroleum industry. The entire industry involves crude storing and mixing tanks, crude distillation units (CDU), vacuum distillations units (VDU), catalytic reforming units, fluid catalytic/hydro cracking units, hydro treaters, visbraker/delayed coker units and off-site storage/blending facilities to store/process the finished products/intermediate streams.

1.2 The Supply Chain of Refinery

Figure 1.2 shows a schematic of a typical petrochemical supply chain (Srinivasan et al. 2006). Crude oil is first produced from either ground fields or offshore platforms. After pretreatment and storage, it is transported via supertankers like VLCCs (Very Large Crude Carriers) and small vessels such as single-parcel vessels to various refineries around the world, unloaded through SBM (Single Buoy Mooring), SPM (Single Point Mooring), or jetty pipelines, stored and blended in storage or charging tanks, or both, and charged to CDU for processing. It is then converted into a variety of intermediate bulk chemicals that are used as feeds to the petrochemical plants globally and consumer products such as fuels that are used in aviation, ground transport, electricity generation, etc. Thus, a refinery supply chain involves three manufacturing centers, namely the oil fields & platforms, and the petroleum refineries, that are surrounded by a host of logistics services in the forms of storage, transportation, distribution, packaging, etc (Srinivasan et al. 2006).

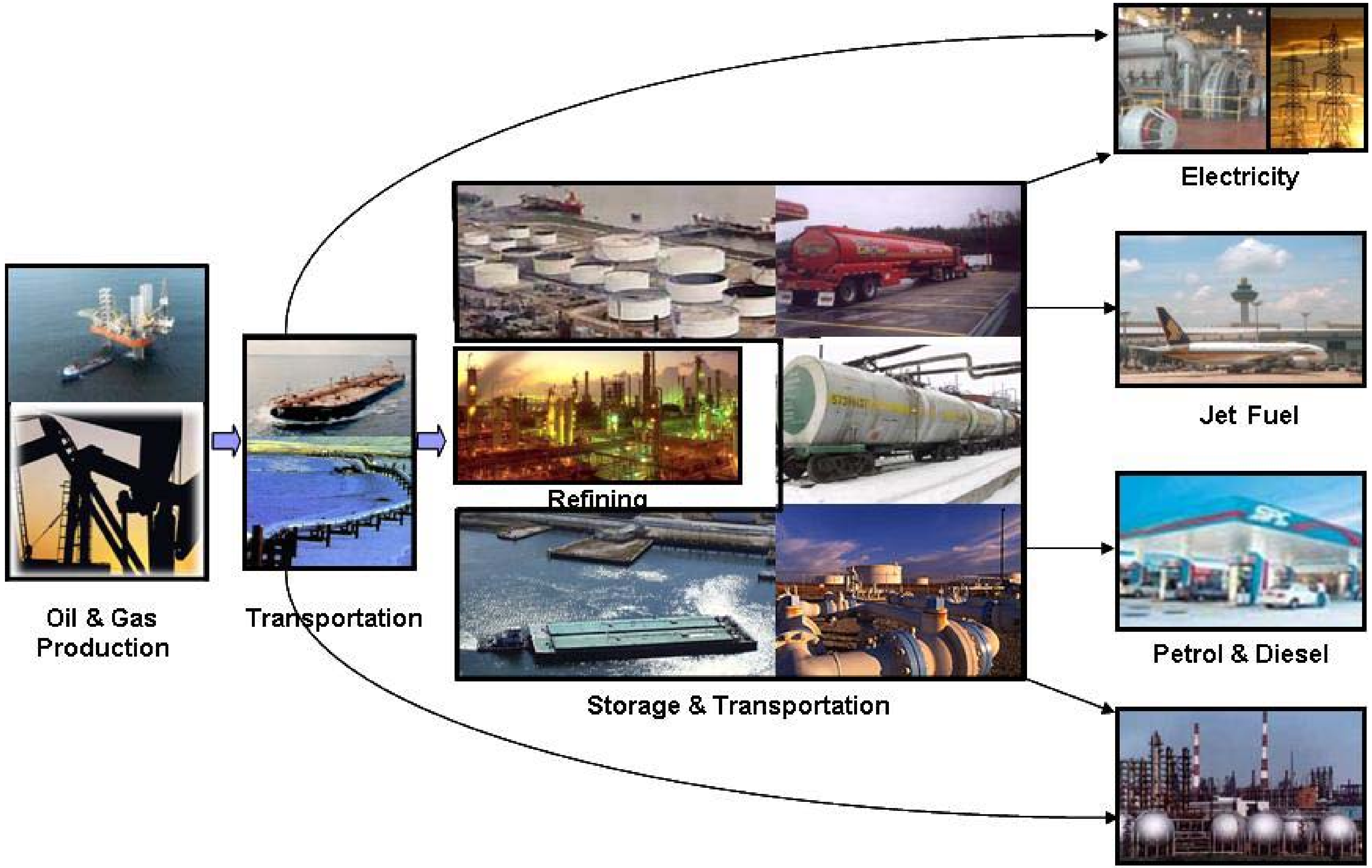

Figure 1.2 Schematic of a typical petrochemical supply chain (Srinivasan et al. 2006)

1.3 Need for Management in Petroleum Industry

In the past, the petroleum industry has succeeded by creating markets and supplying them with suitable products. Today, globalization has become an irreversible trend with the rapid development of Information Technology and the decreasing costs of communication and transportation. Furthermore, new market dynamics such as the proliferation of product grade specifications, the drive for lower inventories, increased capital investments, environmental regulations, refinery retail and transportation asset rationalization, and higher market volatility are all adding to the complexity. To survive financially, refineries have to seek efficient managerial tools and apply new technologies.

In the refining processes, one key challenge is how to best operate the plant under different feed compositions, production rates, energy availability, ambient conditions, fuel heating values, feed and product prices, and many more factors that are changing all the time. Undesirable changes may lead to off-spec products, reduced throughputs, increased equipment wear and tear, uncertainty and more work. Past experience can achieve operating targets in some situations. However, in order to fully exploit the complete spectrum of how the plant can be operated to maximize operating profit, efficient management using advanced computer-aided techniques is also needed in the competitive environment.

1.4 Supply Chain Management of Petroleum Industry

The main managerial activities of a refinery can be divided into three layers: planning, scheduling and unit operations. Optimization plays an important role in managing the oil refinery. Oil refineries have used optimization techniques for a long time, specifically Linear Programs (LPs) for the planning and scheduling of process operations. Planning and scheduling primarily differ in terms of the time frames involved. Planning is generally undertaken for longer time horizons such as months or years and includes management objectives, policies, etc. besides immediate processing requirements. It represents aggregated objectives and usually does not include finer details. The main objective of planning is to maximize the gross refinery profit margin while meeting demand forecast and efficiently using facility resources such as plant capacities, utilities, and manpower. Optimal plan produced in the planning stage forms the basis for scheduling. While scheduling defines the detailed specification of each unit at each time over a short horizon ranging from shifts to weeks to satisfy the targets set at the planning stage, the objective of scheduling is implementation of the plan subject to the variability that occurs in the real world. This variability can be in feed stock supplies, quality, production process, customer requirements or transportation.

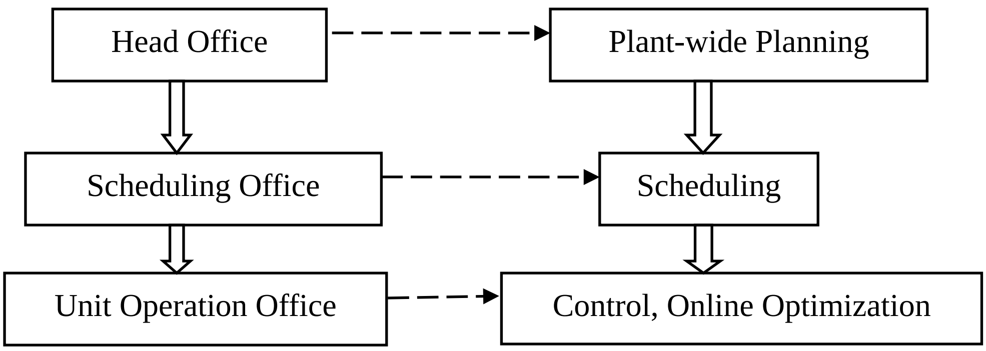

Figure 1.3 A configuration of managerial activities in a refinery (Li, 2004)

The managerial activities in a refinery can be described as shown in Figure 1.3.

First of all, the plant-wide plans issued by the head office who considers plant-wide factors such as market condition, raw material availability, operation capacity and so on are sent to the scheduling office as guidelines. These plans mainly handle business decisions such as which units to operate, which raw materials to process and which products to produce, etc. The objective of planning is to obtain an optimal operation strategy that can maximize the total profit. After analyzing the plan, the scheduling office determines detailed operation schedule for each unit that is to be executed in a plant within the scheduling horizon. The objective of scheduling is to seek a feasible operation strategy that meets the planning requirements while maximizing the total profit. These feasible schedules are sent to the unit operation office as the operation guidelines so that the operators can control the unit operations rigorously to realize the scheduling objective.

With an effective supply chain management, the refinery can reduce costs of purchased crude oils and chemicals, feedstock, their quality issues, optimize and manage crudes and product inventory, increase plant yields, improve visibility of scheduling and inventories across the supply chain, and satisfy customers, etc.

1.5 Research Objectives

This research focuses on refinery planning and scheduling operations. While refinery-planning problems have been extensively studied and are considered well developed, as discussed in the next Chapter, scheduling problems can involve enormous considerations for conceiving an optimum schedule taking into account all the factors. In the most general form, the problem is too complicated to formulate mathematically, let alone solving and obtaining an optimum schedule. And even if the problem is formulated, a simplistic approach of enumeration of alternatives sounds preposterous because of the number of possibilities that might exist (combinatorial nature of the problem). A lot of research has been undertaken in this area in the past decade with a focus on the development of exact and approximate methods to solve short-term scheduling problems. Therefore, this research project focuses on real, large-scale scheduling problems during refinery operations. Furthermore, some disruptions may be unavoidable during the refinery operations. The focus of this research is also to take into account these disruptions, while developing optimal schedules to make them robust and efficient.

With this, the objectives of this research work are to (1) Develop efficient mathematical models for scheduling refinery operations such as crude oil operations, and product blending and distribution operations, which incorporate many real operation features; (2) Develop new robust and efficient algorithms, for instance, decomposition algorithm to solve the developed models, especially for real large-size industrial problems; (3) Define and evaluate robustness and Develop robust schedules for refinery scheduling operations in the presence of uncertainties.

1.6 Outline of the Thesis

This thesis includes eight chapters. After a brief introduction in Chapter 1, Chapter 2 presents a detailed literature review on planning and scheduling of refinery operations.

Based on this detailed review, several gaps in the existing work are summarized.

In Chapter 3, the first part of scheduling operations in a refinery: crude oil scheduling is presented in detail. Some deficiencies of the existing work in the literature are overcome. Some strategies and ways are developed to improve robustness, quality, and solution speed of the algorithm in the literature, and estimate solution quality by means of a tight upper bounding strategy. Twenty-four large simulated examples are used to demonstrate numerically the robustness and effectiveness of the improved algorithm. In addition, the most important nonlinear crude properties that are crucial to crude distillation and downstream processing are identified and incorporated into the problem formulation.

Chapter 4 extends the enhanced formulation and developed solution strategies in Chapter 3 to handle crude oil scheduling in an in-land refinery, in which crudes are stored or stored and blended in storage tanks and blended in charging tanks. Three policies of crude concentrations are analyzed in storage and charging tanks: 1) constant crude composition in storage tanks but variable in charging tanks, 2) variable crude composition in both storage and charging tanks, 3) variable crude composition in storage tanks but prefixed in charging tanks.

In Chapter 5, scheduling of gasoline blending and distribution operations is addressed. A global slot-based continuous-time formulation for simultaneous treatment of recipe, blending, scheduling, and distribution is developed. A schedule adjustment procedure is proposed to solve the nonlinearity arising from ensuring constant blending rates of blenders during bend runs. In addition, nine nonlinear important properties for gasoline such as octane number, Reid vapor pressure (RVP), sulfur, benzene, and aromatics content are also accounted for.

In Chapter 6, a novel unit-slot based continuous time formulation is developed to treat the same problem presented in Chapter 5. The novel formulation incorporates all real-life operation features of the model developed in Chapter 5. The basic formulation is extended to multi-period scenario.

Chapter 7 first uses reactive approach to address several disruptions during scheduling of crude oil operations and compare with the heuristic method of Arief et al. (2007a). Then, schedule robustness is defined as schedule effectiveness, predictability and rescheduling stability. Based on this, schedule robustness index (RI) is defined. A procedure is proposed to evaluate the robustness of a schedule. A scenario-based formulation is developed to obtain robust schedules for demand uncertainty.

Finally, conclusions and recommendations for future research are summarized in Chapter 8.

CHAPTER 2

LITERATURE REVIEW

Chemical manufacturing processes can be classified into two types: batch or continuous based on their operation modes. Planning and scheduling problems for batch chemical plants have been extensively addressed in the literature (Reklaitis, 1992; Floudas and Lin, 2004a,b). However, planning and scheduling problems associated with semicontinuous/continuous process have received less attention. A petroleum refinery is a typical multiunit and multiproduct integrated continuous plant. As discussed in Chapter 1, this research project mainly focuses on planning and scheduling of refinery operation. Therefore, the work so far related with refinery planning and scheduling problems in the literature is first reviewed as follows.

2.1 Planning in Refinery

Refinery planning problems have been studied since the introduction of linear programming (LP) in 1950s (Simon and Azma, 1983; Bodington and Baker, 1990; Zhang et al., 2001). Symonds (1955) and Manne (1956) applied linear programming techniques to the long-term supply and production plan of crude oil and product pooling problems. Bodington and Baker (1990) presented a review on the history of Mathematical Programming (MP) in the petroleum industry. They forecasted that non-linear optimization would gain more wide use, especially in the field of operational planning. Sear (1993) developed a linear programming network model for planning the logistics of a downstream oil company. The model involved crude oil purchase and transportation, processing of products and transportation and depot operation. Coxhead (1994) identified several applications of planning models for refinery and oil industry, including crude selection, crude allocation to multiple refineries, partnership models for negotiating raw material supply and operations planning. Dempster et al. (2000) applied a stochastic programming approach to planning problems for a consortium of oil companies. Iakovou et al. (2001) developed a strategic mixed-integer linear programming (MILP) planning model to decide optimal transportation route for ships that carried crudes or petroleum products incorporating risk assessment.

Most refining processes are nonlinear. Dealing with these nonlinear processes is very challenging. Bodington (1992) pointed out that systematic methodologies were lacking for dealing with nonlinear relations. Despite that, progress in nonlinear programming in the nineties has been achieved. Ramage (1998) presented that nonlinear programming (NLP, MINLP) could be a necessary tool for the refineries of the 21st century. Fieldhouse (1993) studied the pooling problem and solved simultaneously the mass balance equations and quality relations with successive linear approximation. More et al. (1998) presented a planning model for diesel production in which some properties were determined by using nonlinear correlations. The whole problem was formulated as a nonlinear program. Pinto et al. (2000) and Joly et al.

(2002) developed a super structure model for production planning integrating models for processing units such as CDU, FCCU, etc. However, they used a linear model for FCC. The linear model for FCC may not generate accurate yields and properties of FCC distillates because of the nonlinearity of FCC behavior (Decroocq, 1984). Besides modeling processing units like CDU, FCCU, etc, Neiro and Pinto (2004) proposed particular frameworks for modeling storage tanks and pipelines. Li (2004) proposed a plant wide planning model integrating CDU and FCCU model, which were the most complicated and important units in the refining area. In modeling CDU he started from the ASTM boiling ranges of CDU fractions obtained from refineries, CDU designers or literatures such as Watkins (1979), converted the ASTM boiling ranges of CDU fractions into True Boiling Points (TBP) using correlations developed by Watkins (1979) or other correlations presented by Arnold (1985). According to True Boiling Point (TBP) curve of crude oil and different operation modes, he got the range of weight transfer ratios (WTR) for each fraction used to determine the flow rate of each fraction. In modeling FCCU they calculated different weight transfer ratios of FCC fractions corresponding to the conversion level until the conversion level reaches its upper limit. According to these data, FCC fraction weight transfer ratios and FCC conversion levels are correlated and an equation for each FCC fraction weight transfer ratio versus FCC conversion level was now obtained and could be used in refinery planning model to optimize the FCC conversion level.

Additionally some commercial software such as RPMS (Refinery and Petrochemical Modeling System), PIMS (Process Industry Modeling System), GRTMPS (Haverly Systems), Aspen Plus, PRO/II (SimSci-Esscor) and DESIGN II™ (ChemShare) have been also developed for refinery production planning and are commonly used in the petroleum industry. OMEGA (Dewitt et al., 1989) and StarBlend (Rigby et al., 1995) were developed to address planning problems of product blending operation.

Therefore, planning technology can be considered well developed and relevant progress may not be expected (Pelham and Pharris, 1996).

2.2 Scheduling in Refinery Operation

So far, very few models have been developed and applied for scheduling refinery operations compared with refinery planning problems (Ballintjin, 1993; Li and Hui, 2003). Even though some scheduling tools for refinery exist in the market, the state-of-the-art of this technology cannot be considered as mature a solution as that for planning (Magalhaes et al., 1998). This is because scheduling problems using mixed integer optimization models to explicitly model the discrete decisions are very difficult to solve especially for real large-scale industrial problems. Furthermore, very few optimization-based formulations are applied to the scheduling of continuous multi-product plants especially refineries, as opposed to batch plants (Reklaitis, 1992; Pinto and Grossmann, 1995). Ballintjin (1993) also pointed out the low applicability of models based only on continuous variables after comparing continuous and mixed-integer linear formulations. Therefore, scheduling of refinery operation has been receiving more and more attention recently.

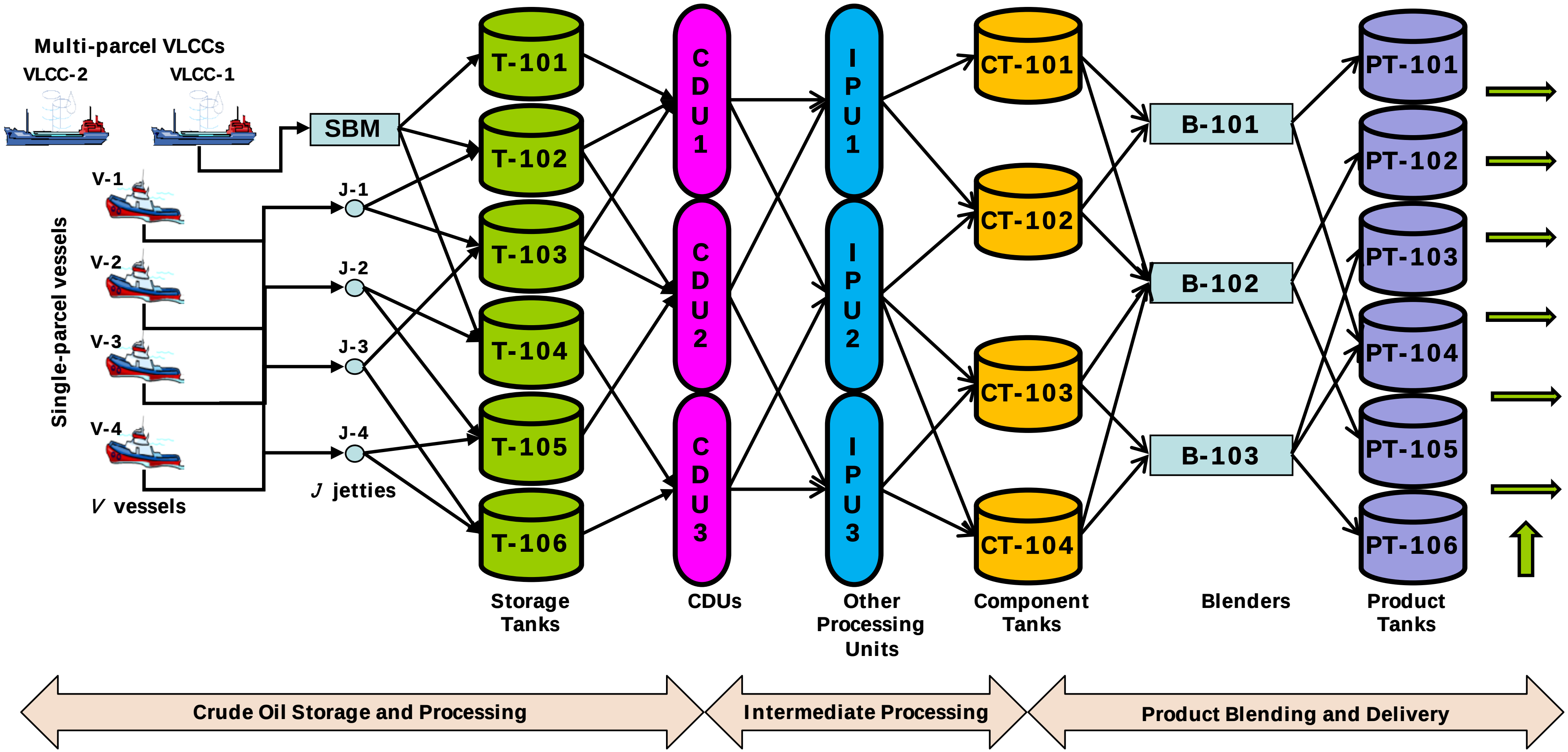

Figure 2.1 Schematic of the overall refinery operation

The overall refinery operation (Figure 2.1) involves crude oil storage and processing, intermediate processing, and product blending and distribution. During crude oil scheduling operation, crude schedulers react to the timing of crude arrivals, determine which tank the crude should be placed in, blend crudes as needed to meet targets for yields and qualities of the crude unit, and determine which tank charging to which CDU, in what amount, and at what time. Intermediate processing and production scheduling is concerned with the operations of major units such as FCCU (Fluid Catalytic Cracking unit), and inventories between the units. The main objective is to have proper control of intermediate inventories. Scheduling of product blending and distribution is concerned with determining the timing, and amounts of blends, selecting components for blends to meet quality specifications of blends, and defining the activities required to move the products out of the refinery while ensuring the inventory control, and customer satisfaction. Next, the details on each of those three scheduling processes are presented.

2.2.1 Crude Oil Scheduling

Scheduling of crude oil operation is an important and complex routine task in a refinery. It involves crude oil unloading, tank allocation, storage and blending of different crudes, and CDU charging. Crude oil costs account for about 80% of refinery turnover, so selecting a cheaper crude oil can have a significant impact on profit margins. However, some crudes may lead to processing problems and have to be mixed with other crudes to meet the operational requirements. Moreover, most refineries have unsteady supply of crude oil and face ever-changing raw material prices and fluctuating demands for products. Optimal crude oil scheduling aims to maximize the profit by using cheaper crudes, minimizing the crude changeovers, avoiding ship demurrage and managing the crude inventory reasonably. However, mathematical modeling of the blending of different crudes in storage tanks results in bilinear terms, which turns the whole problem into a difficult, nonconvex, mixed integer nonlinear program (MINLP).

Shah (1996) presented a mathematical model for crude oil scheduling as an MILP based on a discrete-time representation. The problem was decomposed into two sub-problems: the upstream and downstream problems. The upstream problem consists of portside tanks and offloading and the downstream problem includes allocation of charging tanks and CDU operation. The downstream problem was solved first and the upstream problem was solved subsequently. However, the proposed model lacked many real features such as jetties, multiple parcels vessels, brine settling, crude oil segregation, two tanks feeding CDU and so on. Furthermore, each tank was allowed to store at most one type of crudes. Thus, no crude blending occurred. The objective was to minimize the tank heel not the operating cost or total profit.

Magalhaes and Shah (2003) reported a continuous-time-grid model for the same problem. While the details of algorithm and model were not presented, they incorporated some real-world operational rules such as crude segregation, no simultaneous receipt and delivery of crude by a tank, brine settling and pipeline peak time flow regime. The objective was to minimize the deviation from the planned operation, not the operating cost or profit margin.

Lee et al. (1996) also addressed crude oil scheduling involving unloading from vessels to storage tanks via one jetty, transferring from storage tanks to charging tanks and charging schedule for CDU. Crude oil blending was allowed in charging tanks. They applied bilinear equations to model this mixing operation. However, they used the reformulation linearization technique (RLT) to convert the bilinear terms into linear forms which led to composition discrepancy as shown by Li et al. (2002). Composition discrepancy means the amounts of individual crudes delivered from one tank to one CDU are not proportional to the crude composition in the tank. In addition, they did not consider some real-life operational features such as multiple jetties, brine-settling, multiple-parcel vessels, multiple tanks feeding one CDU, one tank feeding multiple CDUs, etc.

Li et al. (2002) realized the composition discrepancy occurring in Lee et al. (1996) due to the mixing of different types of crudes. They also pointed out that the RLT technique from Quesada and Grossmann (1995) could not eliminate composition discrepancy. They incorporated some new features such as multiple jetties and two tanks feeding one CDU at one time compared to Lee et al. (1996) and proposed an iterative solution algorithm to solve this bilinear problem. They iteratively solved one mixed-integer linear programming (MILP) model and one NLP. However, their algorithm had to solve NLP problems and failed to find feasible solutions even when one existed as shown by Reddy et al. (2004b). They also attempted to reduce the overall number of discrete decision variables by decomposing tri-index discrete variables to bi-index discrete variables. Unfortunately, their variable decomposition may lead to some problems as shown by Reddy et al. (2004b) and they did not consider some real life features such as multiple-parcel vessels, brine settling and one tank feeding multiple CDUs at one time. Moreover, they counted changeovers twice in their formulation as pointed out by Reddy et al. (2004b).

Rodrigo Mas and Pinto (2003) developed a continuous-time model based on event point for short-term crude oil scheduling problems in a distribution complex that contains ports, refineries, and a pipeline infrastructure capable of transferring oil from ports to refineries. In their model, they allowed only one type of crude oil in each tank. Thus, crude oil mixing did not exist in their problem. They incorporated some real features into their model including brine settling, not simultaneous load/unload operation for each tank and crude oil segregations.

Jia and Ierapetritou (2003) also reported the problem of crude oil short-term scheduling operations based on a continuous-time representation. They used the bilinear equations to model crude oil mixing similar to Lee et al. (1996). However, they used CPLEX 7.5 as MIP solver and CONOPT as NLP solver to solve the MINLP problem directly. As with Li et al. (2002), NLP solver may not find a feasible solution. Besides, they did not consider the changeover costs arising from crude class or tank changes and incorporate several operational features such as multiple-parcel vessels, multiple tanks feeding one CDU, single tank feeding multiple CDUs, brine settling, multiple jetties and so on.

Moro and Pinto (2004) addressed the problem of crude oil inventory management in a real world refinery that receives crude oil through a pipeline. They developed two models based on continuous-time representation. The first one relied on a MINLP model, the other MILP by adopting the discretization procedure obtained from Voudouris and Grossmann (1992). However, in their formulation they fixed the start and end times for parcel unloading, which excluded the uncertainty in parcel unloading and assumed that the space for crude receipt was always available. In their discretization procedure, they used the fraction of tank volume to create the output stream that led to the discrete values for the flow rate from tanks to CDU and introduced another tri-index binary variable to decide which fraction was extracted from storage tank. This led to more binary variables and increased the computation difficulty. In addition, they did not guarantee that each CDU would always be fed. The objective function did not contain any quality parameter of crude, so optimal allocation of crude mix feed could not be achieved. Besides, the model seemed tailored for single distillation unit as most of equations relating flow and allocation constraints were defined specifically for a single CDU. Thus, it may be hard to extend to more CDUs. They also lacked some real operational features such as constraints on the total flow rate of any CDU, the key components of any CDU, crude oil segregation and single tank feeding multiple CDUs.

In order to remove the errors occurring in the literature and incorporate more real life operation features, Reddy et al. (2004b) presented a novel solution approach for crude oil operations optimization including unloading crude oil from vessels to storage tanks, mixing crude oil in the storage tanks and charging CDUs with storage tanks.

They incorporated many real-life operational features including SBM, SPM, multiple jetties, multiple-parcel vessels (VLCCs), multiple tanks feeding one CDU at one time, one tank feeding multiple CDUs at one time, brine settling and tank-to-tank transfers. Although their formulation was based on discrete time representation, it had some continuous-time features by allowing two parcels to connect and disconnect in a given period. Furthermore, they decomposed the tri-index decision variables and counted the changeover correctly compared to Li et al. (2002). Most importantly, they proposed a novel solution algorithm to solve the composition discrepancy and obtain the correct concentration in each tank. The new algorithm required solving a series of MILP to avoid solving MINLP or NLP. In their algorithm, they defined the two types of blocks based on tank compositions. One is that tank compositions keep constant. The other is that tank compositions change because of receiving crudes from vessels. They started from the first block with known and constant tank compositions and solved block by block. Once one block was solved, they fixed all the variables and proceeded to the next block. Therefore, as iterations proceeded the problem size was reduced and could be solved with smaller relative optimality gap. Because they fixed all the variables in previous blocks when solving the following blocks, their algorithm may fail to find a feasible solution. Reddy et al. (2004a) developed a continuous-time formulation for crude oil scheduling operations. In their model, they incorporated some real features such as SBM, SPM, multiple-parcel vessels (VLCCs), multiple tanks feeding one CDU at one time, one tank feeding multiple CDUs at one time and brine settling. They also used the algorithm of Reddy et al. (2004b) to avoid composition discrepancy.

Kelly and Forbes (1998) determined the allocation of feed stocks to storage tanks when storage tanks were fewer than feed stocks. Their aim was to ensure maximum flexibility for downstream process operation while keeping the feed storage facilities to a bare minimum.

Furman et al. (2007) developed a robust event-based continuous time formulation for tank transfer scheduling. Their model generally and more robustly handled the synchronization of time events with material balances than previous proposed models in the literature (Jia et al., 2003; Reddy et al., 2004a). Moreover, they modeled the input and output of a tank within the same time event, which potentially reduced the number of binary variables and provided a significant reduction in combinatorial complexity. They used their model to solve the same problem with Jia et al. (2003) and used NLP solver to solve bilinear items. Thus, their model also lacked many real operational features and may lead to infeasibility, which is the same as that of Jia et al. (2003).

From what has been discussed above, crude oil scheduling is a nonconvex MINLP problem because of the mixing of different types of crudes. Besides the above special algorithms, several general global optimization methods for MINLP problem also exist in the literature such as Generalized Benders Decomposition (GBD) (Geoffrion, 1972), Outer Approximation/Equality Relaxation method (OA/ER) (Duran and Grossmann, 1986a, 1986b; Kocis and Grossmann, 1987, 1988), Generalized Cross Decomposition (Holmberg, 1990), and Simplicial Approximation Method (Goyal and Ierapetritou, 2004). All those methods decompose MINLP problem into an MIP problem (master problem) and a NLP problem in which the MIP problem provides a lower bound and the NLP problem produces an upper bound for the whole MINLP problem until some criterion is satisfied. Although those decomposition-based heuristic methods can solve some nonconvex MINLP problem, they may fail to give a feasible solution for many cases even when one exists as asserted by Kelly and Mann (2003a,b). In addition, two general commercial solvers such as DICOPT (Grossmann, 1995) and BARON (Sahinidis, 2002) can be used to solve general MINLP problems. However, they also fail to solve some problems and are horribly slow in solving some problems.

Apart from the optimization-based model presented in the above papers, simulation-based approach for crude oil scheduling is also reported in the literature such as Paolucci et al. (2002) and Chryssolouris et al. (2004). Paolucci et al. (2002) proposed a simulation based decision support system only for allocating crude oil supply to port and refinery tanks. Chryssolouris et al. (2004) integrated refinery short-term scheduling involving the unloading of crude oil to storage tanks, the transfer and blending from storage tanks to charging tanks and crude oil distillation units, and the arrangement of the temperature cut-points for each distillation unit. Although simulation-based approaches can use heuristic knowledge, support what-if analysis and evaluate the performance of alternative solutions, they largely rely on the independent variables specified by users and cannot ensure optimal schedules.

2.2.2 Scheduling of Intermediate Processing

The downstream of crude oil tank area is the refinery production area. Most of the production activities are performed in the refining area where main compositions in crude oil are distilled into lighter fractions and some of these fractions are upgraded and purified to produce intermediate products.

Scheduling of intermediate processing is also an important aspect, but it has received considerably less attention than production planning because of its complexity and limitations in computing technology. Joly and Pinto (1999), Pinto and Joly (2000), Pinto et al. (2000), Joly et al. (2002), and Joly and Pinto (2003) developed an MINLP model for scheduling of fuel oil (FO) and asphalt production. They first modeled the problem as MINLP and then used rigorous linearization of viscosity balance constraints to transform the model to MILP, which provided a rigorous lower bound. Pinto and Moro (2000), Pinto et al. (2000) and Joly et al. (2002) developed a slot-based continuous time MILP formulation for LPG production scheduling.

Lundgren et al. (2002) addressed a production-scheduling problem in an oil refinery company that consists of one distillation unit and two hydro-treatment units, regarding the scheduling of operation modes. They used linear model for those units instead of nonlinear equations, which might lead to inaccurate results. Persson et al. (2004) presented a tabu search heuristic for scheduling the production at an oil refinery. The tabu search heuristic includes the use of variable neighborhood, dynamic penalty and different tabu lists.

Doganis et al. (2005) developed a discrete-time MILP formulation for the optimal scheduling of the lubricant production plant. They considered the situation where storage tanks were not dedicated for each particular lubricant. In other words, each storage tank may hold several different lubricants during the entire scheduling horizon. Luo and Rong (2007) proposed a hierarchical approach with two decision levels for refinery production scheduling problems. The upper level was based on discrete-time optimization formulation which modeled the operations related to processing units and pipelines and was used to decide sequences and timings of the operation modes of these unites. The detailed material movements from/to individual tanks were not taken into account in this level. The lower level was an intelligent simulation system used to control the detailed material movement according to heuristics and operation rules.

2.2.3 Scheduling of Product Blending and Distribution Operation

The purpose of blending processes is to mix several components of different properties to obtain a product meeting the given specifications, so it can be sold on the market or can be processed further, blending processes can be characterized by the following key features (Glismann and Gruhn, 2001a,b):

1. Blending components with different properties are stored in the intermediate storage tanks.
2. Different components are blended according to recipes.
3. The blends are stored in tanks and/or are delivered directly to the customers.
4. Similar products can be blended by applying entirely different recipes.

Scheduling product blending and distribution operations is a critical and complex routine task involving tank allocation, component mixing, blending, product storage, and order delivery. A common field for this process is production and distribution of gasoline, because gasoline is one of the most profitable products of a refinery and can account for as much as 60-70% of total profit (Rigby et al., 1995; Jia and Ierapetritou, 2003). A refinery typically blends several gasoline cuts or fractions from various processes to meet its customer orders of varying specifications. However, this process involves nonlinear blending and complex combinatorics, and can easily result in suboptimal schedules and costly quality give-aways. The large numbers of orders, delivery dates, blenders, blend components, tanks, quality specifications, etc. make this problem highly complex and nonlinear. Optimal scheduling using advanced techniques of mixed-integer programming are imperative for avoiding ship demurrage, improving order delivery and customer satisfaction, minimizing quality give-aways, reducing transitions and slop generation, exploiting low-quality cuts, and reducing inventory costs. Therefore, scheduling of gasoline blending and distribution is very crucial.

The early work related to this problem focused on the optimal blending of various intermediate fractions from the refinery and some additives to meet product quality specifications. Dewitt et al. (1989) developed a decision support system named OMEGA (Optimization Method for the Estimation of Gasoline Attributes) for gasoline blending operations. They used detailed nonlinear models for predicting gasoline attributes. Rigby et al. (1995) improved OMEGA to a multi-period blending model named StarBlend. Some commercial tools such as Aspen Blend™ and Aspen PIMS-MBO™ also address product-blending problems. However, they largely restrict themselves to determining product recipes for the stand-alone blending problem and fail to provide an integrated solution that considers resource allocation and temporal decisions as well.

Pinto et al. (2000) developed an MILP model for scheduling refinery production involving blending operations. While they considered transitions in blending pipelines, they did not integrate the distribution operations with blending, did not ensure constant blending rates, and did not enforce minimum run lengths (Karimi and Macdonald, 1997) for blend runs. In addition, they used linear blending correlations for key elements, while cannot handle most nonlinear product properties such as sulfur. Joly and Pinto (2003) also developed an MINLP formulation for scheduling fuel oil/asphalt production. This also involved blending operation, but they made the same assumptions as Pinto et al. (2000).

Glismann and Gruhn (2001a,b) developed a two-level decomposition approach to integrate short-term scheduling with blend recipe optimization. They first solve a nonlinear programming (NLP) problem to obtain product recipes and quantities, and then use mixed-integer linear programming (MILP) to obtain a schedule for the blending operation for the fixed product recipes. Jia and Ierapetritou (2003b) proposed a continuous-time event-based MILP formulation for scheduling gasoline blending and distribution operations simultaneously. They allowed features such as multi-purpose product tanks, one product tank delivering multiple orders, and multiple product tanks delivering one order. However, their model lacked other key operation features such as multiple parallel non-identical blenders, variable recipes, and product specifications. Moreover, they allowed a blender to feed multiple product tanks at the same time, which may not be a normal practice. However, more importantly, their formulation gives infeasible solutions, and allows a product tank to hold several products at a time (See Appendix A for details).

Recently, Mendez et al. (2006) presented both discrete-time and slot-based continuous-time models for the simultaneous optimization of blending and short-term scheduling. Their model allowed parallel identical blenders and determined optimal blend recipes. However, they employed nonlinear correlations for some product specifications, which resulted in a non-convex MINLP. To solve this MINLP, they proposed an iterative algorithm that first uses linear correlations to obtain component volume fractions in each blend, and computes the correction factor “bias” between the nonlinear and linear estimates of product specifications. Then, the algorithm used this “bias” to amend the linear correlations, until all product specifications meet their limits. Thus, their algorithm solved linearized MILPs. However, they did not integrate the distribution operations with blending, and did not ensure constant blend rates or minimum run lengths.

2.2.4 Scheduling of Product Transportation

Petroleum products such as diesel oil, liquefied petroleum gas (LPG) and gasoline can be sent to distribution centers and markets by road, railroad, vessel and pipeline. Among these transportation modes, pipeline transportation is the most reliable and economical mode for large amounts of liquid and gaseous products, because it can operate continuously compared to other modes (Sasikumar et al., 1997).

Bodin et al. (1983) emphasized that annual transportation costs of consumer goods exceeded US$ 400 billion in the past 20 years. In addition, large amounts of products need to be pumped over large distances and huge energy is consumed by booster stations. These aspects require oil companies to seek efficient scheduling and planning tools to reduce the cost of transportation and storage operation and increase customer satisfaction.

First of all, optimization techniques are used to assist distribution decisions in the literature. Rejowski and Pinto (2002 and 2003) addressed scheduling of a multiproduct pipeline system which was used to transfer large quantities of different products to distribution centers to meet the customer orders. They presented two MILP models based on discrete time representation. The first model (M1) assumed that the pipeline was divided into packs of equal size, whereas the second one (M2) relaxed this assumption. Rejowski and Pinto (2004) addressed the same problem as Rejowski and Pinto (2002 and 2003) and proposed a set of special constraints that minimized product contamination inside the pipeline and delivery cuts to improve the efficiency of the MILP model by Rejowski and Pinto (2002 and 2003). Magatao et al. (2004) developed an MILP based on uniform time discretization for scheduling an oil pipeline transporting different types of oil derivatives from a harbor to an inland refinery over a limited time horizon. Cafaro and Cerda (2004) developed a novel continuous time MILP formulation for scheduling a single pipeline transporting refined petroleum products from a unique oil refinery to several depots based on the problem proposed by Rejowski and Pinto (2003). More recently, Rejowski and Pinto (2008) proposed a novel continuous-time MINLP formulation to address the similar problem presented by Rejowski and Pinto (2003). Besides inventory constraints at the locations connected to the pipeline, interface detection, pipeline operation and demand satisfaction constraints, they also incorporated variable pumping flow and yield rates, which involved nonlinearities.

Other approaches have also been applied to similar problems. Zhao (1986) presents a dynamic programming approach to an oil distribution network through pipelines. Sasikumar et al. (1997) applied the beam search method to solve the scheduling problem of oil derivatives through a pipeline.

2.3 Integration in Petroleum Refinery

The above work only deals with one part of the overall refinery process. Detailed modeling, effective integration and efficient solution of those sections are receiving growing interests. The main reason is that integration of the main business areas such as sales, operations, distribution would lead to higher profits. Shobrys and White (2002) supported the idea of integrating the planning, scheduling and process control functions. They specifically pointed out that about 10 dollars increased margin per ton of product or more would be achieved by integration. Magalhães et al. (1998) proposed an integrated system for production planning (SIPP) including crude management, process plants, management of intermediate stocks and product blending. Zhang and Zhu (2001) presented the overall refinery optimization through integration of hydrogen network and utility systems with the material processing system. Li et al.

(2003) developed a plant-wide scheduling model for the whole refinery, which integrated a crude oil unloading and storage area, a refinery area, a blending area and a product storage area of the refinery. Jia et al. (2003) developed an event-based continuous MILP model to integrate crude oil scheduling, production scheduling and scheduling of product blending and distribution operations.

Although integrated models have been developed, their complexities involving large numbers of binary and continuous variables as well as nonlinear constraints lead to mathematical and computational difficulties that make the integration approach currently impractical. Moreover, the results generated by the integration approach may cause confusion because nobody can understand all the interactions between so many processes and decisions to explain the optimal results (Zhang and Zhu, 2000). Therefore, decomposition approaches are the best-suggested methodology to optimize these integrating models. Zhang and Zhu (2000) presented a novel decomposition strategy to tackle large scale overall refinery optimization problem. The overall plant model was decomposed into a site level model and process level model. The site level model determined common issues among processes such as allocation of raw materials and utilities, feed and yield for each unit and so on, and then process level model was solved by simulation. The solution information from the process level model was then iteratively fed to the site level model for further optimization. The procedure terminated when the convergence criteria were reached. Zhang and Zhu (2001) used a decomposition approach in which material processing was optimized first using LP techniques to maximize the overall profit and then hydrogen network and utility system were optimized to reduce the operating costs for the fixed process conditions determined from the LP optimization. Kelly (2002) proposed a chronological decomposition heuristic (CDH) based algorithm, which was a simple time-based divide and conquer strategy intended to find integer-feasible solutions to production scheduling optimization problems. Li et al. (2003) also used a decomposition method in which they solved the crude oil unloading and storage model by using their algorithm (MIP+NLP) firstly, then solved refining area units especially crude distillation unit and fluided-bed cracker unit (FCCU) using nonlinear algorithms and finally solved the product blending and storage model. Jia et al. (2003b-c) solved their integrated model presented by Jia et al. (2003a and 2004) separately.

2.4 Uncertainty in Refinery Operation

During real refinery operations, especially crude oil scheduling, frequent uncertainties are unavoidable such as ship arrival delays, demand fluctuations, equipment malfunction, etc. In the presence of these uncertainties, an optimal schedule obtained using nominal parameter values may often be suboptimal or even become infeasible. Therefore, how to manage these disruptions is quite critical. In general, two approaches to disruption management exist in the literature, which are predictive and reactive.

2.4.1 Reactive Scheduling

Reactive scheduling is used during the actual execution of the plan or schedule, when a disruption has occurred. For reactive rescheduling, the time required for rescheduling is a critical issue because rescheduling has to be performed in the actual course of the operation execution and any delay in responding to the disruptions could have significant financial impact. Although refinery operations are often interrupted by uncertainties and hence efficient methodologies are needed for rescheduling, there is no significant research work so far in rescheduling refinery operations. In the past, researchers mainly focused on rescheduling noncontinuous operations.

Abumaizar and Svestka (1997) developed Affected Operation Rescheduling (AOR) algorithm, which reduces much of the deviation and computational complexity associated with Total Rescheduling while producing makespan that are not statistically higher. They proposed a rescheduling program which, based on user-keyed random disruption information, produces three alternative schedules and outputs their related performance measures.

Akturk and Gorgulu (1999) proposed a rescheduling strategy to reschedule part of the initial schedule when a machine breakdown. Kunnathur et al. (2004) developed a rescheduling heuristic for operations when any variation from the expected value of flow time happens.

Henseler (1994) proposed an algorithm for reactive scheduling that efficiently repairs broken constraints by iteratively revising the schedule until there were no more violated constraints.

Honkomp et al. (1999) presented a reactive scheduling framework for processing time variations and equipment breakdown by coupling a deterministic schedule optimizer with a simulator developed by Honkomp (1995). They also divided the scheduling horizon into three regions: a fixed region, a flexible region, and a free region based on the disturbances.

Vin and Ierapetritou (2000) proposed a novel solution approach for efficient reactive scheduling of multiproduct batch plants. Their solution approach was based on a two-stage procedure in which the deterministic schedule was obtained with the data at the current time at the first stage, and the optimal reschedule was obtained by fixing all decision variables before the disturbance occurs. They used this solution approach to address two kinds of disturbances: machine breakdown and rush order arrival.

Mendez and Cerda (2003b) employed several rescheduling operations to perform reactive scheduling in multiproduct, sequential batch plants. They considered start time shifting, local reordering, and unit reallocation of old batches as well as insertion of new batches.

Mendez and Cerda (2004) developed an MILP-reactive scheduling algorithm to revise the short-term schedule of resource-constrained multistage batch facilities arising from unexpected disruptions. The size of the problem formulation remained reasonable because a large part of the scheduling decisions were unchanged and rescheduling actions were applied gradually by first reassigning resource items to tasks yet to be processed and then reordering tasks.

Rosenberger et al. (2003) proposed an aircraft selection heuristic for selecting which aircraft was rerouted and they presented an optimization model that rescheduled legs and rerouted aircraft by minimizing an objective function involving rerouting and cancellation costs. They could solve the problem using a Benders' decomposition formulation in which the master problem determined the new aircraft routes for the first stage, and the subproblems rerouted the aircraft for different second stage scenarios.

Subramaniam and Raheja (2003) developed a heuristic-based reactive repair mechanism for job shop schedules. They pressed the fact that the reactive repair of the original schedule was a better alternative to total rescheduling, as the latter not only was time consuming but also led to shop floor nervousness. They studied typical job shop disruptions and decomposed their repair processes into four generic repair steps, which were achieved using the proposed modified affected operation rescheduling (mAOR) heuristic. They conducted an extensive simulation study to evaluate the performance of the mAOR schedule repair heuristic.

Recently, Herroelen and Leus (2004) reviewed methodologies for proactive and reactive project scheduling and presented some hints that should allow project management to identify a proper project scheduling methodology for different project scheduling environments. Aytug et al. (2005) reviewed rescheduling methods in the face of uncertainties and suggested some future directions. They introduced a four-dimensional taxonomy for uncertainties. They pointed out that much work has to be done in spite of some noteworthy work in this area.

Subramaniam et al. (2005) applied modified affected operation rescheduling (mAOR) proposed by Subramaniam and Raheja (2003) for repairing a majority of typical job shop disruptions such as absenteeism of worker, process time variations, and arrival of unexpected jobs. Their results showed that the performance of the mAOR heuristic is superior to the right shift rescheduling heuristic.

Recently, Arief et al. (2004 and 2007a) proposed heuristic-based approach for crude oil operations to reschedule operations of a given schedule to accommodate disruptions. In their paper, they defined *block* as one or more periods involving no intervening change in configuration. Moreover, they proposed four rescheduling principles such as rescheduling every disruption and disrupted block individually and minimizing domino effects and CDU changeover. Their heuristic approach requires much less time to generate efficient schedules compared to total rescheduling. However, their heuristic approach is problem-specific and difficult to extend. To overcome these limitations, Arief et al. (2007b) proposed a general model-based approach in which causal models of the refinery operations describing the effects of a disruption and reveal rectification strategies were used.

Knowledge-based and artificial intelligence approaches have also been proposed for job shop rescheduling, including case-based reasoning (Dorn, 1994), constraint-based scheduling (Miyashita, 1995; Spargg et al. 1997), fuzzy logic (Dorn et al., 1994; Schmidt, 1994), and neural network (Garner and Ridley, 1994; Rovithakis et al., 2001; Qi et al., 2000).

2.4.2 Predictive Scheduling

Predictive scheduling seeks to accommodate possible disruptions while scheduling. In other words, the predictive approach aims to produce inherently robust schedules. The predictive approach includes stochastic programming, robust optimization, fuzzy programming, etc. Stochastic scheduling is extensively used in the literature for predictive scheduling, which uses stochastic variables to model uncertainties and converts the original deterministic scheduling model into stochastic model. The objective is to optimize the expectation of a certain performance criterion, for instance, the expected makespan, and the expected profit. Stochastic programming models can be classified into two-stage or multistage stochastic programming and chance constraint programming models. The stochastic approach has been extensively used in batch plants (Ierapetritou and Pistikopoulos, 1996; Balasubramanian and Grossmann, 2004).

Robust scheduling focuses on minimization of the effects of uncertainties on the performance measure such as profit, and operating cost. Its main objective is to ensure the realized schedules not to differ drastically from uncertainties, while maintaining a high level of schedule performance. Robust optimization has been applied to several fields such as production planning, machine scheduling, and logistics. Recently, Vin and Ierapetritou (2001), Lin et al. (2004), and Janak et al. (2007) used robust optimization to address robust scheduling of batch processes.

Fuzzy programming uses fuzzy set theory and interval arithmetic to describe the imprecision and uncertainties in process parameters. This approach is used in the case where probabilistic models that describe the uncertain parameters, which stochastic programming and robust optimization rely on, are not available. This approach is used in the literature to address scheduling of flow-shop plants and new product development process (Balasubramanian and Grossmann, 2003), product development projects (Wang, 2004), etc. The detailed reviews on stochastic models, robust scheduling models and fuzzy programming can be referred to Sahinidis (2004) and Li and Ierapitritou (2008).

So far, several efforts in the literature have addressed uncertainty in refinery operations. Li et al. (2004) presented a novel approximation-based approach to refinery planning under demand or other economic parameters uncertainties. In their model, loss functions were derived and applied to calculate the expectation of plant revenues. The whole problem was formulated using two-stage stochastic programming approach. In addition, demand and price were assumed to be independent. To address the problem that demand and price are correlated, Li et al. (2005 and 2006) extended the model of Li et al. (2004) to address refinery planning problem under correlated and truncated price and demand uncertainties. Neiro and Pinto (2005) proposed a multiple period optimization model for production-planning of petroleum refineries incorporating product price and demand uncertainties. These uncertainties were included as a set of discrete probabilities. Recently, Cao and Gu (2006) used chance constrained programming to address demand uncertainty during crude oil scheduling. However, they used the approach of Quesada and Grossmann (1995) to linearize bilinear items, which led to composition discrepancy.

2.5 Summary of Research Gaps

According to the above literature review on scheduling of refinery operations, the following research gaps can be summarized.

1. For crude oil scheduling, all existing solution algorithms in the literature may fail to find a feasible solution although one may exist. Moreover, they still need large solution times for solving large, practical problems and cannot guarantee optimality. Therefore, no reliable, robust, and efficient algorithm exists in the literature for this real, practical, and very useful problem. In addition, the models in the literature only incorporated one or two key components, which were assumed to be linearly additive.
2. For crude oil scheduling, all existing models in the literature incorporated some real operation features. In other words, no complete formulation is developed to incorporate all real operation features like SBM, Jetties, VLCCs, single-parcel vessels, storage tanks, charging tanks, crude segregations, brine settling, multiple tanks feeding one CDU, one tank feeding multiple CDUs, fifteen important crude properties. In addition, only one type of crudes were allowed to store in storage tanks in Lee et al. (1996), Li et al. (2002), and Jia et al. (2003c) for in-land refinery. Reddy et al. (2004b) allowed only one parcel in the SBM pipeline.
3. For scheduling of gasoline blending and distribution operations, all existing models in the literature considered only parts of the problems for treating recipe, blending, scheduling and distribution. Furthermore, some of them used nonlinear correlations to predict some product quality specifications, which turns the problem into MINLP and makes difficult to solve. In addition, they did not impose constant blending rate during a run.

4. The existing work in the literature mainly focused on the refinery-planning problem under uncertainties or rescheduling of operations for a given schedule to accommodate disruptions. Although Cao and Gu (2006) used chance constrained programming to address demand uncertainty during crude oil scheduling, they used the approach of Quesada and Grossmann (1995) to linearize bilinear items, which led to composition discrepancy.

2.6 Research Focus

Based on the above challenges, this research project focuses on the following aspects.

1. A robust and efficient algorithm is developed to solve the MINLP problem of crude oil scheduling and new strategies are presented to reduce computational time and improve solution optimality. In addition, fifteen crude properties that are critical to crude distillation and downstream processing are identified and the practical utility of Reddy et al. (2004b)'s MINLP formulation is also enhanced by adding appropriate linear blending correlations for these properties. Moreover, Reddy et al. (2004b)'s formulation is revised to ensure practically realistic schedules with limited changes of feed rates to CDUs.
2. A discrete-time MINLP formulation is developed for crude oil scheduling in an-inland refinery. This model incorporates many real operation features such as SBM, multiple jetties, VLCC, single-parcel vessels, storage tanks, charging tanks, crude segregation, brine settling, multiple charging tanks feeding one CDU, one charging tank feeding multiple CDUs, etc. Furthermore, different types of crudes

are allowed to blend in both storage and charging tanks, which results in three blending polices by considering whether crude compositions in both storage and charging tanks are variables or not. The developed robust and efficient algorithm is used to solve this complete formulation to illustrate the capability of developed formulation and further demonstrate the robustness and efficiency of the developed algorithm.

3. A general process slot-based continuous-time MINLP formulation is developed to address scheduling of gasoline blending and distribution operation and determine optimal product recipes. Several real operation features are incorporated such as several non-identical blenders working in parallel, dedicated component tanks, multi-purpose product tanks, variable product blending recipes, constant blending rate in a run, etc. A schedule adjustment procedure is proposed to solve this MINLP model and compared with commercial solves such as BARON and DICOPT. Additionally, nine important gasoline properties are identified and appropriate linear blending correlations are used for these properties.
4. A general unit-slot MINLP formulation is proposed to improve the efficiency of the process-slot based formulation and all real-life features of the proposed process-slot formulation are incorporated. The schedule adjustment procedure is extended to avoid solving nonconvex MINLP problem.
5. Several possible disruptions such as demand fluctuation, ship arrival delay, and tank unavailability during crude oil scheduling operations are identified. Both reactive and predictive approaches are used to address those disruptions. First, a

novel approach is developed for reactive scheduling of crude oil, in which schedule changes are minimized and computational times are reduced by fixing binary variables involving the time before disruptions occur. Then for predictive approach, schedule robustness is defined using a penalty function and a procedure is developed to evaluate schedule robustness. Finally, a scenario-based formulation is developed to address demand fluctuation to obtain robust schedule.

In next Chapter, crude oil scheduling problem is first addressed to develop a robust algorithm and improve speed and optimality. Before going to the next Chapter, the following briefly introduces time representation in the literature, which is very critical for developing mathematical models for scheduling problems.

2.7 Time Representation

A variety of discrete-time (Kondili et al., 1993; Shah et al., 1993; Mockus and Reklaitis, 1994; Lee et al. 2001) and continuous-time formulations have appeared in the literature (Floudas and Lin, 2004a,b; Mendez et al. 2006; Pitty and Karimi, 2008). In discrete-time representation, the scheduling horizon is divided into a number of intervals of equal duration. Events of any type such as the start or end of processing individual batches of individual tasks, changes in the availability of processing equipment and other resources, etc. are only allowed at the interval boundaries. Its main advantage is that it facilitates the formulation by providing a reference grid against which all operations competing for shared resources are positioned.

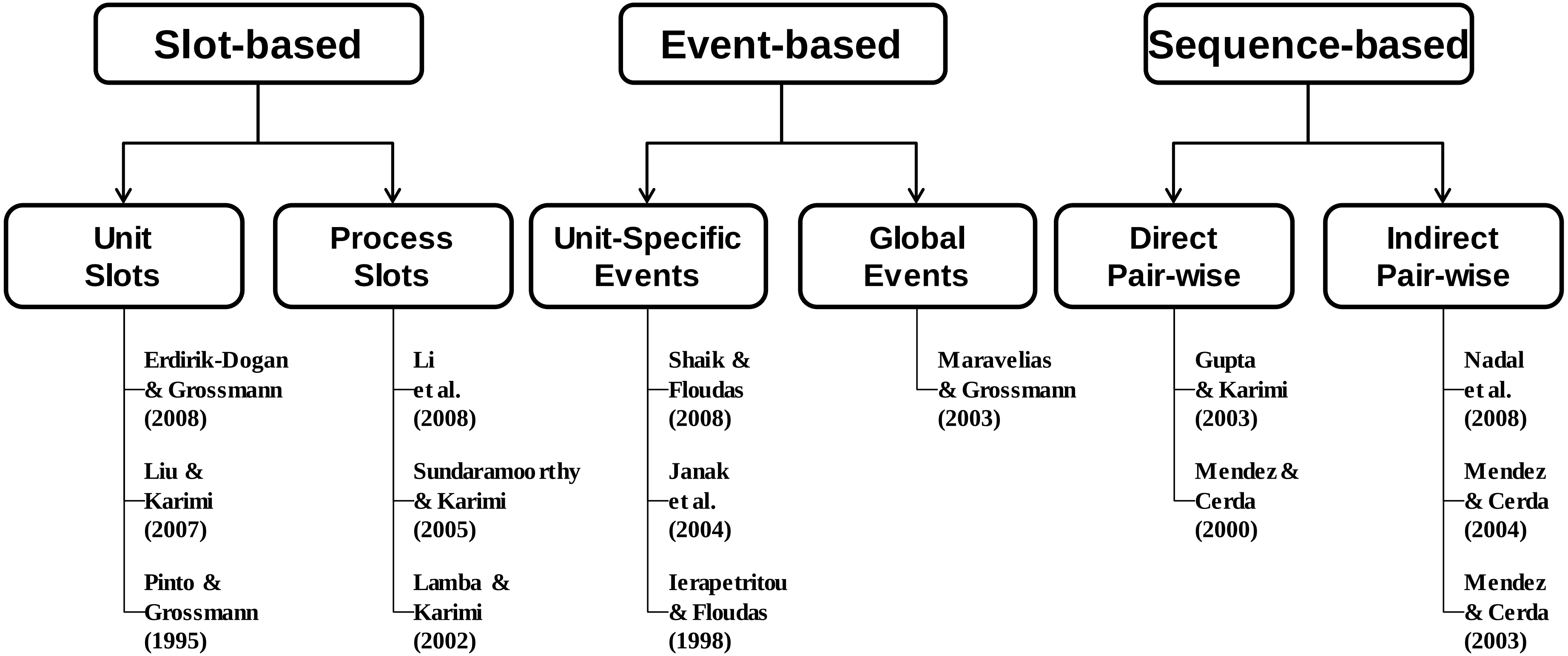

Figure 2.2 Classification of continuous-time scheduling models

A binary variable is used to indicate whether the task is started at the beginning of that interval or not. The main difficulty with this representation is that a model with a very large number of binary variables is needed to represent a process accurately. Another inherent difficulty of discrete-time representation arises in representing continuous processes because a continuous process may start and end somewhere within an equal size interval, not on the interval boundaries. These two limitations are removed by the continuous-time representation. The continuous-time representation accounts for variable processing times and is more realistic than the discrete-time representation. It also requires significantly fewer time intervals and hence leads to smaller problems.

During the last two decades, numerous formulations have been proposed in the literature based on continuous time representation. The different continuous-time models can be broadly classified into three distinct categories: sequence-based model, slot-based model, and event-based model as shown in Figure 2.2.

The slot-based models (Pinto & Grossmann, 1995; McDonald & Karimi, 1997; Karimi & McDonald, 1997; Lamba & Karimi, 2002a-b; Lim & Karimi, 2003; Reddy et al., 2004a; Sundaramoorthy and Karimi, 2005; Liu & Karimi, 2007a-b; Liu & Karimi, 2008; Erdogan & Grossmann, 2008) model time by means of ordered slots of non-uniform unknown lengths to which batches, tasks, or activities are assigned. The literature has used two slot types. If one single common or shared set of slots is employed for all units in a process, then they are called synchronous (Lim & Karimi, 2003) or process slots (Liu & Karimi, 2007a,b). Such a slot design makes it straightforward to deal with shared resources such as storage, utilities, etc., as the relative timings of operations at all units are known without uncertainty. If such a single shared set of slots is not used, but a distinct and independent set of slots is employed for each unit in the process, then they are called asynchronous (Lim and Karimi, 2003) or unit slots (Liu and Karimi, 2007a,b). Because these slots are asynchronous across the units, it is not easy to know the relative timings of activities at various units for accurately monitoring resource level. Lim and Karimi (2003) used binary variables to explicitly identify the relative times at which a resource is consumed or produced. Thus, the use of unit slots for handling shared resources has not received much attention, but some progress was made in this direction in recent work (Susarla et al., 2008; Li et al. 2008).

The unit-specific event-based models (Ierapetritou and Floudas, 1998a-b; Ierapetritou et al., 1999; Lin and Floudas, 2001; Giannelos and Georgiadis, 2002; Lin et al., 2003; Jia and Ierapetritou (2003a,b); Jia et al. (2003); Jia and Ierapetritou (2004); Janak et al., 2004; Shaik et al. 2006; Shaik and Floudas, 2007; Shaik and Floudas, 2008) “introduce an original concept of event points, which are a sequence representing the beginning of a task or utilization of the unit. The locations of event points are different for different units, allowing different tasks to start at different moments in different units for the same event point” (Floudas and Lin, 2004). The global event-based models (Castro et al., 2001; Castro et al., 2004; Maravelias & Grossmann, 2003) use one single set of event points and times for all units in a process, and they are analogous to the models using process slots.

The sequence-based models (Mendez et al., 2000; Mendez et al., 2001; Mendez & Cerda, 2002; Mendez & Cerda, 2003; Mendez & Cerda, 2004; Hui & Gupta, 2000; Hui et al., 2000; Gupta & Karimi, 2003a-b; Pitty & Karimi, 2008; Ferrer-Nadal et al., 2008) use direct (immediate) or indirect (general) pair-wise sequencing (precedence) of tasks on units to define a schedule, thus they do not model time explicitly in terms of slots or event points. This gives them one advantage over the slot-based or event-based models, as they do not need to postulate a number of slots or event points a priori for most problems. However, the work on these models has been limited, and they do suffer from the difficulty in monitoring resource levels.

CHAPTER 3

IMPROVING THE ROBUSTNESS AND EFFICIENCY OF CRUDE SCHEDULING ALGORITHMS

3.1 Introduction

Crude oil costs account for nearly 80% of a refinery's turnover (Kelly & Mann, 2003a). Crude oils vary significantly in compositions, product yields, properties, and prices. Premium crudes such as West Texas Intermediate (WTI), Brent blend, etc. sell roughly \$15 per barrel higher than low-quality crudes such as Arabia heavy, Soudieh, etc. Most refineries use varying blends of several crude oils over time to exploit the higher margins of low-cost crude oils. With declining supplies and increasing prices of premium crude oils, the challenge facing the refiner is how to best exploit the greater margins of the low-cost crudes to increase profits. However, the low-cost crudes are almost always high in less-than-desirable components or traits such as sulfur, aromatics, high residue, etc., and cause processing and/or product quality problems in crude distillation units (or CDUs) and downstream units. Therefore, a key issue in the refinery business is to identify and process optimal blends of low-cost and premium crudes to minimize the operational problems yet maximize profit margins. As noted by Kelly & Mann (2003a,b), scheduling of crude oil operations in a refinery is an critical task that can save millions of dollars per year, if done in an optimal manner. However, that is easier said than done.

This chapter addresses the crude scheduling problem described by Reddy et al. (2004b) for a typical marine-access refinery. As mentioned by them, the task of crude oil scheduling in today's refinery is becoming increasingly complex and involves integrated management of activities such as crude arrivals, unloading, storage, blending, and charging or processing over days to weeks. Crude schedulers react to crude arrivals, schedule crude unloading, assign destination tanks for crude parcels to get clever crude blends, then mix these blends again to charge CDUs at appropriate feed rates to meet product demand and quality targets with minimum give-aways, minimize operational problems, and maximize profits. Clearly, the crude scheduler plays a critical role in determining the bottomline of a refinery. However, his/her task is by no means simple, as evident from the several attempts in the open literature at solving this problem at industrial scale optimally and with reasonable computational effort.

As noted by Reddy et al. (2004b), the schedulers have an enviably tough job in a refinery, which has become even more difficult in recent years. They must continuously watch both crude oil movements, and plant operation status, and match them to fluctuating demands. In most cases, under intense time-pressure and low inventory flexibility, the schedulers rely largely on experience and select the first feasible solution found by a spreadsheet model or some other method. Clearly, that leaves tremendous room for economic and operability improvement. Quantifiable economic benefits from advanced scheduling are improved options, increased utilization and throughput, intelligent use of low-cost crude stocks, capture of quality barrels, reduction of yield and quality giveaways, improved control and predictability of downstream production, reduced demurrage costs, reduced slop generation from changeovers, improved inventory and safety stock control, etc.

The presence of blending in crude oil operations that do not use charge tanks of specified compositions gives rise to bilinear terms in a mathematical formulation for scheduling. Additionally, discrete scheduling decisions such as selecting a tank to unload and the often complex nonlinear nature of crude properties and qualities make such a model difficult, nonlinear, nonconvex mathematical program or a nonconvex, mixed integer nonlinear program (MINLP). As shown later, even the best existing commercial solvers (e.g. DICOPT and BARON) are unable to solve these scheduling problems of practical, industrial size in reasonable time. Hence, in recent years, several researchers have addressed this problem in various forms and reported a variety of continuous-time and discrete-time models, often along with specialized algorithms to solve them with reasonable computational effort. However, the problem remains far from being solved satisfactorily. While attaining a guaranteed globally optimal solution to this nonlinear nonconvex problem is a challenging and important issue, getting good solutions for practical-size problems in reasonable times, even without the guarantee of global optimality, is an even more pressing issue at this time. Thus, it is clear that this paper focuses on the latter and not the former. While trying best to get the best solutions, the aim of this chapter is not to get the guaranteed globally optimal solutions.

To begin with, no existing work on crude oil scheduling has so far considered nonlinear crude/product properties. Thus, most existing models without charge tanks of known compositions have primarily been mixed integer bilinear programs (MIBLPs). With this in mind, the existing work involves four broad approaches, notwithstanding their nature of time representation (continuous-time vs. discrete-time). The first approach (Lee et al., 1996; Jia & Ierapetritou, 2003) comprises approximating the original MIBLP by a pure MILP. Apart from several deficiencies in the existing work, described in detail by Reddy et al. (2004b), the main issue with this approach has been the composition discrepancy pointed out by Li et al. (2002) and Reddy et al. (2004b). For instance, Lee et al. (1996) used reformulation linearization technology (RLT) for bilinear terms, but this linearization approximation leads to a mismatch between the composition of crude delivered by a set of storage tanks to a CDU and that actually received by the CDU. To avoid this composition discrepancy, Li et al. (2002) proposed an iterative decomposition algorithm that solves an alternating series of MILP approximations and NLPs. However, as commented by Kelly & Mann (2003a,b) and shown by Reddy et al. (2004b), this decomposition approach may fail to obtain a feasible solution even when one exists. Moro and Pinto (2004) presented an alternate approach for dealing with bilinear terms by using discrete values for continuous variables such as CDU feed rates. However, in addition to getting approximate optimal solutions, this discretization procedure increases problem size to an extent that makes it almost impossible to solve reasonably sized problems. Almost concurrently, Reddy et al. (2004a,b) proposed a novel rolling-horizon solution algorithm, which not only avoids composition discrepancy without increasing the problem size, but also avoids solving any NLP or MINLP. They used their algorithm to solve several moderate-size and one large practical-size problems. In spite of their relative success in solving this difficult problem, it is shown later that their algorithm can also fail to obtain feasible schedules and needs long solution times for solving large, practical-size problems. Therefore, it is fair to say that the existing open literature still lacks a reliable, robust, and efficient algorithm for this practical and important problem. For a detailed review of work on this problem, please refer Reddy et al. (2004b).

In this Chapter, the work is building on the model and algorithm of Reddy et al. (2004b) for scheduling crude oil operations to overcome some deficiencies of the existing work. As the first step, the most important nonlinear crude properties that are crucial to crude distillation and downstream processing are identified and modeled. Then, a robust, improved, and more efficient version of their algorithm is developed. Finally, a procedure is developed to estimate the quality of schedules by developing a rigorous upper bound for this nonconvex problem. An example is first used to motivate this work, a brief description of the problem is presented, and then some salient features of the formulation proposed by Reddy et al. (2004b), which provides the basis for this work, are reviewed. That formulation is then extended to accommodate nonlinear crude properties. In subsequent sections, strategies are developed to improve robustness, quality, and solution speed of Reddy et al. (2004b)'s algorithm, and present ways to estimate solution quality by means of a tight upper bounding strategy. Finally, twenty-four large simulated examples are used to demonstrate numerically the robustness and effectiveness of the improved algorithm.

3.2 Problem Statement

Figure 3.1 shows a schematic of crude oil operations in a typical marine-access refinery. It comprises offshore facilities for crude unloading such as a single buoy mooring (SBM) or single point mooring (SPM) station, onshore facilities for crude unloading such as one or more jetties, tank farm consisting of crude storage and/or charging tanks, and processing units such as crude distillation units (CDUs). The crude storage tanks hold crude blends rather than pure crudes. Their compositions vary with time. In this work, it is assumed that the refinery has no separate charging tanks; crude storage tanks also act as charging tanks. The unloading facilities supply crude to storage tanks via pipelines. The pipeline connecting the SBM/SPM station with crude tanks is called the SBM/SPM line, and it normally has a substantial holdup.

Two types of ships supply crudes to the refinery. Very large crude carriers (VLCCs) or ultra large crude carriers (ULCCs) carry multiple parcels of several crudes and dock at the SBM/SPM station offshore. Small vessels carry single crudes and berth at the jetties. The entire crude oil operation involves unloading and blending crudes from ships into various storage tanks at various times, and charging CDUs from one or more storage tanks at various rates over time. Thus, crude oil operations in a typical refinery involve both scheduling and allocation decisions. The problem (Reddy et al., 2004b) can be stated as follows:

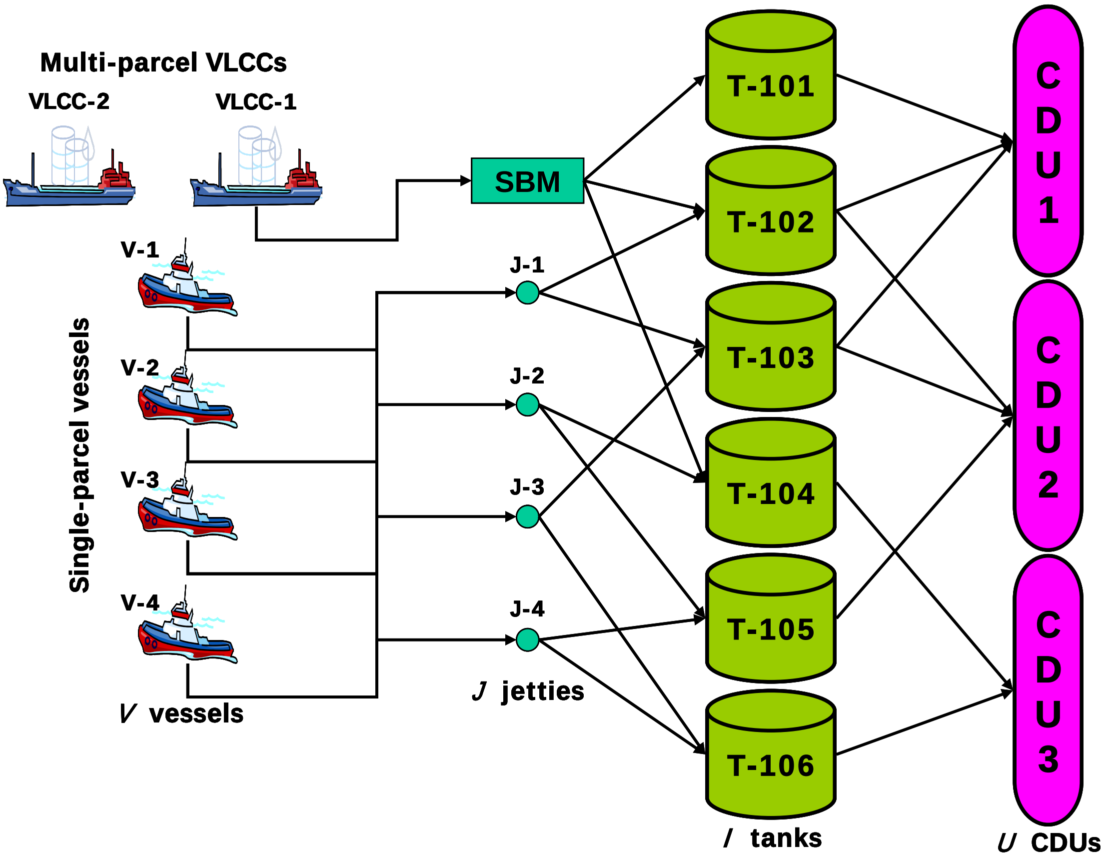

Figure 3.1 Schematic of crude oil unloading, blending, and processing

Given:

1) Crude delivery data: Estimated arrival times of ships, their crude parcels, and parcel sizes.
2) Maritime infrastructure: Jetties, jetty-tank and SBM-tank connections, crude unloading transfer rates, and SBM pipeline holdup volume and its resident crude.
3) Tank farm data: Storage tanks, their capacities, their initial crude stocks and compositions, and crude quality specifications or limits.
4) Crude processing data: CDUs, processing rates, and crude quality specifications.
5) Economic data: demurrage, crude changeover costs, safety stock penalties, crude

margins, and product demands.

Determine:

1) Unloading schedule for each ship including the timings, rates, and tanks for all parcel transfers.
2) Inventory and crude concentration profiles of all storage tanks.
3) Charging schedule for each CDU including the feed tanks, feed rates, and timings.

Subject to the operating practices:

1) A storage tank cannot receive and feed crude at the same time.
2) Each tank needs 8 hours to settle and remove brine after each crude receipt.
3) Multiple tanks can feed a CDU simultaneously and vice versa.
4) Only one VLCC can dock at the SBM station at any time.
5) A parcel can unload to only one storage tank at any moment, but may unload to multiple tanks over time.
6) Sequence in which a VLCC unloads its parcels is known a priori. This is normally fixed when the VLCC loads its parcels and the refinery needs to specify that at the time of shipping.

Assuming:

1) Holdup of the SBM line is far smaller than a typical parcel size. Thus, only one crude resides in the SBM line at the end of each parcel transfer. Crude flow is plug flow in the SBM.
2) Holdup of the jetty pipeline is negligible.
3) Crude mixing is perfect in each storage tank.

4) Crude changeover times are negligible.

5) During operation, CDUs never shut down.

The objective of the scheduling problem is to maximize the gross profit, which is the revenue computed in terms of crude margins minus the operating costs such as demurrage, safety stock penalties, etc.

3.3 Base Formulation

While several discrete-time (Shah, 1996; Lee et al., 1996; Li et al., 2002;Reddy et al., 2004b) and continuous-time models (Jia et al., 2003; Reddy et al., 2004a) exist in the literature for solving varying forms of the crude scheduling problem, one that will form the basis for the proposed algorithmic improvements is needed. While continuous-time formulations have advantages, Reddy et al. (2004b) showed that it is not fully certain, if they are clearly the best for this specific problem. The discrete-time model of Reddy et al. (2004b) is preferred to use because of some advantages. First, it embodies some features of a continuous-time formulation by allowing two parcels to transfer during any period. This partially obviates the need for a continuous-time model. Second, it accommodates some important structural and operational features of a marine-access refinery such as SBM/SPM, jetties, multi-parcel VLCCs, multiple tanks feeding a CDU simultaneously and vice versa, brine settling, crude segregation, accurate demurrage accounting, crude changeovers, etc. Third, its use of 8-h time periods is quite practical and suitable for refinery operations, as it matches with the common brine settling time and shift-based operation of refineries. A shorter time-slot would make the discrete-time models difficult to solve, as the binary variables will increase considerably. On the other hand, a longer time-slot will reduce precision. However, note that Reddy's algorithm is not constrained by the assumption of 8-h slots. Fourth, it accounts accurately for the significant holdup of the SBM line as done by treating the SBM holdup as a distinct single-crude parcel. When the first VLCC parcel is unloaded, this SBM parcel is ejected first. Similarly, when the last VLCC parcel is unloaded, a portion remains in the SBM line and becomes the next SBM parcel.

For the sake of completeness and easier comprehension of this work, the key aspects of the MINLP model of Reddy et al. (2004b) are now presented. Full details are available in their paper. The model represents time in terms of consecutive 8-h periods (t) and converts all ships and the initial crude holdup in the SBM line into a series of single-crude parcels (p) with appropriate arrival times. It uses three primary binary variables to model parcel-to-SBM/Jetty, tank-to-SBM/jetty, and tank-to-CDU connections.

$$XP_{pt} = \begin{cases} 1 & \text{if parcel } p \text{ is connected for transfer during period } t \\ 0 & \text{otherwise} \end{cases}$$

$$XT_{it} = \begin{cases} 1 & \text{if tank } i \text{ is connected to receive crude during period } t \\ 0 & \text{otherwise} \end{cases}$$

$$Y_{iut} = \begin{cases} 1 & \text{if tank } i \text{ feeds CDU } u \text{ during period } t \\ 0 & \text{otherwise} \end{cases}$$

A major problem with some existing models is the composition discrepancy arising from the linearization of the following bilinear constraints related to crude blending in storage tanks.

$$FCTU_{iuct} = f_{ict} \cdot FTU_{iut} \qquad (i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC} \qquad (3.1)$$

$$VCT_{ict} = f_{ict} \cdot V_{it} \qquad (i, c) \in \boldsymbol{IC} \qquad (3.2)$$

where, f_{ict} denotes the fraction of crude c in tank i at the end of period t, $FCTU_{iuct}$ is the amount of crude c from tank i to CDU u during period t, FTU_{iut} is the total amount of crude from tank i to CDU u during t, VCT_{ict} is the amount of crude c in tank i at the end of period t, and V_{it} is the total amount of crude in tank i at the end of period t. $\boldsymbol{IC}$ = {(i, c)| tank i can hold crude c}. $\boldsymbol{IU}$ = {(i, u)| tank i can feed CDU u}. Equations (3.1) and (3.2) are bilinear, f_{ict} is unknown, except at the start of the scheduling horizon (or equivalently the first period), when the initial crude compositions of tanks were known. When these are approximated by linear constraints, the composition of crude received by a CDU may not match with that supplied by the tanks, which has been termed as composition discrepancy by Li et al. (2002) and Reddy et al. (2004b). To avoid both MINLP solution and composition discrepancy, Reddy et al. (2004b) developed an effective heuristic iterative strategy to obtain good schedules, which is discussed later.

3.4 Motivation

Although the model and algorithm of Reddy et al. (2004a,b) are the best in the literature so far for this MINLP problem, they still have some shortcomings. In addition to linear crude properties, long solution times, and lack of quality estimates, a major issue is robustness or the ability to give a feasible solution in large and difficult problems. Consider the following example to illustrate that their algorithm may fail to obtain a feasible solution, even when one exists.

A refinery has one SBM pipeline, four storage tanks (T1-T4), two CDUs (CDU1 and CDU2), and processes four crudes (C1-C4) that are segregated into two classes

(CL1 and CL2). C1 and C2 belong to CL1, can be stored in T1 and T4, and can be processed in CDU1. Similarly, C3 and C4 belong to CL2, can be stored in T2 and T3, and can be processed in CDU2. The scheduling horizon is 3 days, in which one VLC carrying three crude parcels (300 kbbl C1, 300kbbl C4, 350 kbbl C3, unloaded in that sequence) arrives at time zero. At time zero, the SBM pipeline is holding 10 kbbl C2 from the last parcel. For simplicity, only one key component is considered, whose allowable ranges are [0.0045, 0.006] vol% and [0.014, 0.0153] vol% for CDU1 and CDU2 respectively. Other data are shown in Table 3.1.

Reddy’s algorithm fails to find a feasible solution after four iterations. In the four iterations, the algorithm fixes the schedule up to period 4 as shown in Table 3.2. All parcels have been unloaded in this partial schedule, which is now frozen for the fifth iteration. At the end of period 4, T2 has 200 kbbl of crude with 0.015166 vol% of key component and T3 has 790 kbbl with 0.015437 vol% of key component. T2 satisfies the key component specification of [0.014, 0.0153] vol% for CDU2, but T3 does not. Since no crudes will arrive, CDU2 must use crudes in T2 and T3 for periods 5 to 9. Furthermore, T3 just finished receiving a parcel in period 4, hence it cannot supply crude in period 5. Now, the total demand of CDU2 is 550 kbbl, of which 200 kbbl has been met until period 4 and 350 kbbl remain to be processed during periods 5 to 9. Since a low-quality crude is in T3, the best scenario is to feed 50 kbbl to CDU2 from T2 in period 5 and then use a mixture from T2 and T3 for the remaining periods. However, the lower limit on crude holdup in T2 is 50 kbbl, hence 100 kbbl of crude is available from T2 for the last four periods. Thus, one best chance for meeting the demand and quality constraints in periods 6 to 9 is to use 100 kbbl from T2 and 200 kbbl from T3. Unfortunately, this crude mixture has 0.015347 vol% of key component, which is unacceptable crude quality for CDU2, and the algorithm fails in iteration 5. However, when this example is solved manually, a feasible solution is obtained shown in Table 3.2.

Interestingly, both GAMS/DICOPT and the iterative method of Li et al. (2002) fail to solve this example. Thus, there is a need to develop a tailor-made, better, robust, and more efficient algorithm for this difficult crude oil scheduling problem.

3.5 Extensions of Reddy's Model

The model of Reddy et al. (2004b) needs two refinements to extend its practical utility for scheduling crude oil operations. The first relates to the uncontrolled changes in CDU feed rates. Reddy et al. (2004b) allowed the CDU feed rates to fluctuate uncontrolled between successive periods. For instance, in the schedule for their motivating example, the feed to CDU2 is 50 kbbl in period 7, while it is 100 kbbl in period 8; which is a 100% change in 8 hours. Clearly, such drastic changes in feed rates may disrupt CDU operation, may generate off-spec distillation cuts, and may even be impossible to achieve without destabilizing the column. They can be disallowed by simply adding the following two constraints.

$$\gamma_u^L FU_{ut} \le FU_{u(t+1)} \le \gamma_u^U FU_{ut} \tag{3.3a,b}$$

Table 3.1 Data for Example 1

Tank & CDU	Capacity (kbbl)	Heel (kbbl)	Initial Inventory (kbbl)	Allowable Crude (Class)	Initial Crude Amount (kbbl)		Crude Concentration Range (Min-Max)		Total Demand (kbbl)	Processing Limits (kbbl/period) Min-Max	Property Specification Range (Min-Max)
					C1 or C3	C2 or C4	C1 or C3	C2 or C4			
T1	700	50	300	C1, C2 (1)	200	100	0.2-0.8	0.2-0.8	-	-	-
T2	700	50	300	C3, C4 (2)	100	200	0.2-0.8	0.2-0.8	-	-	-
T3	900	50	250	C3, C4 (2)	50	200	0.0-1.0	0.0-1.0	-	-	-
T4	700	50	300	C1, C2 (1)	130	170	0.2-0.8	0.2-0.8	-	-	-
CDU1	-	-	-	C1, C2 (1)	-	-	0.3-0.7	0.3-0.7	550	50-100	0.0045-0.0060
CDU2	-	-	-	C3, C4 (2)	-	-	0.0-1.0	0.0-1.0	550	50-100	0.0140-0.0153

Crude	Property Specification	Margin ($/bbl)
C1	0.0050	3.0
C2	0.0060	4.5
C3	0.0165	5.0
C4	0.0145	6.0

Parcel-Tank flow rate: 10-400 kbbl/period — Tank-CDU flow rate: 0-100 kbbl/period

Demurrage: 100 k$/period — Changeover loss: $5000/instance

Safety stock penalty: 0.2 $/bbl/period — Desired safety stock: 1200 kbbl

One VLCC (300 kbbl C1, 300 kbbl C4, 340 kbbl C3) arrives at time zero

Table 3.2 Schedules for Example 1

		Crude Amount [to CDU No.] (from Vessel No.) in kbbl for Period								
Algorithm	Tank	1	2	3	4	5	6	7	8	9
RA	1	−50.0[1]	−50.0[1]	−8.3[1]	−8.3[1]	−8.3[1]	−8.3[1]	−8.3[1]	−8.3[1]	−100.0[1]
	2			−50.0[2]	−50.0[2]	−50.0[2]				
	3	−50.0[2]	−50.0[2]	+300.0(3) +10.0(3)	+330.0(4)		−100.0[2]	−50.0[2]	−50.0[2]	−100.0[2]
	4	+10.0(1) +300.0(2)		−50.0[1]	−50.0[1]	−50.0[1]	−50.0[1]	−50.0[1]	−50.0[1]	
Manual	1	−50.0[1]	−50.0[1]	−50.0[1]	−50.0[1]	−10.0[1]	−10.0[1]	−10.0[1]	−10.0[1]	−10.0[1]
	2		−50.0[2]	−50.0[2]	+340.0(4)					
	3	−50.0[2]	+300.0(3)		−50.0[2]	−50.0[2]	−50.0[2]	−50.0[2]	−100.0[2]	−100.0[2]
	4	+10.0(1) +290.0(2)	+10.0(2)			−60.0[1]	−60.0[1]	−60.0[1]	−60.0[1]	−60.0[1]
RRA	1	−50.0[1]	−50.0[1]	−54.9[1]	−15.8[1]	−15.8[1]	−15.8[1]	−15.8[1]	−15.8[1]	−15.8[1]
	2		−50.0[2]	−50.0[2]	+340.0(4)					
	3	−50.0[2]	+300.0(3)		−50.0[2]	−50.0[2]	−55.9[2]	−67.1[2]	−80.5[2]	−96.6[2]
	4	+10.0(1) +300.0(2)			−50.0[1]	−50.0[1]	−50.0[1]	−50.0[1]	−50.0[1]	−50.0[1]

'–' sign represents delivery to [CDU], '+' sign represents receipt from (Parcels)

where, FU_{ut} denotes total amount of crude fed to CDU u during period t. parameters γ_u^L and γ_u^U can be suitably set to control period-to-period changes in crude feed flows.

The second model refinement is about ensuring the quality of feeds to CDUs. This is a critical operating requirement in practice, as a feed with poor quality can seriously disrupt the operation of a CDU and even downstream units. Ensuring acceptable feed qualities to CDUs is especially critical in this problem, where the goal is to exploit cheap, poor-quality crudes to enhance profitability. A variety of crude properties are used in practice such as specific gravity, sulfur, nitrogen, oxygen, carbon residue, pour point, flash point, nickel, Reid vapor pressure, asphaltene, aromatics, paraffins, naphthene, wax, and viscosity. Reddy et al. (2004b) used the idea of key component concentrations to model these crude quality specifications. They imposed the following constraints to specify acceptable lower and upper limits on the concentration of a key component.

$$\theta_{ku}^L FU_{ut} \le \sum_i \sum_c FCTU_{iuct}\theta_{kc} \le \theta_{ku}^U FU_{ut} \qquad (i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC} \qquad (3.4a,b)$$

where, θ_{kc} is the concentration of a key component k in crude c, and θ_{ku}^L and θ_{ku}^U respectively are the lower and upper acceptable limits on that in the feed to CDU u. While Reddy et al. (2004b) did mention that the above constraints can be applied to most common crude quality specifications; their approach has some limitations. First, it assumes (eq. 3.4) the key component concentration to be linearly additive. However, many crude properties (e.g. viscosity) involve highly nonlinear mixing rules, where eq. (3.4) cannot work. Second, it also assumes that the key component concentrations are additive on a volume basis. Crude properties such as density and sulfur are additive on a weight basis. Therefore, a better approach is needed to address crude quality specifications.

To this end, it is noted that a linear blending index exists for almost every crude property involving nonlinear mixing correlations. Furthermore, these blending indices are either volume-based or weight-based. Table 3.3 provides the blending indices and their additive bases for the fifteen most commonly used crude properties. Thus, instead of key component concentrations, an appropriate blending index is used for each crude property and the following constraint is defined to accommodate weight-based blending indices as follows.

$$\theta_{ku}^{L}\left(\sum_{i}\sum_{c} FCTU_{iuct}\rho_{c}\right) \le \sum_{i}\sum_{c} FCTU_{iuct}\rho_{c}\theta_{kc} \le \theta_{ku}^{U}\left(\sum_{i}\sum_{c} FCTU_{iuct}\rho_{c}\right)$$

$$(i, u) \in \boldsymbol{IU},\ (i, c) \in \boldsymbol{IC} \qquad (3.5\text{a,b})$$

where, ρ_c as the density of crude c, and θ_{kc} now refers to the blending index for property k of crude c in eqs. (3.4) and (3.5).

With these extensions, a revised MINLP model of Reddy et al. (2004b) with the same scheduling objective as in Reddy et al. (2004b) is obtained. It is called **F**. The remainder of this paper aims to solve **F** by proposing improved algorithms.

3.6 Improving Robustness & Efficiency

First, the key steps of the algorithm of Reddy et al. (2004b) is reviewed, which is called RA (Reddy's algorithm) for the sake of brevity. Recall that Reddy et al. (2004b) used 8-hour periods in their discrete-time model. The success of their algorithm in eliminating composition discrepancy without solving a nonlinear problem hinges on recognizing that eqs. 3.1 and 3.2 would be linear, if f_{ict} or tank composition was known. If a tank receives no crude during several periods, then its composition cannot change. The sets of such periods are called as zones of constant composition, which can be easily identified from the arrival times of ships as explained by Reddy et al. (2004b). Reddy et al. (2004b) combine this observation with the fact that tank compositions are known at time zero to divide the scheduling horizon into two distinct blocks for each tank. The front block (in time) in which the tank composition is known and constant and the rear one in which it is unknown. For all periods in the front block and the first period in the second block, they use eqs. 3.1 and 3.2 with known f_{ict}, and for the remaining periods, they ignore eqs. 3.1 and 3.2. Now, as the first step of their algorithm, they identify the initial zone of periods for each tank for which the tank composition is constant and known. The length of this block may vary from tank to tank. For the remaining periods, the composition in each tank is unknown. They use eqs. 3.1 and 3.2 for this first zone of periods, drop eqs. 3.1 and 3.2 for the remaining periods, and solve the MILP. Because no linearization was used in the first zone for each tank, the corresponding MILP solution is free of composition discrepancy. Then, they identify the longest zone of periods for which compositions are known in *all* tanks. They freeze the schedule for that zone and repeat the procedure again. This rolling-horizon type of strategy ensures that the final schedule has no composition discrepancy.

Table 3.3 Crude properties, their relevance, and corresponding indexes and correlations

Crude Property	Blending Index	Addition Base	Relevance to (Important for)	Index Correlation
Specific Gravity (SG)	DNI	Volume	Crudes, all products	1/SG
Sulfur	SULI	Weight	Crudes, all products	Weighted average
Nitrogen	NITI	Weight	Crudes, residue streams (550+ °C), vacuum gas oil (370-550 °C)	Weighted average
Carbon Residue	CRI	Weight	Crudes, residue streams (550+ °C), vacuum gas oil (370-550 °C)	Weighted average
Pour Point (PP °C)	PIndex	Volume	Crudes, all products	316200×Exp(12.5×Log(0.001(1.8×PP+491.67))
Freeze Point (°C)	FreezeIndex	Volume	Kerosene (150-280 °C)	3162000×Exp(12.5×Log(0.001(1.8×Freeze Point+491.67))
Flash Point (FLP °C)	FPIndex	Volume	All products	Exp((−6.1184+(2414/(FLP+230.56)))×Log(10))
Smoke Point (SMP mm)	SMI	Volume	Kerosene (150-280 °C)	−362+3200/Log(SMP)
Ni	NiIndex	Weight	Crudes, residue streams (550+ °C), vacuum gas oil (370-550 °C)	Weighted average
Reid Vapor Pressure (RVP Bar)	RVI	Volume	Crudes, products up to naphtha range boiling below 200 °C	Exp(1.14×Log(100×RVP))
Asphaltenes	ASPI	Weight	Crudes, residue streams (550+ °C), vacuum gas oil (370-550 °C)	Weighted average
Aromatics	AROI	Volume	Naphtha range boiling below 200 °C	Volumetric average
Paraffins	PARI	Volume	Naphtha range boiling below 200 °C	Volumetric average
Naphthenes	NAPHI	Volume	Naphtha range boiling below 200 °C	Volumetric average
Viscosity @ 50 °C (Visc_cst)	ViscIndex	Weight	Crudes, residue streams (550+ °C), vacuum gas oil (370-550 °C)	79.1+33.47×(Log(Log(Visc_cst+0.8)/Log(10))/Log(10))

Index correlations from Reddy (Singapore Petroleum Company Pte Ltd)

In section 3.4, it is shown that RA may fail to get a feasible schedule. When RA fails to find a feasible schedule at an iteration, one possible culprit is the early (frozen) part of the schedule. This frozen schedule involved fixing of binary variables, and their fixed values may mean an infeasible combination. RA lacks a mechanism to retract from these infeasible combinations. Thus, a simple way to resurrect RA would be to eliminate some infeasible combinations of binary variables by using the well-known integer cut (Balas & Jeroslow, 1972). Using this basic idea, a backtracking strategy is now developed to identify and remove infeasible combinations of frozen binary variables, which enables to proceed forward to obtain a feasible solution.

3.6.1 Backtracking Strategy

RA moves forward by exactly one block of periods at each iteration, where that block represents the first one or more contiguous periods after the frozen schedule, in which tank compositions are constant and known. These blocks are called as composition-based blocks. Thus, if RA fails to get a feasible solution at iteration *n*, then the fixed values of binary variables in one or more of previous blocks (*n*–1 to 1) or some combinations thereof may cause infeasibility. Since which combination is infeasible is not known exactly, one option is to include in the integer cut all the binary variables fixed so far from blocks 1 to (*n*–1). Clearly, this may work for small *n*, because the cut would involve only a few binary variables. However, this would be inefficient for large *n*. An alternate option is to consider previously frozen blocks one at a time backward. In other words, if the algorithm fails to give a feasible solution in block *n*, then it retracts to block (*n*–1), impose an integer cut for the previous binary combination in block *n*–1, and then resume the algorithm from block (*n*–1). If solving block (*n*–1) does not yield a feasible solution, then it retracts to block *n*–2, and repeat the same procedure. It continues to retract, until a feasible solution is got, and then proceed forward as in RA. This method would be effective, when the block that caused infeasibility is not too far away before block *n*.

Often in RA, a block comprises one period only, so the aforementioned strategy will require backtracking by single periods and that would be slow. Therefore, new blocks that are different from those used by RA are defined. To this end, blocks based on the scheduled arrival times of vessels are defined, but each vessel is ensured to unload entirely in exactly one block. These are called as vessel-based blocks to differentiate them from the composition-based blocks used by RA. The first block begins at time zero and ends at the scheduled arrival time of the first vessel. The second block follows immediately after the first, and ends at the scheduled arrival time of the second vessel, if the first vessel is ensured to complete unloading before the second arrives. If this cannot be guaranteed, then the first block will extend to the arrival of the third vessel. It continues likewise to define all blocks such that no vessel will unload in more than one block. For example, if four vessels arrive at the starts of periods 4, 5, 13, and 17, and each vessel needs at least four periods to unload, then four blocks are got. Block 1 spans periods 1-3, block 2 spans 4-12, block 3 spans 13-16, and block 5 spans 17-last. Note that the model of Reddy et al. (2004b) assumes a priori the periods in which a vessel could possibly unload, which enables to define the blocks in the aforementioned manner. Since it estimates these periods in a conservative manner, such blocks can always be defined.

3.6.2 Variables for Integer Cuts

Even if cuts in one block are implemented at a time using the blocks defined above, the cuts can still involve large numbers of variables. To strengthen them, an effective strategy is needed for selecting the best set of variables. The first key consideration in this selection is the number of variables. The fewer the variables in the cuts, the tighter the cuts; therefore the fewest variables is preferred. Second, the variables must have a direct impact on the feasibility of a schedule later. If a decision has no direct impact on the future feasibility of a schedule, then it is not critical, and its corresponding variables serve no useful purpose in the cut. This is because the frozen decisions that led to infeasibility later in time in the schedule are being tried to correct.

With the above two criteria in mind, the binary variables in the model of Reddy et al. (2004b), namely XP_{pt}, XT_{it}, and Y_{iut}, are examined. XP_{pt} denotes parcel-to-SBM/jetty connections. These connections must occur for every parcel to be unloaded. These connections have limited direct impact on the compositions of crude stocks in tanks, because the tank that the parcel is unloaded into also depends on XT_{it}. Thus, XP_{pt} has little direct effect on the feasibility of subsequent scheduling decisions, and it should not be in the cuts. A similar argument can be made for Y_{iut}. The tanks that are used to feed CDUs at a given time have limited direct effect on the future compositions of tanks and feasibility of later schedule. Moreover, the number of Y_{iut} is always more than that of XT_{it} in a vessel-based block, because Y_{iut} may be nonzero even in the constant-composition zones, while XT_{it} must be zero in those zones. Thus, they are not included in cuts. In contrast to these two binary variables, XT_{it} denotes tank-to-SBM/jetty connections. Effectively, these decisions control which tanks receive crudes and largely determine the qualities of crude stocks in the tanks, while XP_{pt} and Y_{iut} do not at least directly. This further illustrates that XT_{it} has a more direct effect on the future composition of tanks and the feasibility of subsequent schedule compared to XP_{pt} and Y_{iut}. A poor blending decision can prove costly later, as it may become impossible to blend the stocks at hand to satisfy the feed quality requirements of CDUs. Thus, it is clear that some combinations of XT_{it} could lead to infeasibility in a problem and these must be eliminated using integer cuts to achieve feasible schedules.

If all XT_{it} variables are included in the integer cuts, then,

$$\sum_{(i,t)\ni XT_{it}=1} XT_{it} - \sum_{(i,t)\ni XT_{it}=0} XT_{it} \le NZ - 1 \tag{3.6}$$

where, NZ is the number of terms in the first summation. As stated earlier, eq. 3.6 is obtained directly from Balas & Jeroslow (1972). It simply eliminates the particular combination of XT_{it} values from being considered again in the MILP. It is possible to reduce the number of binary variables in eq. 3.6 in some cases. Often, refineries segregate crudes into various classes, store them in different tanks, and process them in different CDUs. If the class/es of crudes or tanks that caused infeasibility can be somehow identified, then only the variables associated with those crudes or tanks can be included in eq. 6. Now a procedure for doing this is derived for a refinery practicing such a segregation of crudes and tanks.

Recall that RA uses eqs. 3.1 and 3.2 with known f_{ict} in the current constant-composition block to avoid composition discrepancy. When it fails to find a feasible solution at an iteration, a solution to these equations cannot exist without composition discrepancy. If the equations that cannot be solved can be identified, then the tanks that are causing the infeasibility would be known. To this end, eq. 3.1 is relaxed as follows by using two nonnegative slack variables.

$$FCTU_{iuct} = f_{ict} FTU_{iut} - u^{-}_{iuct} + u^{+}_{iuct} \qquad (i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC} \qquad (3.7)$$

If any of u^{-}_{iuct} or u^{+}_{iuct} is nonzero, then it is clear that composition discrepancy exists. Thus, by adding them to eq. 3.1, RA is essentially used to get a solution with composition discrepancy. Note that introducing slack variables in eq. 3.2 is not necessary for such a solution. If such a solution can be got, then the nonzero slack variables in that solution will tell the class or classes of crudes that are violating eq. 3.1.

There are two ways to achieve the above solution with nonzero slack variables. One is to add a penalty for u^{-}_{iuct} and u^{+}_{iuct} in the original scheduling objective. However, choosing a suitable penalty with proper weights that do not bias the solutions is a non-trivial problem. It is found that it hard to select a penalty that is suitable for different cases. The second method is to solve the same model but with a different objective that tries to force the slack variables to zero. The simplest such objective is to minimize the sum of slack variables.

$$\text{Min} \quad P = \sum_{i} \sum_{u} \sum_{c} \sum_{t} (u^{+}_{iuct} + u^{-}_{iuct}) \qquad (i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC} \qquad (3.8)$$

Recall that **F** is defined as the revised MINLP model of Reddy et al. (2004b) with the original scheduling objective. Now **FL** is defined as **F** augmented with eqs. 3.1 and 3.2 for the periods of known crude compositions in tanks and no eqs. 3.1 and 3.2 for remaining periods. Note that **FL** is MILP model of RA used to solve **F**. Then, **FP** is defined as **FL** but with eq. 3.8 as the objective in place of the original scheduling objective. Note that **FP** is also an MILP problem and its solution may be computationally expensive. Since the goal of **FP** is just to identify a set of nonzero slack variables, obtaining an exact optimal solution is not necessary in all cases. In order to control solution time, an upper limit on time is specified for solving **FP**. By doing this, an optimal solution, feasible solution, or no solution at all for **FP** may be obtained. With this, a revised version of RA is fully described, which is designed for obtaining feasible schedules that RA fails to get.

3.6.3 Revised Reddy's Algorithm (RRA)

Let α denote a composition-based block in RA and β denote a vessel-based block defined in section 3.6.1. The revised algorithm (RRA) follows all the steps in RA, except when it fails to get a feasible solution at some iteration. Let α be the current composition-based block that causes a solution failure for RA and β be the vessel-based block to which α belongs. Then, the slack variables are added for block α in **FL** to get **FP** and solve it to identify the nonzero slack variables and the corresponding class/es of crudes that fail to satisfy eq. 3.1. For each such crude class, one separate integer cut (that includes all crudes in that class) is written for all periods in block β, but before block α. Then, **FL** is updated by adding these cuts permanently. Now, all variables are freed before block α in block β and return to the beginning of block β. With the schedule before block β frozen, RA is restarted as per normal. Note that when RA fails at the first composition-based block α of a block β, then all combinations of integer variables in block β have been examined and it should backtrack to block $\beta - 1$. This procedure is repeated, until the entire schedule is obtained. Figure 3.2 shows the detailed algorithm.

Example 1

Let illustrate RRA using the motivating example. In that example, one VLCC carrying three parcels arrives at the beginning of the scheduling horizon, thus only one vessel-based block (β = 1) is needed, which includes periods 1-9. The first four iterations of RRA proceed smoothly, i.e. feasible solutions are readily obtained. α = 1 includes period 1 only, α = 2 includes period 2, α = 3 includes period 3, α = 4 includes period 4, and β = 1 for all iterations. At iteration 5, α = 5 includes periods 5-9. With the schedule frozen for periods 1-4, **FL** fails to give a solution for α = 5. Then, eq. 1 is replaced by the following constraints and solve **FP**.

$$FCTU_{iuct} = f_{ict} FTU_{iut} - u^{-}_{iuct} + u^{+}_{iuct} \qquad (i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC} \qquad (3.9)$$

Solving **FP** gives $u^{+}_{3236} = 2.16$, $u^{+}_{3237} = 4.844$, $u^{-}_{3246} = 2.16$, and $u^{-}_{3247} = 4.844$. Since these involve crudes 3 and 4, which belong to Class 2, it can be concluded that there is a problem with the way that Class 2 crudes were blended during periods 1-4. To remove this infeasible blending combination, the following integer cut for Class 2 is then used.

$$XT_{33} + XT_{34} - \sum_{t=1}^{4} XT_{2t} - XT_{31} - XT_{32} \leq 1 \tag{3.10}$$

Then, all XT_{it} (t = 1-4) for Class 1 crudes are fixed in **FL**, add eq. 10 to **FL**, and begin RRA from period 1 again. RRA proceeds to completion without any failure. Table 3.2 also shows the final schedule without any composition discrepancy.

3.6.4 Partial Relaxation Strategy

Consider RRA at an arbitrary iteration such that the schedule before composition-based block α is fixed. While the MILP solved at this iteration by RRA will treat all binary variables within α and beyond as binary, it will guarantee composition consistency (i.e. no discrepancy) in block α only. In other words, the portion of this MILP solution for blocks beyond α may have composition discrepancy and is of no use at this iteration. In other words, computational effort to get an integer-feasible solution of the MILP for blocks beyond α is not warranted. In fact, the binary variables for these blocks can be relaxed without affecting the solution for block α. In principle, this can be done for all blocks beyond α, but eq. 6 involves binary variables from vessel-based blocks and may get partially relaxed, losing its cutting power completely. Therefore, only the binary variables beyond the vessel-based block β to which α belongs are relaxed. Figure 3.3 shows this partial relaxation strategy and this algorithm is called as RRA-P.

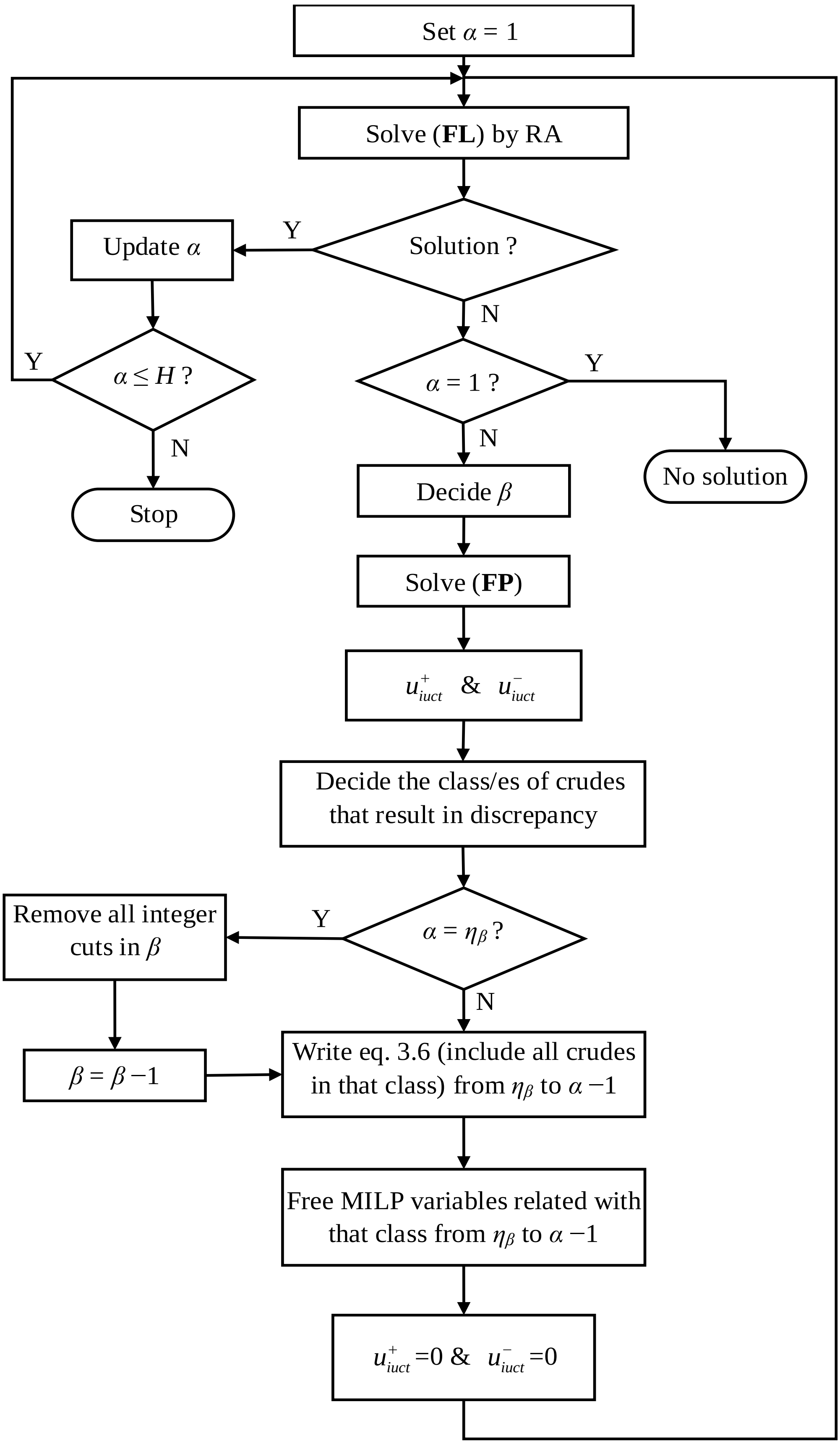

Figure 3.2 Flow chart for RRA [Revised Algorithm of Reddy et al. (2004a,b)], α denotes a composition-based block, β denotes a vessel-based block, η_β denotes the first composition-based block in block β, H denotes the scheduling horizon

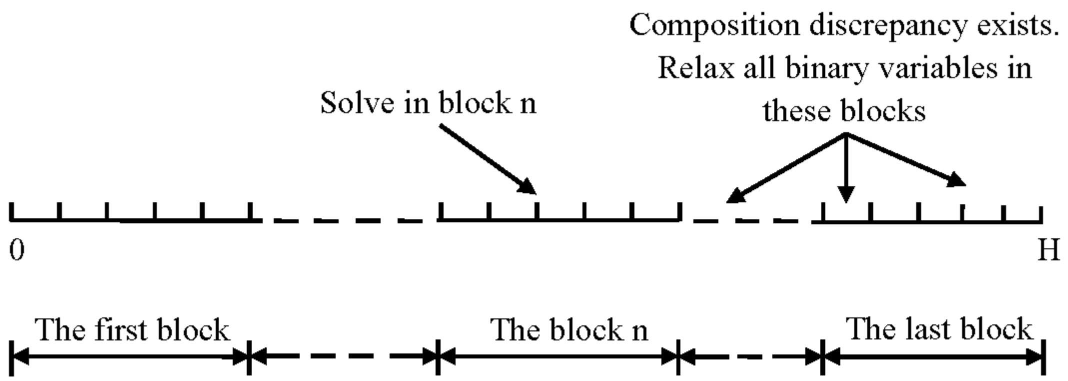

Figure 3.3 Schematic of RRA-P (Partial Relaxation Strategy)

3.6.5 Algorithm Evaluation

Twenty-four examples (including Example 1 or the motivating example) of varying sizes and features are used to evaluate the performance of RRA and RRA-P against those of RA (Reddy et al., 2004b), LA (Li et al., 2002) and commercial MINLP codes such as DICOPT and BARON (in solving **F**). Tables 3.4-3.9 show the data for these examples. While Examples 1-16 and 22-24 use only one, Examples 17-21 use fifteen specifications on crude feed quality. These properties and their corresponding linearly-additive indexes are listed in Table 3.3. While Examples 1 and 2 are relatively small-size, 3-6 are medium-size, and 7-24 are large-size. Examples 7-24 are well-representative of the actual scenarios in most refineries. The scheduling horizon is 2 days (seven 8-h periods) in Example 2, 3 days (nine 8-h periods) in Example 1, 7 days in Examples 3-6; 14 days in Examples 7-21; and 20 days in Examples 22-24. All examples have one SBM, except Example 2 that has four jetties but no SBM. Example 2 has 17 single-parcel vessels, 7 storage tanks, and two CDUs. Examples 3-6 have two VLCCs (8 parcels), 8 tanks, and 3 CDUs. Examples 7-15 and 17-19 have three VLCCs

(12 parcels), 8 tanks, and 3 CDUs. Examples 16 and 20-21 have 3 jetties, 15 parcels, 8 tanks, and 3 CDUs. Finally, Examples 22-24 have four VLCCs (16 parcels), 8 tanks, and 3 CDUs, Except for Examples 14 and 15, the first vessel arrives at time zero in all examples. It should be clear that the test examples vary widely in structure, size, scale, and complexity and are representative of industrial scenarios. CPLEX 9.0/GAMS 21.4 on a Dell workstation PWS650 (Intel® Xeon™ CPU 3.06GHZ, 3.5 GB memory) running Windows XP are used.

Table 3.10 shows the solution statistics for Examples 1-21. Note that DICOPT, BARON, LA, and RA fail to solve most examples. In contrast, RRA and RRA-P are able to solve all examples. Thus, RRA and RRA-P are far more robust than the other algorithms.

Interestingly, while BARON can guarantee global solutions, it can solve Examples 1 and 2 only, and even for them, it requires huge computational times. Although DICOPT cannot guarantee global solutions, it is able to solve Examples 5 and 6, but requires huge computation times. In addition, it also gives inferior solutions compared to RRA. As it can be seen from Examples 14, 19 and 21, RRA reduces to RA, when RA can find feasible solutions.

Table 3.4 Vessel arrival data for Examples 2-24

Example	Arrival Period	Vessel (Crude-Parcel Size kbbl or kton*)
2	1	V1 (C1-3), V2 (C1-3)
	2	V3 (C1-3), V4 (C1-3)
	3	V5 (C1-5), V6 (C1-5), V7 (C1-3), V8 (C1-3)
	4	V9 (C1-3)
	5	V10 (C2-5), V11 (C6-5), V12 (C2, 3.5), V13 (C4-3.5)
	6	V14 (C1-3), V15 (C1-3)
	7	V16 (C4-3), V17 (C2-1.5, C6-1.5)
3	1	VLCC-1 (C2-10, C3-250, C4-300, C5-190)
	14	VLCC-2 (C5-10, C6-250, C3-250, C8-240)
4-6	1	VLCC-1 (C2-10, C3-250, C4-300, C5-190)
	14	VLCC-2 (C5-10, C6-250, C3-250, C8-240)
7-11,	1	VLCC-1 (C2-10, C3-350, C4-200, C5-300)
17-19	16	VLCC-2 (C5-10, C6-200, C8-250, C3-240)
	28	VLCC-3 (C3-10, C6-250, C2-250, C7-190)
12	1	VLCC-1 (C2-10, C3-350, C4-200, C5-300)
	16	VLCC-2 (C5-10, C6-200, C8-250, C3-240)
	28	VLCC-3 (C3-10, C6-250, C2-250, C7-190)
13	1	VLCC-1 (C2-10, C3-350, C4-200, C5-300)
	16	VLCC-2 (C5-10, C6-200, C8-250, C3-240)
	28	VLCC-3 (C3-10, C6-250, C2-250, C7-190)
14	6	VLCC-1 (C2-10, C3-250, C4-200, C5-350)
	21	VLCC-1 (C5-10, C6-250, C8-200, C3-240)
	33	VLCC-3 (C3-10, C6-250, C2-300, C7-190)
15	4	VLCC-1 (C7-10, C1-250, C6-200, C5-240)
	19	VLCC-2 (C5-10, C7-250, C4-300, C2-190)
	31	VLCC-3 (C2-10, C8-300, C3-200, C6-250)
16	1	VLCC-1 (C2-10, C6-100, C8-100, C4-90)
&	3	V1 (C2-125)
20-21	4	V2 (C5-125), V3 (C3-100)
	5	V4 (C7-120)
	21	VLCC-2 (C4-10, C8-130, C3-120, C2-100)
	23	V5 (C6-100), V6 (C1-90)
	24	V7 (C7-125)
22-23	1[4]	VLCC-1 (C3-10, C5-200, C7-250, C2-190)
&	4[7]	V1 (C1-150), V2 (C6-220)
[24]	5[8]	V3 (C2-180), V4 (C4-150)
	6[9]	V5 (C8-230)
	20	VLCC-2 (C2-10, C1-300, C7-240, C5-190)
	20	V6 (C4-160), V7 (C6-210)
	21	V8 (C7-270)
	32	V9 (C8-200), V10 (C4-250), V11 (C2-180), V12 (C3-150)
	46	VLCC-3 (C5-10, C2-200, C8-170, C3-180)
	47	V13 (C1-300), V14 (C7-250)

Arrival periods in [ ] are for Example 24, while the alternatives are for Examples 22-23
* kton for Example 2

Table 3.5 Tank capacities, heels, and initial inventories for Examples 2-24

	Capacity (kbbl or kton*)									Heel (kbbl or kton*)			
Tank	Ex 2	Ex 3-4	Ex 5-9 & 13	Ex 10-11 & 17-18	Ex 12	Ex 14-15	Ex 16	Ex 19-21	Ex 22-24	Ex 2	Ex 3-11, 13 & 16-19	Ex 15 & 20-21	Ex 12, 14 & 22-24
T1	25	570	570	570	980	600	570	570	700	0	60	50	60
T2	25	570	570	570	980	600	570	570	700	0	60	50	60
T3	25	570	570	570	980	600	570	570	700	0	60	50	60
T4	25	980	980	980	980	600	570	980	700	0	110	50	60
T5	40	980	980	980	980	600	570	570	700	0	110	50	60
T6	40	980	570	570	980	600	570	570	700	0	60	50	60
T7	40	570	570	570	980	600	570	570	700	0	60	50	60
T8	0	570	570	980	980	600	570	980	700	0	60	50	60

	Allowable Crude (Class)		Initial Inventory (kbbl or kton*)									
Tank	Ex 2	Ex 3-24	Ex 2	Ex 3-9	Ex 10-11	Ex 12	Ex 13 & 16	Ex 14	Ex 15	Ex 17-18	Ex 19-21	Ex 22-24
T1	C1 (1)	C1-C4 (1)	5	350	400	320	350	420	300	450	350	350
T2	C2 (1)	C5-C8 (2)	6	400	400	400	400	320	350	400	400	300
T3	C3 (1)	C5-C8 (2)	7	350	350	400	350	400	250	350	350	350
T4	C1 (1)	C5-C8 (2)	8	950	950	900	950	280	400	950	950	250
T5	C1 (1)	C5-C8 (2)	8	300	300	280	300	300	300	300	300	210
T6	C2, C4, C6 (1)	C1-C4 (1)	20	80	80	80	80	100	160	80	80	80
T7	C3 (1)	C1-C4 (1)	10	80	80	80	80	80	80	80	80	80
T8	-	C1-C4 (1)	0	450	450	450	450	250	350	450	450	450

* kton for Example 2

Table 3.6 Initial crude amounts (kbbl or kton*) for Examples 2-24

	Ex 2				Ex 3-9, 13, 16 & 19-21				Ex 10-11				Ex 17-18			
Tank	C1 or C5	C2	C3 or C6	C4	C1 or C5	C2 or C6	C3 or C7	C4 or C8	C1 or C5	C2 or C6	C3 or C7	C4 or C8	C1 or C5	C2 or C6	C3 or C7	C4 or C8
T1	5	-	-	-	50	100	100	100	100	100	100	100	90	120	120	120
T2	-	6	-	-	100	100	100	100	100	100	100	100	100	100	100	100
T3	-	-	7	-	100	100	50	100	100	100	50	100	100	100	50	100
T4	8	-	-	-	200	250	200	300	200	250	200	300	200	250	200	300
T5	8	-	-	-	100	100	50	50	100	100	50	50	100	100	50	50
T6	-	10	5	5	20	20	20	20	20	20	20	20	20	20	20	20
T7	-	-	10	-	20	20	20	20	20	20	20	20	20	20	20	20
T8	-	-	-	-	100	100	100	150	100	100	100	150	100	100	100	150

	Ex 12				Ex 14				Ex 15				Ex 22-24			
Tank	C1 or C5	C2 or C6	C3 or C7	C4 or C8	C1 or C5	C2 or C6	C3 or C7	C4 or C8	C1 or C5	C2 or C6	C3 or C7	C4 or C8	C1 or C5	C2 or C6	C3 or C7	C4 or C8
T1	80	80	80	80	100	120	100	100	100	60	90	50	50	100	100	100
T2	100	100	100	100	80	80	80	80	100	50	100	100	50	100	50	100
T3	100	100	100	100	100	100	100	100	70	60	60	60	100	100	50	100
T4	200	250	200	250	60	80	80	60	100	100	100	100	100	50	50	50
T5	60	60	80	80	70	80	80	70	80	70	80	70	60	50	50	50
T6	20	20	20	20	25	25	25	25	45	30	50	35	20	20	20	20
T7	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20
T8	100	100	100	150	80	60	60	50	100	100	100	50	100	100	100	150

* kton for Example 2

Table 3.7 Crude concentration ranges in tanks and CDUs for Examples 2-24

Example	Crude	Concentration Range (Min-Max)	Tank & CDU
2	C1-C3	1.00-1.00	T1, T2, T3 & T4-T5
	C2, C4, C6	0.00-1.00	T6
3	C1-C8	0.00-1.00	T1-T8
	C5-C8	0.00-1.00	CDU1 & CDU2
	C1	0.15-0.85	CDU3
	C2-C4	0.00-1.00	CDU3
4-9 & 21-24	C1-C4	0.00-1.00	T1, T6-T8 & CDU3
	C5-C8	0.00-1.00	T2-T5, CDU1 & CDU2
10-11, 16 & 18	C1-C8	0.00-1.00	T1-T8
	C1-C4	0.10-0.90	CDU3
	C5-C8	0.10-0.90	CDU1 & CDU2
12-13	C1-C8	0.05-0.95	T1-T8
	C1-C3	0.10-0.90	CDU3
	C4	0.00-1.00	CDU3
	C5-C8	0.10-0.90	CDU1 & CDU2
14	C1-C8	0.05-0.95	T1-T8
	C1-C4	0.06-0.94	CDU3
	C5-C8	0.06-0.94	CDU1 & CDU2
15	C1-C4	0.05-0.95	T1, T6-T8 & CDU3
	C5-C8	0.05-0.95	T2-T5
	C5-C8	0.08-0.92	CDU1 & CDU2
17	C1-C8	0.00-1.00	T1-T8
	C1-C4	0.05-0.95	CDU3
	C5-C8	0.05-0.95	CDU1 & CDU2
19	C1-C4	0.00-1.00	T1, T6-T8 & CDU3
	C5-C8	0.00-1.00	T2-T5, CDU1 & CDU2
20	C1-C8	0.02-0.98	T1-T8
	C1-C4	0.05-0.95	CDU3
	C5-C8	0.05-0.95	CDU1 & CDU2

Table 3.8 Transfer rates, processing limits, operating costs, crude margins, and demands for Examples 2-24

Example	Flow Rate Limits (kbbl or kton*/period) Parcel-Tank Min-Max	Tank-CDU Min-Max	Demurrage (k$/period) or Sea Waiting Cost* (kYuan/period)	Changeover Loss (k$ or kYuan* /instance)	Inventory Penalty ($/bbl/period)	Inventory Cost (Yuan/ton/period)	Unloading Cost (kYuan/period)	Desired Safety Stock (kbbl)	Crude	Margin ($/bbl or kYuan/ton*) Ex 2	Ex 3-4, 7-14 & 17-19	Ex 5	Ex 6	Ex 15	Ex 16, 20-21 & 22-24
2	0-5	0-5	5	1	-	0.05	7	-	C2	1	1.70	1.70	1.50	1.75	1.75
3-6	10-400	20-45	25	10	0.20	-	-	1500	C3	1	1.50	1.50	1.50	1.55	1.85
7-14,17-19	10-400	20-40	25	10	0.02	-	-	1500	C4	1	1.60	1.60	1.50	1.80	1.25
15	10-450	20-100	20	15	0.05	-	-	1600	C5	1	1.45	1.45	1.50	1.45	1.45
16	10-250	20-50	15	5	0.20	-	-	1200	C6	1	1.60	1.60	1.50	1.70	1.65
20-21	10-250	20-50	15	5	0.20	-	-	1200	C7	-	1.55	1.55	1.50	1.60	1.55
22-24	10-300	10-80	15	5	0.20	-	-	1200	C8	-	1.60	0.50	1.50	1.65	1.60

CDU	Processing Limits (kbbl or kton*/period) Ex 2	Ex 3-6	Ex 7-14	Ex 16-21	Ex 15 & 22-24	Total Demand (kbbl) Ex 3-6	Ex 7-11	Ex 12	Ex 13	Ex 14	Ex 15	Ex 16	Ex 17 & 18	Ex 19-21	Ex 22-24
CDU 1	2-8	20-45	20-40	20-40	20-60	750	1000	900	1000	900	960	1000	1000	1000	1600
CDU 2	1-3	20-45	20-40	20-40	20-60	750	1000	900	1000	900	960	1000	1000	1000	1600
CDU 3	-	20-45	20-40	20-40	20-60	750	1000	900	900	900	960	1000	900	1000	1600

* for Example 2

Table 3.9a Specific gravities, sulfur contents, nitrogen contents, carbon residues for crudes and acceptable ranges for feeds to CDUs

Crude & CDU		Specific Gravity		Sulfur								Nitrogen		Carbon Residue	
		Ex 17 [18 & 20]	Ex 19 & [21]	Ex 1	Ex 3, 5, 6 & [4]	Ex 7 & [9-11,13]	Ex 8 & [15]	Ex 12 & [14]	Ex 16 & [22-24]	Ex 17 [18 & 20]	Ex 19 & 21	Ex 17 [18 & 20]	Ex 19 & 21	Ex 17 [18 & 20]	Ex 19 & 21
	C1	1.2576 [1.1477]	1.2057	0.01	0.0020	0.0020	0.0020 [0.0010]	0.0020	0.0050	0.0135 [0.0320]	0.0095	32.00 [30.00]	55.00	0.0620 [0.0320]	0.0450
	C2	1.2646 [1.1546	1.2339	0.01	0.0025	0.0025	0.0025	0.0025	0.0080 [0.0020]	0.0115 [0.0280]	0.0085	21.00 [25.00]	45.00	0.0450 [0.0270]	0.0420
	C3	1.2466 [1.1703]	1.2113	0.02	0.0015	0.0015	0.0015 [0.0035]	0.0015	0.0040	0.0150 [0.0276]	0.0080	45.00 [15.00]	50.00	0.0750 [0.0200]	0.0436
	C4	1.2599 [1.1852]	1.2749	0.01	0.0060	0.0060	0.0060	0.0060	0.0060	0.0120 [0.025]	0.0090	25.00 [10.00]	40.00	0.0500 [0.0150]	0.0350
	C5	1.0892 [1.0800]	1.0375	-	0.0120	0.0120	0.0120 [0.0180]	0.0120	0.0150 [0.0120]	0.0075 [0.0180]	0.0250	79.00 [98.00]	93.00	0.2400 [0.1200]	0.1880
	C6	1.1207 [1.0959]	1.0615	-	0.0130	0.0130	0.0130 [0.0150]	0.0130 [0.0100]	0.0100	0.0050 [0.0165]	0.0235	62.00 [95.00]	88.00	0.1000 [0.0850]	0.1730
	C7	1.1105 [1.105]	1.0664	-	0.0090	0.0130 [0.0090]	0.0090	0.0090	0.0200	0.0070 [0.0135]	0.0225	73.00 [85.00]	84.00	0.1800 [0.0800]	0.1540
	C8	1.1148 [1.1124]	1.0968	-	0.0150	0.0150	0.0150 [0.0120]	0.0150	0.0160	0.0065 [0.0120]	0.0210	65.00 [82.00]	78.00	0.1300 [0.0700]	0.1260
CDU1	Min	1.0000	1.0000	0.00	0.0010	0.0010	0.0010	0.0013	0.0100	0.0050 [0.0130]	0.0200	60.00 [80.00]	75.00	0.1000 [0.0500]	0.1000
	Max	1.1200 [1.1100]	1.0850 [1.0920]	0.01	0.0130	0.0135	0.0135 [0.0165]	0.0145	0.0165	0.0071 [0.0170]	0.0242	75.00 [96.00]	92.00	0.2000 [0.1000]	0.1800
CDU2	Min	1.0000	1.0000	0.01	0.0010	0.0010	0.0010	0.0010	0.0100	0.0050 [0.0125]	0.0200	60.00 [80.00]	75.00	0.1000 [0.0500]	0.1000
	Max	1.1200 [1.1100]	1.0900	0.02	0.0125	0.0130	0.0130 [0.0160]	0.0140	0.0150 [0.0160]	0.0070 [0.0172]	0.0245	78.00 [96.50]	91.50	0.2200 [0.1150]	0.1850
CDU3	Min	1.0000 [1.1000]	1.2000	-	0.0010	0.0010	0.0010	0.0010	0.0040 [0.0010]	0.0100 [0.0250]	0.0060	20.00 [10.00]	10.00	0.0100 [0.0100]	0.0100
	Max	1.2625 [1.1780]	1.2700	-	0.0035 [0.0030]	0.0040	0.0030 [0.0045]	0.0050	0.0075 [0.0045]	0.0138 [0.0290]	0.0092	40.00 [28.00]	54.00	0.0720 [0.0300]	0.0440

Data in [ ] are for corresponding [Examples]

Table 3.9b Pour points, freeze points, flash points, smoke points, Ni contents and Reid vapor pressures for crudes and acceptable ranges for feeds to CDUs

Crude & CDU		Pour Point		Freeze Point		Flash Point		Smoke Point		Ni		Reid Vapor Pressure	
		Ex 17 [18 & 20]	Ex 19 & 21	Ex 17 [18 & 20]	Ex 19 & 21	Ex 17 [18 & 20]	Ex 19 & 21	Ex 17 [18 & 20]	Ex 19 & 21	Ex 17 [18 & 20]	Ex 19 & 21	Ex 17 [18 & 20]	Ex 19 & 21
	C1	13.3290 [8.3994]	58.0549	36.3479 [133.2897]	270.2996	341.6801 [19.7932]	207.7017	588.3175 [538.0525]	548.1218	0.900 [0.095]	0.075	133.1031 [162.1270]	153.6366
	C2	27.7594 [14.0162]	12.0466	10.7753 [102.0263]	211.3251	337.8010 [28.0269]	551.5897	706.1862 [561.3248]	588.8047	0.760 [0.085]	0.062	92.4005 [138.0196]	120.4380
	C3	10.3347 [25.2344]	21.8409	54.8962 [77.0306]	248.0304	357.1811 [38.9152]	311.3055	557.2127 [578.8451]	567.6004	1.000 [0.082]	0.050	158.3207 [119.7402]	144.8884
	C4	18.8718 [58.0549]	10.3347	26.2774 [46.0414]	168.4381	341.4362 [58.3937]	661.2327	608.9218 [632.1359]	626.5365	0.850 [0.070]	0.035	116.6061 [110.6941]	113.5842
	C5	2.5140 [3.9516]	5.1896	812.1963 [701.2393]	1412.5240	2.1225 [2.1719]	16.5062	419.5659 [478.6314]	431.4538	12.500 [0.030]	19.000	28.6244 [73.9050]	24.1774
	C6	9.3209 [5.4896]	4.6626	712.1419 [589.7132]	1286.6348	4.5253 [2.7114]	21.3079	536.3974 [530.977]	455.4485	5.400 [0.0235]	18.300	19.4676 [67.2142]	22.5324
	C7	3.9516 [7.9708]	48.4716	757.3304 [536.1761]	1015.0334	2.6029 [3.1074]	29.5074	427.1329 [561.3248]	477.3611	10.300 [0.012]	17.500	26.2276 [59.0405]	21.1838
	C8	7.1733 [10.3347]	7.5624	744.1575 [512.9720]	768.6957	3.3367 [3.6938]	39.4486	457.4626 [578.8451]	503.5443	7.900 [0.0092]	16.700	24.1774 [48.0583]	13.8983
CDU1	Min	2.5000 [1.0000]	4.0000	700.0000 [500.0000]	700.0000	2.0000 [2.0000]	15.0000	400.0000 [450.0000]	400.0000	5.000 [0.001]	15.000	15.0000 [40.0000]	10.0000
	Max	9.0000 [10.0000]	45.0000	810.0000 [700.0000]	1405.0000	4.5000 [3.5000]	39.0000	530.0000 [560.0000]	475.0000	12.200 [0.027]	18.800	28.3000 [70.0000]	24.0000
CDU2	Min	2.5000 [2.0000]	4.0000	700.0000 [500.0000]	700.0000	2.0000 [2.0000]	15.0000	400.0000 [450.0000]	400.0000	5.000 [0.001]	15.000	15.0000 [40.0000]	10.0000
	Max	9.2000 [10.2000]	48.0000	810.0000 [690.0000]	1410.0000	4.4800 [3.6000]	39.2000	520.0000 [570.0000]	470.0000	12.100 [0.028]	18.600	28.5000 [72.0000]	23.9000
CDU3	Min	10.0000 [8.0000]	10.0000	10.0000 [40.0000]	150.0000	300.0000 [15.0000]	200.0000	500.0000	500.0000	0.500 [0.050]	0.010	90.0000 [100.0000]	100.0000
	Max	27.5000 [50.0000]	58.0000	54.0000 [130.0000]	270.0000	350.0000 [58.0000]	650.0000	700.0000 [620.0000]	600.0000	0.980 [0.092]	0.072	155.0000 [160.0000]	150.0000

Data in [ ] are for corresponding [Examples]

Table 3.9c Asphaltenes, aromatics, paraffins, naphthenes, and viscosities for crudes and acceptable ranges for feeds to CDUs

		Asphaltenes		Aromatics		Paraffins		Naphthenes		Viscosity	
Crude & CDU		Ex 17 [18 & 20]	Ex 19 & 21	Ex 17 [18 & 20]	Ex 19 & 21	Ex 17 [18 & 20]	Ex 19 & 21	Ex 17 [18 & 20]	Ex 19 & 21	Ex 17 [18 & 20]	Ex 19 & 21
	C1	0.0350 [0.1400]	0.0850	0.0892 [0.4500]	0.2972	0.7273 [0.3140]	0.3844	0.2892 [0.2360]	0.3414	71.7767 [79.2854]	76.8625
	C2	0.0250 [0.1250]	0.0650	0.0465 [0.4200]	0.2793	0.7366 [0.3400]	0.4222	0.2835 [0.2400]	0.3203	64.5435 [78.7725]	76.2073
	C3	0.0150 [0.1200]	0.0500	0.0642 [0.3500]	0.2756	0.6341 [0.3650]	0.3614	0.2281 [0.2850]	0.3022	74.7458 [78.5639]	75.7175
	C4	0.0200 [0.1100]	0.0700	0.0635 [0.3030]	0.2713	0.7335 [0.4000]	0.4004	0.2901 [0.2970]	0.3443	68.1196 [78.4192]	76.5457
	C5	0.1150 [0.0870]	0.2000	0.3216 [0.2980]	0.5216	0.3282 [0.3540]	0.2400	0.2767 [0.3480]	0.2384	83.1872 [76.9637]	82.6218
	C6	0.0900 [0.0820]	0.1890	0.3130 [0.2960]	0.4942	0.4035 [0.3750]	0.3244	0.2200 [0.3290]	0.2302	79.3574 [76.6536]	81.5636
	C7	0.1350 [0.0500]	0.1750	0.4437 [0.2760]	0.4577	0.3500 [0.4220]	0.2756	0.2085 [0.3020]	0.2407	82.5847 [75.7175]	81.1988
	C8	0.1005 [0.0420]	0.1500	0.3599 [0.2500]	0.4317	0.3892 [0.4500]	0.3016	0.1990 [0.2950]	0.2439	81.5982 [75.4539]	80.3514
CDU1	Min	0.0900 [0.0400]	0.1500	0.3000 [0.2500]	0.4000	0.3000 [0.3500]	0.2400	0.1800 [0.2500]	0.2000	70.0000 [75.0000]	80.0000
	Max	0.1300 [0.0850]	0.1960	0.4200 [0.2960]	0.5000	0.4000 [0.4500]	0.3200	0.2700 [0.3400]	0.2420	83.0000 [76.7000]	82.5000
CDU2	Min	0.0900 [0.0400]	0.1500	0.3000 [0.2000]	0.4000	0.3000 [0.3500]	0.2400	0.1800 [0.2500]	0.2000	70.0000 [75.0000]	80.0000
	Max	0.1320 [0.0840]	0.1950	0.4000 [0.2950]	0.5200	0.3950 [0.4400]	0.3200	0.2750 [0.3450]	0.2410	82.9000 [76.8000]	82.6000
CDU3	Min	0.0150 [0.1000]	0.0500	0.0400 [0.3000]	0.2500	0.6000 [0.3000]	0.3500	0.2000	0.3000	60.0000 [70.0000]	70.0000
	Max	0.0320 [0.1360]	0.0800	0.0850 [0.4350]	0.2950	0.7350 [0.3900]	0.4200	0.2900 [0.2950]	0.3440	74.5000 [79.0000]	76.8000

Data in [ ] are for corresponding [Examples]

The solution times for RRA range from 28 CPU min to 5.5 h for large examples (7-21). For examples (17-21) with 15 quality specifications, they range from 49.8 min to 5.5 h. In contrast, the solution times for RRA-P range from 47.0 min to 2.1 h for Examples 7-21. Except for Example 7, RRA-P is much faster than RRA and reduces solution times by anywhere from 0% to 99.3%. For instance, RRA-P requires 1.6 h for Example 16 compared to 4.1 h for RRA. As expected, RRA-P is faster than RRA especially for large examples (22-24). A profit of $ 7370.13K is obtained within 16 min for Example 22, $ 7349.32K in 31 min for Example 23, and $ 7426.09K in 14 min for Example 24, while RRA and other algorithms fail to solve these problems. This clearly shows the merit of using the proposed partial relaxation strategy and demonstrates that RRA-P is in fact much faster and more robust than RRA. However, note that the quality of RRA-P solution need not be better than that of RRA. For instance, the best profit for RRA-P is $ 4016.65K compared to $ 4045.79K for RRA in Example 14.

3.7 Solution Quality

Having achieved substantial improvements in robustness and solution efficiency, schedule quality should be taken into account. As mentioned at the outset, while it is not the intention to seek guaranteed globally optimal solutions in this work, the best is tried to get the best solutions possible and more importantly would like to develop a way to estimate the quality of the obtained solutions. To this end, how to improve the schedule given by RRA-P is first discussed.

Table 3.10 Solution statistics for various algorithms/codes

Example	Statistics	DICOPT	BARON	LA	RA	RRA	RRA-P	RRA-P1	IRRA-P1	RLA
1	CPU Time (s)	200	1824	-	-	2	2	2	-	1
	Profit (k$)	N/A	5069.94	N/A	N/A	5069.94	5069.94	5069.94	5069.94	5069.94
2	CPU Time (s)	-	6001	-	-	3	3	3	-	3
	Profit (k$)	N/A	1.0145E+08	N/A	N/A	1.0119E+08	1.0119E+08	1.0119E+08	1.0121E+08	1.0119E+08
3	CPU Time (s)	42689	60000	-	-	255	223	258	-	20263
	Profit (k$)	N/A	N/A	N/A	N/A	3417.29	3437.62	3456.67	3466.43	3367.10
4	CPU Time (s)	30669	60000	-	-	174	56	64	-	802
	Profit (k$)	N/A	N/A	N/A	N/A	3422.12	3391.40	3427.90	3432.44	3360.96
5	CPU Time (s)	22384	60000	-	-	152	79	149	-	912
	Profit (k$)	3000.13	N/A	N/A	N/A	3065.86	2993.80	3086.59	3128.00	3054.10
6	CPU Time (s)	17061	60000	-	-	277	28	40	-	201
	Profit (k$)	3295.00	N/A	N/A	N/A	3315.00	3325.00	3345.00	3355.00	3250.00
7	CPU Time (s)	-	60000	-	-	1694	7511	7614	-	1978
	Profit (k$)	N/A	N/A	N/A	N/A	4514.60	4594.03	4622.76	4639.40	4604.09
8	CPU Time (s)	38033	60000	-	-	6354	47	332	-	5252
	Profit (k$)	N/A	N/A	N/A	N/A	4533.35	4590.41	4605.16	4623.16	4549.67
9	CPU Time (s)	55226	60000	-	-	1796	52	110	-	1453
	Profit (k$)	N/A	N/A	N/A	N/A	4567.58	4642.37	4644.75	4670.13	4599.14
10	CPU Time (s)	33404	60000	-	-	5407	1566	1598	-	800
	Profit (k$)	N/A	N/A	N/A	N/A	4578.57	4611.29	4611.40	4630.38	4575.80
11	CPU Time (s)	40155	60000	-	-	5494	999	1484	-	3878
	Profit (k$)	N/A	N/A	N/A	N/A	4542.78	4588.19	4611.25	4615.79	4507.42

N/A = a feasible solution was either not obtained at all or not obtained within the specified time

Example	Statistics	DICOPT	BARON	LA	RA	RRA	RRA-P	RRA-P1	IRRA-P1	RLA
12	CPU Time (s)	49143	60000	-	-	8583	188	226	-	2802
	Profit (k$)	N/A	N/A	N/A	N/A	4150.61	4146.09	4165.28	4177.43	4130.48
13	CPU Time (s)	51538	60000	-	-	12584	5208	5416	-	19427
	Profit (k$)	N/A	N/A	N/A	N/A	4414.09	4429.52	4443.48	4451.45	4393.72
14	CPU Time (s)	70000	60000	-	6864	6864	373	693	-	58831
	Profit (k$)	N/A	N/A	N/A	4045.79	4045.79	4016.65	4078.24	4100.39	3946.39
15	CPU Time (s)	166131	60000	-	-	14589	1958	2395	-	31455
	Profit (k$)	N/A	N/A	N/A	N/A	4424.47	4468.43	4508.06	4539.64	4514.32
16	CPU Time (s)	68157	60000	-	-	14815	5860	5976	-	10425
	Profit (k$)	N/A	N/A	N/A	N/A	4718.75	4716.21	4745.88	4770.26	4684.60
17	CPU Time (s)	72672	60000	-	-	3394	411	476	-	19641
	Profit (k$)	N/A	N/A	N/A	N/A	4442.74	4492.33	4492.99	4516.79	4453.49
18	CPU Time (s)	56123	60000	-	-	9497	1576	1616	-	51267
	Profit (k$)	N/A	N/A	N/A	N/A	4450.70	4450.36	4466.38	4492.75	4416.77
19	CPU Time (s)	35403	60000	-	4639	4639	328	899	-	8446
	Profit (k$)	N/A	N/A	N/A	4600.02	4600.02	4617.17	4640.52	4653.24	4580.67
20	CPU Time (s)	40356	60000	-	-	19962	827	901	-	14092
	Profit (k$)	N/A	N/A	N/A	N/A	4695.91	4715.17	4744.35	4747.38	4719.93
21	CPU Time (s)	63002	60000	-	2988	2988	244	336	-	26496
	Profit (k$)	N/A	N/A	N/A	4730.45	4730.45	4735.18	4753.10	4762.09	4688.54

N/A = a feasible solution was either not obtained at all or not obtained within the specified time

Table 3.11 Operation schedule from RRA-P1 for Example 16

Tank	Crude Amount [to CDU No.] (from Vessel No.) in kbbl for Period											
	1	2	3	4	5	6	7	8	9	10	11	12
1	−20.0[3]	−20.0[3]	−20.0[3]	−20.0[3]	−20.0[3]	−10.2[3]	−23.2[3]					
2	+100.0(2)	+10.0(3)		−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]
3	−20.0[2]	−20.0[2]	−20.0[2]	+125.0(6)		+120.0(8)						
4	−27.7[1]	−23.5[1]	−20.0[1]	−23.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−22.9[1]	−26.4[1]	−30.3[1]	−34.9[1]	−29.7[1]
5		+90.0(3)										
6	+10.0(1)		+80.0(4)									
			+10.0(5)									
7				+18.3(5)				−26.7[3]	−30.7[3]	−35.3[3]	−30.0[3]	−25.5[3]
				+10.0(7)	+90.0(7)							
8			+10.0(4)									
			+96.7(5)									
	13	14	15	16	17	18	19	20	21	22	23	24
1									+10.0(9)	+110.0(11)		+10.0(14)
2	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]
3									−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]
4	−34.0[1]	−39.2[1]	−40.0[1]	−40.0[1]	−34.0[1]	−28.9[1]	−24.6[1]	−20.9[1]	+130.0(10)		+100.0(13)	
5												
6										+10.0(11)	+100.0(12)	
7												+80.0(14)
8	−21.7[3]	−20.0[3]	−20.0[3]	−20.0[3]	−20.0[3]	−20.0[3]	−20.0[3]	−20.0[3]	−20.0[3]	−20.0[3]	−20.0[3]	−20.0[3]
	25	26	27-31	32	33	34	35	36	37	38	39	40-42
1						−20.3[3]	−22.2[3]	−25.5[3]	−28.9[3]	−32.7[3]	−37.0[3]	−40.0[3]
2	+125.0(15)											
3	−20.0[1]	−20.0[1]										
4	−20.0[2]	−20.0[2]	−20.0[2]	−20.8[2]	−23.9[2]	−27.5[2]	−31.6[2]	−36.3[2]	−40.0[2]	−40.0[2]	−40.0[2]	−40.0[2]
5			−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]
8	−20.0[3]	−20.0[3]	−20.0[3]	−20.0[3]	−20.0[3]							

'−' sign represents delivery to [CDU], '+' sign represents receipt from (Parcels)

Let **S** denote the solution from RRA-P. two series of feasible schedules (i.e. with no composition discrepancy) from **S** can be obtained by solving MILPs and BLPs repeatedly. First, the values of the binary variables is taken from **S**, and fixed in the exact MIBLP (involving eqs. 3.1 and 3.2) to those values to get a BLP. Clearly, a solution **S**1 to this BLP is a schedule with no composition discrepancy. The compositions of tanks are extracted from **S**1 and fixed in the MIBLP to get an MILP. This alternating series of MILP and BLP is continued, until the solutions of successive BLPs converge. Now, instead of solving a BLP first, an MILP based on **S** could be also solved and another series of solutions could be obtained. Figure 3.4 shows the complete iterative improvement procedure. The procedure normally takes about 8-10 (MILP+BLP) solutions and the best of all these solutions is taken. This enhanced RRA-P is called as RRA-P1. Table 3.10 shows that RRA-P1 improves the solutions from RRA-P by an average of 0.60% in all examples and they are better than those from RRA by an average of 0.90%. Furthermore, the additional (MILP+BLP) solutions do not demand excessive additional computing effort. For illustration, the best crude schedule from RRA-P1 for Example 16 is presented in Table 3.11.

While RRA-P1 improves the quality from a given solution from RRA-P, RRA-P1 is also used repeatedly to get alternate and better solutions. Given the best solution from the previous iterations of RRA-P1, an integer cut is imposed to prohibit the best combination of XT_{it} from recurring. However, this cannot guarantee a better solution. If a better solution is got, then a lower bound on the profit is imposed during the last phase of RRA-P1, where MILPs and BLPs are solved. If a better solution is not got, then imposing a lower bound on profit will most likely result in no solution, so an integer cut is merely used to eliminate this solution and return to RRA-P1. This algorithm is called IRRA-P1. IRRA-P1 is computationally more expensive than RRA-P1 by a factor of around 10, and it needs to check if it improves quality substantially. Examples 1-21 are solved with IRRA-P1 to compare with RRA-P1. Table 3.10 shows that IRRA-P1 improves solution quality by 0% to 1.34% for all examples and 0.37% on an average. Although attractive, this improvement may not be worth the additional substantial computational effort of IRRA-P1.

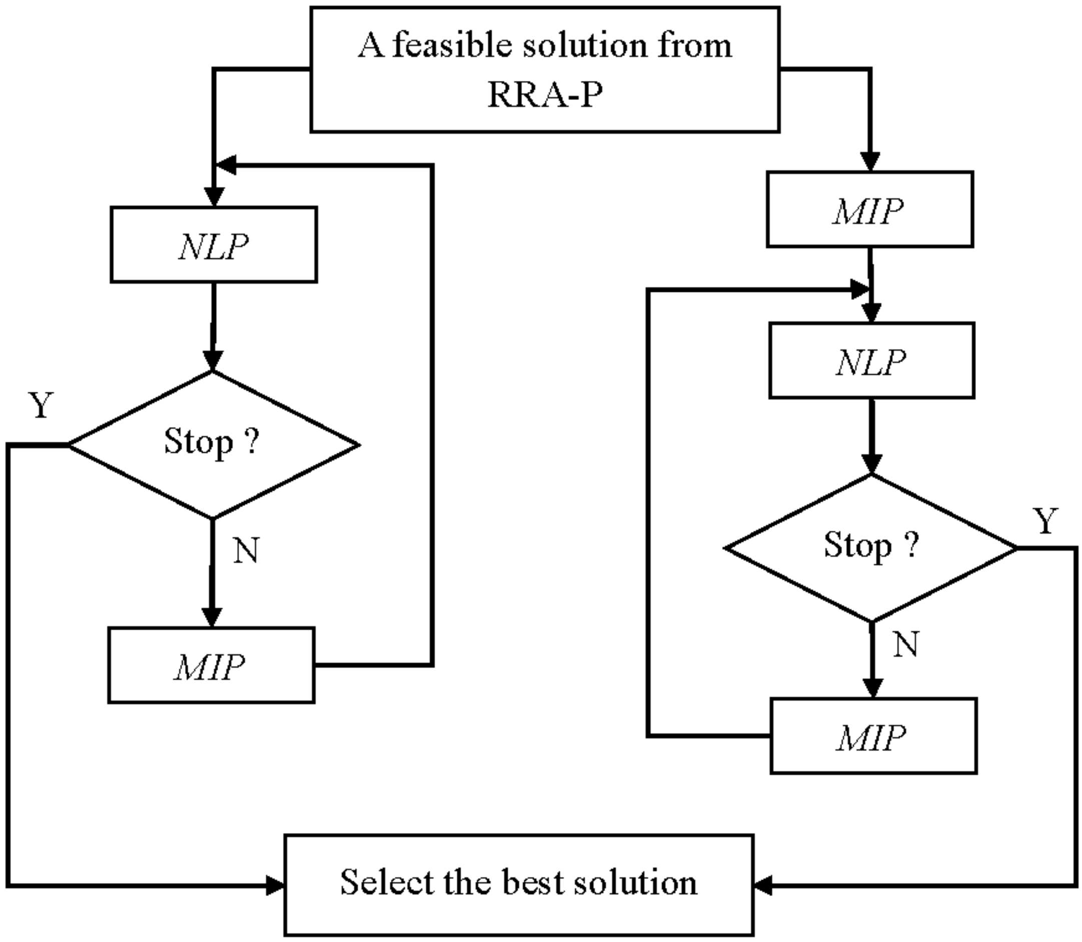

Figure 3.4 Flow chart for RRA-P1 (Partial Relaxation Refinement Strategy)

3.8 Upper Bound on Profit

The algorithms presented so far in this paper carry no guarantee of optimality with respect to the original and full MIBLP. It is reasonable to ask how close their solutions are to the global optima of the original MIBLP. Of course, a global optimization algorithm for this problem would be desirable, but considering the large sizes of practical problems and the need for quick solutions, that will require considerable effort and is best left for the future. However, even without such an algorithm, reasonable conservative estimates of solution quality can be obtained based on a theoretical upper bound on the profit. Recall that in the first MILP of RRA, the bilinear constraints (eqs. 3.1 & 3.2) are linearized for most periods after the arrival period of the first vessel. For periods before the arrival of the first vessel, the bilinear constraints (eqs. 3.1 & 3.2) become linear because the initial tank compositions are known. Since the first MILP of RRA is a linear relaxation of **F**, its optimal solution is a valid upper bound for **F**. However, the first MILP of RRA is the largest and the most difficult to solve and often cannot be solved to zero relative gap for medium and large-size problems. Therefore, the best possible integer solution from the first MILP of RRA is used as the upper bound, because the best possible integer solution is a valid upper bound on the optimal solution. Because the linearization is not very tight, other constraints are needed to improve this upper bound. To this end, several novel tightening constraints are now presented. The only reason why these novel constraints are not in RRA and RRA-P is that their inclusion makes MILPs intractable for most problems.

Apart from the initial periods in which all tank compositions are known (= f_{ic0}) in the first iteration, RRA approximates eqs. 3.1 and 3.2 in all the remaining periods. The linear estimators of eqs. 3.1 and 3.2 are not as tight as the best known linearizations of McCormick (1976) for bilinear constraints. Thus, the first tightening constraints are the following derived from Relaxation Linearization Technology (McCormick, 1976).

$$FCTU_{iuct} \geq f_{ict}FTU_{iu}^{L} + xt_{ic}^{L}FTU_{iut} - xt_{ic}^{L}FTU_{iu}^{L} - FTU_{iu}^{L}(1-Y_{iut})$$

$$(i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC} \quad (3.11a)$$

$$FCTU_{iuct} \geq f_{ict}FTU_{iu}^{U} + xt_{ic}^{U}FTU_{iut} - xt_{ic}^{U}FTU_{iu}^{U} \quad (i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC} \quad (3.11b)$$

$$FCTU_{iuct} \leq f_{ict}FTU_{iu}^{U} + xt_{ic}^{L}FTU_{iut} - xt_{ic}^{L}FTU_{iu}^{U} \quad (i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC} \quad (3.11c)$$

$$FCTU_{iuct} \leq f_{ict}FTU_{iu}^{L} + xt_{ic}^{U}FTU_{iut} - xt_{ic}^{U}FTU_{iu}^{L} + FTU_{iu}^{L}(1-Y_{iut})$$

$$(i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC} \quad (3.11d)$$

$$VCT_{ict} \geq f_{ict}V_{i}^{L} + xt_{ic}^{L}V_{it} - xt_{ic}^{L}V_{i}^{L} \quad (i, c) \in \boldsymbol{IC} \quad (3.12a)$$

$$VCT_{ict} \geq f_{ict}V_{i}^{U} + xt_{ic}^{U}V_{it} - xt_{ic}^{U}V_{i}^{U} \quad (i, c) \in \boldsymbol{IC} \quad (3.12b)$$

$$VCT_{ict} \leq f_{ict}V_{i}^{U} + xt_{ic}^{L}V_{it} - xt_{ic}^{L}V_{i}^{U} \quad (i, c) \in \boldsymbol{IC} \quad (3.12c)$$

$$VCT_{ict} \leq f_{ict}V_{i}^{L} + xt_{ic}^{U}V_{it} - xt_{ic}^{U}V_{i}^{L} \quad (i, c) \in \boldsymbol{IC} \quad (3.12d)$$

where, xt_{ic}^{L} and xt_{ic}^{U} are the limits on the composition of crude *c* in tank *i*, FTU_{iu}^{L} and FTU_{iu}^{U} are the limits on the amount of crude charge per period from tank *i* to CDU *u*, and V_{i}^{L} and V_{i}^{U} are the limits on crude inventory in tank *i*. Note that these are not written for initial periods in which the tank compositions are known to be f_{ic0}, but only for those periods in which they are unknown.

In most scenarios, not all tanks may receive crude, when the first ship arrives. Thus, their compositions will remain as f_{ic0} even after that. However, the first iteration of RRA uses exact linearizations of eqs. 3.1 and 3.2 only for those periods in which the tank compositions are f_{ic0} without a doubt. This requirement can be relaxed further by forcing a tank composition to be f_{ic0}, until it receives a crude. The transfer of crude to a tank can be detected by looking at the value of $\sum_{\tau \le t} XT_{i\tau}$. As long as this remains zero, the composition of tank i must remain f_{ic0}. To enforce this idea, the following two constraints are used.

$$FTU_{iu}^{U} xt_{ic}^{U} \sum_{\tau \le t} XT_{i\tau} + FCTU_{iuct} \ge FTU_{iut} f_{ic0} \qquad (i,u) \in \boldsymbol{IU},\ (i,c) \in \boldsymbol{IC} \qquad (3.13a)$$

$$FTU_{iu}^{U} xt_{ic}^{U} \sum_{\tau \le t} XT_{i\tau} + FTU_{iut} f_{ic0} \ge FCTU_{iuct} \qquad (i,u) \in \boldsymbol{IU},\ (i,c) \in \boldsymbol{IC} \qquad (3.13b)$$

Similar constraints can also be written for f_{ict} in those periods, where the tank composition is unknown.

$$f_{ict} \ge f_{ic0} - xt_{ic}^{U} \sum_{\tau \le t} XT_{i\tau} \qquad (i,c) \in \boldsymbol{IC} \qquad (3.14a)$$

$$f_{ict} \le f_{ic0} + xt_{ic}^{U} \sum_{\tau \le t} XT_{i\tau} \qquad (i,c) \in \boldsymbol{IC} \qquad (3.14b)$$

If a tank i receives no crude during a period t, then its composition must remain constant. In other words,

$$f_{ict} \ge f_{ic(t-1)} - xt_{ic}^{U} XT_{it} \qquad (i,c) \in \boldsymbol{IC} \qquad (3.15a)$$

$$f_{ict} \le f_{ic(t-1)} + xt_{ic}^{U} XT_{it} \qquad (i,c) \in \boldsymbol{IC} \qquad (3.15b)$$

Furthermore, the individual crude fractions in each tank must sum to 1.

$$\sum_{c} f_{ict} = 1 \qquad (i,c) \in \boldsymbol{IC} \qquad (3.16)$$

If only one tank i feeds CDU u during period t, then the crude quality in tank i at the end of t must satisfy the crude quality constraints for CDU u.

$$V_i^{U} \theta_{ku}^{L} \left(1 - Y_{iut} + \sum_{ii \ne i} Y_{(ii)ut} \right) + \sum_{c} VCT_{ict} \theta_{kc} \ge V_{it} \theta_{ku}^{L}$$

$$(ii,u) \in \boldsymbol{IIU},\ (i,u) \in \boldsymbol{IU},\ (i,c) \in \boldsymbol{IC} \qquad (3.17a)$$

$$\sum_c VCT_{ict}\theta_{kc} \le V_{it}\theta_{ku}^U + \left(\sum_c V_i^U xt_{ic}^U \theta_{kc}\right)\left(1 - Y_{iut} + \sum_{ii \ne i} Y_{(ii)ut}\right)$$

$(ii, u) \in \boldsymbol{IIU}$, $(i, u) \in \boldsymbol{IU}$, $(i, c) \in \boldsymbol{IC}$ (3.17b)

$$\theta_{ku}^L \left(\sum_c V_i^U xt_{ic}^U \rho_c\right)\left(1 - Y_{iut} + \sum_{ii \ne i} Y_{(ii)ut}\right) + \sum_c VCT_{ict}\rho_c\theta_{kc} \ge \theta_{ku}^L \left(\sum_c VCT_{ict}\rho_c\right)$$

$(ii, u) \in \boldsymbol{IIU}$, $(i, u) \in \boldsymbol{IU}$, $(i, c) \in \boldsymbol{IC}$ (3.17c)

$$\sum_c VCT_{ict}\rho_c\theta_{kc} \le \theta_{ku}^U \left(\sum_c VCT_{ict}\rho_c\right) + \left(\sum_c V_i^U xt_{ic}^U \rho_c\theta_{kc}\right)\left(1 - Y_{iut} + \sum_{ii \ne i} Y_{(ii)ut}\right)$$

$(ii, u) \in \boldsymbol{IIU}$, $(i, u) \in \boldsymbol{IU}$, $(i, c) \in \boldsymbol{IC}$ (3.17d)

where, $\boldsymbol{IIU}$ = {(ii, u)| tank ii can feed CDU u}.

Similarly, the following can also be obtained by using f_{ict}.

$$\sum_c f_{ict}\theta_{kc} \ge \theta_{ku}^L - \theta_{ku}^L \left(1 - Y_{iut} + \sum_{ii \ne i} Y_{(ii)ut}\right) \quad (ii, u) \in \boldsymbol{IIU}, (i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC} \quad (3.18a)$$

$$\sum_c f_{ict}\theta_{kc} \le \theta_{ku}^U + \left(\sum_c xt_{ic}^U \theta_{kc}\right)\left(1 - Y_{iut} + \sum_{ii \ne i} Y_{(ii)ut}\right)$$

$(ii, u) \in \boldsymbol{IIU}$, $(i, u) \in \boldsymbol{IU}$, $(i, c) \in \boldsymbol{IC}$ (3.18b)

$$\sum_c f_{ict}\rho_c\theta_{kc} \ge \theta_{ku}^L \left(\sum_c f_{ict}\rho_c\right) - \theta_{ku}^L \left(\sum_c xt_{ic}^U \rho_c\right)\left(1 - Y_{iut} + \sum_{ii \ne i} Y_{(ii)ut}\right)$$

$(ii, u) \in \boldsymbol{IIU}$, $(i, u) \in \boldsymbol{IU}$, $(i, c) \in \boldsymbol{IC}$ (3.18c)

$$\sum_c f_{ict}\rho_c\theta_{kc} \le \theta_{ku}^U \left(\sum_c f_{ict}\rho_c\right) + \left(\sum_c xt_{ic}^U \rho_c\theta_{kc}\right)\left(1 - Y_{iut} + \sum_{ii \ne i} Y_{(ii)ut}\right)$$

$(ii, u) \in \boldsymbol{IIU}$, $(i, u) \in \boldsymbol{IU}$, $(i, c) \in \boldsymbol{IC}$ (3.18d)

If only one tank i feeds a CDU u during t, then its concentration must satisfy the crude quality constraints of CDU u,

$$f_{ict} \ge xc_{cu}^L - xc_{cu}^L \left(1 - Y_{iut} + \sum_{ii} Y_{(ii)ut}\right) \quad (ii, u) \in \boldsymbol{IIU}, (i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC} \quad (3.19a)$$

$$f_{ict} \le xc_{cu}^{U} + xt_{ic}^{U}\left(1 - Y_{iut} + \sum_{ii} Y_{(ii)ut}\right) \qquad (ii, u) \in \boldsymbol{IIU}, (i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC} \qquad (3.19b)$$

$$VCT_{ict} \ge V_{it} xc_{cu}^{L} - V_{i}^{U} xc_{cu}^{L}\left(1 - Y_{iut} + \sum_{ii} Y_{(ii)ut}\right)$$

$$(ii, u) \in \boldsymbol{IIU}, (i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC} \qquad (3.19c)$$

$$VCT_{ict} \le V_{it} xc_{cu}^{U} + V_{i}^{U} xt_{ic}^{U}\left(1 - Y_{iut} + \sum_{ii} Y_{(ii)ut}\right)$$

$$(ii, u) \in \boldsymbol{IIU}, (i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC} \qquad (3.19d)$$

Where, xc_{cu}^{L} and xc_{cu}^{U} are limits on the composition of crude c in feed to CDU u.

While eqs. 3.12a to 3.19d are non-redundant, many of them are big-M constraints. The sheer number of these constraints makes the problem too large to be practical for routine use. This is why these constraints are only used for obtaining tight upper bounds only and not inside RRA-P1.

3.8.1 Deviations from Upper Bounds

Examples 1-21 are solved by adding eqs. 3.12a to 3.19d to obtain upper bounds (Table 3.12). Their addition makes the medium- to large-size problems hard to solve to zero relative gaps. Therefore, they have to be solved within some small relative gaps (e.g. 3% for medium-size and 5% for large-size problems). In such cases, the best possible integer solution is used as the upper bound. Clearly, these are conservative estimates of upper bounds. However, from Table 3.12, it can be seen that eqs. 3.12a to 3.19d do produce upper bounds lower than those obtained from the first iterations of RRA.

Solution quality is assessed as a percent deviation of the best solution from its upper bound. The deviations are below 3% and 6% for medium-size and large-size problems respectively except Example 5 whose deviation is about 8.8%. While the deviation for Example 1 is 1.59%, the solution from RRA-P1 is indeed the global optimum as confirmed by BARON. The solution for Example 2 is very near the global optimum, because the deviation is about 0.40%.

It is possible to explain the large deviation (8.8%) for Example 5. As it can be seen from Tables 3.4-3.9, the margin for crude 8 is very different from those of crudes 1-7. Since the MILP for the upper bound problem does involve relaxation and cannot guarantee a solution free from composition discrepancy, the solver has the freedom to feed crudes 1-7 preferentially over crude 8 during the upper bound problem. This naturally makes the upper bound and causes a large deviation. Indeed, the results of Example 6 lend further support to the explanation. The crude margins (Tables 3.4-3.9) are very similar in Example 6, and the deviation is indeed small (0.53%).

3.9 NLP-Based Strategy

It is clear that the linear approximation of bilinear constraints is the root cause of composition discrepancy. The algorithms discussed so far used the rolling-horizon type procedure of Reddy et al. (2004a,b) to avoid discrepancy. An alternate strategy that does not “decompose” the problem along the time dimension is to solve an alternating series of MILP and NLP as proposed by Li et al. (2002). This strategy is very similar to the final MILP-NLP refinement used in RRA-P1. Li et al. (2002) solve an MILP approximation (MIP I) in the first iteration. They examine this solution to see if the concentration of crudes in each storage tank is the same as that of the feed from that tank. They call this as condition I in Li et al. (2002). If condition I is satisfied, then an optimal solution is found and their procedure terminates. Otherwise, they fix the integer map from MIP I into the original MINLP model to get an NLP, whose solution ensures composition consistency. If the gap between the objectives of NLP and MIP I satisfies a tolerance (condition II), then the procedure terminates. Otherwise, the tank compositions obtained from the NLP are fixed in the original MINLP model to get a new MILP (MIP II in Li et al., 2002). If the objective value from MIP II is better than that from NLP and the integer map obtained from MIP II is different from the fixed integer map in NLP (condition III), then the integer map from MIP II is fixed into the original MINLP model to get another NLP, and the iterations are repeated. Otherwise, the procedure again terminates. However, as pointed out by Reddy et al. (2004a,b), this algorithm fails, whenever the integer map from MIP I yields an infeasible NLP.

Some key ideas described and used earlier in RRA-P, namely integer cuts, tightening constraints, and slack variables, can also be employed successfully in the NLP-based algorithm (LA) of Li et al. (2002). To see if such a strategy can be more efficient or superior, LA is modified to obtain a revised algorithm (RLA), and tested it along with other algorithms in this paper. RLA adds eqs. 3.11, 3.12, 3.13, and 3.17 to tighten the MILP approximation in RRA and uses the same integer cuts as in RRA. This modified MILP approximation in RLA is called as MIP-1.

Table 3.12 The upper bound for Examples 1-21

Example	Profit (k$)	Upper Bound1	Upper Bound 2	Solution Deviation
1	5069.94	5152.09	5185.63	0.0159
2	1.0121E+08	1.0161E+08	1.0161E+08	0.0040
3	3466.43	3554.60	3580.34	0.0248
4	3432.44	3537.27	3570.67	0.0296
5	3128.00	3429.59	3508.74	0.0879
6	3355.00	3373.00	3375.00	0.0053
7	4639.40	4759.79	4788.96	0.0253
8	4623.16	4767.26	4803.97	0.0302
9	4670.13	4797.80	4820.27	0.0266
10	4630.38	4780.07	4782.97	0.0313
11	4615.79	4776.10	4786.96	0.0336
12	4177.43	4298.61	4306.18	0.0282
13	4451.45	4614.17	4627.54	0.0353
14	4100.39	4290.00	4299.29	0.0442
15	4539.64	4802.58	4810.46	0.0547
16	4770.26	4900.82	4911.14	0.0266
17	4516.79	4639.73	4654.29	0.0265
18	4492.75	4620.00	4626.04	0.0275
19	4653.24	4804.46	4811.10	0.0315
20	4747.38	4912.52	4920.63	0.0336
21	4762.09	4949.71	4963.65	0.0379

Profit = the solution from IRRA-P1

Upper bound 1 = the best possible integer solution obtained from the first iteration of RRA with eqs. 12-19

Upper bound 2 = the best possible integer solution obtained from the first iteration of RRA

Solution deviation = the relative gap between profit and upper bound 1

It is clear that a composition discrepancy may still exist in a solution of MIP-1, which will lead to an infeasible NLP. Two slack variables in eq. 3.7 are used to get an NLP solution to avoid such infeasibility. Here, in addition, two more positive slack variables (s^{+}_{ukt} and s^{-}_{ukt}) corresponding to eqs. 3.4 and 3.5 in the NLP are also used. These relate to the other source of infeasibility, namely the feed quality requirement. Thus, eqs. 4 and 5 in the NLP are replaced by eqs. 3.20-3.21 and use the following objective in the NLP.

$$\text{Profit} = \sum_i \sum_u \sum_c \sum_t FCTU_{iuct} CP_{cu} - \sum_v DC_v - COC \sum_u \sum_t CO_{ut} - \sum_t SC_t$$
$$- M \sum_i \sum_u \sum_c \sum_t (u^{+}_{iuct} + u^{-}_{iuct}) - M \sum_u \sum_k \sum_t (s^{+}_{ukt} + s^{-}_{ukt})$$

$$\theta^{L}_{ku} FU_{ut} - s^{-}_{ukt} \le \sum_i \sum_c FCTU_{iuct} \theta_{kc} \qquad (i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC} \qquad (3.20a)$$

$$\sum_i \sum_c FCTU_{iuct} \theta_{kc} \le s^{+}_{ukt} + \theta^{U}_{ku} FU_{ut} \qquad (i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC} \qquad (3.20b)$$

$$\theta^{L}_{ku} \left(\sum_i \sum_c FCTU_{iuct} \rho_c \right) - s^{-}_{ukt} \le \sum_i \sum_c FCTU_{iuct} \rho_c \theta_{kc}$$
$$(i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC} \qquad (3.21a)$$

$$\sum_i \sum_c FCTU_{iuct} \rho_c \theta_{kc} \le s^{+}_{ukt} + \theta^{U}_{ku} \left(\sum_i \sum_c FCTU_{iuct} \rho_c \right)$$
$$(i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC} \qquad (3.21b)$$

where, M is a large number to force the slack variables to zero, whenever possible. If all the slack variables (u^{+}_{iuct}, u^{-}_{iuct}, s^{+}_{ukt}, and s^{-}_{ukt}) are zero, then the integer map obtained from MIP-1 is feasible. If not, the values of the slack variables are used to identify the sources of infeasibility, add the following slack cuts, and fix the tank concentrations obtained from the NLP (see Li et al., 2002 for further details) in MIP-1 to obtain a revised MILP approximation (MIP-2R). Note that MIP-2R does not include the tightening constraints in MIP-1.

$$\sum_{i'} FCTU_{i'uct} \geq [u^{-}_{iuct}] \qquad (i, u, c, t) \in \boldsymbol{IF11}_{iuct},\ i' \in \boldsymbol{IF12}_{i(i')uct} \tag{3.22}$$

$$\sum_{i'} FCTU_{i'uct} \geq [u^{+}_{iuct}] \qquad (i, u, c, t) \in \boldsymbol{IF21}_{iuct},\ i' \in \boldsymbol{IF22}_{i(i')uct}, \tag{3.23}$$

$$\sum_{ii} FTU_{(ii)ut} \geq \varepsilon \qquad (ii, u) \in \boldsymbol{IIU},\ ii \in \boldsymbol{IE1}_{(ii)ukt} \tag{3.24}$$

where, ε is a small number, $[u^{-}_{iuct}]$ and $[u^{+}_{iuct}]$ are the values of slack variables u^{-}_{iuct} and u^{+}_{iuct} respectively, and sets $\boldsymbol{IF11}_{iuct}$, $\boldsymbol{IF12}_{i(i')uct}$ are defined dynamically as shown in Figure 3.5. If $f_{ict} > xc^{U}_{cu}$, then the NLP solver makes $[u^{-}_{iuct}]$ positive. This reduces $FCTU_{iuct}$ and keeps the composition of crude c in the feed to CDU u below xc^{U}_{cu}. In this case, tank i cannot feed CDU u alone, which is included in set $\boldsymbol{IF11}_{iuct}$. The possible modifications of the integer map related to tank i are: (a) set $Y_{iut} = 0$, so tank i cannot charge CDU u, (b) change the value of $Y_{i'ut}$ (i' denotes a tank that can charge CDU u and $f_{i'ct} < xc^{U}_{cu}$), so that other tanks can replace tank i or charge CDU u together with tank i during period t. In this way, the values of Y_{iut} and $Y_{i'ut}$ will change and FPT_{pit} and XT_{it} will change accordingly. In practice, setting $Y_{iut} = 0$ may cause infeasibility, because this excludes the possibility that tank i may charge CDU u together with other tanks. Thus, it has to look for tanks i' with $f_{i'ct} < xc^{U}_{cu}$. These tanks are in $\boldsymbol{IF12}_{i(i')uct}$ $\boldsymbol{IF21}_{iuct}$ and $\boldsymbol{IF22}_{i(i')uct}$ are defined similarly, except that $[u^{+}_{iuct}]$ >0 (f_{ict} $< xc^{L}_{cu}$) and $f_{i'ct} > xc^{L}_{cu}$.

Eq. 3.22 addresses the situation, when the concentration of crude c in tank i exceeds the upper acceptable limit for CDU u. When u^{-}_{iuct} in eq. 3.7 is nonzero, the concentration of crude c in tank i exceeds the upper limit of concentration to CDU u. Clearly, this tank i cannot feed CDU u alone. In other words, it must charge CDU u

together with some other tank/s in $\boldsymbol{IF12}_{i(i')uct}$. Thus, the total amount of crude from tanks in $\boldsymbol{IF12}_{i(i')uct}$ must be nonzero. Similarly, eq. 3.23 addresses the situation, when the concentration of crude *c* in tank *i* is below the lower limit for CDU *u*.

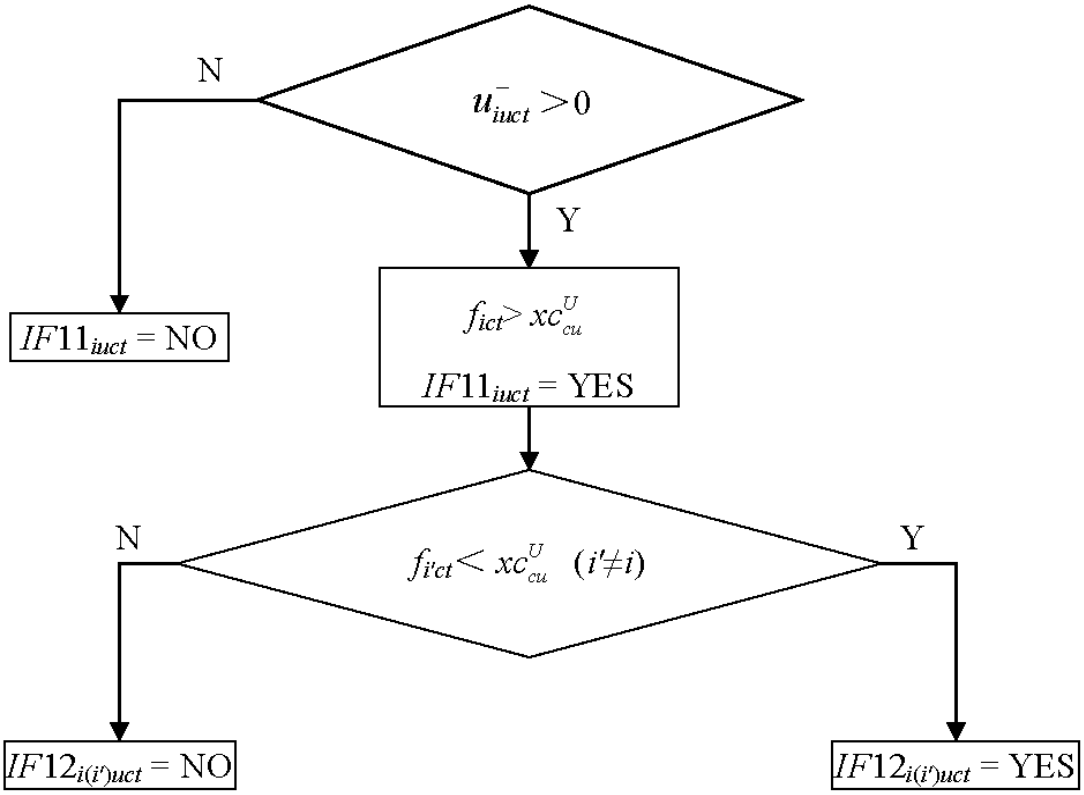

Figure 3.5 Definition of sets for slack cuts

Eq. 3.24 handles the feed quality requirements for CDU *u* during *t*. When $s^+_{ukt} > 0$, property *k* in the feed to CDU *u* exceeds the upper limit for CDU *u*. To bring this within the limits for CDU *u*, it is found tanks *ii* with crude qualities for property *k* below the upper limit for CDU *u*, and include them in $\boldsymbol{IE1}_{(ii)ukt}$ Then, to ensure that one or more of these tanks feed CDU *u*, the sum of the flows from these tanks during period *t* is forced to be nonzero.

Figure 3.6 illustrates the procedure for RLA. MIP-1 is first solved. Then, composition discrepancy in the solution is checked using condition I from Li et al.

(2002). If no composition discrepancy exists, then the algorithm stops. Otherwise, the integer map is fixed and the NLP problem incorporating eqs. 3.7 and 3.20-3.21 is solved. If some slack variables are nonzero, then sets $\boldsymbol{IF11}_{iuct}$, $\boldsymbol{IF12}_{i(i')uct}$, $\boldsymbol{IF21}_{iuct}$, $\boldsymbol{IF22}_{i(i')uct}$ and $\boldsymbol{IE1}_{(ii)ukt}$ are identified as explained earlier and resolve MIP-2R involving eqs. 3.22-3.24 and NLP repeatedly, until a feasible solution is obtained. If the slack variables are zero, then a feasible solution is obtained. Once a feasible solution is obtained, then the tank concentrations is fixed in MIP-1 and redundant constraints are removed (no discrepancy exists so all tightening constraints will be redundant) to get MIP-2 and do the iterative refinement as in RRA-P1. Note that the iteration conditions (I, II, and III) in Figure 3.6 refer to Li et al. (2002).

3.9.1 Evaluation of RLA

First, RLA is illustrated using Example 1. The MILP approximation with eqs. 3.11a-d, 3.12a-d, 3.13a-b, and 3.17a-d gives the first integer map. This integer map results in an infeasible NLP. Therefore, the slack variables are added, and get $u^-_{4115} = 0.268$, $u^-_{4116} = 0.294$, $u^-_{4117} = 0.396$, $u^-_{4118} = 0.564$, $u^+_{4115} = 0.268$, $u^+_{4116} = 0.294$, $u^+_{4117} = 0.396$, $u^+_{4118} = 0.564$ by solving the NLP with slack variables. Using Figure 3.5, $\boldsymbol{IF11}_{iuct}$={(4, 1, 1, 5), (4, 1, 1, 6), (4, 1, 1, 7), (4, 1, 1, 8)} and $\boldsymbol{IF12}_{i(i')uct}$ ={(4, 1, 1, 1, 5), (4, 1, 1, 1, 6), (4, 1, 1, 1, 7), (4, 1, 1, 1, 8)} are got. Then, MIP-2R is solved by using eqs. 22-24 and fixing the tank concentrations obtained from the NLP. The solution of MIP-2R gives an integer map that yields a feasible NLP with a profit of $ 5066.7K. Using this feasible solution, MIP-2 is obtained, then the iterative refinement as in RRA-P1 is done to get the final profit of $ 5069.94K.

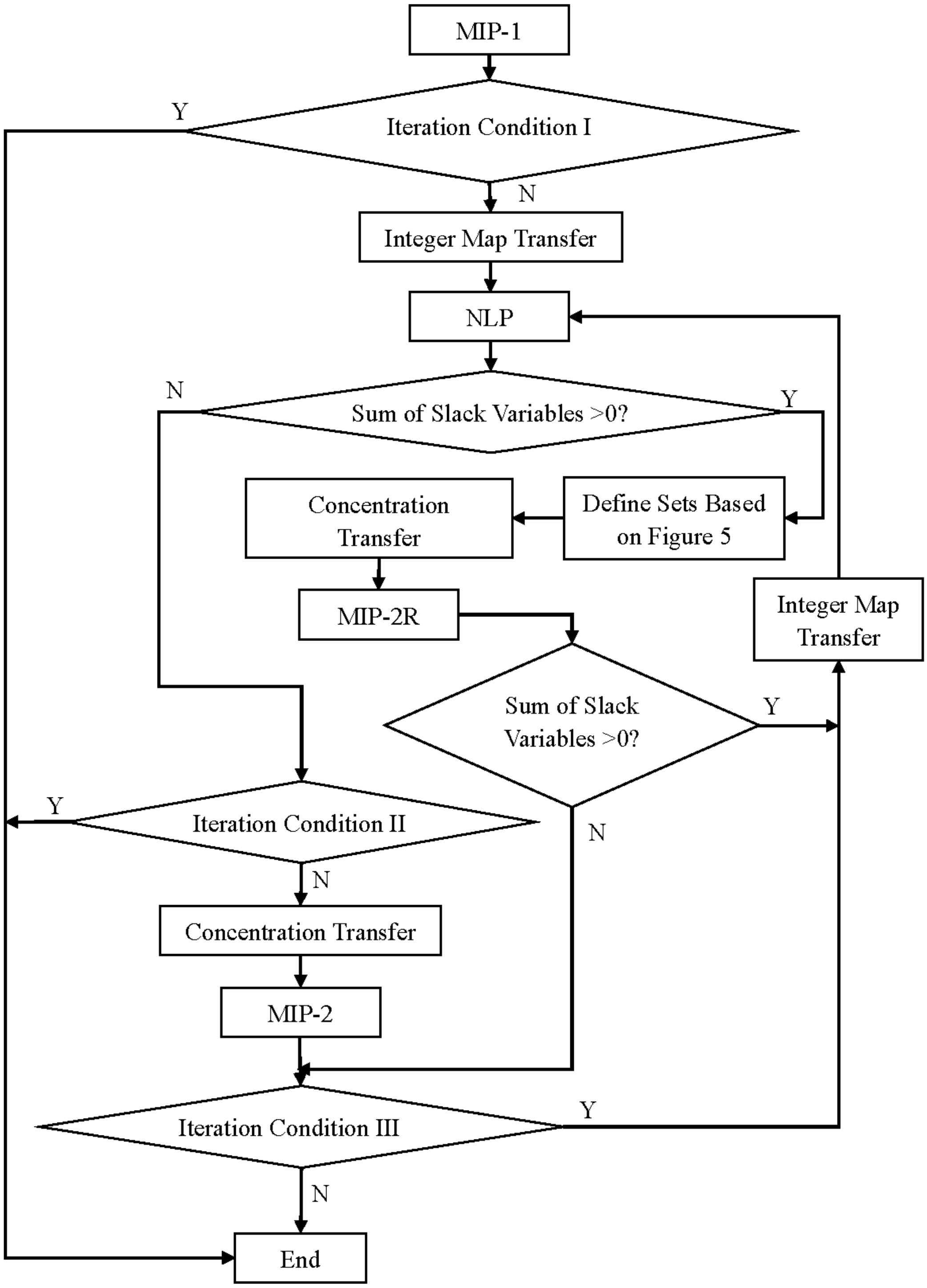

Figure 3.6 Flow chart for RLA [Revised Algorithm of Li et. al. (2002)]

To evaluate RLA more rigorously, Examples 1-21 were solved using RLA. Table

3.10 also shows the performance of RLA along with other algorithms. RLA does improve on LA (Li et al., 2002) and gives feasible solutions for all examples, but requires much longer solution times than RRA-P1. This is because RLA must solve much larger MILPs due to eqs. 3.11a-d, 3.12a-d, 3.13a-b, and 3.17a-d. Surprisingly, RLA fails to match the solution quality of RRA-P1.

3.10 Summary

In this chapter, the formulation and algorithm of Reddy et al. (2004a,b) were revised for the difficult and nonconvex MINLP problem of scheduling crude oil operations involving blending in a refinery. Although the algorithm does not guarantee globally optimal schedules, it successfully solves problems with several ship/VLCC arrivals and up to 20 days of scheduling horizon, and gives schedules with profits within 6% of a conservative upper bound. It is superior to existing literature algorithms (Li et al., 2002; Reddy et al., 2004a,b, and others) and general-purpose software (BARON, DICOPT) in several respects.

1. It enhances the practical utility of crude scheduling algorithms by identifying and modeling fifteen crude quality specifications currently used by the refinery industry. It reports relevant indexes and linear (weight-based or volume-based) blending correlations to address nonlinear crude properties.

2. It is far more robust in getting a good feasible schedule. While it successfully solved all the twenty-four industry-scale examples that are tested in this paper, other algorithms and software failed to solve most of them.

3. It is significantly faster. A clever partial relaxation strategy enables it to solve large problems, which others fail to solve in reasonable time, without compromising solution quality, causing any composition discrepancy, or requiring NLP solutions. It is also much faster (by a factor of nearly five on an average) than an NLP-based algorithm that are devised by improving the algorithm of Li et al. (2002).

4. It improves schedule quality by employing an iterative refinement strategy. While for small problems, its solutions are very near-optimal, for medium-size examples, they are within 3% of a conservative upper bound on the profit

The proposed algorithm is timely and useful in this era of increasing crude prices and decreasing crude qualities. Further work is now appearing (Karuppiah et al., 2006 and 2008) and is desirable on global optimization algorithms for solving this difficult scheduling problem, and also on addressing disruptions and uncertainty in crude scheduling (Adhitya et al., 2007a-b; Li et al., 2005a-b; Li et al, 2006). While the revised algorithm is intended for a marine-access refinery, the algorithmic strategy is applicable to other types of refineries such as in-land refineries. Thus, in the next Chapter, this algorithmic strategy is extended for in-land refineries.

CHAPTER 4

A DISCRETE-TIME MODEL WITH DIFFERENT CRUDE BLENDING POLICIES FOR CRUDE OIL SCHEDULING

4.1 Introduction

As mentioned in Chapter 3, crude oil scheduling operation is an important and complicated routine task in a refinery. The cost of crude oil accounts for about 80% of a refinery's turnover. Efficient crude oil scheduling can reduce overall operation costs significantly by minimizing changeovers, reducing safety-stock penalties and avoiding vessel demurrage costs, etc. However, Mathematical modeling this operation involves many discrete and continuous variables, especially modeling crude blending in storage tanks or charging tanks, which results in bilinear terms to make sure composition consistency. Thus, the entire problem turns to a complicated non-convex MINLP problem. In Chapter 3, a robust and efficient algorithm was successfully developed to solve this non-convex MINLP problem. While the proposed algorithm is intended for a marine-access refinery, the algorithmic strategy is applicable to other types of refineries such as in-land refineries.

Therefore, the purpose of this Chapter is to extend that robust and efficient algorithm to an in-land refinery where crudes are first unloaded into storage tanks, allowed or not allowed to be blended in storage tanks, and then blended in charging tanks. Furthermore, a discrete-time formulation is also developed for this in-land refinery incorporating many realistic features such as SBM, multiple-parcel vessels (VLCCs), multiple jetties, single-parcel vessels, crude segregation, brine settling, multiple tanks feeding one CDU at a time, one tank charging multiple CDUs at a time, crude blending only in charging tanks or in both storage tanks and charging tanks, and multiple crudes in the SBM pipeline. The state-task network (STN) representation proposed by Kondili et al. (1993) is used to clearly demonstrate the refinery configurations and flow streams among these configurations throughout this Chapter.

This Chapter is organized as follows. At the first step, the problem is described in detail. Then, a complete formulation based on discrete-time representation is developed. Finally, several examples are used to illustrate the capability of the presented formulation and simultaneously demonstrate the robustness and efficiency of the algorithm proposed in Chapter 3.

4.2 Problem Definition

Figure 4.1 shows a schematic of crude oil operations in a typical in-land refinery. It comprises offshore facilities for crude unloading such as a single buoy mooring (SBM) or single point mooring (SPM) station, onshore facilities for crude unloading such as one or more jetties, tank farm consisting of crude storage and/or charging tanks, and processing units such as crude distillation units (CDUs). Each storage tank can hold pure crude or crude blend. Crudes from storage tanks are blended in charging tanks to adjust the crude concentration and quality to meet CDU processing requirements. While the compositions in storage tanks may be constant or vary with time, they in charging tanks vary with time. The unloading facilities supply crude to storage tanks via pipelines. The pipeline connecting the SBM/SPM station with crude tanks is called the SBM/SPM line, and it normally has a substantial holdup.

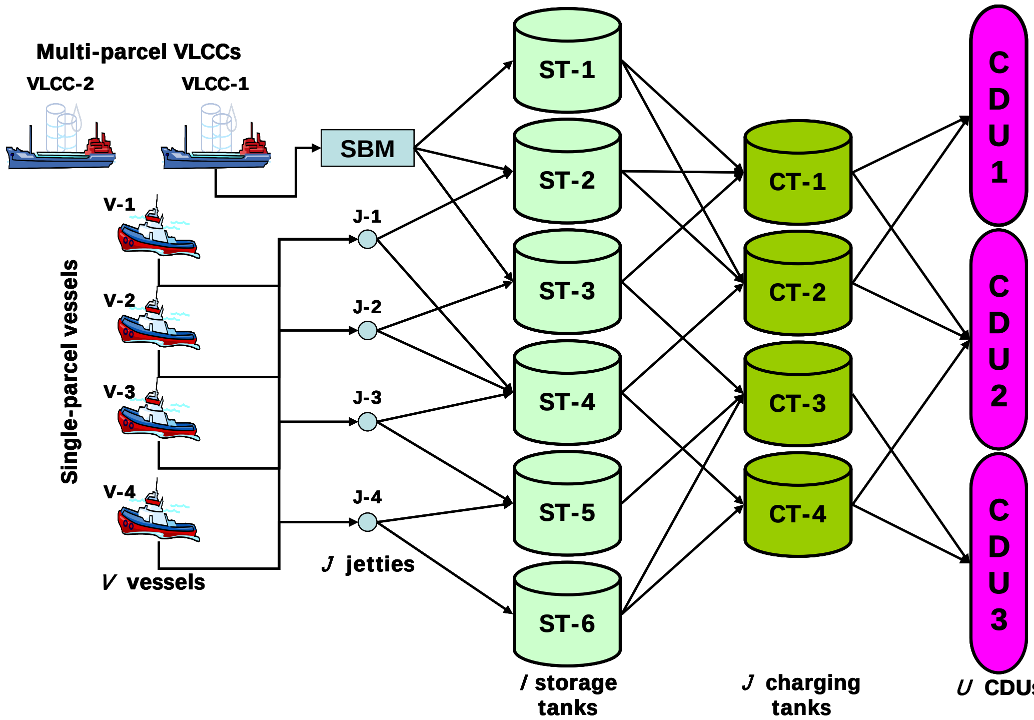

Figure 4.1 Schematic of crude oil unloading, storage, blending and processing

Two types of ships supply crudes to the refinery. Very large crude carriers (VLCCs) or ultra large crude carriers (ULCCs) carry multiple parcels of several crudes and dock at the SBM/SPM station offshore. Small vessels carry single crudes and berth at the jetties. The entire crude oil operation involves unloading and blending crudes from ships into various storage and charging tanks at various times, and charging

CDUs from one or more charging tanks at various rates over time. With this, the problem can be stated as follows:

Given:

1. Crude delivery data: estimated arrival times of ships, their crude parcels, crude types, and parcel sizes.
2. Maritime infrastructure: jetties, jetty-storage tank and SBM-storage tank connections, crude unloading transfer rates, and SBM pipeline holdup volume and its resident crude (s).
3. Tank farm data: storage and charging tanks, their capacities, their initial crude stocks and compositions, allowable crudes, crude quality specifications or limits, and limits on crude transfer rates from storage tanks to charging tanks, and compositions and quality specifications in storage and charging tanks.
4. Crude processing data: CDUs, limits on processing rates and feed rates from charging tanks, and limits on crude compositions and quality specifications.
5. Crude demands
6. Economic data: demurrage, crude changeover costs, safety stock penalties, and crude margins.

Determine:

1. Crude unloading schedule for each ship including the timings, rates, and tanks for all parcel transfers.
2. Inventory and crude concentration profiles of all storage and charging tanks.
3. Charging schedule for each CDU including the feed tanks, feed rates, and timings.

Subject to the operating practices:

1. A storage tank cannot receive and feed crude at the same time.
2. Each tank needs 8 hours to settle and remove brine after each crude receipt.
3. Multiple tanks can feed a CDU simultaneously and vice versa.
4. Only one VLCC can dock at the SBM station at any time.
5. Sequence in which a VLCC unloads its parcels is known a priori. This is normally fixed when the VLCC loads its parcels and the refinery needs to specify that at the time of shipping (Reddy et al. 2004b).

Assumptions:

1. Holdup of the Jetty line is small and its effect on the concentration of the receiving tanks is negligible, but it is not negligible in the SBM line.
2. One or multiple crudes reside in the SBM line and no mixing between two different adjacent crudes in SBM line.
3. Crude mixing is perfect in each storage tank.
4. Crude changeover times are negligible.

The objective of the entire scheduling problem is to maximize the gross profit, which is the revenue computed in terms of crude margins minus the operating costs such as demurrage, safety stock penalties, changeover cost, etc.

4.3 Mathematical Formulation

So far two types of time representations exist in the literature including discrete-time and continuous time representation. Pinto et al. (2000) pointed out that although continuous-time representation reduces the combinatorial complexity substantially, discrete-time models easily handle resource constraints and provide tighter formulations. Besides this, Reddy et al. (2004b) suggested other three advantages for their discrete-time model. First, if slots of 8 h duration in a discrete-time formulation can be successfully achieved in this problem, then the complexity of a continuous-time formulation is not necessary. Second, use of a discrete-time formulation can deal effectively with the inherent nonlinearity of this problem without solving a single NLP. Third, their discrete-time formulation embodied key features of a continuous-time formulation. This partially obviates the need for a continuous-time model.

Therefore, discrete-time representation is still used in this Chapter to develop formulation for a typical in-land refinery involving SBM, multiple jetties, storage tanks, charging tanks, and multiple CDUs. The approaches of Reddy et al. (2004b) in dividing horizon, identifying time periods, creating the order list for parcels and segregating parcels into SBM and VLCC parcels and Jetty parcels are also used in this Chapter. Reddy et al. (2004b) considered only one SBM parcel by assuming the size of the SBM pipeline is far smaller than a typical parcel size because of marine-access refinery. However, the situation where multiple parcels may exist in the SBM pipeline is considered in this Chapter since the line may be a long-distance pipeline with a huge holdup.

Consider a refinery with one SBM line, *jp* jetties, *i* storage tanks, *j* charging tanks, *u* CDUs. *v* VLCCs and *jp* single-parcel vessels arrive at different times during the scheduling horizon. Now, a complete mathematical formulation is developed in the following.

4.3.1 Parcel-to-SBM/Jetty and SBM/Jetty-to-Storage Tank Connections

To model parcel-to-SBM/Jetty and SBM/Jetty-to-storage tank connections, the following binary variables XP_{pt}, XF_{pt} and XL_{pt}, XT_{it}, X_{pit} are defined (Reddy et al. 2004b) as follows,

$$XP_{pt} = \begin{cases} 1 & \text{if parcel } p \text{ is connected for transfer during period } t \\ 0 & \text{otherwise} \end{cases}$$

$$XF_{pt} = \begin{cases} 1 & \text{if parcel } p \text{ is first connected for transfer at the start of period } t \\ 0 & \text{otherwise} \end{cases}$$

$$XL_{pt} = \begin{cases} 1 & \text{if parcel } p \text{ is disconnected at the end of period } t \\ 0 & \text{otherwise} \end{cases}$$

$$XT_{it} = \begin{cases} 1 & \text{If tank } i \text{ is connected to receive crude during period } t \\ 0 & \text{Otherwise} \end{cases}$$

$$X_{pit} = \begin{cases} 1 & \text{if parcel } p \text{ is transferred to storage tank } i \text{ during period } t \\ 0 & \text{otherwise} \end{cases}$$

The relationship of XP_{pt}, XF_{pt} and XL_{pt} can be expressed as follows,

$$XP_{pt} = XP_{p(t-1)} + XF_{pt} - XL_{p(t-1)} \qquad (p, t) \in \boldsymbol{PT} \tag{4.1}$$

$$XP_{pt} \geq XL_{pt} \qquad (p, t) \in \boldsymbol{PT} \tag{4.2}$$

where, $\boldsymbol{PT} = \{(p, t) \mid \text{parcel } p \text{ may be connected to SBM/Jetty line in period } t\}$.

Each parcel connects to and disconnects from SBM/Jetty line once and only once during the entire scheduling horizon.

$$\sum_t XF_{pt} = 1 \qquad (p, t) \in \boldsymbol{PT} \tag{4.3}$$

$$\sum_t XL_{pt} = 1 \qquad (p, t) \in \boldsymbol{PT} \tag{4.4}$$

Eqs. 4.1-4.4 enforce XF_{pt} and XL_{pt} to be binary variables automatically, when XP_{pt} are so.

The time (TF_p) at which parcel p first connects to SBM/Jetty line and the time (TL_p) at which it disconnects are:

$$TF_p = \sum_t (t-1) \cdot XF_{pt} \qquad (p, t) \in \boldsymbol{PT} \tag{4.5}$$

$$TL_p = \sum_t t \cdot XL_{pt} \qquad (p, t) \in \boldsymbol{PT} \tag{4.6}$$

The discrete-time formulation developed by Reddy et al. (2004b) embedded some continuous-time features by allowing two parcels to unload during a period t. This utilized fully the time available in a period. At most two parcels are allowed to be connected to SBM during period t, while at most one parcel is allowed to be connected to one Jetty during period t.

$$\sum_p XP_{pt} \le 2 \qquad (p, t) \in \boldsymbol{PT}, p \in \boldsymbol{SP} \tag{4.7a}$$

$$\sum_p XP_{pt} \le NJ \qquad (p, t) \in \boldsymbol{PT}, p \in \boldsymbol{JP} \tag{4.7b}$$

$$TF_{(p+1)} \ge TL_p - 1 \qquad p \in \boldsymbol{SP} \tag{4.8}$$

where, $\boldsymbol{JP}$ denotes the Jetty parcels. $\boldsymbol{SP}$ denotes SBM parcels. NJ is the number of Jetties.

Each parcel should begin to unload after its arrival.

$$TF_p \ge ETA_p \tag{4.9}$$

As mentioned in eq. 4.7a, at most two SBM parcels are allowed to be connected to the SBM line during one period, the same is done for storage tank-to-SBM connections:

$$\sum_i XT_{it} \le 2 \tag{4.10}$$

Although eq. 4.10 is developed for storage tank to SBM connections, it is also effective for storage tank to jetty connections because one jetty parcel is allowed to be unloaded at most two storage tanks in the same period.

The relationship of X_{pit} with XP_{pt} and XT_{it} is given as follows (Reddy et al. 2004b):

$$X_{pit} \geq XP_{pt} + XT_{it} - 1 \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI} \qquad (4.11)$$

$$\sum_{i} X_{pit} \leq 2XP_{pt} \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI} \qquad (4.12)$$

$$\sum_{p} X_{pit} \leq 2XT_{it} \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI} \qquad (4.13)$$

$$\sum_{p}\sum_{i} X_{pit} \leq 2 \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI}, p \in \boldsymbol{SP} \qquad (4.14a)$$

$$\sum_{p}\sum_{i} X_{pit} \leq 2NJ \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI}, p \in \boldsymbol{JP} \qquad (4.14b)$$

where $\boldsymbol{PI} = \{(p, i) \mid \text{tank } i \text{ may receive crude from parcel } p\}$.

4.3.2 Storage Tank-to-Charging Tank Connections

Because brine settling and removal operation is carried in storage tanks, a storage tank cannot feed any charging tank after it receives crudes from parcels. It is assumed that the brine settling and removal needs 8 hours, i.e. one period. To model this, a 0-1 continuous variable XSB_{ijt} is defined as follows,

$$XSB_{ijt} = \begin{cases} 1 & \text{If storage tank } i \text{ feeds charging tank } j \text{ during period } t \\ 0 & \text{Otherwise} \end{cases}$$

The brine settling and removal operation can be modeled as follows,

$$2XT_{it} + XSB_{ijt} + XSB_{ij(t+1)} \leq 2 \qquad (i, j) \in \boldsymbol{IJ} \qquad (4.15)$$

where $\boldsymbol{IJ} = \{(i, j) \mid \text{Storage tank } i \text{ that can feed charging tank } j\}$. Eq. 4.15 ensures that if tank i is receiving crudes from vessels during period t, then this tank cannot feed any charging tank during period t and $(t + 1)$.

4.3.3 Charging Tank-to-CDU Connections

To feed crudes to CDU for processing, a charging tank must connect to one or more CDUs. To model this connection, a binary variable Y_{jut} is defined as follows,

$$Y_{jut} = \begin{cases} 1 & \text{If charging tank } j \text{ feeds CDU } u \text{ during period } t \\ 0 & \text{Otherwise} \end{cases}$$

According to the operating rules, a charging tank may not feed more than two CDUs simultaneously at the same time and vice versa.

$$\sum_{u} Y_{jut} \leq 2 \qquad (j, u) \in \boldsymbol{JU} \qquad (4.16)$$

$$\sum_{j} Y_{jut} \leq 2 \qquad (j, u) \in \boldsymbol{JU} \qquad (4.17)$$

where $\boldsymbol{JU}= \{(j, u)|$ charging tank j can feed CDU $u\}$.

A charging tank cannot charge any CDU simultaneously, when it is fed by any storage tank and vice versa.

$$XSB_{ijt} + Y_{jut} \leq 1 \qquad (i, j) \in \boldsymbol{IJ}, (j, u) \in \boldsymbol{JU} \qquad (4.18)$$

Eq. 4.18 ensures that once a charging tank j is feeding any CDU, then this charging tank j cannot receive crudes from any storage tanks.

4.3.4 Crude Unloading

In the above constraints, binary variables are defined to model SBM/Jetty-to-storage tank, storage tank-to-charging tank, charging tank-to-CDU connections. Now the amount of crudes unloaded from parcels to storage tanks during period t (FPT_{pit}) can be modeled as follows.

$$FPT_{pi}^{L} X_{pit} \leq FPT_{pit} \leq FPT_{pi}^{U} X_{pit} \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI} \qquad (4.19a,b)$$

where, FPT_{pi}^{L} and FPT_{pi}^{U} are the limits on unloading rate per period from parcel p

to storage tank i.

If only SBM occurs in the refinery configuration, then the following constraint is used.

$$\sum_p \sum_i \frac{FPT_{pit}}{FPT_{pi}^U} \le 1 \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI} \qquad (4.20a)$$

If only Jetties occur, then

$$\sum_i \frac{FPT_{pit}}{FPT_{pi}^U} \le 1 \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI} \qquad (4.20b)$$

$$\sum_p \frac{FPT_{pit}}{FPT_{pi}^U} \le 1 \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI} \qquad (4.20c)$$

For both SBM and multiple Jetties,

$$\sum_i \frac{FPT_{pit}}{FPT_{pi}^U} \le 1 \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI} \qquad (4.20d)$$

$$\sum_{p \in SP} \frac{FPT_{pit}}{FPT_{pi}^U} + \sum_{p \in JP} \frac{FPT_{pit}}{FPT_{pi}^U} \le 1 \qquad (4.20e)$$

Each parcel p should be unloaded completely within the scheduling horizon.

$$\sum_i \sum_t FPT_{pit} = PS_p \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI} \qquad (4.21)$$

4.3.5 Crude Transfer from Storage Tank to Charging Tank

Let FSB_{ijt} be the amount of crude transferred from storage tank i to charging tank j during period t. This amount should meet some upper and lower limits fixed by the maximum and minimum pumping rates of crudes from storage tanks to charging tanks.

$$FSB_{ij}^L XSB_{ijt} \le FSB_{ijt} \le FSB_{ij}^U XSB_{ijt} \qquad (i, j) \in \boldsymbol{IJ} \qquad (4.22)$$

If different types of crudes are allowed to be mixed in storage tanks, then the amount of crude c delivered from storage tanks to charging tanks should also be considered. To enforce this, $FCSB_{ijct}$ is defined as the amount of crude c fed by storage tank i to charging tank j during period t.

$$FSB_{ijt} = \sum_{c} FCSB_{ijct} \qquad (i, j) \in \boldsymbol{IJ}, (i, c) \in \boldsymbol{IC} \qquad (4.23)$$

where, $\boldsymbol{IC} = \{(i, c)|$ crude c can be stored in tank $i\}$.

4.3.6 Crude Charging to CDU and Processing

As with storage and charging tanks, most refineries segregate CDUs too. To see if a charging tank can feed a CDU, sets $\boldsymbol{JU} = \{(j, u)$ | charging tank j charges CDU $u\}$ and $\boldsymbol{JC} = \{(j, c)$ | crude c can be stored in charging tank $j\}$ are defined. $FCTU_{juct}$ is defined as the amount of crude c charged by charging tank j to CDU u during period t. Then, the total amount (FTU_{jut}) that charging tank j feeds CDU u during period t is given as follows,

$$FTU_{jut} = \sum_{c} FCTU_{juct} \qquad (j, u) \in \boldsymbol{JU}, (j, c) \in \boldsymbol{JC} \qquad (4.24)$$

The total amount of crudes charged from charging tank j to CDU u should satisfy some lower (FTU_{ju}^{L}) and upper (FTU_{ju}^{U}) limits:

$$FTU_{ju}^{L} Y_{jut} \le FTU_{jut} \le FTU_{ju}^{U} Y_{jut} \qquad (j, u) \in \boldsymbol{JU} \qquad (4.25)$$

Eq. 4.25 makes sure that if charging tank j does not charge CDU u, then no amount of crudes is fed CDU u. Let FU_{ut} as the total amount of crudes charged to CDU u during period t. Then,

$$FU_{ut} = \sum_{j} FTU_{jut} \qquad (j, u) \in \boldsymbol{JU} \qquad (4.26)$$

Also some lower and upper processing limits (FU_{ut}^{L} and FU_{ut}^{U}) are imposed on CDU u.

$$FU_{ut}^{L} \le FU_{ut} \le FU_{ut}^{U} \qquad (4.27)$$

In the real refinery operation, the fraction of crude mixtures that are processed in CDU u should be within some allowable values.

$$FU_{ut} \cdot xcu_{cu}^{L} \leq \sum_{j} FCTU_{juct} \leq FU_{ut} \cdot xcu_{cu}^{U} \quad (j, u) \in \boldsymbol{JU}, (j, c) \in \boldsymbol{JC} \qquad (4.28a,b)$$

Where, xcu_{cu}^{L} and xcu_{cu}^{U} are the lower and upper limits on the fraction of crude c for CDU u.

4.3.7 Two Refinements

In Chapter 3, two refinements were done to extend the practical utility of Reddy et al. (2004b)'s model for scheduling real-life crude operations. The first relates to the uncontrolled changes in CDU feed rates. Clearly, drastic changes in feed rates may disrupt CDU operation, generate off-spec distillation cuts, and even be impossible to achieve without destabilizing the column. They can be disallowed by simply adding the following two constraints.

$$\gamma_{u}^{L} FU_{ut} \leq FU_{u(t+1)} \leq \gamma_{u}^{U} FU_{ut} \qquad (4.29a,b)$$

where, parameters γ_{u}^{L} and γ_{u}^{U} can be suitably set to control period-to-period changes in crude feed flows.

Similarly, the other refinement is to ensure the quality of feeds to CDUs. This is a critical operating requirement in practice, as a feed with poor quality can seriously disrupt the operation of a CDU and even downstream units. In Chapter 3, a variety of crude properties were presented such as specific gravity, sulfur, nitrogen, oxygen, carbon residue, pour point, flash point, nickel, Reid vapor pressure, asphaltene, aromatics, paraffins, naphthene, wax, and viscosity and Table 3.3 gives their linear blending indices which are volume-based or weight-based. Let θ_{kc} represent the specification index of property k in crude c. The following constraint ensures the feed quality of CDU.

$$FU_{ut}\theta_{ku}^{L} \le \sum_{j}\sum_{c} FCTU_{juct}\theta_{kc} \le FU_{ut}\theta_{ku}^{U} \qquad (j, u) \in \boldsymbol{JU}, (j, c) \in \boldsymbol{JC} \qquad (4.30a,b)$$

$$\theta_{ku}^{L}\left(\sum_{i}\sum_{c} FCTU_{iuct}\rho_{c}\right) \le \sum_{i}\sum_{c} FCTU_{iuct}\rho_{c}\theta_{kc} \le \theta_{ku}^{U}\left(\sum_{i}\sum_{c} FCTU_{iuct}\rho_{c}\right)$$

$$(i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC} \qquad (4.31a,b)$$

where, θ_{ku}^{L} and θ_{ku}^{U} are the allowable lower and upper limits on property specification indexes. ρ_c as the density of crude c.

4.3.8 Changeovers

One main objective of refinery operations is to minimize the occurrence of changeovers. Changeover happens when the composition of crudes charged to CDU changes. This change will perturb the CDU operation and may lower the product quality. Moreover, additional cost should be imposed for each changeover. This is undesirable for a refinery and should be avoided as possible. Several definitions of changeover have been reported in the literature (Lee et al. 1996; Li et al. 2002; Reddy et al. 2004a,b). Reddy et al. (2004b) proposed that Lee et al. (1996) and Li et al. (2002) defined changeover as a change in composition of feed to CDU. They ignored that composition may change even when flows from two tanks feeding to one CDU change. Moreover, Li et al. (2002) counted changeover twice. To remove these errors, Reddy et al. (2004a,b) defined changeover accurately, which is also presented here. To detect changeover changes, a 0-1 continuous variable $YY_{jut} = Y_{jut}Y_{ju(t+1)}$ is defined, which is one, if charging tank j is connected to CDU u during both periods t and (t + 1). The linearization of YY_{jut} is given as follows,

$$YY_{jut} \geq Y_{jut} + Y_{ju(t+1)} - 1 \qquad (j, u) \in \boldsymbol{JU} \qquad (4.32a)$$

$$YY_{jut} \leq Y_{ju(t+1)} \qquad (j, u) \in \boldsymbol{JU} \qquad (4.32b)$$

$$YY_{jut} \leq Y_{jut} \qquad (j, u) \in \boldsymbol{JU} \qquad (4.32c)$$

Then, to detect the presence of a changeover on a CDU u, the following constraint is imposed.

$$CO_{ut} \geq Y_{jut} + Y_{ju(t+1)} - 2YY_{jut} \qquad (j, u) \in \boldsymbol{JU} \qquad (4.33)$$

Eq. 4.33 means that if CDU u is charged by any different tank during periods t and (t + 1), then changeover incurs. The detailed comparison of different definitions of changeover is shown later in Example 1.

When multiple tanks feed a CDU, the change of the feed rates can cause the change of feed composition. To avoid this, the feed flow rates of individual tanks should remain constant when two tanks are feeding a CDU.

$$M[2 - \sum_{j} YY_{jut}] + FTU_{jut} \geq FTU_{ju(t+1)} \qquad (j, u) \in \boldsymbol{JU} \qquad (4.34a)$$

$$M[2 - \sum_{j} YY_{jut}] + FTU_{ju(t+1)} \geq FTU_{jut} \qquad (j, u) \in \boldsymbol{JU} \qquad (4.34b)$$

4.3.9 Crude Inventory in Storage Tanks

Different parcels may contain different types of crudes based on crude quality specifications. Set $\boldsymbol{PC}$ = {(p, c) | parcel p contains crude c} is defined to identify crude c in parcel p. Recall that each parcel contains only one type of crude. Thus, the amount of crude c unloaded from vessel v to storage tank i during period t should be equivalent to the total amount of crudes unloaded from vessel v to storage tank i during period t. Different types of crudes is allowed to be blended or is not allowed in storage tanks, which is discussed respectively as follows,

(1) If different types of crudes are not allowed to be mixed in storage tanks, each storage tank stores only one type of crudes. Using VST_{it} to denote the amount of crudes in storage tank i at the end of period t, then the mass balance of the inventory of storage tank i in period t can be computed as:

$$VST_{it} = VST_{i(t-1)} + \sum_p FPT_{pit} - \sum_j FSB_{ijt} \qquad (p, i) \in \boldsymbol{PI}, (p, \mathrm{c}) \in \boldsymbol{PC}, (i, j) \in \boldsymbol{IJ} \qquad (4.35)$$

(2) If different types of crudes are allowed to be mixed in storage tanks, then let $FCSB_{ijct}$ be the amount of crude c transferred from storage tank i to charging tank j. $VCST_{ict}$ is defined as the amount of crude c in storage tank i at the end of period t. The mass balance of crude c in storage tank i is given as follows,

$$VCST_{ict} = VCST_{ic(t-1)} + \sum_p FPT_{pit} - \sum_j FCSB_{ijct}$$

$$(p, i) \in \boldsymbol{PI}, (p, c) \in \boldsymbol{PC}, (i, j) \in \boldsymbol{IJ} \qquad (4.36)$$

$$VST_{it} = \sum_c VCST_{ict} \qquad (i, c) \in \boldsymbol{IC} \qquad (4.37)$$

At any time, the inventory level of a storage tank must satisfy its minimum and maximum capacities (VST_i^L and VST_i^U).

$$VST_i^L \le VST_{it} \le VST_i^U \qquad (4.38a,b)$$

When different types of crudes are allowed to be blended in storage tanks, the crude concentration in storage tanks may also satisfy some limits because of processing and operational constraints.

$$VST_{it} xcs_{ci}^L \le VCST_{ict} \le VST_{it} xcs_{ci}^U \qquad (i, c) \in \boldsymbol{IC} \qquad (4.39a,b)$$

The operating limitations on crude quality specifications in storage tanks are calculated as,

$$VST_{it} xks_{ki}^{L} \le \sum_{c} VCST_{ict} \theta_{kc} \le VST_{it} xks_{ki}^{U} \qquad (i, c) \in \boldsymbol{IC} \qquad (4.40a,b)$$

where, xcs_{ci}^{L} and xcs_{ci}^{U} are the lower and upper limits on the composition of crude c in storage tank i, respectively. xks_{ki}^{L} and xks_{ki}^{U} are the lower and upper limits on crude specification indexes of property k in storage tank i, respectively.

4.3.10 Crude Inventory of Charging Tanks

Different types of crudes are blended in charging tanks. $VCBT_{jct}$ is defined as the amount of crude c in charging tank j at the end of period t. VBT_{jt} is defined to denote the total amount of crudes in charging tank j at the end of period t. The crude balance in charging tank j is given as follows,

$$VCBT_{jct} = VCBT_{jc(t-1)} + \sum_{i} FCSB_{ijct} - \sum_{u} FCTU_{juct}$$

$$(i, j) \in \boldsymbol{IJ}, (i, c) \in \boldsymbol{IC}, (j, u) \in \boldsymbol{JU}, (j, c) \in \boldsymbol{JC} \qquad (4.41)$$

$$VBT_{jt} = \sum_{c} VCBT_{jct} \qquad (j, c) \in \boldsymbol{JC} \qquad (4.42)$$

Also, the crude inventory of charging tanks should meet some upper and lower limits as,

$$VBT_{j}^{L} \le VBT_{jt} \le VBT_{j}^{U} \qquad (4.43)$$

where, VBT_{j}^{L} and VBT_{j}^{U} are the lower and upper limits on crude inventory in charging tank j, respectively.

Similar to storage tanks, crude concentrations in charging tanks may also need to keep in some limits as follows,

$$VBT_{jt} xcb_{cj}^{L} \leq VCBT_{jct} \leq VBT_{jt} xcb_{cj}^{U} \qquad (j, c) \in \boldsymbol{JC} \qquad (4.44)$$

Operating constraints on crude quality specification in charging tanks are given as follows,

$$VBT_{jt} xkb_{kj}^{L} \leq \sum_{c} VCBT_{jct} \theta_{kc} \leq VBT_{jt} xkb_{kj}^{U} \qquad (j, c) \in \boldsymbol{JC} \qquad (4.45)$$

where, xcb_{cj}^{L} and xcb_{cj}^{U} are lower and upper limits on the fraction of crude c in charging tank j, respectively. xkb_{kj}^{L} and xkb_{kj}^{U} are lower and upper limits the specification indices of crude property k in charging tank j, respectively.

4.3.11 Composition Consistency

If different types of crudes are allowed to be mixed in storage tanks, the composition of crude c in the outflow of tank i to charging tank j should be equal to the concentration of crude c in tank i during period t. Let fs_{ict} be the composition of crude c in storage tank i at the end of period t.

$$FCSB_{ijct} = fs_{ict} FSB_{ijt} \qquad (i, j) \in \boldsymbol{IJ}, (i, c) \in \boldsymbol{IC}, (j, c) \in \boldsymbol{JC} \qquad (4.46a)$$

$$VCST_{ict} = fs_{ict} VST_{it} \qquad (i, c) \in \boldsymbol{IC} \qquad (4.46b)$$

Similarly, the composition of crude c in the outstream of a charging tank j to CDU u should be also consistent with its composition:

$$FCTU_{juct} = fb_{jct} FTU_{jut} \qquad (j, u) \in \boldsymbol{JU}, (j, c) \in \boldsymbol{JC} \qquad (4.47a)$$

$$VCBT_{jct} = fb_{jct} VBT_{jt} \qquad (j, c) \in \boldsymbol{JC} \qquad (4.47b)$$

Where, fb_{jct} means the composition of crude c in charging tank j at the end of period t.

It should be noted that if different types of crudes are not allowed to be mixed in storage tanks, then eq. 4.46 should be dropped. Eqs. 4.46-4.47 are bilinear items, which turn this problem into non-convex MINLP problem and very difficult to solve to global optimality.

4.3.12 Production Requirements

The throughput demand over the entire scheduling horizon for each CDU should meet the demand of each CDU.

$$\sum_t FU_{ut} = D_u \qquad (4.48)$$

4.3.13 Sea Waiting Cost

Two definitions of sea waiting cost exist in the literature. One is the cost for the waiting time of a vessel from its arrival to its start time to unload (Lee et al., 1996; Li et al., 2002). The other is the cost for a vessel which harbors beyond the stipulated time in logistics contract (Reddy et al., 2004a,b). Let SWC_v (\$ per unit time) be the sea waiting cost for a vessel v. ETD_v denotes the estimated time of departure of a vessel v as agreed in the logistics contract.

The definition of sea waiting cost from Lee et al. (1996) and Li et al. (2002) is,

$$DC_v \geq (TF_p - ETA_p)SWC_v \qquad (p, v) \in \boldsymbol{FPV} \qquad (4.49a)$$

The definition from Reddy et al. (2004a,b) is,

$$DC_v \geq (TL_p - ETA_p - ETD_v)SWC_v \qquad (p, v) \in \boldsymbol{LPV} \qquad (4.49b)$$

Where, ***FPV***= {(p, v)| parcel p is the first parcel in vessel v}, ***LPV***= {(p, v)| parcel p is the last parcel in vessel v}.

4.3.14 Safety Stock or Inventory Cost

Safety stock is used to maintain a minimum stock of crude to avert uncertainty, because the refiner usually makes decisions to purchase crudes far advance of scheduling activity. While Reddy et al. (2004a,b) incorporated safety stock in their model, Lee et al. (1996) and Li et al. (2002) calculated crude inventory cost. Let *SS* be the desired safety stock of crude and *SSP* as the penalty ($ per unit volume per period) for violate crude safety stock. SC_t is defined as the stock penalty in period t. The safety stock cost is computed as follows,

$$SC_t \geq SSP(SS - \sum_i VST_{it} - \sum_j VBT_{jt}) \tag{4.50}$$

When the crude inventory is higher than the safety stock, no penalty is imposed. Otherwise, penalty should be imposed on this violation.

Inventory costs of storage and charging tanks (INVCOST) can be calculated as,

$$\text{INVCOST} = \sum_i \sum_t CIVS_i (VST_{it} + VST_{i(t-1)}) / 2 + \sum_j \sum_t CINB_j (VBT_{jt} + VBT_{j(t-1)}) / 2 \tag{4.51}$$

4.3.15 Scheduling Objective

So far, two types of scheduling objectives for crude oil scheduling problem exist in the literature. One is to minimize the total operating cost (Lee et al., 1996; Li et al., 2002). The operating cost consists of unloading cost, sea waiting cost, inventory cost for storage tanks and charging tanks, and changeover cost. The total operating cost can be formulated as,

$$\text{Cost} = \sum_p CULD_p (TL_v - TP_v) + \sum_v DC_v + \text{INVCOST} + COC \sum_u \sum_t CO_{ut} \tag{4.52a}$$

The other is to maximize total profit (Li et al., 2002; Reddy et al., 2004a,b). Li et al. (2002) defined total profit as revenues of crudes minus the cost of raw materials and operating cost.

$$\text{Profit} = \sum_i \sum_{u \in IU} \sum_{c \in IC} \sum_t CPROD_c FCTU_{iuct} - \sum_c \sum_{p \in PC} CRAW_c PS_p - \sum_v CULD_v (TF_v - TL_v) - \sum_v DC_v - COC \sum_u \sum_{\text{t}} CO_{ut} - \text{INVCOST} \tag{4.52b}$$

Reddy et al. (2004b) defined the total profit as crude profit margin minus operating cost (sea waiting cost, changeover cost, and penalty for under-running crude safety stock).

$$\text{Profit} = \sum_i \sum_u \sum_c \sum_t FCTU_{iuct} CP_{cu} - \sum_v DC_v - COC \sum_u \sum_t CO_{ut} - \sum_t SC_t \tag{4.52c}$$

As mentioned before, a storage tank can hold pure crude or crude blend, whereas different types of crudes are blended in charging tanks. In some real operations, crude composition in each charging tank may be prefixed in order to stabilize CDU operations. Thus, crude composition in each charging tank is variable or prefixed. Since crudes are blended in charging tanks, prefixing crude composition in each storage tank is useless. Therefore, four policies for crude composition in each storage and charging tanks are involved. They are:

1. Each storage tank holds pure crude, and crude composition in each charging tank is variable (not prefixed).
2. Each storage tank holds crude blends, and crude composition in each charging tank is variable.
3. Each storage tank holds crude blends, and crude composition in each charging

tank is prefixed.

4. Each storage tank holds pure crude, and crude composition in each charging tank is prefixed.

Note that crude composition in both storage and charging tanks is known for policy 4. Then, the problem turns to be MILP problem, which can be solved using CPLEX directly. Therefore, policies 1, 2 and 3 are considered in this Chapter, because they result in nonconvex MINLP problems. The complete MINLP formulation for different refinery configurations comprises eqs. 4.1-4.52. Table 4.1 summarizes these equations for their corresponding refinery configurations.

4.4 Solution Method

Recall that this crude oil scheduling is also a complex non-convex MINLP problem, which is very difficult to solve optimally. Although several general commercial solvers such as DICOPT and BARON are developed to solve general MINLP problems, they fail for most cases and need horrible large time to get a feasible solution, as illustrated in Chapter 3. The algorithm named RRA-P1 in Chapter 3 is more robust and efficient than the existing algorithms in the literature (Lee et al. 1996; Li et al. 2002; Reddy et al. 2004a,b; Moro and Pinto 2004). In the following, this RRA-P1 algorithm is described in brief. The more details can be referred to Chapter 3.

Table 4.1 Constraints for different refinery configurations and crude blending policies

		Constraint Equations for			
SBM & Jetty	Policy	Parcel-to-SBM/Jetty Connection	SBM/Jetty-to-Storage Tank Connection	Storage Tank-to- Charging Tank and Charging Tank-to-CDU Connection	Crude Delivery and Processing
SBM Only	1	4.1-4.6, 4.7a, 4.8-4.9	4.10-4.13, 4.14a	4.15-4.18	4.19, 4.20a, 4.21-4.22, 4.24-4.34
	2	4.1-4.6, 4.7a, 4.8-4.9	4.10-4.13, 4.14a	4.15-4.18	4.19, 4.20a, 4.21-4.34
	3	4.1-4.6, 4.7a, 4.8-4.9	4.10-4.13, 4.14a	4.15-4.18	4.19, 4.20a, 4.21-4.23, 4.25-4.27, 4.30-4.34
Jetty Only	1	4.1-4.6, 4.7b, 4.8-4.9	4.10-4.13, 4.14b	4.15-4.18	4.19, 4.20b,c, 4.21-4.22, 4.24-4.34
	2	4.1-4.6, 4.7b, 4.8-4.9	4.10-4.13, 4.14b	4.15-4.18	4.19, 4.20b,c, 4.21-4.34
	3	4.1-4.6, 4.7b, 4.8-4.9	4.10-4.13, 4.14b	4.15-4.18	4.19, 4.20b,c, 4.21-4.23, 4.25-4.27, 4.30-4.34
SBM & Jetties	1	4.1-4.6, 4.7a,b, 4.8-4.9	4.10-4.13, 4.14a,b	4.15-4.18	4.19, 4.20d,e, 4.21-4.22, 4.24-4.34
	2	4.1-4.6, 4.7a,b, 4.8-4.9	4.10-4.13, 4.14a,b	4.15-4.18	4.19, 4.20d,e, 4.21-4.34
	3	4.1-4.6, 4.7a,b, 4.8-4.9	4.10-4.13, 4.14a,b	4.15-4.18	4.19, 4.20d,e, 4.21-4.23, 4.25-4.27, 4.30-4.34
SBM & Jetty	Policy	Crude Inventory in Storage Tank and Charging Tanks	Composition Consistency	Production Requirement	Scheduling Objective
SBM Only	1	4.35, 4.38, 4.41-4.45	4.47a,b	4.48	4.49-4.52
	2	4.35-4.45	4.46a,b, 4.47a,b	4.48	4.49-4.52
	3	4.35-4.43, 4.45	4.46a,b, 4.47a,b	4.48	4.49-4.52
Jetty Only	1	4.35, 4.38, 4.41-4.45	4.47a,b	4.48	4.49-4.52
	2	4.35-4.45	4.46a,b, 4.47a,b	4.48	4.49-4.52
	3	4.35-4.43, 4.45	4.46a,b, 4.47a,b	4.48	4.49-4.52
SBM & Jetties	1	4.35, 4.38, 4.41-4.45	4.47a,b	4.48	4.49-4.52
	2	4.35-4.45	4.46a,b, 4.47a,b	4.48	4.49-4.52
	3	4.35-4.43, 4.45	4.46a,b, 4.47a,b	4.48	4.49-4.52

Let α denote a composition-based block in the algorithm of Reddy et al. (2004b) denoted as RA, β denote a vessel-based block defined in Chapter 3 and **F** denote the MILP model with known compositions for eqs. 4.46-4.47. The algorithm (RRA-P) follows all the steps in RA (Reddy et al. (2004b)'s algorithm) and relaxes only the binary variables beyond the current vessel-based block to which current composition-based block belongs, except when it fails to get a feasible solution at some iteration. Let *α* be the current composition-based block that causes a solution failure for RA and β be the vessel-based block to which α belongs. Then, four slack variables (u^{+}_{juct}, u^{-}_{juct}, v^{+}_{ijct}, and v^{-}_{ijct}) for block α in **F** as follows,

$$FCTU_{juct} = fb_{jct}FTU_{jut} - u^{-}_{juct} + u^{+}_{juct} \qquad (j, u)\in \boldsymbol{JU},\ (j, c)\in \boldsymbol{JC} \qquad (4.53a)$$

$$FCSB_{ijct} = fs_{ict}FSB_{ijt} - v^{-}_{ijct} + v^{+}_{ijct} \qquad (i, j)\in \boldsymbol{IJ},\ (i, c)\in \boldsymbol{IC},\ (j, c)\in \boldsymbol{JC} \qquad (4.53b)$$

To know the values of slack variables u^{-}_{juct}, u^{+}_{juct}, v^{+}_{ijct}, and v^{+}_{ijct}, another MILP model is defined, denoted as **FP** as follows,

$$\text{Min} \quad P = \sum_{j}\sum_{u}\sum_{c}\sum_{t}(u^{+}_{juct} + u^{-}_{juct}) + \sum_{i}\sum_{j}\sum_{c}\sum_{t}(v^{+}_{ijct} + v^{-}_{ijct}) \qquad (4.54)$$

FP is defined as **F** but eq. 4.54 as the objective in place of the scheduling objective. Model **FP** is solved to identify the nonzero slack variables and the corresponding class/es of crudes that fail to satisfy eqs. 4.46a and 4.47a. For each such crude class, one separate integer cut (that includes all crudes in that class) i.e. eq. 3.6 is written for all periods in block β, but before block α. Then, **F** is updated by adding these cuts permanently. Now, the schedule is erased for block β and onwards. With the schedule before block β frozen, RA is restarted as per normal. Note that when RA fails at the first composition-based block α of a block β, then all combinations of integer variables in block β have been examined and it should backtrack to block $\beta-1$. This procedure (RRA-P) is repeated until the entire schedule is obtained.

Let **S** denote the solution from the above procedure. Two series of feasible schedules (i.e. with no composition discrepancy) can be obtained from **S** by solving MILPs and BLPs repeatedly. First, the values of the binary variables from **S** is taken and the binary variables are fixed in the exact MIBLP (involving eqs. 4.46 and 4.47) to those values to get a BLP. Clearly, a solution **S**1 to this BLP is a schedule with no composition discrepancy. The compositions of tanks are extracted from **S**1 and fixed in the MIBLP to get an MILP. This alternating series of MILP and BLP continue, until the solutions of successive BLPs converge. Now, instead of solving a BLP first, an MILP based on **S** could also be solved and another series of solutions could be obtained. The procedure normally takes about 8-10 (MILP+BLP) solutions and the best of all these solutions is taken. The flowchart of RRA-P1 is shown in Figure 4.2.

4.5 Case Studies

Four examples from the literature are first used to evaluate the developed formulation and demonstrate the robustness and efficiency of the algorithm proposed in Chapter 3. Example 1 is taken from Lee et al. (1996), Li et al. (2002) and Jia et al. (2003). Examples 2-4 come from Lee et al. (1996) and Jia et al. (2003). All examples are solved on a Dell workstation PWS650 (Inter® Xeron™ CPU 3.06GHZ, 3.5 GB memory) running Windows NT using solver CPLEX 9.0.

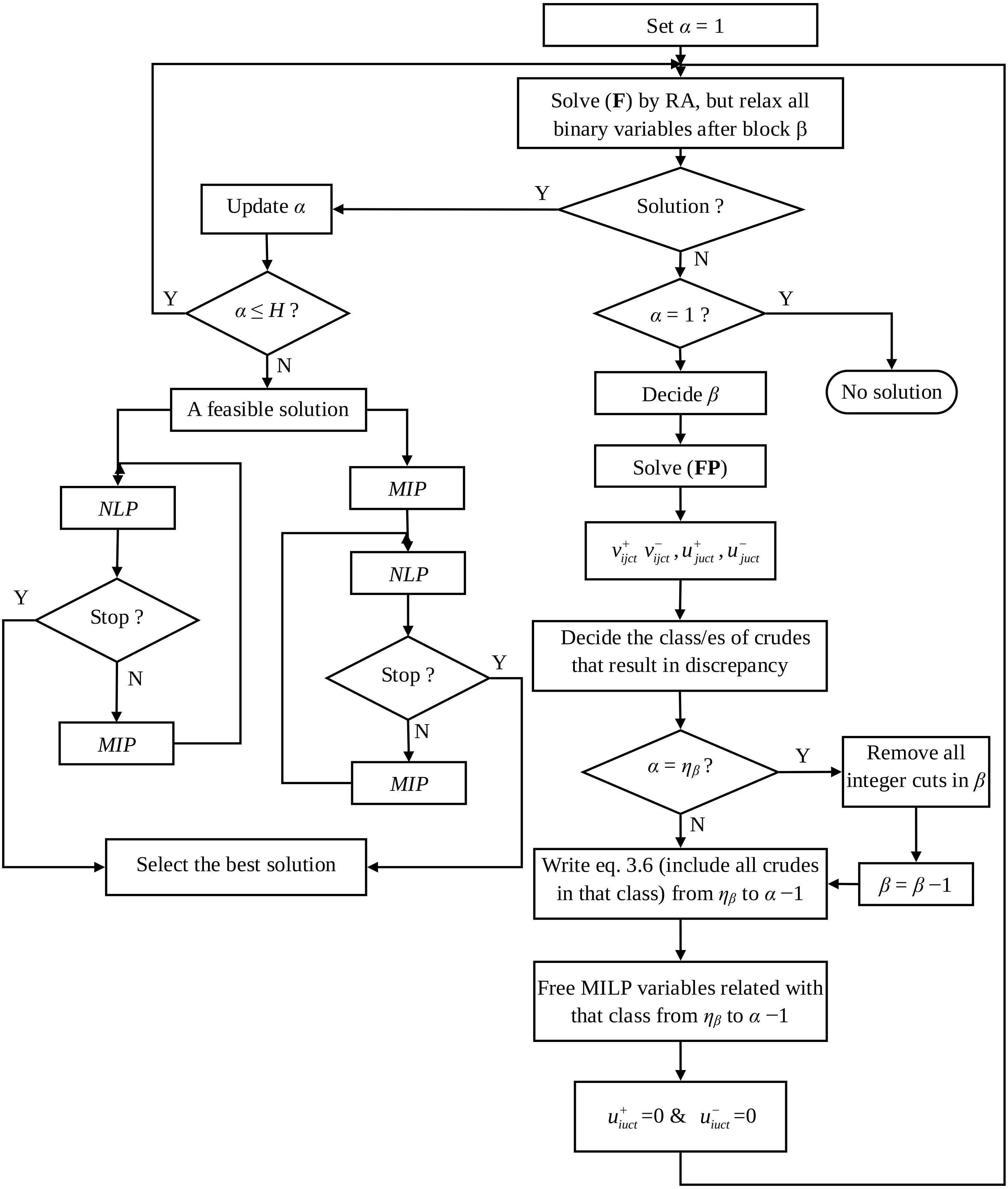

Figure 4.2 Flow chart for RRA-P1 (Partial Relaxation Refinement Strategy)

Example 1

This example (Figure 4.3) is the motivating example of Lee et al. (1996), and Example 1 of Li et al. (2002) and Jia et al. (2003). It involves two single-parcel vessels (V1 and V2), two storage tanks (ST1 and ST2), two charging tanks (CT1 and CT2) and one CDU. Different types of crudes are not allowed to be mixed in each storage tank. Table 4.2 gives the complete data. To compare fairly, the problem feature, operating rules, the definition of changeovers, and objective are the same as those of Lee et al. (1996), Li et al. (2002), and Jia et al. (2003). The result is shown in Table 4.3. The proposed approach yields the same result as the approach of Li et al. (2002) with definition 1 of changeover and gives a better solution than Lee et al. (1996). The approach of Li et al. (2002) with definition 2 of changeover obtains the worst solution among the three approaches because the changeover of CDU *u* is calculated twice during time period *t*.

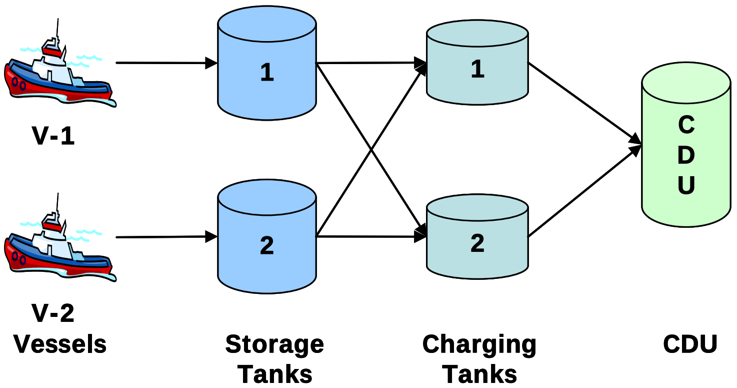

Figure 4.3 Oil flow network for Example 1

Table 4.2 Data for Example 1

Single-Parcel Vessel	Arrival Time	Amount of Crude (kbbl)	Sulfur Specification	
V-1	0	1000	0.01	
V-2	4	1000	0.06	
Tank	Capacity (kbbl)	Initial Inventory (kbbl)	Initial Sulfur Specification	Sulfur Specification Range (Min-Max)
ST1	1000	250	0.01	-
ST2	1000	750	0.06	-
CT1	1000	500	0.02	[0.015, 0.025]
CT2	1000	500	0.05	[0.045, 0.055]

Parcel-Storage tank flow rate: 0-500 kbbl/period

Charging tank-CDU flow rate: 100-500 kbbl/period

Storage tank-Charging tank flow rate: 0-500 kbbl/period

Changeover loss: 50 k$/instance

Unloading cost: 8 k$/period

Sea waiting cost: 5 k$/period

Inventory cost of storage tank: 0.008 k$/(period kbbl)

Inventory cost of charging tank: 0.005 k$/(period kbbl)

Demand of mixed oil by CDUs: Oil mix 1: 1000 kbbl, Oil mix 2: 1000 kbbl

Table 4.3 Model and solution statistics for Example 1 with different cases

Case	Single Equations	Single Variables	Discrete Variables	Objective	CPU Time (s)	Relative MILP Gaps % (Periods)
1	331	192	36	217.7	17.1	0% (1-8)
2	331	192	36	211.2	1.6	0% (1-8)
3	394	257	32	211.2	1.2	0% (1-8)
4	394	257	32	211.2	1.0	0% (1-8)
5	408	257	32	311.2	0.9	0% (1-8)
6	436	263	31	211.2	0.8	0% (1-8)

Case 1: Lee et al. (1996); Case 2: Li et al. (2002)

Case 3: Proposed formulation with changeover definition of Lee et al. (1996)

Case 4: Proposed formulation with changeover definition 1 of Li et al. (2002)

Case 5: Proposed formulation with changeover definition 2 of Li et al. (2002)

Case 6: Proposed formulation with changeover definition of Reddy et al. (2004a,b)

Table 4.4 Data for Example 2

Single-Parcel Vessel	Arrival Time	Amount of Crude (kbbl)	Component 1 Specification	Component 2 Specification		
V-1	1	1000	0.01	0.04		
V-2	4	1000	0.03	0.02		
V-3	7	1000	0.05	0.01		
Tank	Capacity (kbbl)	Initial Inventory (kbbl)	Initial Component 1 Specification	Initial Component 2 Specification	Component 1 Specification Range (Min-Max)	Component 1 Specification Range (Min-Max)
ST1	1000	200	0.01	0.04	-	-
ST2	1000	500	0.03	0.02	-	-
ST3	1000	700	0.05	0.01	-	-
CT1	1000	300	0.0167	0.0333	[0.01, 0.02]	[0.03, 0.038]
CT2	1000	500	0.03	0.023	[0.025, 0.035]	[0.018, 0.027]
CT3	1000	300	0.0433	0.0133	[0.04, 0.048]	[0.01, 0.018]

Parcel-Storage tank flow rate: 0-500 kbbl/period
Charging tank-CDU flow rate: 50-500 kbbl/period
Storage tank-Charging tank flow rate: 0-500 kbbl/period
Changeover loss: 50 k$/instance
Unloading cost: 8 k$/period
Sea waiting cost: 5 k$/period
Inventory cost of storage tank: 0.008 k$/(period kbbl)
Inventory cost of charging tank: 0.005 k$/(period kbbl)
Demand of mixed oil by CDUs: Oil mix 1: 1000 kbbl, Oil mix 2: 1000 kbbl, Oil mix 3: 1000 kbbl

Table 4.5 Data for Example 3

Single-Parcel Vessel	Arrival Time	Amount of Crude (kbbl)	Component Specification	
V-1	1	500	0.01	
V-2	5	500	0.085	
V-3	9	500	0.06	
Tank	**Capacity (kbbl)**	**Initial Inventory (kbbl)**	**Initial Component Specification**	**Component Specification Range (Min-Max)**
ST1	1000	200	0.02	[0.01, 0.03]
ST2	1000	200	0.05	[0.04, 0.06]
ST3	1000	200	0.08	[0.07, 0.09]
CT1	1000	300	0.03	[0.025, 0.035]
CT2	1000	500	0.05	[0.045, 0.065]
CT3	1000	300	0.08	[0.075, 0.085]

Parcel-Storage tank flow rate: 10-400 kbbl/period
Charging tank-CDU flow rate: 10-400 kbbl/period
Storage tank-Charging tank flow rate: 10-400 kbbl/period
Changeover loss: 50 k$/instance
Unloading cost: 10 k$/period
Sea waiting cost: 5 k$/period
Inventory cost of storage tank: 0.005 k$/(period kbbl)
Inventory cost of charging tank: 0.008 k$/(period kbbl)
Demand of mixed oil by CDUs: Oil mix 1: 500 kbbl, Oil mix 2: 500 kbbl, Oil mix 3: 500 kbbl

Table 4.6 Data for Example 4

Single-Parcel Vessel	Arrival Time		Amount of Crude (kbbl)	Component Specification	
V-1	1		600	0.03	
V-2	6		600	0.05	
V-3	11		600	0.65	
Tank	Capacity (kbbl)	Heel (kbbl)	Initial Inventory (kbbl)	Initial Component Specification	Component Specification Range (Min-Max)
ST1	900	100	600	0.031	[0.025, 0.038]
ST2	1100	100	100	0.03	[0.02, 0.04]
ST3	1100	100	500	0.05	[0.04, 0.06]
ST4	1100	100	400	0.065	[0.06, 0.07]
ST5	900	100	300	0.075	[0.07, 0.08]
ST6	900	100	600	0.075	[0.07, 0.08]
CT1	800	0	50	0.0317	[0.03, 0.035]
CT2	800	0	300	0.0483	[0.043, 0.05]
CT3	800	0	300	0.0633	[0.06, 0.065]
CT4	800	0	300	0.075	[0.071, 0.08]

Parcel-Storage tank flow rate: 0-600 kbbl/period
Charging tank-CDU flow rate: 50-500 kbbl/period
Storage tank-Charging tank flow rate: 0-500 kbbl/period
Changeover loss: 30 k$/instance
Unloading cost: 7 k$/period
Sea waiting cost: 5 k$/period
Inventory cost of storage tank: 0.005 k$/(period kbbl)
Inventory cost of charging tank: 0.006 k$/(period kbbl)
Demand of mixed oil by CDUs: Oil mix 1: 600 kbbl, Oil mix 2: 600 kbbl, Oil mix 3: 600 kbbl, Oil mix 4: 600 kbbl

Table 4.7 Computational performance for Examples 2-4

Example	Single Equations	Single Variables	Discrete Variables	Objective	CPU Time (s)	Relative MILP Gap % (Period)
2	1082	595	58	337.25	3.7	0% (1-10)
3	1890	943	108	224.2	14.1	5% (1-5), 3.5 (6-10), 1% (10-12)
4	2745	1390	108	411.775	37.2	5% (1-5), 3.5 (6-10), 1% (10-15)

4.5.2 Examples 2-4

Example 2 involves three single-parcel vessels (V1-V3), three storage tanks (ST1-ST3), three charging tanks (CT1-CT2) and two CDUs with 10-day scheduling horizon. Each storage tank stores only one type of crudes. Example 3 consists of three single-parcel vessels (V1-V3), three storage tanks (ST1-ST3), three charging tanks (CT1-CT3) and two CDUs with 12-day scheduling horizon. Crude mixing is involved in storage tanks. Example 4 presents an industry size problem involving three single-parcel vessels (V1-V3), six storage tanks (ST1-ST6), four charging tanks (CT1-CT4) and three CDUs with 15-day scheduling horizon. Tables 4.4-4.6 give the data for Examples 2-4 and Figures 4.3-4.6 show their oil flow networks respectively. The computational performance for Examples 2-4 is given in Table 4.7. Table 4.8 presents obtained operation schedule for Example 4.

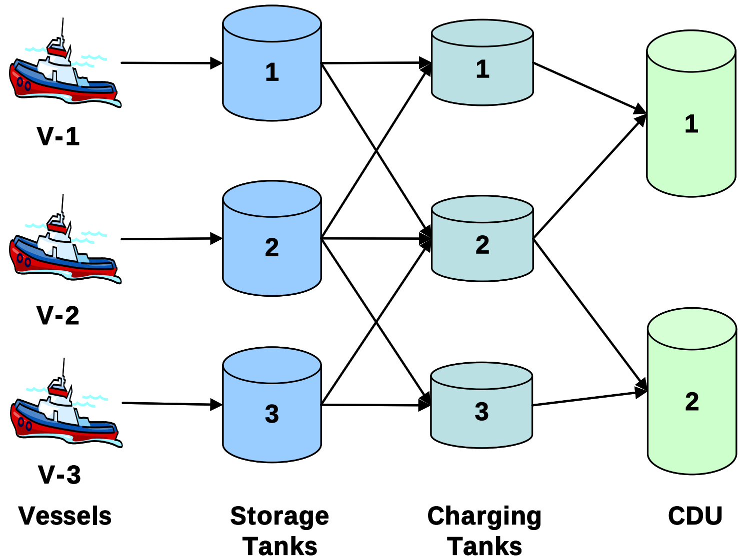

Figure 4.4 Oil flow network for Example 2

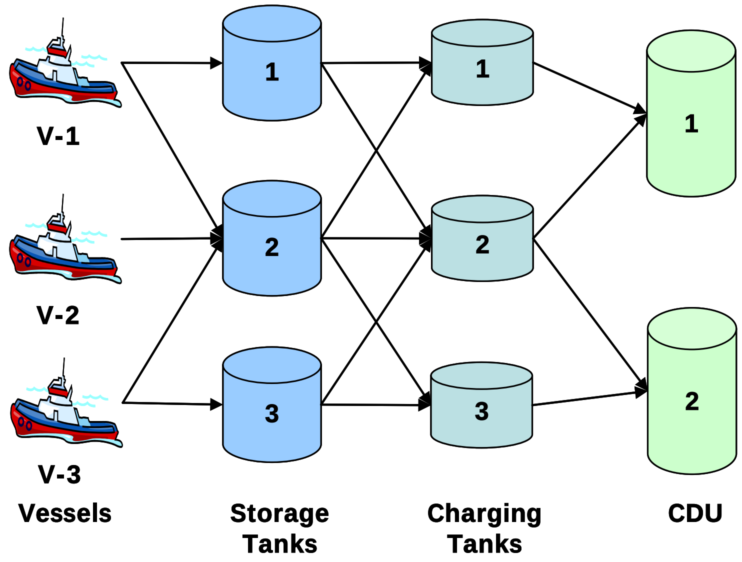

Figure 4.5 Oil flow network for Example 3

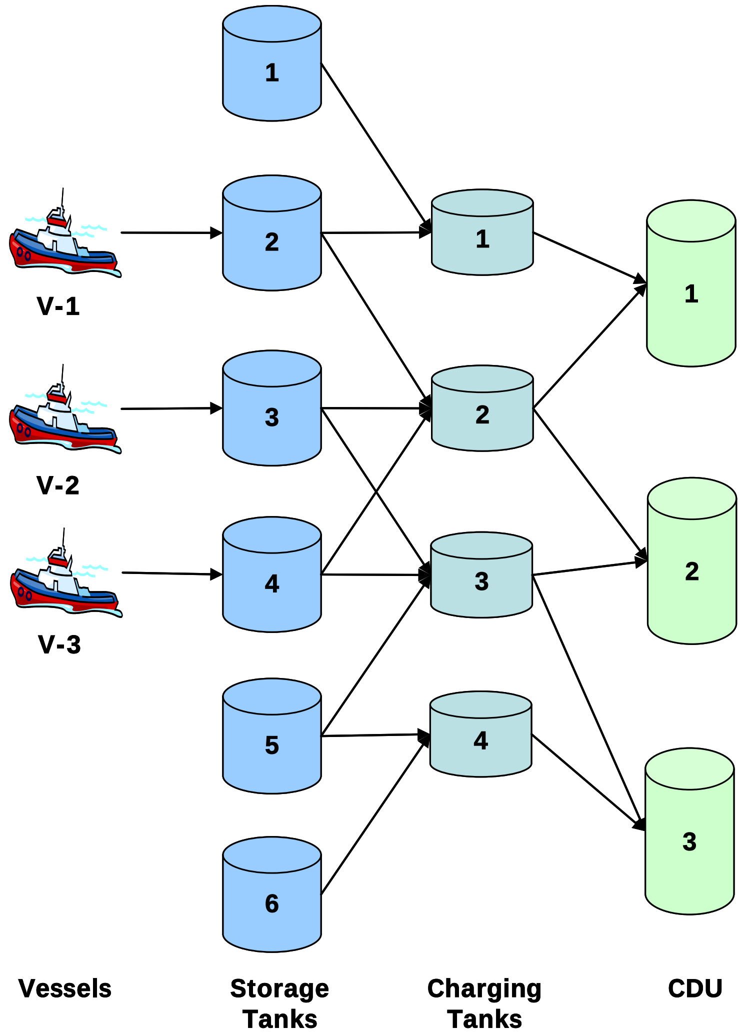

Figure 4.6 Oil flow network for Example 4

Table 4.8 Proposed operation schedule for Example 4

	Crude Amount [to Chargint Tank or CDU No.] (from Vessel No.) in kbbl for Period												
Tank	1	2	3	4	5	6	7	8	9	10-11	12	13	14-15
ST1				−50[1]									
ST2			+60(1)	−5[1]									
ST3					−30[2]			+60(2)					
ST4						−15[3]					−15[3]	+60(3)	
ST5									−20[4]				
ST6									−10[4]				
CT1					−10[1]	−5[1]	−5[1]	−5[1]	−5[1]	−5[1]	−5[1]	−5[1]	−5[1]
CT2	−10[1]	−5[1]	−5[1]	−5[1]		−5[2]	−5[2]	−5[2]	−5[2]	−5[2]	−5[2]		
CT3	−10[2]	−5[2]	−5[2]	−5[2]	−5[2]		−5[3]	−5[3]	−5[3]			−5[2]	−5[2]
CT4	−5[3]	−5[3]	−5[3]	−5[3]	−5[3]	−5[3]				−5[3]	−5[3]	−5[3]	−5[3]

'−' sign represents delivery to [Charging Tank], '+' sign represents receipt from (Parcels)

Table 4.9 Ship arrival data for Examples 5-22

Example	Arrival Period	Vessel (Crude-Parcel Size kbbl or kton*)
5, 11, & 17	1	VLCC-1 (C2-10, C1-300, C4-300, C3-340)
6, 12, & 18	1	VLCC-1 (C3-10, C4-250, C2-300, C1-190)
	14	VLCC-2 (C1-10, C3-250, C2-250, C4-240)
7, 13, & 19	1	VLCC-1 (C3-10, C5-350, C4-200, C2-300)
	16	VLCC-2 (C2-10, C6-200, C1-250, C5-240)
	28	VLCC-3 (C5-10, C1-250, C3-250, C4-190)
8	1	VLCC-1 (C3-10, C5-350, C4-200, C2-300)
	3	V2 (C6-210)
	4	V3 (C7-200)
	16	V4 (C4-250), V5 (C5-240), V6 (C6-270)
	17	V7 (C4-180), V8 (C8-300)
	28	VLCC-2 (C2-10, C1-250, C3-250, C7-190)
9	3	VLCC-1 (C3-10, C5-230, C4-200, C2-150)
	3	V2 (C6-210)
	4	V3 (C7-200)
	16	V4 (C4-250), V5 (C5-240), V6 (C6-270)
	17	V7 (C4-180), V8 (C8-300)
	28	VLCC-2 (C2-10, C1-250, C3-250, C7-190)
10	3	VLCC-1 (C1-30, C5-250, C3-200, C5-350, C4-20)
	3	V2 (C6-200)
	4	V3 (C7-250)
	16	V4 (C4-240), V5 (C7-150), V6 (C6-180)
	17	V7 (C3-200), V8 (C8-210)
	28	VLCC-2 (C4-180, C2-300, C1-210)
14-[15, 21]	1[3]	VLCC-1 (C3-10, C5-350, C4-200, C2-300)
	3	V2 (C6-210)
	4	V3 (C7-200)
	16	V4 (C1-250), V5 (C8-240), V6 (C6-270)
	17	V7 (C4-180), V8 (C5-300)
	28	VLCC-2 (C2-10, C1-250, C3-250, C7-190)
16, [22]	3	VLCC-1 (C1-30, C5-250, C3-200, C5-350, C4-20)
	3	V2 (C6-200) [V2 (C6-180)]
	4	V3 (C7-250) [V3 (C8-190)]
	16	V4 (C4-240), V5 (C7-270), V6 (C6-180)
		[V4 (C4-240], V5 (C7-160), V6 (C6-180)]
	17	V7 (C3-300), V8 (C8-210)
	28	VLCC-2 (C4-180, C2-300, C1-210)
20	1	VLCC-1 (C3-10, C5-350, C4-200, C2-300)
	3	V2 (C6-210)
	4	V3 (C7-200)
	16	V4 (C1-190), V5 (C8-240), V6 (C6-250)
	17	V7 (C4-180), V8 (C5-200)
	28	VLCC-2 (C2-10, C1-250, C3-250, C7-190)

Data in [ ] for Examples 15, 21, & 22

Table 4.10 Tank capacities, heels, and initial inventories for Examples 5-22

	Capacity (kbbl or kton*)				Heel (kbbl)				Initial Inventory (kbbl or kton*)					
Tank	Ex 5, 11 & 17	Ex 6, 12 & 18	Ex 7, 13 & 19	Ex 8-10 14-16 & 20-22	Ex 5, 11 & 17	Ex 6, 12 & 18	Ex 7, 13 & 19	Ex 8-10 14-16 & 20-22	Ex 5, 11 & 17	Ex 6	Ex 7, 13 & 19	Ex 8-10 & 16	Ex 12 & 18	Ex 14-15 20-22
ST1	700	600	600	600	50	60	60	60	300	250	250	250	150	250
ST2	700	600	600	600	50	60	60	60	250	300	300	300	250	300
ST3	700	600	600	600	50	60	60	60	300	200	200	200	200	200
ST4	700	600	600	600	50	60	60	60	320	350	350	350	300	350
ST5	-	-	600	600	-	-	60	60	-	-	210	210	-	240
ST6	-	-	600	600	-	-	60	60	-	-	320	200	-	200
ST7	-	-	-	600	-	-	-	60	-	-	-	150	-	280
ST8	-	-	-	600	-	-	-	60	-	-	-	240	-	270
CT1	700	700	700	700	50	50	50	50	300	350	300	300	350	300
CT2	700	700	700	700	50	50	50	50	300	400	400	400	400	400
CT3	900	700	700	700	50	50	50	50	250	350	300	300	350	300
CT4	700	700	700	700	50	50	50	50	300	450	450	450	450	200
CT5	-	-	-	700	-	-	-	50	-	-	-	150	-	350
CT6	-	-	-	700	-	-	-	50	-	-	-	80	-	400
CT7	-	-	-	700	-	-	-	50	-	-	-	80	-	300
CT8	-	-	-	700	-	-	-	50	-	-	-	250	-	250

Table 4.11a Initial compositions of crudes C1, C2, C5, and C6 for Examples 5-22

	C1 or C5									C2 or C6								
Tank	Ex 5	Ex 6	Ex 7	Ex 8-9 & 10	Ex 11 & 17	Ex 12 & 18	Ex 13 & 19	Ex 14-15 [20-21]	Ex 16 & [22]	Ex5	Ex 6	Ex 7	Ex 8-9 & 10	Ex 11 & 17	Ex 12 & 18	Ex 13 & 19	Ex 14-15 & [20-21]	Ex 16 & [22]
ST1	300	250	250	250	100	40	50	70	100 [70]	-	-	-	-	200	30	100	60	50 [60]
ST2	-	-	-	-	150	50	100	70	90 [70]	250	300	300	300	100	50	100	80	100 [80]
ST3	-	-	-	-	-	50	100	60	50 [60]	-	-	-	-	-	50	50	50	50
ST4	-	-	-	-	-	50	150	90	50 [90]	-	-	-	-	-	50	100	80	100 [80]
ST5	-	-	210	210	-	-	70	60	60 [50]	-	-	-	-	-	-	70	70	40 [70]
ST6	-	-	-	-	-	-	100	50	50	-	-	320	200	-	-	120	50	50
ST7	-	-	-	-	-	-	-	70	40 [70]	-	-	-	-	-	-	-	70	40 [70]
ST8	-	-	-	-	-	-	-	60	90 [50]	-	-	-	-	-	-	-	65 [50]	50 [60]
CT1	200	50	100	100	120	50	100	100	100	100	100	100	50	180	100	100	50	50
CT2	-	100	0	100	-	100	200	100	100	-	100	0	100	-	100	100	100	100
CT3	-	100	0	100	-	100	100	100 [60]	100 [60]	-	100	0	100	-	100	80	80	100 [80]
CT4	130	100	150	150	180	100	150	50	150 [50]	170	100	200	100	120	100	200	50	100 [50]
CT5	-	-	-	50	-	-	-	50	50 [100]			-	50	-	-	-	100	50 [100]
CT6	-	-	-	20	-	-	-	120 [100]	20 [80]			-	20	-	-	-	100 [120]	20 [120]
CT7	-	-	-	20	-	-	-	80	20 [80]			-	20	-	-	-	90	20 [90]
CT8	-	-	-	50	-	-	-	50	50			-	100	-	-	-	100	100

Table 4.11b Initial compositions of crudes C3, C4, C7, and C8 for Examples 5-22

	C3 or C7									C4 or C8								
Tank	Ex 5	Ex 6	Ex 7	Ex 8-9 & 10	Ex 11 & 17	Ex 12 & 18	Ex 13 & 19	Ex 14-15 & [20-21]	Ex 16 & [22]	Ex 5	Ex 6	Ex 7	Ex 8-9 & 10	Ex 11 & 17	Ex 12 & 18	Ex 13 & 19	Ex 14-15 & [20-21]	Ex 16 & [22]
ST1	-	-	-	-	-	60	100	60	50 [60]	-	-	-	-	-	20	-	60	50 [60]
ST2	-	-	-	-	-	100	100	70	50 [70]	-	-	-	-	-	50	-	80	60 [80]
ST3	300	200	200	200	150	50	50	50	50	-	-	-	-	150	50	-	40	50 [40]
ST4	-	-	-	-	200	100	-	90	100 [90]	320	350	350	350	120	100	100	90	100 [90]
ST5	-	-	-	-	-	-	-	50	70 [60]	-	-	-	-	-	-	70	60	40 [60]
ST6	-	-	-	-	-	-	-	50	50	-	-	-	-	-	-	100	50	50
ST7	-	-	-	150	-	-	-	70	30 [70]	-	-	-	-	-	-	-	70	40 [70]
ST8	-	-	-	-	-	-	-	65 [100]	50 [100]	-	-	-	240	-	-	-	60	50 [60]
CT1	-	100	100	100	-	100	100	100	100	-	100	0	50	-	100	-	50	50
CT2	100	100	0	100	90	100	-	100	100	200	100	100	100	210	100	100	100	100
CT3	50	50	0	60	125	50	-	60	60	200	100	120	40	125	100	120	60 [100]	40 [100]
CT4	-	100	100	100	-	100	100	50	100 [50]	-	150	0	100	-	150	-	50	100 [50]
CT5				20	-	-	-	100	20 [50]				30	-	-	-	100	30 [100]
CT6				20	-	-	-	80	20 [100]				20	-	-	-	100	20 [100]
CT7				20	-	-	-	60	20 [60]				20	-	-	-	70	20 [70]
CT8				50	-	-	-	50	50				50	-	-	-	50	50

Table 4.12 Crude concentration ranges in tanks and CDUs for Examples 5-20

Example	Crude: Concentration Range [Min, Max] for Tanks and CDUs
5	C1: [1, 1] for ST1, C2: [1, 1] for ST2, C3: [1, 1] for ST3, C4: [1, 1] for ST4
	C1: [0.2, 0.8] for CT1 & CT4; C2: [0.2, 0.8] for CT1 & CT4
	C3: [0.2, 0.8] for CT2, [0, 1] for CT3; C4: [0.2, 0.8] for CT2, [0, 1] for CT3
	C1-C2: [0.3, 0.7] for CDU 1; C3-C4: [0, 1] for CDU 2
6	C1: [1, 1] for ST1, C2: [1, 1] for ST2, C3: [1, 1] for ST3, C4: [1, 1] for ST4
	C1-C4: [0, 1] for ST1-ST4
	C1-C4: [0.06, 0.90] for CDU 1; C1- C4: [0.05, 0.92] for CDU 2
7	C1: [1, 1] for ST1, C2: [1, 1] for ST2, C3: [1, 1] for ST3
	C4: [1, 1] for ST4, C5: [1, 1] for ST5, C6: [1, 1] for ST6
	C1-C3: [0, 1] for CT1 and CT4; C4-C6: [0, 1] for CT2 and CT3
	C1-C3: [0.1, 0.9] for CDU 1, C4-C6: [0.16, 0.90] for CDU 2
8-10	C1: [1, 1] for ST1, C2: [1, 1] for ST2, C3: [1, 1] for ST3, C4: [1, 1] for ST4
	C5: [1, 1] for ST5, C6: [1, 1] for ST6, C7: [1, 1] for ST7, C8: [1, 1] for ST8
	C1-C4: [0, 1] for CT1, and CT6-CT8; C5-C8: [0, 1] for CT2-CT5
	C1-C4: [0, 1] for CDU 3; C5-C8: [0, 1] for CDU 1 and CDU 2
11, 17	C1-C2: [0, 1] for ST1 and ST2; C3-C4: [0, 1] for ST3 and ST4
	C1-C2: [0.1, 0.9] for CT1 and CT4;
	C3-C4: [0.1, 0.9] for CT2; C3-C4: [0, 1] for CT3
	C1-C2: [0.2, 0.8] for CDU 1; C3-C4: [0, 1] for CDU 2
12, 18	C1-C4: [0, 1] for ST1-ST4
	C1-C4: [0, 1] for CT1-CT4
	C1-C4: [0.05, 0.95] for CDU 1
	C1-C4: [0.15, 0.85] for CDU 2
13	C1-C3: [0, 1] for ST1-ST3; C4-C6; [0, 1] for ST4-ST6
[19]	C1-C3: [0, 1] for CT1 and CT4; C4-C6: [0, 1] for CT2-CT3
	C1-C3: [0.1, 0.9] for CDU 1 [C1-C3: [0, 1] for CDU 1]
	C4-C6: [0.1, 0.9] for CDU 2 [C4-C6: [0, 1] for CDU 2]
14-16	C1-C4: [0, 1] for ST1-ST4; C5-C8: [0, 1] for ST5-ST8
[20-22]	C1-C4: [0, 1] for CT1, CT6-CT8; C5-C8: [0, 1] for CT2-CT5
	C1-C4: [0, 1] for CDU 3
	C5-C8: [0, 1] for CDU 1 and CDU 2

Data in [] for Examples 19-22

Table 4.13 Transfer rates, processing limits, operating costs, crude margins, and demands for Examples 5-22

Example	Flow Rate Limits (kbbl or kton*/period) Parcel-Tank Min-Max	Storage Tank - Charing Tank Min-Max	Tank-CDU Min-Max	Demurrage (k$/period)	Changeover Loss (k$ /instance)	Inventory Penalty ($/bbl/ period)	Desired Safety Stock (kbbl)	Crude	Margin ($/bbl) Ex 5 11 & 17	Ex 6, 12, & 18	Ex 7 13, & 19	Ex 8-10 14-16 20-22
5, 11	10-400	0-80	10-60	100	5	0.2	1200	C1	3	1.50	1.90	1.50
6, 12	10-400	0-50	20-45	25	10	0.2	1500	C2	4.5	1.70	1.70	1.70
7-10, 13	10-400	0-100	10-80	25	10	0.2	1500	C3	5	1.50	1.80	1.50
14-16	10-400	0-100	10-80	25	10	0.2	1500	C4	6	1.60	1.40	1.60
Ex 17	10-400	0-80	10-60	100	5	0.2	1200	C5	-	-	1.45	1.45
Ex 18	10-400	0-50	20-45	25	10	0.2	1500	C6	-	-	1.35	1.60
Ex 19	10-400	0-100	10-80	25	10	0.2	1500	C7	-	-	-	1.55
Ex 20-22	10-400	0-100	10-80	25	10	0.2	1500	C8	-	-	-	1.60

CDU	Processing Limits (kbbl or kton*/period) Ex 5, 11 & 17	Ex 6, 12 & 18	Ex 7, 13 & 19	Ex 8-10	Ex 14-16	Ex 20-22	Total Demand (kbbl) Ex 5, 11 & 17	Ex 6, 12 & 18	Ex 7, 13 & 19	Ex 8-10	Ex 14-16	Ex 20-22
CDU 1	50-100	20-45	20-50	20-50	20-50	20-50	550	750	1000	900	900	900
CDU 2	50-100	20-45	20-50	20-50	20-50	20-50	550	750	1000	900	900	900
CDU 3	-	-	-	20-50	20-50	20-50	-	-	-	900	900	900

Table 4.14a Specific gravities, sulfur contents, nitrogen contents, carbon residues, pour point, freeze point, and flash point for crudes and acceptable ranges for feeds to CDUs

Crude & CDU		Specific Gravity	Sulfur				Nitrogen	Carbon Residue	Pour Point	Freeze Point	Flash Point
		Ex 8-10 14-16 20-22	Ex 5 11, & 17	Ex 6 12, & 18	Ex 7 13, & 19	Ex 8-10 14-16 20-22	Ex 8-10 14-16 20-22	Ex 8-10 14-16 20-22	Ex 8-10 14-16 20-22	Ex 8-10 14-16 20-22	Ex 8-10 14-16 20-22
	C1	1.2576	0.0050	0.0120	0.0015	0.0135	32.00	0.0620	13.3290	36.3479	341.6801
	C2	1.2646	0.0065	0.0130	0.0028	0.0115	21.00	0.0450	27.7594	10.7753	337.8010
	C3	1.2466	0.0165	0.0090	0.0040	0.0150	45.00	0.0750	10.3347	54.8962	357.1811
	C4	1.2599	0.0145	0.0150	0.0140	0.0120	25.00	0.0500	18.8718	26.2774	341.4362
	C5	1.0892	-	-	0.0090	0.0075	79.00	0.2400	2.5140	812.1963	2.1225
	C6	1.1207	-	-	0.0150	0.0050	62.00	0.1000	9.3209	712.1419	4.5253
	C7	1.1105	-	-	-	0.0070	73.00	0.1800	3.9516	757.3304	2.6029
	C8	1.1148	-	-	-	0.0065	65.00	0.1300	7.1733	744.1575	3.3367
CDU1	Min	1.0000	0.0040	0.0010	0.0010	0.0050	60.00	0.1000	2.5000	700.0000	2.0000
	Max	1.1200	0.0058	0.0124	0.0027	0.0071	75.00	0.2000	9.0000	810.0000	4.5000
CDU2	Min	1.0000	0.0100	0.0010	0.0030	0.0050	60.00	0.1000	2.5000	700.0000	2.0000
	Max	1.1200	0.0156	0.0127	0.0120	0.0070	78.00	0.2200	9.2000	810.0000	4.4800
CDU3	Min	1.0000	-	-	-	0.0100	20.00	0.0100	10.0000	10.0000	300.0000
	Max	1.2625	-	-	-	0.0138	40.00	0.0720	27.5000	54.0000	350.0000

Table 4.14b Smoke point, Ni, Reid vapor pressure, asphaltenes, aromatics, paraffins, naphthenes, viscosity for crudes and acceptable ranges for feeds to CDUs

Crude & CDU		Smoke Point Ex 8-10 14-16 20-22	Ni Ex 8-10 14-16 20-22	RVP Ex 8-10 14-16 20-22	Asphaltenes Ex 8-10 14-16 20-22	Aromatics Ex 8-10 14-16 20-22	Paraffins Ex 8-10 14-16 20-22	Naphthenes Ex 8-10 14-16 20-22	Viscosity Ex 8-10 14-16 20-22
	C1	588.3175	0.900	133.1031	0.0350	0.0892	0.7273	0.2892	71.7767
	C2	706.1862	0.760	92.4005	0.0250	0.0465	0.7366	0.2835	64.5435
	C3	557.2127	1.000	158.3207	0.0150	0.0642	0.6341	0.2281	74.7458
	C4	608.9218	0.850	116.6061	0.0200	0.0635	0.7335	0.2901	68.1196
	C5	419.5659	12.500	28.6244	0.1150	0.3216	0.3282	0.2767	83.1872
	C6	536.3974	5.400	19.4676	0.0900	0.3130	0.4035	0.2200	79.3574
	C7	427.1329	10.300	26.2276	0.1350	0.4437	0.3500	0.2085	82.5847
	C8	457.4626	7.900	24.1774	0.1005	0.3599	0.3892	0.1990	81.5982
CDU1	Min	400.0000	5.000	15.0000	0.0900	0.3000	0.3000	0.1800	70.0000
	Max	530.0000	12.200	28.3000	0.1300	0.4200	0.4000	0.2700	83.0000
CDU2	Min	400.0000	5.000	15.0000	0.0900	0.3000	0.3000	0.1800	70.0000
	Max	520.0000	12.100	28.5000	0.1320	0.4000	0.3950	0.2750	82.9000
CDU3	Min	500.0000	0.500	90.0000	0.0150	0.0400	0.6000	0.2000	60.0000
	Max	700.0000	0.980	155.0000	0.0320	0.0850	0.7350	0.2900	74.5000

Table 4.15 Different operation features for Examples 5-22

Ex	SBM	Jetty	Holdups in SBM	VLCC	Single-Parcel Vessel	Storage Tank	Charging Tank	CDU	Arrival Time of the 1st Vessel	Crude Segregation	Crude Property	Blending in Storage Tank	Composition in Charging Tank	Period
5	1	0	One	1	0	4	4	2	0	YES	1	NO	Variable	9
6	1	0	One	2	0	4	4	2	0	NO	1	NO	Variable	20
7	1	0	One	3	0	6	4	2	0	YES	1	NO	Variable	42
8	1	3	One	2	7	8	8	3	0	YES	15	NO	Variable	42
9	1	3	One	2	7	8	8	3	2	YES	15	NO	Variable	42
10	1	3	Multiple	2	7	8	8	3	2	YES	15	NO	Variable	42
11	1	0	One	1	0	4	4	2	0	YES	1	YES	Variable	9
12	1	0	One	2	0	4	4	2	0	NO	1	YES	Varialbe	20
13	1	0	One	3	0	6	4	2	0	YES	1	YES	Variable	42
14	1	3	One	2	7	8	8	3	0	YES	15	YES	Variable	42
15	1	3	One	2	7	8	8	3	2	YES	15	YES	Variable	42
16	1	3	Multiple	2	7	8	8	3	2	YES	15	YES	Variable	42
17	1	0	One	1	0	4	4	2	0	YES	1	YES	Prefixed	9
18	1	0	One	2	0	4	4	2	0	NO	1	YES	Prefixed	20
19	1	0	One	3	0	6	4	2	0	YES	1	YES	Prefixed	42
20	1	3	One	2	7	8	8	3	0	YES	15	YES	Prefixed	42
21	1	3	One	2	7	8	8	3	2	YES	15	YES	Prefixed	42
22	1	3	Multiple	2	7	8	8	3	2	YES	15	YES	Prefixed	42

Table 4.16 Model and solution statistics for Examples 5-22

Ex	Single Equations	Continuous Variables	Discrete Variables	Non Zero Elements	Objective (k$)	CPU Time (s)	Relative MILP Gaps for Feasible Solution (Period)	Relatvie MILP Gaps for Quality Improvement (Period)
5	1235	541	51	3239	5071.67	7.81	0% (1-9)	0% (1-9)
6	3669	2305	140	13960	2371.06	48.77	5% (1-5), 3.5% (6-10), 1% (11-15), 0% (16-20)	0.5% (1-20)
7	4496	3197	96	14163	3209.51	34.59	7% (1-18), 4% (19-28), 2% (29-35), 0% (36-42)	1% (1-42)
8	13272	8261	234	97291	4179.50	318.75	7% (1-18), 4% (19-28), 2% (29-35), 0% (36-42)	1% (1-42)
9	13270	8265	234	97295	4130.71	614.25	7% (1-18), 4% (19-28), 2% (29-35), 0% (36-42)	1% (1-42)
10	13290	8270	239	98684	4183.19	410.89	7% (1-18), 4% (19-28), 2% (29-35), 0% (36-42)	1% (1-42)
11	1424	768	54	3950	5172.85	16.01	0% (1-10)	0% (1-9)
12	4472	3970	136	18911	2368.07	121.82	5% (1-5), 3.5% (6-10), 2% (11-15), 0% (16-20)	0.5% (1-20)
13	5362	5499	90	19101	3098.12	76.07	7% (1-18), 4% (19-28), 2% (29-35), 0% (36-42)	2% (1-42)
14	16414	15176	261	117248	4171.14	2826.70	7% (1-18), 4% (19-28), 2% (29-35), 0% (36-42)	2% (1-42)
15	166660	15188	253	119108	4181.02	1290.89	7% (1-18), 4% (19-28), 2% (29-35), 0% (36-42)	2% (1-42)
16	16681	15197	260	117878	4127.09	2411.02	7% (1-18), 4% (19-28), 2% (29-35), 0% (36-42)	2% (1-42)
17	1344	768	54	3754	5115.00	4.31	0% (1-11)	0% (1-9)
18	4840	3970	136	17967	2350.32	156.98	5% (1-5), 3.5% (6-10), 1% (11-15), 0% (16-20)	0.5% (1-20)
19	6022	5499	90	19749	3089.03	158.38	7% (1-18), 4% (19-28), 2% (29-35), 0% (36-42)	3% (1-42)
20	18886	15176	261	121016	4057.39	278.69	7% (1-18), 4% (19-28), 2% (29-35), 0% (36-42)	3% (1-42)
21	19132	15188	253	121532	4158.61	3402.81	7% (1-18), 4% (19-28), 2% (29-35), 0% (36-42)	3% (1-42)
22	19153	15197	260	120302	4146.18	1159.63	7% (1-18), 4% (19-28), 2% (29-35), 0% (36-42)	3% (1-42)

Note: Pure crude in each storage tank and crude blend in each charging tank for Examples 5-10
Crude blend in each storage and charging tank for Examples 11-16
Crude blend in each storage and charging tank, but prefixed crude composition in each charging tank for Examples 17-22

Table 4.17a Operation schedule for vessel unload, storage tank receipt and feed for Example 7

	Crude Amount [to Charging Tank No.] (from Vessel No.) in kbbl for Period													
Tank	1	2	3	4	5	6	7	8	9	10	11	12	13	14
ST2	−100[6]	−100[6]	+300(4)		−72.0[6]			−17.6[7]						−100[6]
	−100[7]	−100[7]												
ST3	+10(1)													
ST4		+200(3)												
ST5	+350(2)													
ST6	−100[3]			−5.0[5]							−0.43[3]		−100[3]	−14.6[3]
	−100[5]													
ST8	−5.4[3]	−5.4[3]	−5.4[3]	−5.4[3]	−5.4[3]	−5.4[3]	−5.4[3]	−5.4[3]	−0.02[3]	−25.4[3]	−100[3]	−5.4[3]	−5.4[3]	−5.4[3]
	15	16	17	18	19	22	24	25	26	27	28	29	30	42
ST1											+250(10)			
ST2		+10(5)			−88.73[1]				−92.7[1]	−89.0[1]	−90[1]			
ST3												+250(11)		
ST4														−97.6[8]
ST5				+240(8)							+10(9)			
ST6	−100[4]	−100[4]	+200(6)		−20.00[4]	−100[4]		−100[4]			−100[3]			
								−100[5]						
ST7													+190(12)	
ST8	−3.12[4]		+90(7)	+160(7)		−100[5]	−100[5]	−100[5]	−100[4]		−100[3]			
	−45.0[5]													

'–' sign represents delivery to [Charging Tank], '+' sign represents receipt from (Parcels)

Table 4.17b Operation schedule for charging tank feed to CDU for Example 7

	Crude Amount [to CDU No.] (from Storage Tank No.) in kbbl for Period									
Tank	1-5	6-7	8	9	10	11	12	13	14	15
CT1	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]
CT2	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]
CT3										−20.0[1]
CT4	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	
CT8	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]
	16-18	19	20-22	23-25	26	27	28	29-31	32-41	42
CT1								−20.0[3]	−10.0[3]	−10.0[3]
CT3	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]		−10.0[1]	−10.0[1]	−10.0[1]
	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]		−10.0[2]	−10.0[2]	−10.0[2]
CT4							−20.2[1]	−14.3[1]	−14.3[1]	−14.3[1]
CT5							−20.2[2]	−14.3[2]	−14.3[2]	−14.3[2]
CT6	−12.2[3]	−12.2[3]	−12.2[3]	−12.2[3]	−12.2[3]	−12.2[3]	−12.2[3]		−12.8[3]	−12.8[3]
CT7	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]			

Table 4.18a Operation schedule for vessel unload, and storage tank receipt and feed for Example 16

	Crude Amount [to Charging Tank No.] (from Vessel No.) in kbbl for Period															
Tank	1	2	3	4	5	6	7	8	10	11	13	15	16	17	18	19
ST1							+20(5)								+300(11)	
ST2						–100[7]	–1.02[7]			–100[7]						
ST3			+30(1)													
ST4	–28.7[7]	–61.4[7]			+200(3)							–100[1] –100[8]	–100[1] –100[8]	+240(8)		–40[6]
ST5	–100[5]			–50[5]	+250(7)					–28.95[5]			–91[2]	+180(10)		–100[3]
ST6			–40[3]		–100[3]	+340(4)							+200(9)			
ST7			–10[3]	+10(2)		+10(4)		–81.6[5]	–18.3[5]		–0.05[5]				+210(12)	
ST8	–100[5]		+200(6)	+240(2)						–50[5]			+70(9)			

Tank	20	21	22	23	27	28	29	30	31	32	33	34	35	36-37	42
ST1								+30(15)							
ST2	–38.98[6]					+180(13)		–86.7[7]	–0.23[8]						–0.53[1]
ST3								+170(15)							–6.74[1]
ST4					–100[6]	–100[6]	+300(14) +10(15)		–29.31[1] –71.43[8]	–29.31[1]	–29.3[1]	–92.8[1]	–29.3[1]		–19.4[1]
ST5	–100[4]	–11.1[4]	–99[4]												
ST7	–100[2]		–100[4]	–9.9[4]	–0.05[5]										
ST8								–63.4[5]	–22.83[2]	–0.36[2]	–0.36[2]	–0.36[2]	–0.36[2]	–0.36[2]	

'–' sign represents delivery to [CDU], '+' sign represents receipt from (Storage Tank)

Table 4.18b Operation schedule of charging tank feed to CDU for Example 16

Tank	Crude Amount [to CDU No.] in kbbl for Period												
	1	2-3	4	5	6-14	15	16	17	18	19	20	21	22
CT1	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]								
CT2	−20.0[1]	−10.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]						−20.0[1]	−20.0[1]
CT3											−25.3[2]	−30.4[2]	−34.6[2]
CT4	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20[2]	−22[2]	−26.4[2]	−31.7[2]			
CT5		−10[1]					−20[1]	−20[1]	−20.0[1]	−20.0[1]	−20.0[1]		
CT7						−20.0[3]	−20[3]	−20[3]	−20.0[3]	−20.0[3]	−20.0[3]	−20.0[3]	−20.0[3]
CT8	−14.3[3]	−14.3[3]	−14.3[3]		−14.3[3]								
	23	24	25-28	29	30	31	32	33	34	35	36	37	38-42
CT2	−20.0[1]	−20.0[1]	−20.0[1]	−10.0[1]									
CT3	−27.6[2]	−22.1[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20[2]	−20[2]	−20[2]	−20[2]
CT4					−24.0[1]	−28.8[1]	−34.6[1]	−28.7[1]	−34.4[1]	−27.5[1]	−22.0[1]	−20[1]	−20[1]
CT5				−10.0[1]									
CT6					−20.0[3]	−20.0[3]	−20.0[3]	−20.0[3]	−20.0[3]	−20[3]	−20[3]	−20[3]	−20[3]
CT7	−20.0[3]	−20.0[3]	−20.0[3]	−20.0[3]									
CT8													

Table 4.19a Operation schedule for vessel unload, and storage tank receipt and feed for Example 22

	Crude Amount [to Charging Tank No.] (from Vessel No.) in kbbl for Period							
Tank	2	3	4	5	6	13	14	16
ST1					+20(5)			+240(8)
ST2		+30(1)						
ST3				+200(3)				
ST5				+190(7)		–100[3]	–20.7[4]	
ST6	–20.0[4]	+180(6)				–55.3[3]	–17.3[4]	
ST7		–100[4]		–13.9[4]	+350(4)	–18.8[3]	–11.1[4]	
ST8			+250(2)			–39.2[3]	–20.8[4]	+160(9)
	17	18	19	20-21	26	28	29	30
ST1						+180(13)		+110(15)
ST2		+300(11)						
ST3							+300(14)	
							+100(15)	
ST4						–100[6]		
			–10.5[3]					
ST5	+180(10)		–12.4[3]	–84.8[3]	–37.4[4]	–40.1[4]		
ST6	+210(12)			–100[3]	–93.4[4]	–100[4]		
ST7			–5.11[3]	–41.1[3]	–60.6[4]	–64.9[4]		
ST8			–6.83[3]	–55.0[3]	–77.2[4]	–82.7[4]		

'–' sign represents delivery to [CDU], '+' sign represents receipt from (Storage Tank)

Table 4.19b Operation schedule for charging tank feed to CDU for Example 22

	Crude Amount [to CDU No.] in kbbl for Period											
Tank	1	2	3	4	5-11	12	13	14	15	16	17	18
CT2	−10.0[1]	−10.0[1]	−10.0[1]	−10.0[1]	−10.0[1]	−10.0[1]	−10[1]	−10[1]	−10[1]	−10[1]	−10[1]	−10[1]
CT3	−20.0[2]	−22.2[2]	−26.6[2]	−21.3[2]	−20.0[2]	−20.0[2]		−24.0[2]	−28.8[2]	−34.6[2]	−31.3[2]	−25[2]
CT4							−20[2]					
CT5	−10.0[1]	−10.0[1]	−10.0[1]	−10.0[1]	−10.0[1]	−10.0[1]	−10[1]	−10[1]	−10[1]	−10[1]	−10[1]	−10[1]
CT6	−20[3]	−20[3]	−20[3]	−20[3]	−20[3]	−20[3]	−20[3]	−20[3]	−20[3]	−20[3]	−20[3]	−10[3]
CT8												–11.4[3]
	19-25	26-29	30-31	32	33	34	35	36	37	38	39	40-42
CT1	−14.3[3]											
CT2	−10.0[1]	−10.0[1]										
CT3			−20.0[1]	−24.0[1]	−28.8[1]	−28.0[1]	−33.6[1]	−26.9[1]	−32.3[1]	−25.8[1]	−20.6[1]	−20.0[1]
		−20.0[2]	−20.0[2]	−24.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]
CT4	−20.0[2]											
CT5	−10.0[1]	−10.0[1]										
CT7	−11.4[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]	−10.0[3]
CT8		−11.1[3]	−11.1[3]	−11.1[3]	−11.1[3]	−11.1[3]	−11.1[3]	−11.1[3]	−11.1[3]	−11.1[3]	−11.1[3]	−11.1[3]

4.5.3 Examples 5-22

While Examples 5-10 involve policy 1, Examples 11-16 incorporate policy 2, and Examples 17-22 consider policy 3. Tables 4.9-4.14 give the data for these examples (Examples 5-22). These examples vary widely in structure, size, scale, and complexity (Table 4.15) and are representative of different industrial scenarios. All examples are solved on a Dell workstation PWS650 (Inter® Xeron™ CPU 3.06GHZ, 3.5 GB memory) running Windows NT using solver CPLEX 9.0.

Table 4.16 shows the model performance and solution statistics for Examples 5-22. For illustration, operation schedules for Examples 7, 13, and 19 for policies 1, 3, and 4 are given in Tables 4.17-4.19, respectively. These results show that the proposed algorithm in Chapter 3 successfully solves various problems in an inland refinery.

4.6 Summary

In this Chapter, a discrete-time MINLP formulation was developed for crude oil scheduling. It considered an in-land refinery configurations involving SBM, multiple jetties, storage tanks and charging tanks, and incorporated many real-life features such as multiple-parcel vessels (VLCCs), crude segregation, brine settling, crude blending, multiple tanks feeding one CDU at one time, one tank feeding multiple CDUs at one time and crude mixing in storage tanks or charging tanks or both. The proposed algorithm (RRA-P1) in Chapter 3 successfully solved four literature examples and other eighteen tested examples varying in structures, size, scale, complexity, and blending policies. Therefore, the proposed algorithm in Chapter 3 was successfully applicable to in-land refineryies, although it was intended for marine-access refineries.

As discussed in Chapter 2, intermediate processing is not addressed in this Ph.D project because of the highly complex and non-analytical models for production units such as crude distillation column (CDU), fluid catalytic cracking unit (FCCU), etc. which makes it the most difficult to solve. Therefore, in the next Chapter, the final component of overall refinery operation, i. e. product blending and distribution, will be addressed.

CHAPTER 5

RECIPE DETERMINATION AND SCHEDULING OF GASOLINE BLENDING AND DISTRIBUTION OPERATIONS

5.1 Introduction

Rising crude prices, deteriorating crude qualities, and growing environmental concerns are squeezing the profit margins of modern oil refineries like never before. Optimal scheduling of various operations in a refinery offers significant potential for saving costs and increasing profits. The overall refinery operations (Pinto et al., 2000; Jia and Ieraptritou, 2003b) involve three main segments, namely crude oil storage and processing, intermediate processing, and product blending and distribution. Scheduling of crude oil operations (Reddy et al., 2004a,b; Li et al., 2007) has received the most attention so far. However, only limited work exists on the scheduling of product blending and distribution operations.

Gasoline is one of the most profitable products of a refinery and can account for as much as 60-70% of total profit. A refinery typically blends several gasoline cuts or fractions from various processes to meet its customer orders of varying specifications. However, this process involves nonlinear blending and complex combinatorics, and can easily result in suboptimal schedules and costly quality give-aways. The large numbers of orders, delivery dates, blenders, blend components, tanks, quality specifications, etc. make this problem highly complex and nonlinear. Optimal scheduling using advanced techniques of mixed-integer programming are imperative for avoiding ship demurrage, improving order delivery and customer satisfaction, minimizing quality give-aways, reducing transitions and slop generation, exploiting low-quality cuts, and reducing inventory costs. Therefore, scheduling of gasoline blending and distribution is very crucial.

As discussed in Chapter 2, several works in the literature (Dewitt et al., 1989; Rigby et al., 1995; Pinto et al., 2000; Joly and Pinto, 2003; Glismann and Gruhn, 2001a,b; Mendez et al., 2006) have addressed the problem of product blending operations and incorporated some real-life operation features such as variable recipes, identical parallel blenders, etc. However, these works did not integrate the distribution operations with blending, and did not force the blending rate to be constant in a run and minimum run length. Jia and Ierapetritou (2003b) proposed a continuous-time event-based MILP formulation for scheduling gasoline blending and distribution operations simultaneously. However, their model lacked many key operation features such as multiple parallel non-identical blenders, variable recipes, and product specifications, etc. More importantly, their formulation gives infeasible solutions, and allows a product tank to hold several products at a time (Appendix A). Therefore, previous works have considered only pieces of the full product blending and distribution problem. An integrated treatment of recipe, blending, scheduling, and distribution is missing. Furthermore, works that did address product specifications used nonlinear correlations, which made the problem more difficult.

Therefore, in this Chapter, most of the drawbacks of the existing work are addressed, and general formulations are developed that incorporate several real-life operation features such as multi-purpose tanks, parallel non-identical blenders, constant rates during blending runs, minimum run lengths, changeovers, linear property indices (Li et al., 2005), piecewise constant profiles for blend component qualities and feed rates, etc. The blend component flow rates and qualities are allowed to be piecewise constant over the horizon by means of a multi-period formulation. Although the formulation is nonlinear and non-convex, a novel schedule adjustment procedure that solves only MILPs and no MINLP is developed.

A detailed problem statement is first presented. In section 5.3, a general single-period mathematical formulation is developed. Following that, a novel procedure that addresses the nonlinearity arising from forcing constant blending rates is proposed. In section 5.5, the single-period formulation is extended to multi-period scenario. Next, the proposed schedule adjustment procedure is illustrated with a small example, and the developed model and procedure are evaluated with thirteen additional examples. Lastly, the proposed procedure is compared with commercial MINLP solvers (DICOPT/GAMS and BARON/GAMS).

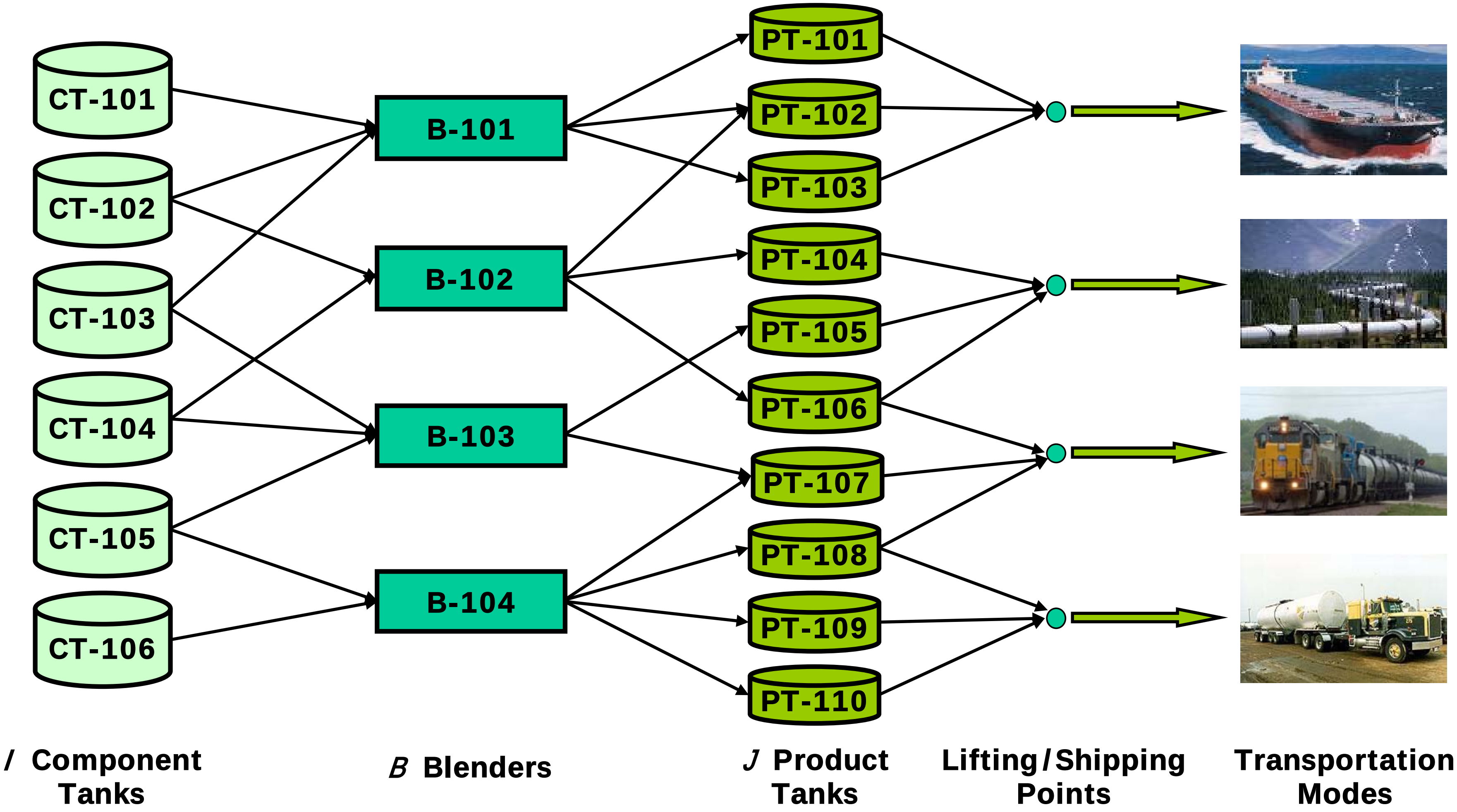

Figure 5.1 Schematic of gasoline blending and distribution

5.2 Problem Statement

Consider a gasoline blending and distribution unit (GBDU) in a typical refinery (Figure 5.1). It employs I component tanks (i = 1, 2, …, I), B blenders (b = 1, 2, …, B), J product tanks (j = 1, 2, …, J), and some lifting ports. The GBDU uses I components (i = 1, 2, …, I) to make P possible products (p = 1, 2, …, P). These components are gasoline fractions from various processes in a refinery such as atmospheric distillation, FCCU (Fluid Catalytic Cracking Unit), FCRU (Fluid Catalytic Reforming Unit), AKU (Alkanisation Unit), IFU (Isoforming Unit), CHU (Catalytic Hydrogenation Unit), ARU (Aromatization Unit), and various additives such as MTBE (Methyl Tert-Butyl Ether) and Butane to enhance octane rating or act as corrosion inhibitors or lubricators. Thus, different grades of gasoline contain different components. Each component has a pre-fixed, distinct, and known quality or specification, and component i is stored in its own dedicated component tank i. Note that even if several component tanks store one component, then they can be treated as one tank with no loss of generality. At time zero, it has O orders (o = 1, 2, …, O) to fulfill during the coming scheduling horizon [0, H]. Each order o has a time window $[DD_o^L, DD_o^U]$ for delivery. An order may involve multiple products, but it can be assumed with no loss of generality that each involves a single product, as each multi-product order can be broken into several single-product orders. Any delivery after DD_o^U incurs a demurrage cost (DM_o).

The quality of blend components is specified in terms of various property indices such as RBN (Research Octane Number Index), RVI (Reid Vapor Pressure Index), FPI

(Flash Point Index), VI (Viscosity Index), etc. The flow profiles over time of various blend components into respective component tanks are known a priori. The blend components from various component tanks are fed to the blenders in some proportions to make various products of desired quality at various times. The blenders are semi-continuous units that process products one at a time. The blended products from these units flow to assigned product tanks that may hold different products over time. The products from product tanks are loaded into vehicles or ships at appropriate times for the delivery of various orders.

The operation of this typical GBDU involves decisions such as recipe determination, allocation of component tanks to blenders, assignment of product tanks to products over time, and scheduling of blending, transfer, and delivery operations. With this, the gasoline blending and distribution problem addressed in this paper can be stated as:

Given:

(1) A scheduling horizon $[0, H]$.

(2) I components and their property indices.

(3) I component tanks, their initial inventories, limits on their holdups, flow profiles of feeds into the tanks, and limits on the flows out of the tanks.

(4) P products and specification limits on their property indices.

(5) B blenders, the products that each blender can process, lower limits on the blend times of these products, and limits on their blending rates.

(6) Product tanks, the products that each tank can store, limits on their holdups, the

products and holdups at time zero, and delivery (lifting) rates for various products.

(7) *O* orders, their constituent products, amounts, and delivery time windows.

(8) Revenues from product sales, component costs, inventory costs (for components and products), and demurrage costs for orders.

Determine:

1. The blenders that each component tank should feed over time, and at what flow rates.
2. The products that each blender should produce over time, and at what rates.
3. The products that each product tank should receive over time, from which blender, and at what flow rates.
4. The orders that each product tank should deliver over time and their amounts.
5. The inventory profiles of various tanks (component and product).

Assuming:

1. Flow rate profile of each component from the upstream process is piecewise constant.
2. Component inventories are sufficient for blending through the entire scheduling horizon.
3. Mixing in each blender is perfect.
4. Changeover times between products are negligible for both blenders and product tanks.
5. Each order involves only one product. As discussed earlier, each multi-product order can be decomposed into several single-product orders.

6. Each order is completed during the scheduling horizon.

Subject to the operating rules:

1. A blender can process at most one product at any time. Once it begins processing a product, it must operate for some minimum time, before it can switch to another product.
2. A blender can feed at most one product tank at any time. In addition to being the industry practice, this helps to decrease the number of tanks in use and increase their utilization.
3. A product tank cannot receive and deliver a product simultaneously.

Allowing:

1. A component tank may feed multiple blenders, and a blender may receive from multiple component tanks at the same time.
2. A blender may feed multiple product tanks during the scheduling horizon.
3. A product tank may deliver multiple orders at the same time.
4. Multiple tanks may deliver an order at the same time.

An MILP formulation is now developed for the above general problem. However, for the sake of simplicity, instead of presenting the most general formulation, the simplest scenario in which the flow rates of all components are constant over the entire scheduling horizon is first considered. This is also what most existing work assumes. In addition, most existing work considers a single or multiple identical blenders.

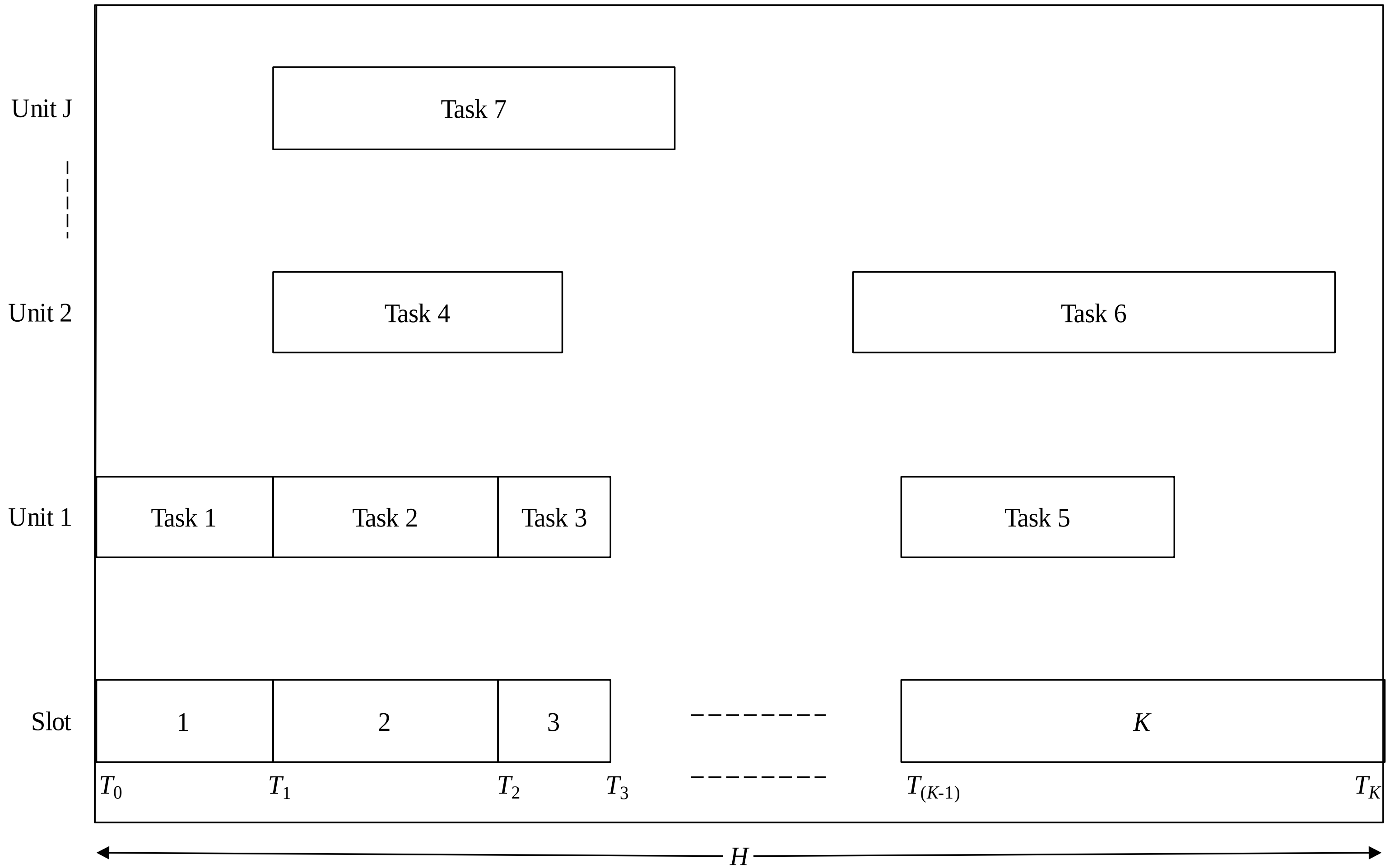

Figure 5.2 Schematic of slot design

5.3 Single-Period MILP

The horizon H is divided into K (k = 1, 2, …, K) process-slots (Liu and Karimi, 2007b) of variable lengths (SL_k). k = 0 denotes the time just before zero time. The process-slots are common to or synchronized across all units (tanks and blenders). Denoting T_0 as the start of the horizon and the end of slot k = 0, and T_k as the time at which slot k ends,

$$T_k = T_{(k-1)} + SL_k \qquad T_0 = 0,\ 0 < k \le K \tag{5.1}$$

with H as the upper bound of T_k for all k > 0. Figure 5.2 shows the schematic of the slot design. Each new operation (except idling) on a unit (tank or blender) is assumed to begin at the start of a slot, but may end at any time within a slot.

Throughout this Chapter, each variable is defined with specific ranges of its indices, and each constraint, unless otherwise indicated, is written for all valid values of the indices of its constituent variables.

5.3.1 Blending and Storage

At any time, a blender must be either running or idle. When running, it must be connected to a product tank. If idle, then it is connected to a dummy product tank (j = 0). Thus, there have J real product tanks (j = 1, 2, …, J) and one (j = 0) dummy product tank. Now, $\boldsymbol{BJ}$ = {(b, j) | blender b can feed product tank j} and a binary variable (v_{bjk}) are defined.

$$v_{bjk} = \begin{cases} 1 & \text{If blender } b \text{ feeds product tank } j \text{ during slot } k \\ 0 & \text{Otherwise} \end{cases} \qquad (b, j) \in \boldsymbol{BJ},\ 0 \le j \le J,\ 0 \le k \le K$$

v_{bjk} (j = 1, 2, …, J) is treated as binary and v_{b0k} as 0-1 continuous variable.

Each blender must feed exactly one product tank (real or dummy) in each slot.

$$\sum_{j=0}^{J} v_{bjk} = 1 \qquad (b, j) \in \boldsymbol{BJ},\ 0 < k \le K \qquad (5.2)$$

Let G_{bjk} be the volume that blender b feeds product tank j during slot k. If blender b does not feed tank j during slot k, then G_{bjk} must be zero.

$$G_{bjk} \le VP_j^U v_{bjk} \qquad (b, j) \in \boldsymbol{BJ},\ 0 < j \le J,\ 0 < k \le K \qquad (5.3)$$

where, VP_j^U is the maximum capacity of tank j.

To model the holdup in product tanks, $\boldsymbol{PJ}$ = {(p, j) | product tank j can hold product p} and one binary variable (u_{jpk}) are defined as follows,

$$u_{jpk} = \begin{cases} 1 & \text{If product tank } j \text{ holds product } p \text{ during slot } k \\ 0 & \text{Otherwise} \end{cases} \qquad (p, j) \in \boldsymbol{PJ},\ 0 < j \le J,\ 0 \le k \le K$$

Note that u_{jpk} = 1 allows tank j to have a zero holdup of p during k. A variable is saved for modeling the state of an empty tank. Thus, each tank must hold exactly one product in each slot.

$$\sum_{p=1}^{P} u_{jpk} = 1 \qquad (p, j) \in \boldsymbol{PJ},\ 0 < j \le J,\ 0 < k \le K \qquad (5.4)$$

Note that u_{jp0} must have an appropriate value based on what was inside tank j before time zero. To model product transitions in product tank j, a 0-1 continuous variable ue_{bk} is defined.

$$ue_{jk} = \begin{cases} 1 & \text{If tank } j \text{ switches products at the end of slot } k \\ 0 & \text{Otherwise} \end{cases} \qquad 0 < j \le J,\ 0 \le k < K$$

$$ue_{jk} \ge u_{jpk} - u_{jp(k+1)} \qquad (p, j) \in \boldsymbol{PJ},\ 0 \le k < K,\ 0 < j \le J \qquad (5.5a)$$

$$ue_{jk} \ge u_{jp(k+1)} - u_{jpk} \qquad (p, j) \in \boldsymbol{PJ},\ 0 \le k < K,\ 0 < j \le J \qquad (5.5b)$$

It is no need to force ue_{jk} = 0, as a penalty for product changeovers will be imposed in the objective. Now, a product transition cannot occur, unless the tank holdup (VP_{jk}, $0 \le VP_{jk} \le VP_j^U$) at the end of slot k is zero. Thus,

$$VP_{jk} \le VP_j^U (1 - ue_{jk}) \qquad 0 < k < K \qquad (5.6)$$

To model the blending operation, $\boldsymbol{BP}$ = {(b, p) | blender b can process product p}, and the following 0-1 continuous variables are defined.

$$x_{bpk} = \begin{cases} 1 & \text{If blender } b \text{ processes product } p \text{ during slot } k \\ 0 & \text{Otherwise} \end{cases} \qquad (b, p) \in \boldsymbol{BP},\ 0 \le k \le K$$

$$xe_{bk} = \begin{cases} 1 & \text{If blender } b \text{ ends the current product run during slot } k \\ 0 & \text{Otherwise} \end{cases} \qquad 0 \le k < K$$

Now x_{bpk}, u_{jpk}, and v_{bjk} are related to force x_{bpk} to be binary. First, if blender b is feeding a real tank j and that tank is holding product p during slot k, then b must be processing p in slot k.

$$x_{bpk} \ge u_{jpk} + v_{bjk} - 1 \qquad (b, p) \in \boldsymbol{BP},\ (b, j) \in \boldsymbol{BJ},\ (p, j) \in \boldsymbol{PJ},\ 0 < j \le J,\ 0 < k \le K \quad (5.7a)$$

Similarly, if b is processing p and feeding j during slot k, then j must hold p during k.

$$u_{jpk} \ge x_{bpk} + v_{bjk} - 1 \qquad (b, p) \in \boldsymbol{BP},\ (b, j) \in \boldsymbol{BJ},\ (p, j) \in \boldsymbol{PJ},\ 0 < j \le J,\ 0 < k \le K \quad (5.7b)$$

Note that eq. 5.7 are not written for p that j cannot store, i.e. (p, j) $\notin$ $\boldsymbol{PJ}$. For such products,

$$x_{bpk} + \sum_{j \notin PJ} v_{bjk} \le 1 \qquad (b, p) \in \boldsymbol{BP},\ (b, j) \in \boldsymbol{BJ},\ 0 < j \le J,\ 0 < k \le K \qquad (5.8a)$$

$$\sum_{p \notin PJ} x_{bpk} + v_{bjk} \le 1 \qquad (b, p) \in \boldsymbol{BP},\ (b, j) \in \boldsymbol{BJ},\ 0 < j \le J,\ 0 < k \le K \qquad (5.8b)$$

If a blender is not idle, then it must process exactly one product. In other words,

$$\sum_{p=1}^{P} x_{bpk} + v_{b0k} = 1 \qquad (b, p) \in \boldsymbol{BP},\ 0 < k \le K \qquad (5.9)$$

Eqs. 5.2, 5.4, 5.7 and 5.9 make x_{bpk} binary (Proof in Appendix B). Note that proper values must be assigned for x_{bp0} based on the product that blender b was processing before time zero.

Using x_{bpk}, xe_{bk} is computed as follows. If a blender processes the same product in two consecutive slots (k and k+1), then the current blend run cannot end at slot k, and vice versa. In other words,

$$xe_{bk} + x_{bpk} + x_{bp(k+1)} \le 2 \qquad (b, p) \in \boldsymbol{BP},\ 0 \le k < K \qquad (5.10)$$

On the contrary, if a product does not continue in the next slot, then its run must end at slot k.

$$xe_{bk} \ge x_{bpk} - x_{bp(k+1)} \qquad (b, p) \in \boldsymbol{BP},\ 0 \le k < K \qquad (5.11a)$$

$$xe_{bk} \ge x_{bp(k+1)} - x_{bpk} \qquad (b, p) \in \boldsymbol{BP},\ 0 \le k < K \qquad (5.11b)$$

Next, RL_{bk} is defined as the length of the current run of blender b at the end of slot k, if the run does not end during slot k, and zero otherwise. In other words,

$$RL_{bk} = \begin{cases} \text{Current run length} & \text{If the current run of blender } b \text{ does not end during slot } k \\ 0 & \text{Otherwise} \end{cases}$$

$$1 \le b \le B,\ 0 \le k \le K$$

Thus, $RL_{b0} = 0$, if a run has ended at time zero, otherwise it is the current run length at time zero. To compute RL_{bk}, it has,

$$RL_{bk} \le RL_{b(k-1)} + SL_k \qquad 0 < k \le K \qquad (5.12)$$

$$RL_{bk} \le H(1 - xe_{bk}) \qquad 0 \le k \le K \qquad (5.13)$$

Then, to ensure a minimum length (RL^L_{bp}) for each blend run, it demands,

$$RL_{b(k-1)} + SL_k + RL^L_b(1 - xe_{bk}) \ge \sum_{p=1}^{P} RL^L_{bp} x_{bpk} \qquad (b, p) \in \boldsymbol{BP},\ 0 < k \le K \qquad (5.14)$$

where, $RL_b^L = \max_p \left(RL_{bp}^L \right)$.

Blending requires components. Let q_{ibk} be the volume of component i used by blender b during slot k. Recall that G_{bjk} is the volume that blender b feeds product tank j during slot k. Then, the volume (Q_{bk}) processed in blender b during slot k is,

$$Q_{bk} = \sum_{i=1}^{I} q_{ibk} \qquad 0 < k \le K \qquad (5.15a)$$

$$Q_{bk} = \sum_{j=1}^{J} G_{bjk} \qquad (b, j) \in \boldsymbol{BJ},\ 0 < k \le K \qquad (5.15b)$$

If blender b is idle during slot k, then this volume must be zero.

$$Q_{bk} \le M_b \cdot (1 - v_{b0k}) \qquad 0 < k \le K \qquad (5.16)$$

where, M_b is the most volume that blender b can process during a slot. There are several ways of estimating M_b. One estimate is $H \cdot F_b^U$. Another is the maximum number in the maximum capacities of all product tanks because a blender can feed at most one product tank in a slot. The minimum possible estimate should be used as M_b. In other words, $M_b = \min \left\{ H \cdot F_b^U, \max_j \left(VP_j^U \right) \right\}$.

If blender b is not idle during slot k, then Q_{bk} must be limited by the maximum processing rate (F_b^U) of blender b.

$$Q_{bk} \le F_b^U \cdot SL_k \qquad 0 < k \le K \qquad (5.17a)$$

On the other hand, it must also respect the minimum processing rate (F_b^L) of blender b for each product, unless the current run is ending during slot k. In other words,

$$Q_{bk} + F_b^L \cdot H \cdot (v_{b0k} + xe_{bk}) \ge F_b^L \cdot SL_k \qquad 0 < k \le K \qquad (5.17b)$$

To compute the volume of product processed during a run up to slot k, CQ_{bk}

analogous to RL_{bk} is defined as follows.

$$CQ_{bk} = \begin{cases} \text{Volume processed} & \text{If blender } b \text{ does not end its run during slot } k \\ 0 & \text{Otherwise} \end{cases}$$

$$CQ_{bk} \le CQ_{b(k-1)} + Q_{bk} \qquad 0 < k \le K \qquad (5.18)$$

$$CQ_{bk} \le F_b^U \cdot H \cdot (1 - xe_{bk}) \qquad 0 \le k \le K \qquad (5.19)$$

$$CQ_{b(k-1)} + Q_{bk} + F_b^L \cdot RL_b^L \cdot (1 - xe_{bk}) \ge F_b^L \sum_{p=1}^{P} RL_{bp}^L x_{bpk} \qquad (b, p) \in \boldsymbol{BP},\ 0 < k \le K \qquad (5.20)$$

Note that eq. 5.17 allows the blending rate to vary from slot to slot during a run. Normally, this is not done in practice. However, enforcing this makes the formulation nonlinear and nonconvex. Therefore, this issue is dealt with later.

Clearly, each blend run must ensure product quality. Several gasoline properties such as Octane Number (ON), Reid Vapor Pressure (RVP), Specific Gravity (SG), Sulfur (S), Benzene (B), Aromatics (A), Olefin (O), Viscosity (Visc), Flash Point (FLP), Freeze Point (FZ), Residue (R), Flammability Limit (FL), Oxygenates (OX), etc. are used in practice. Many of these properties (e.g. viscosity) involve highly nonlinear mixing rules. However, as noted by Li et al. (2007), a linear blending index usually exists and is used for almost every hydrocarbon property with nonlinear mixing correlations. These blending indices are linearly additive on either volume or weight basis. Table 5.1 lists the twelve most commonly used gasoline indices and their additive bases. Let θ_{is} be the known blending index for a property s of component i, ρ_i be the density of component i, $[\theta_{ps}^L, \theta_{ps}^U]$ be the desired limits on property s of product p, and ρ_{max} be the maximum possible density among all products. Then, the following ensure the desired product quality.

Table 5.1 Gasoline properties, corresponding indices, and correlations

Gasoline Property	Blending Index	Addition Basis	Index Correlation
Research Octane Number (RON)	RBN	Volume	RON + 11.5 $(0 \le RON \le 85)$ Exp(0.0135× RON + 3.422042) (RON > 85)
Reid Vapor Pressure (RVP Bar)	RVI	Volume	Exp(1.14×Log(100×RVP))
Specific Gravity (SG)	DNI	Volume	1/SG
Sulfur (S ppm)	SULI	Weight	Weighted average
Benzene (B)	BI	Volume	Volumetric average
Aromatics (A)	AROI	Volume	Volumetric average
Olefin (O)	OI	Volume	Volumetric average
Viscosity @ 50 °C (Visc, cst)	VI	Weight	79.1+33.47×(Log(Log(Visc_cst+0.8)/Log(10))/Log(10))
Flash Point (FLP °C)	FPI	Volume	Exp((−6.1184+(2414/(FLP+230.56)))×Log(10))
Freeze Point (FZ °C)	FRI	Volume	3162000×Exp(12.5×Log(0.001(1.8×Freeze Point+491.67))
Flammability Limit (FL)	FLI	Volume	Volumetric average
Oxygenates (OX)	OXI	Weight	Weighted average

Note: Index correlations from Singapore Petroleum Company (SPC)

$$\sum_{i=1}^{I} q_{ibk}\theta_{is} \geq Q_{bk}\theta_{ps}^{L} - M_b \cdot \left\{\theta_{ps}^{L} - \min_p\left(\theta_{ps}^{L}\right)\right\} \cdot (1 - x_{bpk}) \qquad (b, p) \in \boldsymbol{BP},\ 0 < k \leq K \qquad (5.21a)$$

$$\sum_{i=1}^{I} q_{ibk}\theta_{is} \leq Q_{bk}\theta_{ps}^{U} + M_b \cdot \left\{\max_p\left(\theta_{ps}^{U}\right) - \theta_{ps}^{U}\right\} \cdot (1 - x_{bpk}) \qquad (b, p) \in \boldsymbol{BP},\ 0 < k \leq K \qquad (5.21b)$$

$$\sum_{i=1}^{I} q_{ibk}\rho_i\theta_{is} \geq \left(\sum_{i=1}^{I} q_{ibk}\rho_i\right)\theta_{ps}^{L} - M_b \cdot \left\{\theta_{ps}^{L} - \min_p\left(\theta_{ps}^{L}\right)\right\} \cdot \rho_{\max} \cdot (1 - x_{bpk})$$

$$(b, p) \in \boldsymbol{BP},\ 0 < k \leq K \qquad (5.22a)$$

$$\sum_{i=1}^{I} q_{ibk}\rho_i\theta_{is} \leq \left(\sum_{i=1}^{I} q_{ibk}\rho_i\right)\theta_{ps}^{U} + M_b \cdot \left\{\max_p\left(\theta_{ps}^{U}\right) - \theta_{ps}^{U}\right\} \cdot \rho_{\max} \cdot (1 - x_{bpk})$$

$$(b, p) \in \boldsymbol{BP},\ 0 < k \leq K \qquad (5.22b)$$

where, eq. 5.21 is for volume-based indices, and eq. 5.22 for weight-based indices.

Often, a practitioner may impose limits $[r_{pi}^{L}, r_{pi}^{U}]$ on the volume fraction of component *i* in product *p*. In such a case, it uses,

$$Q_{bk}r_{pi}^{L} - M_b \cdot \left\{r_{pi}^{L} - \min_p\left(r_{pi}^{L}\right)\right\}(1 - x_{bpk}) \leq q_{ibk} \leq Q_{bk}r_{pi}^{U} + M_b \cdot \left\{\max_p\left(r_{pi}^{U}\right) - r_{pi}^{U}\right\}(1 - x_{bpk})$$

$$(b, p) \in \boldsymbol{BP},\ 0 < k \leq K \qquad (5.23a,b)$$

5.3.2 Order Delivery

JO = {(*j*, *o*) | product tank *j* can deliver order *o*} and one binary variable (z_{jok}) are defined as follows,

$$z_{jok} = \begin{cases} 1 & \text{If product tank } j \text{ is delivering order } o \text{ during slot } k \\ 0 & \text{Otherwise} \end{cases}$$

$$(j, o) \in \boldsymbol{JO},\ 0 < j \leq J,\ 0 < k \leq K$$

Since each order must be filled during the scheduling horizon, there must be at least one delivery for each order.

$$\sum_{j=1}^{J}\sum_{k=1}^{K} z_{jok} \geq 1 \qquad (j, o) \in \boldsymbol{JO} \qquad (5.24)$$

A tank j cannot receive and deliver products simultaneously:

$$v_{bjk} + z_{jok} \leq 1 \qquad (b, j) \in \boldsymbol{BJ}, (j, o) \in \boldsymbol{JO}, 0 < j \leq J, 0 < k \leq K \qquad (5.25)$$

If tank j is delivering order o during slot k, then it must be holding the product corresponding to that order in both slots (k–1) and k. This is true, because a tank cannot receive and deliver at the same time, so it must be holding the product in (k–1), before it delivers in slot k.

$$z_{jok} \leq u_{jpk} \qquad (p, j) \in \boldsymbol{PJ}, (j, o) \in \boldsymbol{JO}, (o, p) \in \boldsymbol{OP}, 0 < j \leq J, 0 < k \leq K \qquad (5.26a)$$

$$z_{jok} \leq u_{jp(k-1)} \qquad (p, j) \in \boldsymbol{PJ}, (j, o) \in \boldsymbol{JO}, (o, p) \in \boldsymbol{OP}, 0 < j \leq J, 0 < k \leq K \qquad (5.26b)$$

where, $\boldsymbol{OP}$ = {(o, p) | order o is for product p}. Recall that each order has a single product.

If a product tank j switches products at the end of a slot k, then its holdup must be zero. Thus, it cannot deliver any order in (k + 1).

$$z_{jo(k+1)} + ue_{jk} \leq 1 \qquad (j, o) \in \boldsymbol{JO}, 0 < j \leq J, 0 \leq k < K \qquad (5.27)$$

Let DQ_{jok} be the volume of order o delivered by tank j during slot k. If tank j is not delivering o during k, then the delivery amount must be zero:

$$DQ_{jok} \leq TQ_o \cdot z_{jok} \qquad (j, o) \in \boldsymbol{JO}, 0 < j \leq J, 0 < k \leq K \qquad (5.28)$$

where, TQ_o is the required amount for order o. In general, depending on its pump/valve infrastructure, each tank will have some limits on the delivery rates of orders. Let assume that the maximum possible delivery rate for an order o is DR_{jo} and the maximum cumulative rate for all orders is DR_j^U. Then,

$$DQ_{jok} \leq DR_{jo}SL_k \qquad (j, o) \in \boldsymbol{JO}, 0 < j \leq J, 0 < k \leq K \qquad (5.29a)$$

$$\sum_{o\in \boldsymbol{JO}} DQ_{jok} \le DR_j^U SL_k \qquad 0 < j \le J,\ 0 < k \le K \qquad (5.29b)$$

To ensure full delivery for each order,

$$\sum_{j=1}^{J}\sum_{k=1}^{K} DQ_{jok} = TQ_o \qquad (j, o) \in \boldsymbol{JO} \qquad (5.30)$$

Recall that each order o has a delivery window (DD_o^L, DD_o^U). A tank j cannot begin delivering o before DD_o^L:

$$T_{k-1} \ge DD_o^L z_{jok} \qquad (j, o) \in \boldsymbol{JO},\ 0 < j \le J,\ 0 < k \le K \qquad (5.31)$$

If an order is not fully delivered before DD_o^U, then a delivery delay is given by,

$$d_o \ge T_{k-1} + \frac{DQ_{jok}}{DR_{jo}} - DD_o^U - H(1 - z_{jok}) \qquad (j, o) \in \boldsymbol{JO},\ 0 < j \le J,\ 0 < k \le K \qquad (5.32)$$

Recall that all orders are completed within the scheduling horizon. Therefore, the delivery delay (d_o) of order o must have the upper bound of ($H - DD_o^U$).

Note that order delivery was allowed to be intermittent from a tank. As it was mentioned for blending, a simple adjustment procedure is proposed later to correct this situation.

5.3.3 Inventory Balance

Let V_{ik} ($V_i^L \le V_{ik} \le V_i^U$) denote the inventory of component i at the end of slot k and V_i ($V_i^L \le V_i \le V_i^U$) denote the inventory of component i at the end of scheduling horizon. Within any period, the feed rate (F_i) of i from upstream units is constant, so

$$V_{ik} = V_{i(k-1)} + F_i SL_k - \sum_{b=1}^{B} q_{ibk} \qquad 0 < k \le K \qquad (5.33a)$$

$$V_i = V_{iK} + F_i (H - T_K) \qquad (5.33b)$$

Similarly, for product tank j, it has:

$$VP_{jk} = VP_{j(k-1)} + \sum_{b=1}^{B} G_{bjk} - \sum_{o=1}^{O} DQ_{jok} \qquad (j, o) \in \boldsymbol{JO},\ 0 < k \le K \qquad (5.34)$$

5.3.4 Transitions in Blenders

At the beginning, the amount of an order and its product are known. Then, the total amount of each product needed can be calculated. Having the initial amount of each product, then it is known whether the product is needed to process in the blender or not. Let N denote the number of products that are needed to process in blenders during the scheduling horizon and N_b as the number of blenders. Recall that xe_{bk} is used as the product transition in blender b at the end of slot k and xe_{bK} is equal to one. The total minimum transitions during the scheduling horizon are calculated as follows,

$$\sum_{b=1}^{B}\sum_{k=1}^{K-1} xe_{bk} \ge N - N_b \qquad (5.35)$$

5.3.5 Objective Function

For similar problems, Mendez et al. (2006) maximized total profit, while Jia and Ierapetritou (2003) minimized the makespan. Neither considered the transition costs for blenders and product tanks. For a given set of orders, it feels that minimizing the total operating cost including material (component), demurrage, transition, and backorder costs is more meaningful in practice. Assuming that the transition costs are product and sequence-independent (this is reasonable as product qualities are quite similar), the scheduling objective is:

$$\text{Minimize TC} = \sum_{i=1}^{I}\sum_{b=1}^{B}\sum_{k=1}^{K} c_i \cdot q_{ibk} + \sum_{b=1}^{B}\sum_{k=1}^{K-1} CB_b \cdot xe_{bk} + \sum_{j=1}^{J}\sum_{k=1}^{K-1} CT_j \cdot ue_{jk} + \sum_{o=1}^{O} DM_o \cdot d_o \qquad (5.36)$$

where, c_i is the price ($ per unit volume) of component i, CB_b is the cost ($ per occurrence) of transition on blender b, CT_j is the cost ($ per occurrence) of transition in product tank j, and DM_o is the demurrage cost ($ per unit time) of order o.

This completes proposed single-period model (**SPM**) for scheduling blending operations, which comprises eqs. (5.1-5.36). As mentioned before, it allows the blending rate to vary from slot to slot and order delivery to be discontinuous, which is undesirable in practice. Therefore, a procedure is needed to adjust the solution from **SPM** to obtain a realistic schedule.

5.4 Schedule Adjustment

The optimal solution from **SPM** gives the values of x_{bpk}, xe_{bk}, v_{bjk}, SL_k, RL_{bk}, Q_{bk}, and CQ_{bk}. Note that RL_{bk} and CQ_{bk} respectively may not be the correct length and volume of product processed by blender b when a run is in progress. Therefore, the correct values for run lengths (CRL_{bk}) and volumes (CCQ_{bk}) are computed as follows.

$$CRL_{bk} = CCQ_{bk} = 0 \qquad \text{if } xe_{bk} = 1 \qquad (5.37\text{a,b})$$

$$CRL_{bk} = CRL_{b(k-1)} + SL_k \qquad \text{if } xe_{bk} = 0 \qquad (5.38\text{a})$$

$$CCQ_{bk} = CCQ_{b(k-1)} + Q_{bk} \qquad \text{if } xe_{bk} = 0 \qquad (5.38\text{b})$$

Then, the total volume (TCQ_{bk}) processed by a blender in a run is computed as:

$$TCQ_{bk} = 0 \qquad \text{if } xe_{bk} = 0 \qquad (5.39\text{a})$$

$$TCQ_{bk} = CCQ_{b(k-1)} + Q_{bk} \qquad \text{if } xe_{bk} = 1 \qquad (5.39\text{b})$$

Using the above parameters, the blending rate (R_{bk}) for each blending run at the slot where it ends is computed as follows,

$$R_{bk} = \max\left(F_b^L,\ \frac{CCQ_{b(k-1)} + Q_{bk}}{CRL_{b(k-1)} + SL_k}\right) \text{ for } k \text{ with } xe_{bk} = 1 \,\&\, v_{b0k} = 0 \qquad (5.40)$$

Then, R_{bk} is set for all slots within each run to be the same as the one computed above. Thus, if a run spans slots 3-6 inclusive, then $R_{bk} = R_{b6}$ is set for k = 3-5.

Now, to obtain a realistic schedule with the constant blend rates computed above, x_{bpk}, xe_{bk}, and v_{b0k} are fixed. This allows to fix, remove, or change some variables and constraints in SPM. To this end, $[x_{bpk}]$, $[xe_{bk}]$, and $[v_{b0k}]$ are used respectively to denote the optimal values of x_{bpk}, xe_{bk}, and v_{b0k} obtained from SPM.

Eqs. 5.16-5.20 become:

$$Q_{bk} = 0 \quad \text{for } (b, k) \text{ with } [v_{b0k}] = 1 \qquad (5.41a)$$

$$Q_{bk} = R_{bk} \cdot SL_k \quad \text{for } (b, k) \text{ with } [xe_{bk}] = [v_{b0k}] = 0 \qquad (5.41b)$$

$$Q_{bk} \le R_{bk} \cdot SL_k \quad \text{for } (b, k) \text{ with } [xe_{bk}] = 1 \,\&\, [v_{b0k}] = 0 \qquad (5.41c)$$

$$CQ_{bk} = 0 \quad \text{for } (b, k) \text{ with } [xe_{bk}] = 1 \qquad (5.42a)$$

$$CQ_{bk} = CQ_{b(k-1)} + Q_{bk} \quad \text{for } (b, k) \text{ with } [xe_{bk}] = [v_{b0k}] = 0 \qquad (5.42b)$$

$$CQ_{b(k-1)} + Q_{bk} \ge TCQ_{bk} \quad \text{for } (b, k) \text{ with } [xe_{bk}] = 1 \,\&\, [v_{b0k}] = 0 \qquad (5.42c)$$

Fixing the values of x_{bpk}, xe_{bk}, and v_{b0k}, eqs. 5.2, 5.7-5.8, 5.21-5.23, and 5.36 become:

$$\sum_{j=1}^{J} v_{bjk} = 1 - [v_{b0k}] \qquad (b, j) \in \boldsymbol{BJ},\ 0 < j \le J,\ 0 < k \le K \qquad (5.43)$$

$$[x_{bpk}] \ge u_{jpk} + v_{bjk} - 1$$

$$(b, p) \in \boldsymbol{BP},\ (b, j) \in \boldsymbol{BJ},\ (p, j) \in \boldsymbol{PJ},\ 0 < j \le J,\ 0 < k \le K \qquad (5.44a)$$

$$u_{jpk} \ge [x_{bpk}] + v_{bjk} - 1$$

$$(b, p) \in \boldsymbol{BP},\ (b, j) \in \boldsymbol{BJ},\ (p, j) \in \boldsymbol{PJ},\ 0 < j \le J,\ 0 < k \le K \qquad (5.44b)$$

$$[x_{bpk}] + \sum_{j \notin PJ} v_{bjk} \le 1 \qquad (b, p) \in \boldsymbol{BP},\ (b, j) \in \boldsymbol{BJ},\ 0 < j \le J,\ 0 < k \le K \qquad (5.45a)$$

$$\sum_{p \notin PJ} [x_{bpk}] + v_{bjk} \le 1 \qquad (b, p) \in \boldsymbol{BP}, (b, j) \in \boldsymbol{BJ}, 0 < j \le J, 0 < k \le K \tag{5.45b}$$

$$\sum_{i=1}^{I} q_{ibk}\theta_{is} \ge Q_{bk}\theta_{ps}^{L} - M_b \cdot \left\{\theta_{ps}^{L} - \min_p\left(\theta_{ps}^{L}\right)\right\} \cdot (1-[x_{bpk}])$$

for (b, k) with $[v_{b0k}] = 0$, $(b, p) \in \boldsymbol{BP}$, $0 < k \le K$ (5.46a)

$$\sum_{i=1}^{I} q_{ibk}\theta_{is} \le Q_{bk}\theta_{ps}^{U} + M_b \cdot \left\{\max_p\left(\theta_{ps}^{U}\right) - \theta_{ps}^{U}\right\} \cdot (1-[x_{bpk}])$$

for (b, k) with $[v_{b0k}] = 0$, $(b, p) \in \boldsymbol{BP}$, $0 < k \le K$ (5.46b)

$$\sum_{i=1}^{I} q_{ibk}\rho_i\theta_{is} \ge \left(\sum_{i=1}^{I} q_{ibk}\rho_i\right)\theta_{ps}^{L} - M_b \cdot \left\{\theta_{ps}^{L} - \min_p\left(\theta_{ps}^{L}\right)\right\} \cdot \rho_{\max} \cdot (1-[x_{bpk}])$$

for (b, k) with $[v_{b0k}] = 0$, $(b, p) \in \boldsymbol{BP}$, $0 < k \le K$ (5.47a)

$$\sum_{i=1}^{I} q_{ibk}\rho_i\theta_{is} \le \left(\sum_{i=1}^{I} q_{ibk}\rho_i\right)\theta_{ps}^{U} + M_b \cdot \left\{\max_p\left(\theta_{ps}^{U}\right) - \theta_{ps}^{U}\right\} \cdot \rho_{\max} \cdot (1-[x_{bpk}])$$

for (b, k) with $[v_{b0k}] = 0$, $(b, p) \in \boldsymbol{BP}$, $0 < k \le K$ (5.47b)

$$Q_{bk}r_{pi}^{L} - M_b \cdot \left\{r_{pi}^{L} - \min_p\left(r_{pi}^{L}\right)\right\}(1-[x_{bpk}]) \le q_{ibk} \le Q_{bk}r_{pi}^{U} + M_b \cdot \left\{\max_p\left(r_{pi}^{U}\right) - r_{pi}^{U}\right\}(1-[x_{bpk}])$$

for (b, k) with $[v_{b0k}] = 0$, $(b, p) \in \boldsymbol{BP}$, $0 < k \le K$ (5.48a,b)

$$\mathrm{TC} = \sum_{i=1}^{I}\sum_{b=1}^{B}\sum_{k=1}^{K} c_i \cdot q_{ibk} + \sum_{b=1}^{B}\sum_{k=1}^{K-1} CB_b \cdot [xe_{bk}] + \sum_{j=1}^{J}\sum_{k=1}^{K-1} CT_j \cdot ue_{jk} + \sum_{o=1}^{O} DM_o \cdot d_o \tag{5.49}$$

The revised model (RSPM) comprises eqs. 5.1, 5.3-5.6, 5.15, 5.24-5.34, and 5.41-5.49, whose solution ensures that blending campaigns have constant blend rates that are within the limits on the blending rates and minimum run lengths at the same time. Appendix C gives the proof.

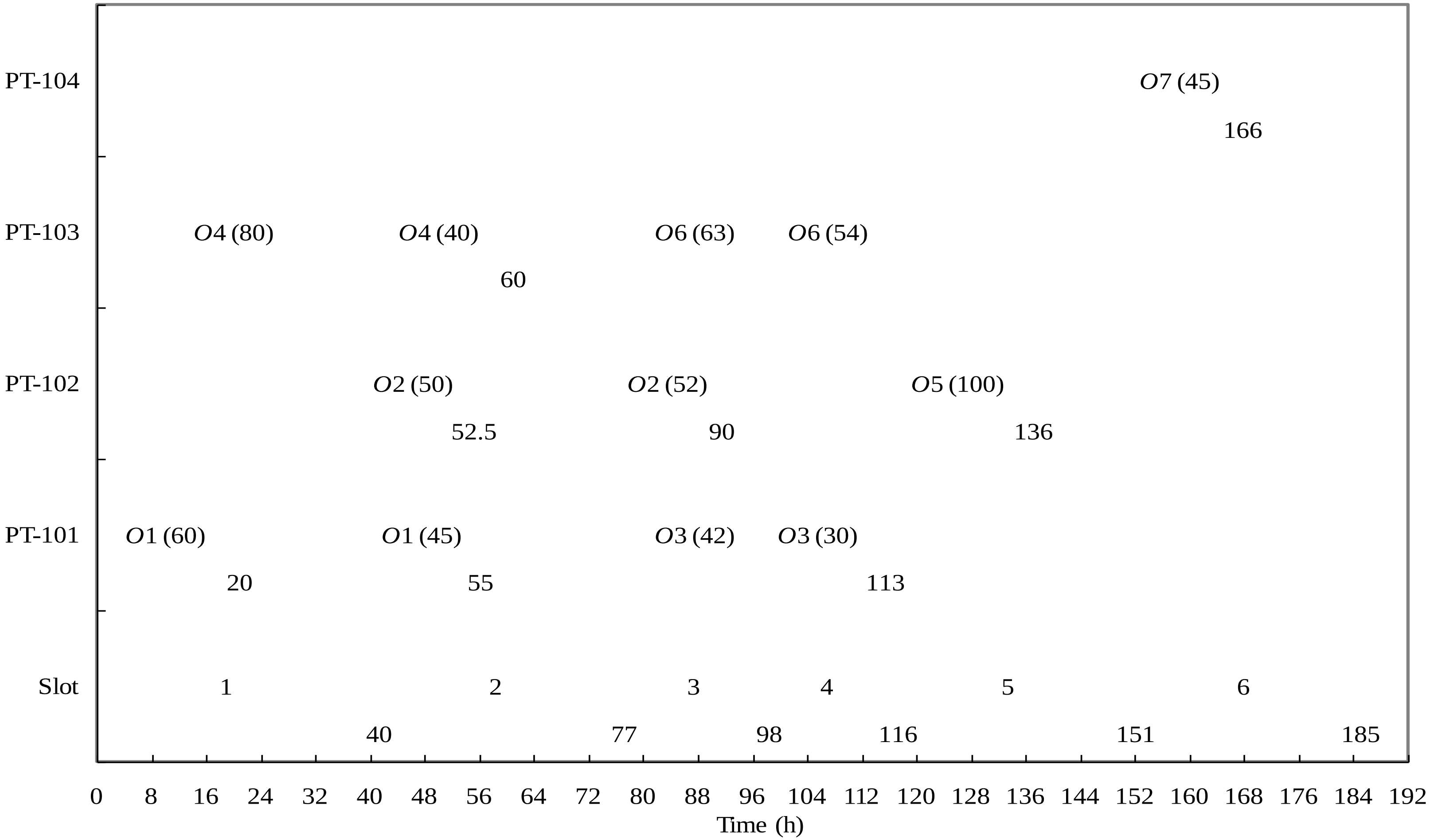

Figure 5.3 An example schedule to illustrate intermittent delivery of orders O1 and O2 by PT-101 and PT-120 respectively

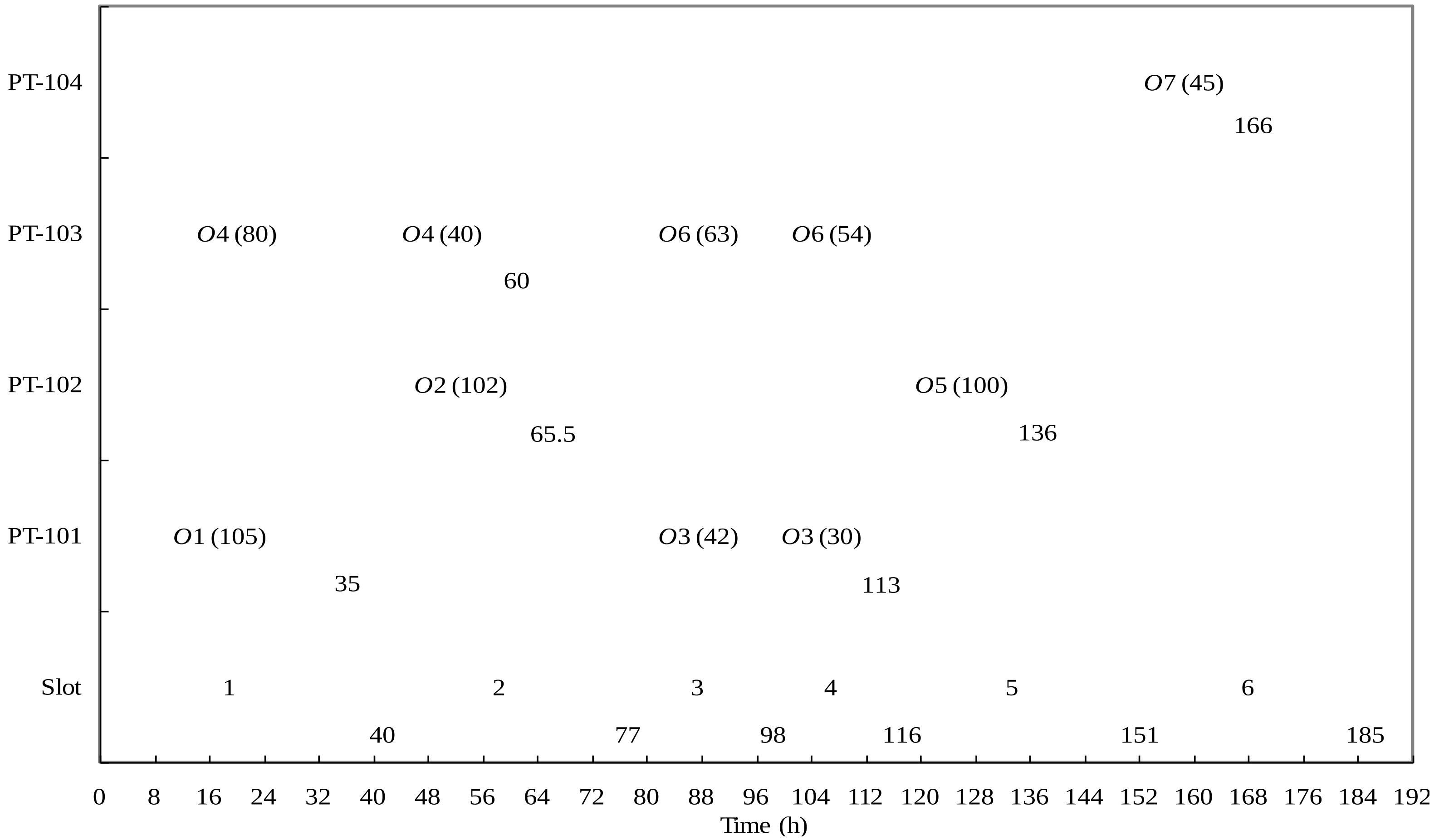

Figure 5.4 The schedule of Figure 3 revised by the proposed algorithm where PT-101 and PT-102 deliver O1 and O2 continuously

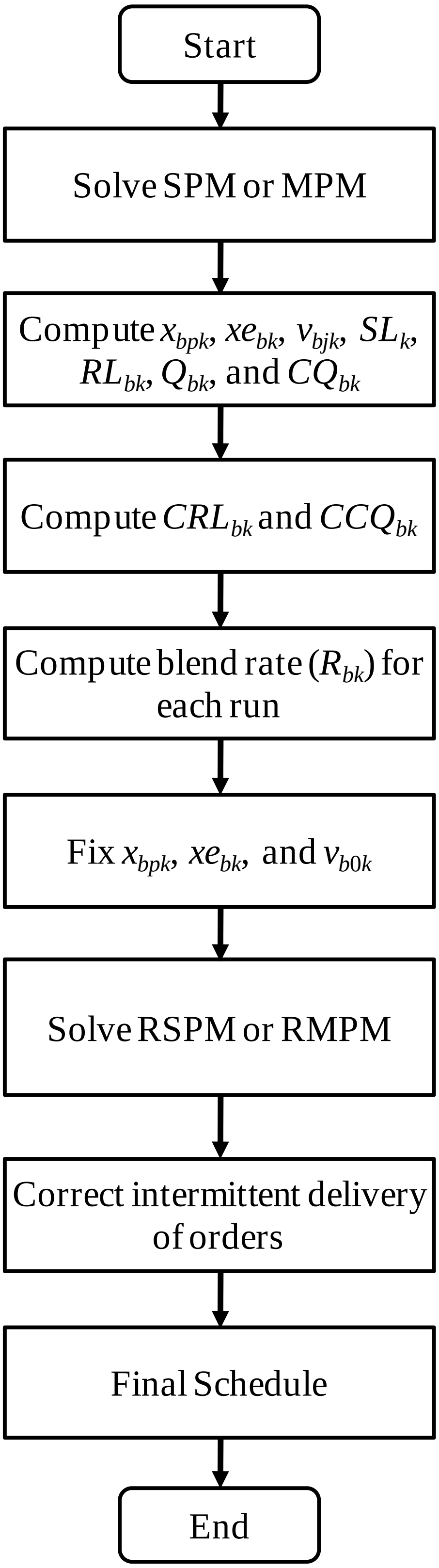

Figure 5.5 Flowchart for the schedule adjustment procedure

The schedule from RSPM may still show intermittent delivery of orders (Figure 5.3). In Figure 5.3, product tanks PT-101 and PT-102 deliver orders *O*1 and *O*2 intermittently. When the delivery is over contiguous slots, then this can be easily revised by simply delivering at a constant rate until the entire order, which is distributed over contiguous slots, is fully delivered. Figure 5.4 shows such a revised schedule for Figure 5.3, where deliveries of *O*1 and *O*2 are uninterrupted. Figure 5.5 shows the complete algorithm for the adjustment procedure.

The models and procedures discussed so far were for a single period with constant feed rates to component tanks. The extension of this to a multi-period scenario, where the entire horizon can be divided into multiple periods of constant feed rates, is straightforward as it can be seen next.

5.5 Multi-Period Formulation (MPM)

Given the rate profiles of feeds into component tanks, the entire scheduling horizon is divided into T periods (t = 1, 2, …, T) of lengths H_t such that the flow rates of components are constant within each period and $H = H_1 + H_2 + \ldots + H_T$. Let F_{it} be the flow rate of component i in period t. The approach used by Karimi and McDonald (1997) in their second model (M2) is followed. Thus, each period is divided into several slots of unknown lengths. Let $\boldsymbol{TK} = \{(t, k) \mid \text{slot } k \text{ is in period } t\}$. For this model, some T_k are first fixed to be the period ends. For instance, if periods 1, 2, and 3 have three slots each, then $T_3 = H_1$, $T_6 = H_1 + H_2$, and $T_9 = H_1 + H_2 + H_3$ with the upper bound of T_K being H. Clearly, eq. 5.1 is also effective for this model (MPM), while eq.

5.33 becomes,

$$V_{ik} = V_{i(k-1)} + \sum_{t \in TK} F_{it} SL_k - \sum_{b=1}^{B} q_{ibk} \qquad 0 < k \le K \tag{5.50a}$$

$$V_i = V_{iK} + F_{iT}(H - T_K) \tag{5.50b}$$

In addition, component properties may vary from period to period. For instance, refinery may often generate slops of lower quality, which can be used in blending. The component properties are assumed to be known and constant during each period, but may vary with periods. Let θ_{ist} denote the known blending index for a property *s* of component *i* during period *t*, ρ_{it} be the density of component *i* during period *t*, and $\rho_t^{\max}$ be the maximum possible density among all products during period *t*. Then, eqs. 5.21-5.22 change to:

$$\sum_{i=1}^{I} q_{ibk}\theta_{ist} \ge Q_{bk}\theta_{ps}^{L} - M_b \cdot \left\{\theta_{ps}^{L} - \min_p\left(\theta_{ps}^{L}\right)\right\} \cdot (1 - x_{bpk})$$

$$(b, p) \in \boldsymbol{BP},\ (t, k) \in \boldsymbol{TK},\ 0 < k \le K \tag{5.51a}$$

$$\sum_{i=1}^{I} q_{ibk}\theta_{ist} \le Q_{bk}\theta_{ps}^{U} + M_b \cdot \left\{\max_p\left(\theta_{ps}^{U}\right) - \theta_{ps}^{U}\right\} \cdot (1 - x_{bpk})$$

$$(b, p) \in \boldsymbol{BP},\ (t, k) \in \boldsymbol{TK},\ 0 < k \le K \tag{5.51b}$$

$$\sum_{i=1}^{I} q_{ibk}\rho_{it}\theta_{ist} \ge \left(\sum_{i=1}^{I} q_{ibk}\rho_{it}\right)\theta_{ps}^{L} - M_b \cdot \left\{\theta_{ps}^{L} - \min_p\left(\theta_{ps}^{L}\right)\right\} \cdot \rho_t^{\max} \cdot (1 - x_{bpk})$$

$$(b, p) \in \boldsymbol{BP},\ (t, k) \in \boldsymbol{TK},\ 0 < k \le K \tag{5.52a}$$

$$\sum_{i=1}^{I} q_{ibk}\rho_{it}\theta_{ist} \le \left(\sum_{i=1}^{I} q_{ibk}\rho_{it}\right)\theta_{ps}^{U} + M_b \cdot \left\{\max_p\left(\theta_{ps}^{U}\right) - \theta_{ps}^{U}\right\} \cdot \rho_t^{\max} \cdot (1 - x_{bpk})$$

$$(b, p) \in \boldsymbol{BP},\ (t, k) \in \boldsymbol{TK},\ 0 < k \le K \tag{5.52b}$$

Thus, MPM comprises eqs. 5.1-5.20, 5.23-5.32, 5.34-5.36, and 5.50-5.52.

For RMPM, eqs. 5.46-5.47 become:

$$\sum_{i=1}^{I} q_{ibk}\theta_{ist} \geq Q_{bk}\theta_{ps}^{L} - M_b \cdot \left\{\theta_{ps}^{L} - \min_p\left(\theta_{ps}^{L}\right)\right\} \cdot (1-[x_{bpk}])$$

for (b, k) with $[v_{b0k}] = 0$, $(b, p) \in \boldsymbol{BP}$, $(t, k) \in \boldsymbol{TK}$, $0 < k \leq K$ (5.53a)

$$\sum_{i=1}^{I} q_{ibk}\theta_{ist} \leq Q_{bk}\theta_{ps}^{U} + M_b \cdot \left\{\max_p\left(\theta_{ps}^{U}\right) - \theta_{ps}^{U}\right\} \cdot (1-[x_{bpk}])$$

for (b, k) with $[v_{b0k}] = 0$, $(b, p) \in \boldsymbol{BP}$, $(t, k) \in \boldsymbol{TK}$, $0 < k \leq K$ (5.53b)

$$\sum_{i=1}^{I} q_{ibk}\rho_{it}\theta_{ist} \geq \left(\sum_{i=1}^{I} q_{ibk}\rho_{it}\right)\theta_{ps}^{L} - M_b \cdot \left\{\theta_{ps}^{L} - \min_p\left(\theta_{ps}^{L}\right)\right\} \cdot \rho_t^{\max} \cdot (1-[x_{bpk}])$$

for (b, k) with $[v_{b0k}] = 0$, $(b, p) \in \boldsymbol{BP}$, $(t, k) \in \boldsymbol{TK}$, $0 < k \leq K$ (5.54a)

$$\sum_{i=1}^{I} q_{ibk}\rho_{it}\theta_{ist} \leq \left(\sum_{i=1}^{I} q_{ibk}\rho_{it}\right)\theta_{ps}^{U} + M_b \cdot \left\{\max_p\left(\theta_{ps}^{U}\right) - \theta_{ps}^{U}\right\} \cdot \rho_t^{\max} \cdot (1-[x_{bpk}])$$

for (b, k) with $[v_{b0k}] = 0$, $(b, p) \in \boldsymbol{BP}$, $(t, k) \in \boldsymbol{TK}$, $0 < k \leq K$ (5.54b)

Then RMPM comprises eqs. 5.1, 5.3-5.6, 5.15, 5.24-5.32, 5.34, 5.41-5.45, 5.48-5.50 and 5.53-5.54.

Next, the proposed solution approach is illustrated in detail using a very small example (Example 1) and the solution approach is evaluated using additional thirteen larger examples.

5.6 Example 1

This example involves five orders (O1-O5), three grades (products P1-P3), nine component tanks (CT-101 to CT-109), one blender, five product tanks (PT-101 to PT-105), and one product property (Octane number). Components from atmospheric distillation, FCCU (Fluid Catalytic Cracking Unit), FCRU (Fluid Catalytic Reforming Unit), AKU (Alkanisation Unit), IFU (Isoforming Unit), CHU (Catalytic

Hydrogenation Unit), ARU (Aromatization Unit), and various additives such as MTBE (Methyl Tert-Butyl Ether) and Butane are stored in the nine component tanks. The blender's operating range is [2, 15 kbbl/h] with a minimum run length of 6 h for each product. At time zero, the blender is idle. PT-101, PT-102, and PT-103 can store P2 and P3, whereas PT-104 and PT-105 can store P1 and P2. At time zero, PT-101 holds 90.20 kbbl of P3, PT-102 is empty, PT-103 holds 14.08 kbbl of P2, PT-104 holds 28.49 kbbl of P2, and PT-105 holds 20.20 kbbl of P1. Tables 5.2-5.7 list other data. The scheduling horizon is 72 hours with constant flows into component tanks over that period, so the developed single-period model is appropriate. It is solved on a Dell precision PWS690 (Intel® Xeon^R 5160 with CPU 3 GHZ and 16 GB memory) running Windows XP using CPLEX 10.0.1/GAMS 22.2.

The optimal solution (Figure 5.6) has a cost of \$ 5149.73K. The model needed 4 slots and 0.77 CPU s. P3 is not processed at all, because its initial inventory is sufficient to satisfy all orders. The blender has two runs of 24 h and 34.78 h durations. The first run spans slot 1 and processes 23.43 kbbl of P2. The second run spans slots 2-4, and processes 22.05 kbbl, 93.75 kbbl, and 0.00 kbbl of P1 respectively. Thus, it processes 23.43 kbbl of P2 in run 1, and 115.80 kbbl of P1 in run 2. These give the following blending rates.

R_{11} = max [23.43/24, 2] = 2.0 kbbl/h

R_{12} = R_{13} = R_{14} = max[115.80/34.78, 2] = 3.329 kbbl/h

Table 5.2a Order data for Examples 1-9

		Amount (kbbl)							Delivery Rate (kbbl/h)							Delivery Window						
																Example						
o	p	1	2	3	4-5	6	7-8	9	1	2	3	4-5	6	7-8	9	1	2	3	4-5	6	7-8	9
O1	P1	11	10	10	10	10	10	10	3	5	5	5	5	5	5	[0,12]	[0,24]	[0,24]	[0,24]	[0,24]	[0,24]	[0,24]
O2	P2	3	3	3	3	3	3	3	3	3	3	3	3	3	3	[0,12]	[0,24]	[0,24]	[0,24]	[0,24]	[0,24]	[0,24]
O3	P2	60	3	3	3	3	3	3	5	3	3	3	3	3	3	[24,48]	[24,50]	[0,24]	[0,24]	[0,24]	[0,24]	[0,24]
O4	P1	125	10	10	10	10	10	10	5	5	5	5	5	5	5	[24,72]	[24,50]	[0,24]	[0,24]	[0,24]	[0,24]	[0,24]
O5	P2	3	3	3	3	3	3	3	3	3	3	3	3	3	3	[24,48]	[48,72]	[24,48]	[24,48]	[24,48]	[24,48]	[24,48]
O6	P1	-	10	10	10	10	10	10	-	5	5	5	5	5	5	-	[48,72]	[24,48]	[24,48]	[24,48]	[24,48]	[24,48]
O7	P2	-	3	3	3	3	3	3	-	3	3	3	3	3	3	-	[48,120]	[24,48]	[24,48]	[24,48]	[24,48]	[24,48]
O8	P1	-	120	100	100	90	100	100	-	5	5	5	5	5	5	-	[118,190]	[118,190]	[118,190]	[118,190]	[118,190]	[118,190]
O9	P2	-	3	3	3	3	3	3	-	3	3	3	3	3	3	-	[144,168]	[144,168]	[144,168]	[144,168]	[144,168]	[144,168]
O10	P4	-	150	150	150	150	150	150	-	5	5	5	5	5	5	-	[150.5,185.5]	[150.5,185.5]	[150.5,185.5]	[150.5,185.5]	[150.5,185.5]	[150.5,185.5]
O11	P3	-	-	20	20	45	60	60	-	-	5	5	5	5	5	-	-	[144,168]	[144,168]	[144,168]	[144,168]	[144,168]
O12	P2	-	-	30	30	30	20	20	-	-	5	5	5	5	5	-	-	[24,48]	[24,48]	[24,48]	[24,48]	[24,48]
O13	P4	-	-	-	60	45	60	60	-	-	-	5	5	5	5	-	-	-	[0,56]	[0,56]	[0,56]	[0,56]
O14	P3	-	-	-	10	15	15	20	-	-	-	5	5	5	5	-	-	-	[48,72]	[48,72]	[48,72]	[48,72]
O15	P2	-	-	-	20	15	20	20	-	-	-	4	4	4	4	-	-	-	[0,72]	[0,75]	[0,72]	[0,72]
O16	P2	-	-	-	-	10	20	20	-	-	-	-	4	5	5	-	-	-	-	[48,72]	[48,72]	[48,72]
O17	P1	-	-	-	-	10	10	10	-	-	-	-	5	5	5	-	-	-	-	[48,96]	[48,72]	[48,72]
O18	P1	-	-	-	-	10	10	10	-	-	-	-	5	5	5	-	-	-	-	[48,96]	[48,72]	[48,72]
O19	P2	-	-	-	-	-	60	60	-	-	-	-	-	5	5	-	-	-	-	-	[0,50]	[0,50]
O20	P2	-	-	-	-	-	40	40	-	-	-	-	-	5	5	-	-	-	-	-	[144, 168]	[144,168]
O21	P1	-	-	-	-	-	-	30	-	-	-	-	-	-	5	-	-	-	-	-	-	[96,120]
O22	P5	-	-	-	-	-	-	40	-	-	-	-	-	-	5	-	-	-	-	-	-	[144,168]
O23	P3	-	-	-	-	-	-	20	-	-	-	-	-	-	5	-	-	-	-	-	-	[144,168]

Table 5.2b Order data for Examples

	Product		Amount (kbbl)					Delivery Rate (kbbl/h)					Delivery Window				
	Example																
o	12	The Rest	10	11	12	13	14	10	11	12	13	14	10	11	12	13	14
O1	P1	P1	10	10	10	10	10	5	5	5	5	5	[0,24]	[0,24]	[0,24]	[0,24]	[0,24]
O2	P2	P2	3	3	3	3	3	3	3	3	3	3	[0,24]	[0,24]	[0,24]	[0,24]	[0,24]
O3	P2	P2	3	3	3	3	3	3	3	3	3	3	[0,24]	[0,24]	[0,24]	[0,24]	[0,24]
O4	P1	P1	10	10	10	10	10	5	5	5	5	5	[0,24]	[0,24]	[0,24]	[0,24]	[0,24]
O5	P2	P2	3	3	3	3	3	3	3	3	3	3	[24,48]	[24,48]	[24,48]	[24,48]	[24,48]
O6	P1	P1	10	10	10	10	10	5	5	5	5	5	[24,48]	[24,48]	[24,48]	[24,48]	[24,48]
O7	P2	P2	3	3	3	3	3	3	3	3	3	3	[24,48]	[24,48]	[24,48]	[24,48]	[24,48]
O8	P1	P1	100	100	100	100	100	5	5	5	5	5	[118,190]	[118,190]	[118,190]	[118,190]	[118,190]
O9	P2	P2	3	3	3	3	3	3	3	3	3	3	[144,168]	[144,168]	[144,168]	[144,168]	[144,168]
O10	P4	P4	150	150	100	150	150	5	5	5	5	5	[150.5,185.5]	[150.5,185.5]	[150.5,185.5]	[150.5,185.5]	[150.5,185.5]
O11	P3	P3	60	60	60	60	60	5	5	5	5	5	[144,168]	[144,168]	[144,168]	[144,168]	[144,168]
O12	P2	P2	20	20	20	20	20	5	5	5	5	5	[24,48]	[24,48]	[24,48]	[24,48]	[24,48]
O13	P4	P4	60	60	60	60	60	5	5	5	5	5	[0,56]	[0,56]	[0,56]	[0,56]	[0,56]
O14	P3	P3	20	20	15	20	20	5	5	5	5	5	[48,72]	[48,72]	[48,72]	[48,72]	[48,72]
O15	P2	P2	20	20	20	20	20	4	4	4	4	4	[0,72]	[0,72]	[0,72]	[0,72]	[0,72]
O16	P2	P2	20	20	20	20	20	5	5	5	5	5	[48,72]	[48,72]	[48,72]	[48,72]	[48,72]
O17	P1	P1	10	10	10	10	10	5	5	5	5	5	[48,72]	[48,72]	[48,72]	[48,72]	[48,72]
O18	P1	P1	10	10	10	10	10	5	5	5	5	5	[48,72]	[48,72]	[48,72]	[48,72]	[48,72]
O19	P2	P2	60	60	60	60	60	5	5	5	5	5	[0,50]	[0,50]	[0,50]	[0,50]	[0,50]
O20	P2	P2	40	40	40	40	40	5	5	5	5	5	[144,168]	[144,168]	[144,168]	[144,168]	[144,168]
O21	P5	P1	30	30	30	30	30	5	5	5	5	5	[96,120]	[96,120]	[96,120]	[96,120]	[96,120]
O22	P5	P5	40	40	40	40	40	5	5	5	5	5	[144,168]	[144,168]	[144,168]	[144,168]	[144,168]
O23	P3	P3	20	20	20	20	20	5	5	5	5	5	[144,168]	[144,168]	[144,168]	[144,168]	[144,168]
O24	P5	P5	6	6	6	6	6	3	3	3	3	3	[96,120]	[96,120]	[96,120]	[96,120]	[96,120]
O25	P5	P5	20	20	20	20	20	5	5	5	5	5	[144,168]	[144,168]	[144,168]	[144,168]	[144,168]

O26	P3	P1	-	10	30	10	10	-	4	4	4	4	-	[0,76]	[144,168]	[0,76]	[0,76]
O27	P3	P4	-	20	20	20	20	-	5	4	5	5	-	[120,144]	[72,96]	[120,144]	[120,144]
O28	P4	P1	-	25	3	25	25	-	5	3	5	5	-	[120,144]	[72,96]	[120,144]	[120,144]
O29	P4	P5	-	10	15	10	10	-	5	3	5	5	-	[120,144]	[96,120]	[120,144]	[120,144]
O30	P1	P4	-	15	15	15	15	-	5	3	5	5	-	[120,144]	[96,120]	[120,144]	[120,144]
O31	P2	P1	-	-	15	15	15	-	-	5	5	5	-	-	[96,120]	[120,144]	[120,144]
O32	P5	P1	-	-	20	20	20	-	-	2	5	5	-	-	[96,120]	[144,168]	[144,168]
O33	P1	P4	-	-	20	20	20	-	-	5	5	5	-	-	[0,76]	[144,168]	[144,168]
O34	P3	P4	-	-	20	20	20	-	-	5	5	5	-	-	[120,144]	[168,192]	[168,192]
O35	P3	P5	-	-	30	30	30	-	-	5	5	5	-	-	[120,144]	[168,192]	[168,192]
O36	-	P2	-	-	-	3	3	-	-	-	3	3	-	-	-	[168,192]	[168,192]
O37	-	P1	-	-	-	10	10	-	-	-	5	5	-	-	-	[168,192]	[168,192]
O38	-	P1	-	-	-	40	40	-	-	-	5	5	-	-	-	[168,192]	[168,192]
O39	-	P4	-	-	-	10	10	-	-	-	5	5	-	-	-	[168,192]	[168,192]
O40	-	P5	-	-	-	10	10	-	-	-	5	5	-	-	-	[168,192]	[168,192]
O41	-	P1	-	-	-	-	15	-	-	-	-	5	-	-	-	-	[168,192]
O42	-	P2	-	-	-	-	20	-	-	-	-	3	-	-	-	-	[168,192]
O43	-	P3	-	-	-	-	15	-	-	-	-	5	-	-	-	-	[144,168]
O44	-	P5	-	-	-	-	20	-	-	-	-	4	-	-	-	-	[168,192]
O45	-	P4	-	-	-	-	10	-	-	-	-	5	-	-	-	-	[96,120]

Table 5.3 Product and component tank data for Examples 1-14

	Initial Contain		Initial Stock (kbbl)		Capacity (kbbl)		Storable Products					Maximum Delivery Rate/Feed Rate (kbbl/h)							
	Example																		
Tank	1	2-14	1	2-14	1	2-14	1	2-8	9-11	13-14	12	1	2-5	6	7-8	9-10	11	12	13-14
PT-101	P3	P3	90.20	30.00	100	150	P2, P3	P2, P3	P2, P3, P5	P2, P3, P5	P2, P3, P5	15	20	25	20	30	30	30	30
PT-102	P3	P3	0.00	0.00	100	150	P2, P3	P2, P3	P2, P3, P5	P2, P3, P5	P2, P3, P5	15	20	25	20	30	30	30	30
PT-103	P2	P2	14.08	14.08	150	150	P2, P3	P2, P3	P2, P3, P5	P2, P3, P5	P2, P3, P5	15	20	25	20	30	30	30	30
PT-104	P2	P4	28.49	25.00	100	200	P1, P2	P2- P4	P2- P4	P2- P4	P2- P5	15	20	25	20	30	30	30	30
PT-105	P1	P2	20.20	28.49	100	200	P1, P2	P2, P3	P2, P5	P2, P5	P2, P3, P5	15	20	25	20	30	30	30	30
PT-106	-	P2	-	57.59	-	150	-	P2, P3	P2, P5	P2, P5	P2, P3, P5	-	20	25	20	30	30	30	30
PT-107	-	P1	-	13.79	-	200	-	P1, P4	P1, P4	P1, P4	P1, P4	-	20	25	20	30	30	30	30
PT-108	-	P1	-	12.36	-	150	-	P1, P4	P1, P4	P1, P4	P1, P4	-	20	25	20	30	30	30	30
PT-109	-	P4	-	23.96	-	200	-	P1, P4	P1, P4	P1, P4	P1, P4	-	20	25	20	30	30	30	30
PT-110	-	P1	-	60.00	-	150	-	P1, P4	P1, P4	P1, P4	P1, P4	-	20	25	20	30	30	30	30
PT-111	-	P1	-	12.36	-	150	-	P1, P4	P1, P4	P1, P4	P1, P4	-	20	25	20	30	30	30	30
CT-101	C1	C1	27.38	26.46	200	250	-	-	-	-	-	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
CT-102	C2	C2	34.58	67.90	200	300	-	-	-	-	-	1.00	0.50	0.50	0.50	0.50	0.50	0.50	0.50
CT-103	C3	C3	59.44	59.44	250	300	-	-	-	-	-	1.20	1.00	1.00	1.00	1.00	1.00	1.00	1.00
CT-104	C4	C4	28.29	44.44	250	300	-	-	-	-	-	1.20	1.00	1.00	1.00	1.00	1.00	1.00	1.00
CT-105	C5	C5	10.59	10.59	200	200	-	-	-	-	-	1.00	0.80	0.80	0.80	0.80	0.70	0.80	0.70
CT-106	C6	C6	19.53	19.53	200	250	-	-	-	-	-	1.00	0.50	0.50	0.50	0.50	0.50	0.50	0.50
CT-107	C7	C7	27.30	46.91	100	250	-	-	-	-	-	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.50
CT-108	C8	C8	49.34	49.47	100	250	-	-	-	-	-	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.50
CT-109	C9	C9	13.84	44.58	200	250	-	-	-	-	-	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

Table 5.4 Component and product property indices for Examples 1-14

Comp. & P	RBN			RVI		SULI		BI		AROI		OI	
	Example												
	1	2-8	9-14	2-8	9-14	2-8	9-14	2-8	9-14	2-8	9-14	4-8	9-14
C1	86.5000	86.5000	86.5000	140.4650	140.4650	80.0000	80.0000	0.7800	0.7800	25.0000	25.0000	1.0000	1.0000
C2	103.6600	103.6600	103.6600	68.9213	68.9213	40.0000	40.0000	0.9800	0.9800	31.7000	31.7000	23.8000	23.8000
C3	111.3500	111.3500	111.3500	87.6804	87.6804	0.0000	0.0000	1.2000	1.2000	48.0000	48.0000	0.8500	0.8500
C4	113.9300	113.9300	113.9300	51.4659	51.4659	5.0000	5.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
C5	94.5000	94.5000	94.5000	175.5886	175.5886	0.0000	0.0000	0.0960	0.0960	0.0000	0.0000	0.4000	0.4000
C6	118.1600	118.1600	118.1600	19.9115	19.9115	0.0800	0.0800	0.0050	0.0050	0.0000	0.0000	0.7200	0.7200
C7	144.6800	144.6800	144.6800	12.5522	12.5522	7.5000	7.5000	0.0078	0.0078	0.0500	0.0500	0.00038	0.00038
C8	150.6600	150.6600	150.6600	110.5925	110.5925	2.0000	2.0000	0.2500	0.2500	19.2000	19.2000	0.1500	0.1500
C9	92.5000	92.5000	92.5000	436.3383	436.3383	30.0000	30.0000	0.0920	0.0920	24.0000	24.0000	0.0600	0.0600
P1	[110.45,+∞]	[110.45,+∞]	[110.45,+∞]	[15,170]	[15,170]	[0,45]	[0,45]	[0,0.86]	[0,0.86]	[0,35.00]	[0,35.00]	[0,20.00]	[0,20.00]
P2	[111.95,+∞]	[111.95,+∞]	[111.95,+∞]	[15,170]	[15,170]	[0,50]	[0,50]	[0,0.92]	[0,0.92]	[0,36.00]	[0,36.00]	[0,18.00]	[0,18.00]
P3	[108.97,+∞]	[108.97,+∞]	[108.97,+∞]	[15,170]	[15,170]	[0,44]	[0,44]	[0,0.94]	[0,0.94]	[0,42.00]	[0,42.00]	[0,20.00]	[0,20.00]
P4	-	[103.24,+∞]	[103.24,+∞]	[15,170]	[15,170]	[0,50]	[0,50]	[0,0.90]	[0,0.90]	[0,40.00]	[0,40.00]	[0,18.00]	[0,18.00]
P5	-	-	[115.01,+∞]	-	[15,170]	-	[0,48]	-	[0,0.93]	-	[0,40.00]	-	[0,20.00]

Comp. & P	DNI			VI		FPI		FRI		OXI		FLI	
	Example												
	4	5-8	9-14	4-8	9-14	7-8	9-14	7-8	9-14	7-8	9-14	7-8	9-14
C1	1.4850	1.4850	1.4850	46.1247	46.1247	17.0286	17.0286	15.8409	15.8409	0.2500	0.2500	3.4500	3.4500
C2	1.3340	1.3340	1.3340	42.8839	42.8839	12.7248	12.7248	10.3290	10.3290	0.7500	0.7500	6.2500	6.2500
C3	1.2200	1.2200	1.2200	64.4694	64.4694	4.8294	4.8294	1.0929	1.0929	2.0000	2.0000	2.3600	2.3600
C4	1.5800	1.5800	1.5800	76.9637	76.9637	3.5443	3.5443	1.4811	1.4811	1.2500	1.2500	3.5600	3.5600
C5	1.4980	1.4980	1.4980	69.7708	69.7708	0.2220	0.2220	0.2992	0.2992	0.0800	0.0800	1.9600	1.9600
C6	1.4360	1.4360	1.4360	47.9459	47.9459	0.05215	0.05215	0.0885	0.0885	0.0000	0.0000	3.6500	3.6500
C7	1.1500	1.1500	1.1500	46.1727	46.1727	0.0591	0.0591	0.2192	0.2192	0.0005	0.0005	2.9600	2.9600
C8	1.3480	1.3480	1.3480	47.5274	47.5274	0.0671	0.0671	0.0761	0.0761	18.2000	18.2000	5.4600	5.4600
C9	1.6050	1.6050	1.6050	40.8311	40.8311	0.0384	0.0384	0.0320	0.0320	0.8500	0.8500	7.9500	7.9500
P1	[1.190,1.667]	[1.190,1.667]	[1.190,1.667]	[40,72.0]	[40,72.0]	[0,14.0]	[0,14.0]	[0,12.0]	[0,12.0]	[0,0.028]	[0,2.8]	[1.4,7.60]	[1.4,7.60]
P2	[1.199,1.667]	[1.199,1.667]	[1.199,1.667]	[40,72.5]	[40,72.5]	[0,15.2]	[0,15.2]	[0,15.0]	[0,15.0]	[0,0.0275]	[0,2.75]	[1.4,7.25]	[1.4,7.25]
P3	[1.182,1.667]	[1.182,1.667]	[1.182,1.667]	[40,70.5]	[40,70.5]	[0,15.8]	[0,15.8]	[0,15.5]	[0,15.5]	[0,0.029]	[0,2.9]	[1.4,7.20]	[1.4,7.20]
P4	[1.190,1.667]	[1.190,1.667]	[1.190,1.667]	[40,72.0]	[40,72.0]	[0,16.5]	[0,16.5]	[0,12.0]	[0,12.0]	[0,0.027]	[0,2.7]	[1.4,7.50]	[1.4,7.50]
P5	-	-	[1.200,1.667]	-	[40,72.0]	-	[0,16.0]	-	[0,15.0]	-	[0,3.0]	-	[1.4,7.40]

Table 5.5 Allowable composition ranges for components in products of Examples 1-14

	C1			C2			C3			C4					
	Example														
Product	1	2-8	9-14	1	2-8	9-14	1	2-8	9-14	1	2-4	5-7	8	9-12	13-14
P1	[0, 0.22]	[0, 0.22]	[0, 0.22]	[0.1, 1]	[0.10, 1]	[0.10, 1]	[0, 1]	[0, 1]	[0, 1]	[0, 0.40]	[0, 0.40]	[0, 0.40]	[0, 0.40]	[0, 0.40]	[0, 0.40]
P2	[0, 0.24]	[0, 0.24]	[0, 0.24]	[0.1, 1]	[0.10, 1]	[0.10, 1]	[0, 1]	[0, 1]	[0, 1]	[0, 0.45]	[0, 0.45]	[0, 0.45]	[0, 0.45]	[0, 0.45]	[0, 0.45]
P3	[0, 0.25]	[0, 0.25]	[0, 0.25]	[0.1, 1]	[0.10, 1]	[0.10, 1]	[0, 1]	[0, 1]	[0, 1]	[0, 0.43]	[0, 0.43]	[0, 0.43]	[0, 0.43]	[0, 0.43]	[0, 0.43]
P4	-	[0, 0.24]	[0, 0.24]	-	[0.10, 1]	[0.10, 1]	-	[0, 1]	[0, 1]	-	[0, 0.44]	[0, 0.44]	[0, 0.44]	[0, 0.44]	[0, 0.44]
P5	-	-	[0, 0.30]	-	-	[0.15, 1]	-	-	[0, 1]	-	-	-	-	[0, 0.40]	[0, 0.40]

	C5			C6			C7			C8			C9		
	Example														
Product	1	2-8	9-14	1	2-8	9-14	1	2-8	9-14	1	2-8	9-14	1	2-8	9-14
P1	[0, 0.25]	[0, 0.25]	[0, 0.25]	[0, 0.20]	[0, 0.20]	[0, 0.20]	[0, 0.25]	[0, 0.25]	[0, 0.25]	[0, 0.30]	[0, 0.30]	[0, 0.30]	[0, 0.15]	[0, 0.15]	[0, 0.15]
P2	[0, 0.25]	[0, 0.25]	[0, 0.25]	[0, 0.22]	[0, 0.22]	[0, 0.22]	[0, 0.25]	[0, 0.25]	[0, 0.25]	[0, 0.30]	[0, 0.30]	[0, 0.30]	[0, 0.18]	[0, 0.18]	[0, 0.18]
P3	[0, 0.25]	[0, 0.25]	[0, 0.25]	[0, 0.18]	[0, 0.18]	[0, 0.18]	[0, 0.25]	[0, 0.25]	[0, 0.25]	[0, 0.30]	[0, 0.30]	[0, 0.30]	[0, 0.20]	[0, 0.20]	[0, 0.20]
P4	-	[0, 0.25]	[0, 0.25]	-	[0, 0.20]	[0, 0.20]	-	[0, 0.25]	[0, 0.25]	-	[0, 0.30]	[0, 0.30]	-	[0, 0.16]	[0, 0.16]
P5	-	-	[0, 0.25]	-	-	[0, 0.20]	-	-	[0, 0.25]	-	-	[0, 0.30]	-	-	[0, 0.17]

Table 5.6 Blender and economic data for Examples 1-14

	$RL_{b\,0}$, h & $CQ_{b\,0}$, kbbl					Minimum & Maximum Blending Rate (kbbl/h)							Allowable Product				
	Example																
Blender	1-4	5	6-7	8-12	13-14	1	2	3-6	7	8-9	10-12	13-14	1	2-7	8	9-12	13-14
B-1	0 & 0	10 & 150	0 & 0	0 & 0	0 & 0	2.0-15	1.5-15	1.5-20	1.5-25	1.5-25	1.5-30	1.5-30	P1-P3	P1-P4	P1-P4	P1-P5	P1-P5
B-2	-	-	-	0 & 0	0 & 0	-	-	-	-	1.5-25	1.5-30	1.5-30	-	-	P1-P4	P1-P5	P1-P5
B-3	-	-	-	-	0 & 0	-	-	-	-	-	-	1.5-25	-	-	-	-	P1-P5
	Minimum Run Length of Each Product in Blender (h)																
	P1			P2				P3			P4				P5		
	Example																
	1-7	8-12	13-14	1-7	8-12	13	14	1-7	8-12	13-14	2-7	8-11	12	13-14	9-11	12	13-14
B-1	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	5	5
B-2	-	6	6	-	6	6	6	-	6	6	-	6	6	6	6	5	5
B-3	-	-	6	-	-	6	6	-	-	6	-	-	-	6	-	-	5
	Ex	C1	C2	C3	C4	C5	C6	C7	C8	C9	Ex	Demurrage Cost (k$/h)			Scheduling Horizon (h)		
Cost	1	20	40	55	45	25	50	58	60	30	1	2.1			72		
($/bbl)	2-14	20	24	30	25	22	27	50	50	22.5	2-14	2.5			192		
Transition	Ex	PT-101	PT-102	PT-103	PT-104	PT-105	PT-106	PT-107	PT-108	PT-109	PT-110	PT-111	Transition Cost in blender (k$/instance)				
Cost	1	9.8	9.8	14.5	9.8	9.8	-	-	-	-	-	-	20				
(k$/instance)	2-14	14.5	14.5	14.5	19	19	14.5	19	14.5	19	14.5	14.5					

Table 5.7 Periods, slots, and feed flow rates to component tanks for Examples 1-14

		Period		Feed Flow Rate to Component Tank (kbbl/h)								
Ex	Period	Duration	Slot	CT-101	CT-102	CT-103	CT-104	CT-105	CT-106	CT-107	CT-108	CT-109
1	1	40	1-3	1.2	0.8	1.2	1.2	0.5	0.8	0.0	0.0	1.0
	2	32	4-5	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
2	1	60	1-2	1.2	0.8	1.2	1.2	0.5	0.8	0.0	0.0	1.0
	2	132	3-5	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
3-5	1	100	1-3	1.2	0.8	1.2	1.2	0.5	0.8	0.0	0.0	1.0
	2	92	4-5	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
6	1	120	1-3	1.2	0.8	1.2	1.2	0.7	0.8	0.0	0.0	1.0
	2	72	4-5	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
7	1	80	1-3	1.2	0.8	1.2	1.2	0.7	0.8	0.0	0.0	1.0
	2	70	4-6	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
	3	42	7-9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
8	1	80	1-3	1.2	0.8	1.2	1.2	0.5	0.8	0.0	0.0	1.0
	2	70	4-6	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
	3	42	7-8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
9	1	80	1-3	1.0	0.5	1.0	1.0	0.5	0.5	0.0	0.0	1.0
	2	70	4-7	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
	3	42	8-9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
10	1	80	1-3	1.0	0.5	1.0	1.0	0.8	0.5	0.0	0.0	1.0
	2	60	4-7	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
	3	52	8-10	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
11	1	80	1-4	1.0	0.5	1.0	1.0	0.7	0.5	0.0	0.0	1.0
	2	60	5-9	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
	3	52	10-14	0.5	0.5	0.5	0.5	0.5	0.5	0.0	0.0	0.5
12	1	50	1-5	1.0	0.5	1.0	1.0	0.8	0.5	0.0	0.0	1.0
	2	50	6-10	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
	3	50	11-14	0.5	0.5	0.5	0.5	0.5	0.5	0.0	0.0	0.5
	4	42	15-17	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
13-14	1	50	1-5	1.0	0.5	1.0	1.0	0.7	0.5	0.5	0.5	1.0
	2	50	6-10	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
	3	50	11-14	0.5	0.5	0.5	0.5	0.5	0.5	0.0	0.0	0.5
	4	42	15-17	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

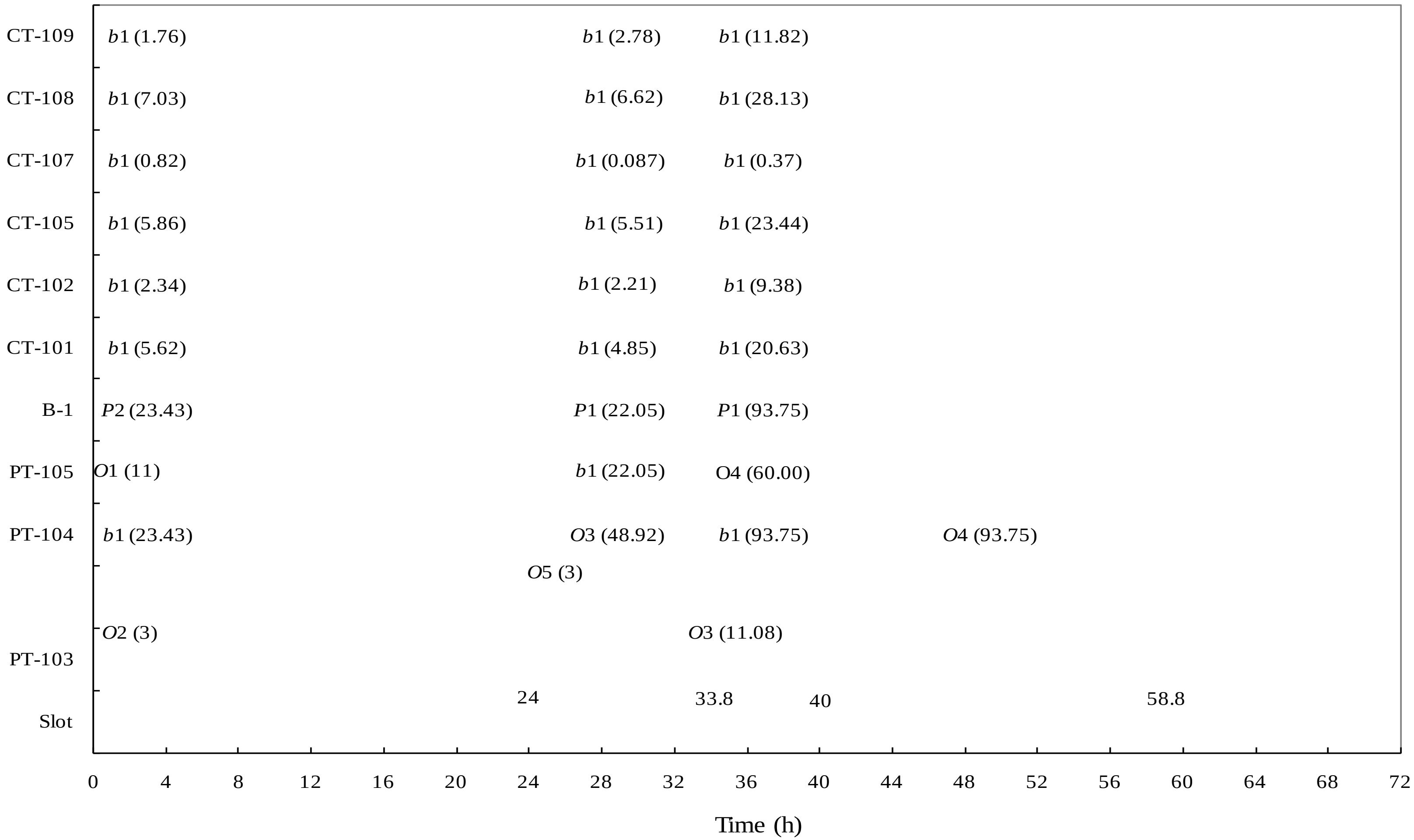

Figure 5.6 Optimal schedule for Example 1 (5 orders) from SPM

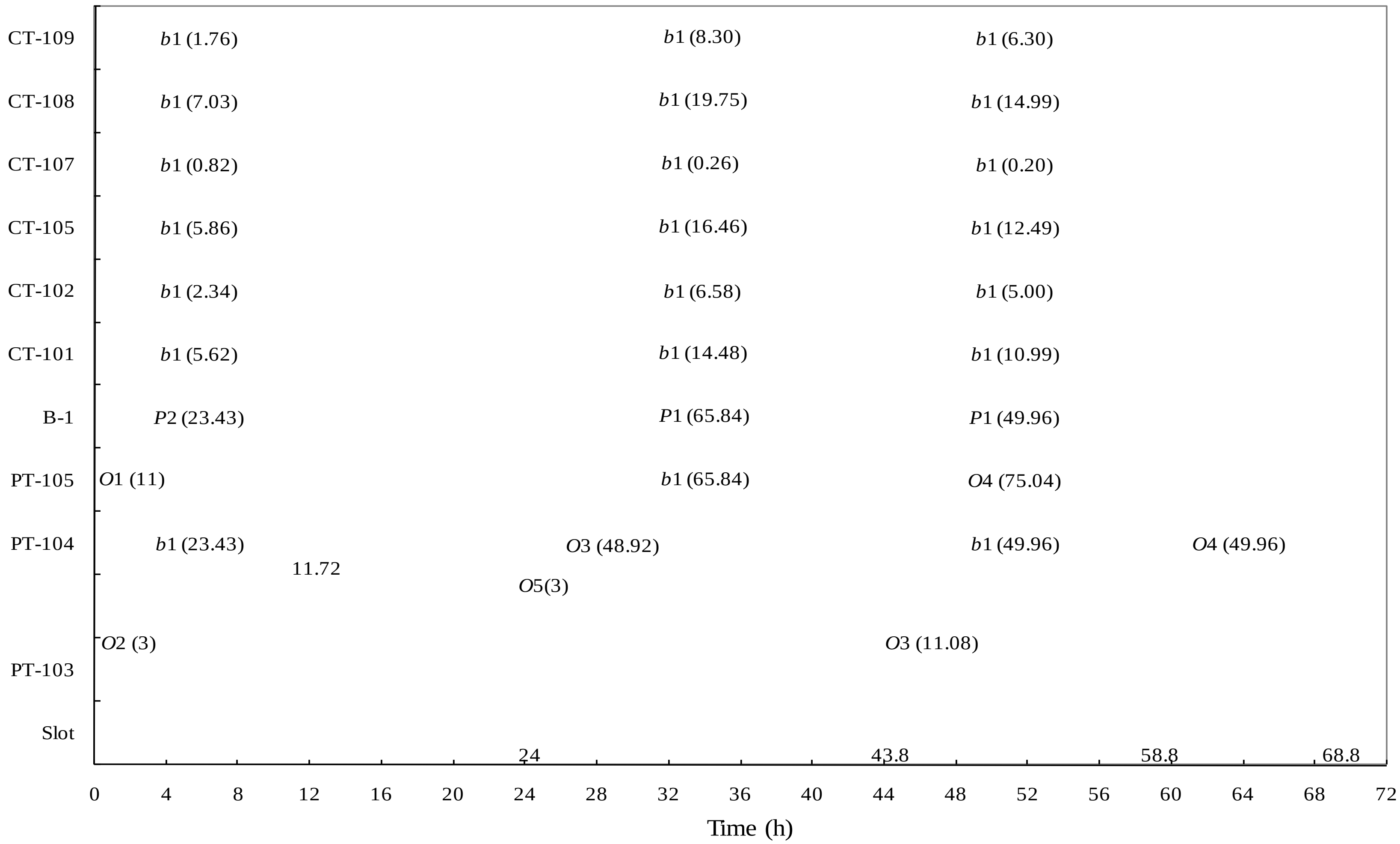

Figure 5.7 Optimal schedule for Example 1 (5 orders) from RSPM

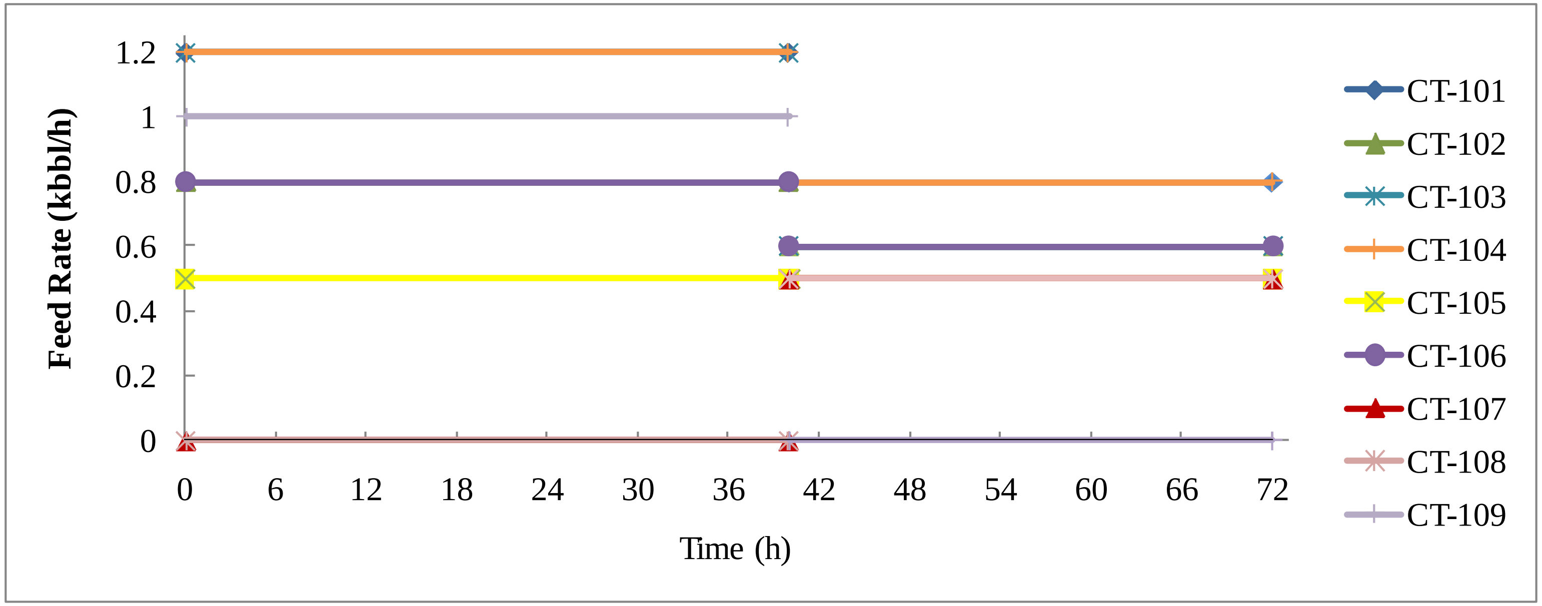

Figure 5.8 Feed rate profiles of blend components from component tanks for Example 1

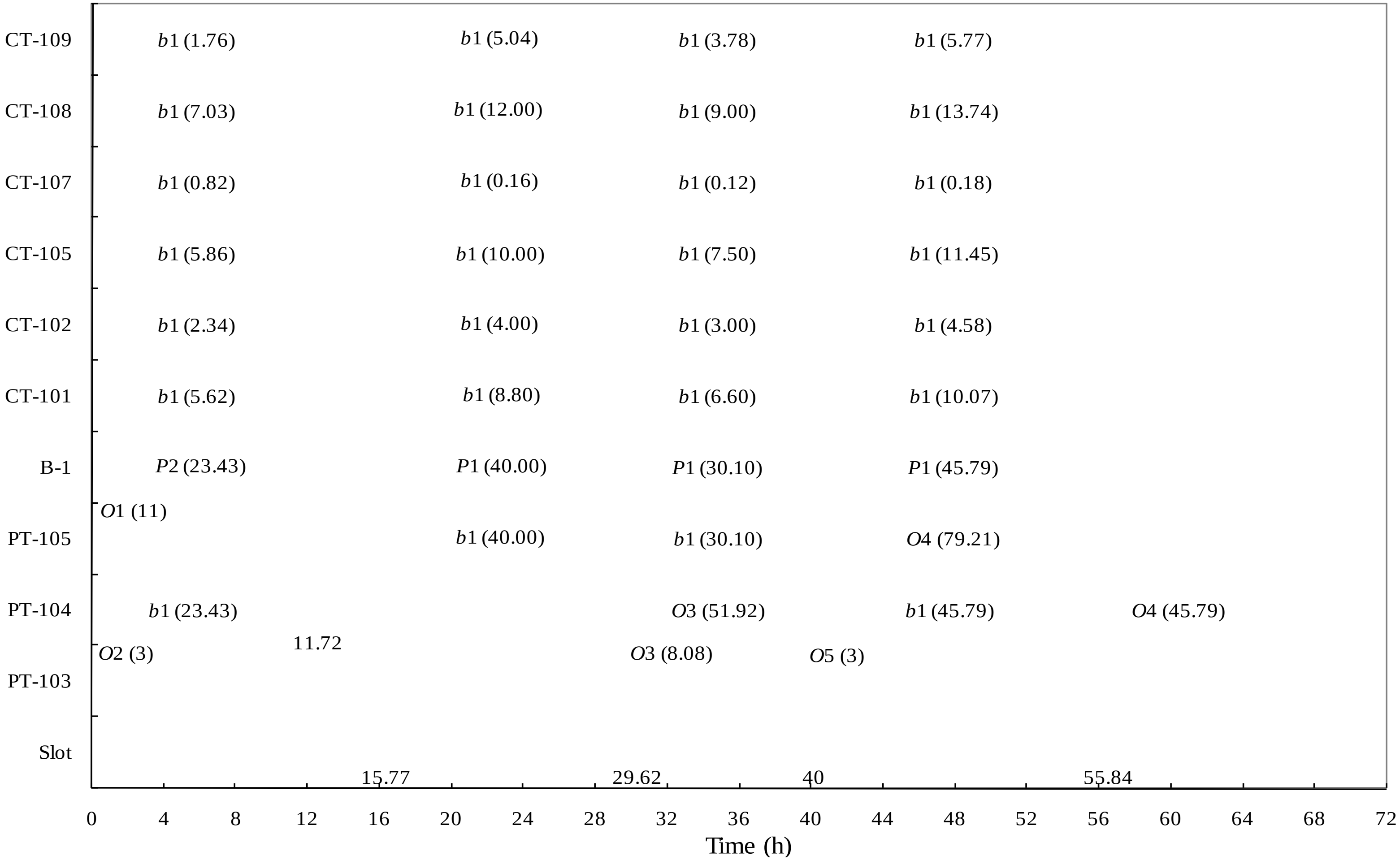

Figure 5.9 Optimal schedule for Example 1 (5 orders) from RMPM

Using the above blend rates, RSPM is solved to obtain the optimal solution of 5149.73 kbbl within 0.06 CPU s. Figure 5.7 shows the optimal schedule from RSPM. Because the solutions from RSPM and SPM have the same cost, it is a guaranteed global optimal solution. Each order is delivered before its due date, so no demurrage incurs in this example. The blender has one product transition (P2-P1) after slot 1, and PT-104 transitions from P2 to P1 after slot 2. PT-104 delivers O3 and O5 simultaneously during slot 2, but there are no intermittent deliveries. Thus, Figure 5.7 is a realistic optimal schedule for this example.

Now, two periods of 40 h and 32 h in the scheduling horizon are considered. Figure 5.8 shows the feed rate profiles to component tanks. The feed rates to CT-101 through CT-109 in the first period are 1.2 kbbl/h, 0.8 kbbl/h, 1.2 kbbl/h, 1.2 kbbl/h, 0.5 kbbl/h, 0.8 kbbl/h, 0.0 kbbl/h, 0.0 kbbl/h, and 1.0 kbbl/h respectively. In the second period, they are 0.8 kbbl/h, 0.6 kbbl/h, 0.6 kbbl/h, 0.8 kbbl/h, 0.5 kbbl/h, 0.6 kbbl/h, 0.5 kbbl/h, 0.5 kbbl/h, and 0.0 kbbl/h respectively. Three slots are used for the first period and two slots for the second. The optimal solution remains the same as $ 5149.73K, but the solution time increases to 12.6 CPU s due to the presence of one more slot in MPM. Figure 5.9 is an optimal schedule for this Example from RMPM.

5.7 Detailed Evaluation

Tables 5.2-5.7 show the data for Examples 2-14 that is used for a detailed evaluation of the proposed methodology. Nine components, eleven product tanks, and 192-hour (8-day) scheduling horizon are used. Examples 2-7 have one blender, Examples 8-12 have two, and Examples 13-14 have three blenders. 10-45 orders are used in these examples as shown in Tables 5.2-5.7. The test examples are designed with widely varying structure, size, scale, and complexity, and they mimic the real-life industrial scenarios very well. For all examples except Example 5, all blenders are idle at time zero. In Example 5, the blender is processing P1 before time zero. All examples are solved on a Dell precision PWS690 (Intel® Xeon^{R} CPU 3 GHz, 16 GB RAM) running Windows XP using solver CPLEX 10.0.1/GAMS22.2.

Tables 5.8-5.9 give the computational performance of SPM and MPM. The proposed procedure obtains guaranteed global optimal solutions for Examples 1-5, because both RSPM and SPM give the same solution. Optimal solutions can not be got or proved to be optimality for Example 6-14, so upper limits on CPU times are set for these examples as shown in Tables 8-9. For instance, 3 h (10800 s) are used for SPM and 1 h (3600 s) for RSPM of Examples 6-10. As it can be seen from Tables 8-9, solutions are typically obtained quite quickly, but proving optimality takes much time. For instance, SPM gives a solution of $ 5213.88K for Example 6 in just 101 CPU s, but spends the remaining 2.97 h to prove optimality. In fact, it fails to prove optimality even after 3 days of CPU time.

Table 5.10 gives the RMIP values and best possible solutions from SPM for Examples 1-14. The best possible solutions, which are the better lower bounds than RMIP values, do not improve much over time for Examples 7-14. For instance, SPM gives an RMIP and best solution of $ 7797.02K for Example 7, but fails to prove optimality even after 3 h. Similarly, RMIP from SPM for Example 9 is $ 10004.27K.

After 3 h, the best possible solution is \$ 10016.35K, which is an improvement of only 0.12%. It seems that lower bounds for SPM and MPM improve very slowly for large problems, and it is difficult to solve SPM or MPM to optimality. One possible reason for this increased complexity is the transitions in the product tanks, which result in very low RMIP values and difficult to improve. To illustrate this point, Example 12 with 35 orders is used, in which some product tanks are allowed to hold one more product compared to other examples. For instance, PT-105 is allowed to hold P2, P3, and P5 in Example 12, while only P2 and P5 in other examples. 30 h are needed for this example to obtain solutions with a relative gap of 10.64%.

RSPM improved solution qualities of SPM solutions by 1.36% on an average for Examples 10, 11, and 13. For instance, \$16858.99K from SPM in Example 13 was improved to \$ 16239.39K by RSPM. However, it did not do so for Examples 9, 12, and 14. For instance, SPM gave \$ 10612.65K for Example 9, but \$10781.40K from RSPM.

The observations from Table 5.9 are similar to those from Table 5.8. For illustration, the final schedules for Examples 4 (15 orders), 5 (15 orders), 9 (23 orders), and 12 (35 orders) in Figures 10-13, are given respectively. The following features of these schedules are noteworthy.

1) In Figure 11, although the blender has already been processing 100 kbbl of P1 at time zero for 10 h, it continues with P1 for another 9.54 h during slot 1 to satisfy the minimum run length.
2) There is no demurrage in Figures 10-12. Figure 13b has demurrage. *O*5 is delivered in slot 2, incurring a delay of 1 h compared to its due date at 48 h.

Table 5.8 Computational performance of SPM

Ex	Order	Slot	Discrete Variables	Continuous Variables	Constraints	Non Zero Elements	CPU Time for SPM (s)	CPU Time for RSPM (s)	Cost from SPM (k$)	Cost from RSPM (k$)
1	5	4	100	269	1098	3094	0.77	0.06	5149.73	5149.73
2	10	4	252	598	2847	9037	48.3	0.28	3658.11	3658.11
3	12	4	315	633	3199	10074	177	0.14	3159.12	3159.12
4	15	4	372	723	3823	12555	544	0.50	4556.67	4556.67
5	15	4	372	723	3835	12579	250	1.0	4556.67	4556.67
6	18	5	576	950	5438	17673	101^{+}	12.1	5213.88	5213.88
7	20	6	775	1197	7319	24924	9838^{+}	21.8	8100.35	8100.35
8	20	4	523	959	6256	22473	10800^{*}	1.9	8100.35	8100.35
9	23	7	1079	1673	12556	46034	10222^{+}	3600^{*}	10612.65	10781.40
10	25	9	1503	2209	17032	61746	10800^{*}	3600^{*}	11371.09	11327.20
11	30	12	2328	3216	25475	90261	36000^{*}	4065	13304.72	13300.30
12	35	16	3851	4958	40480	139101	99426^{+}	10800^{*}	15305.41	15367.18
13	40	16	4142	5630	50133	175721	108000^{*}	10800^{*}	16858.99	16239.39
14	45	17	4830	6439	57657	199007	108000^{*}	10800^{*}	18641.72	19780.33

Note: CPU time limit for SPM is set at 10800 s for Examples 1-10, 36000 s for Example 11, and 108000 s for Examples 12-14
CPU time limit for RSPM is set at 3600 s for Examples 1-10, and 10800 s for Examples 11-14
* Reached CPU time limit
+ The time when final solution is found, but termination at CPU time limit

Table 5.9 Computational performance of MPM

Ex	Order	Discrete Variables	Continuous Variables	Constraints	Non Zero Elements	CPU Time for MPM (s)	CPU Time for RMPM (s)	Cost from MPM (k$)	Cost from RMPM (k$)
1	5	130	328	1384	3912	12.5	0.14	5149.73	5149.73
2	10	329	729	3600	11423	137	0.16	3658.11	3658.11
3	12	411	769	4048	12736	82.6	0.14	3179.12	3179.12
4	15	486	877	4840	15865	800	0.11	4556.67	4556.67
5	15	486	877	4852	15889	95.2	0.37	4556.67	4556.67
6	18	576	950	5438	17676	264^{+}	0.62	5213.88	5213.88
7	20	1219	1725	11138	37790	9842^{+}	4.1	8100.35	8100.35
8	20	1159	1779	12848	45843	9859^{+}	20.6	8100.35	8329.13
9	23	1423	2107	16248	59457	10800^{*}	292	10695.65	10636.40
10	25	1685	2436	18968	68702	10800^{*}	2415	11383.04	11649.17
11	30	2328	3216	25475	90266	36000^{*}	402	13359.51	13357.69
12	35	4099	5258	43046	147832	108000^{*}	10800^{*}	15426.84	15326.00
13	40	4414	5966	53305	186759	108000^{*}	1050	16306.08	16282.08
14	45	4830	6439	57657	198958	108000^{*}	10800^{*}	19522.23	19207.48

Note: CPU time limit for MPM is set at 10800 s for Examples 1-10, 36000 s for Example 11, and 108000 s for Examples 12-14
CPU time limit for RMPM is set at 3600 s for Examples 1-10, and 10800 s for Examples 11-14
* Reached CPU time limit
+ The time when final solution is found, but termination at CPU time limit

Table 5.10 RMIPs and best possible solutions for Examples 1-14 from SPM

Ex	RMIP from SPM (k$)	Best Possible Solution from SPM (k$)	Dev-1 (%)	Cost from SPM (k$)	Cost from RSPM (k$)	Dev-2 (%)
1	4988.52	5149.73	3.23	5149.73	5149.73	0.00
2	3570.65	3658.11	2.45	3658.11	3658.11	0.00
3	3105.08	3159.12	1.74	3159.12	3159.12	0.00
4	4501.78	4556.67	1.22	4556.67	4556.67	0.00
5	4501.78	4556.67	1.22	4556.67	4556.67	0.00
6	4407.80	5103.74	15.79	5213.88	5213.88	2.11
7	7797.02	7797.02	0.00	8100.35	8100.35	3.74
8	7777.02	7777.02	0.00	8100.35	8100.35	3.99
9	10004.27	10016.35	0.12	10612.65	10781.40	7.10
10	10608.50	10621.19	0.12	11371.09	11327.20	6.23
11	12472.99	12486.17	0.11	13804.72	13300.30	6.12
12	13731.95	13732.02	0.00	15305.41	15367.18	10.64
13	14682.38	14695.68	0.09	18858.99	16239.39	9.51
14	16583.16	16583.82	0.00	18641.72	19780.33	16.16

Dev-1: The relative gap between columns 2 and 3

Dev-2: The relative gap between columns 3 and 6

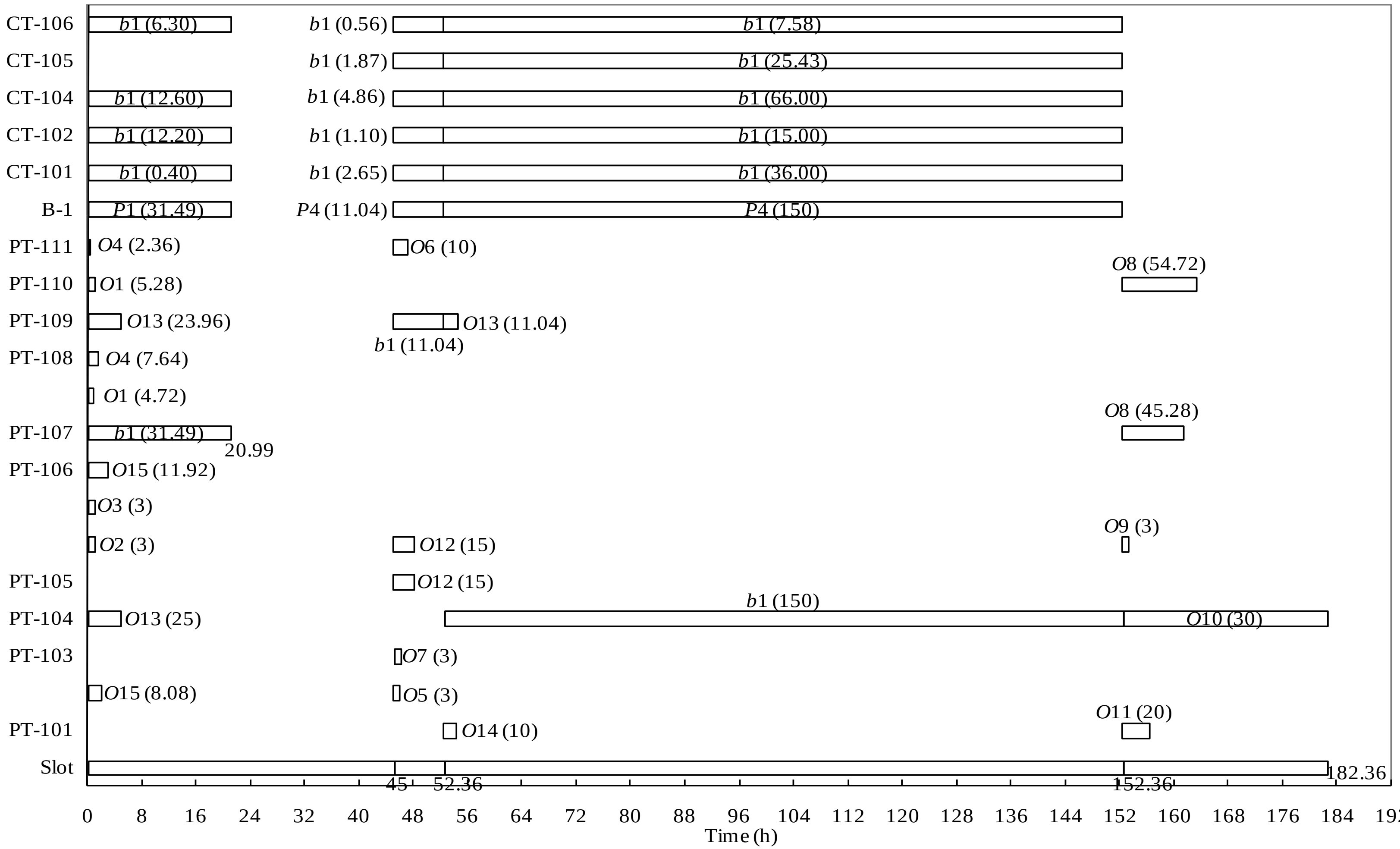

Figure 5.10 Optimal schedule for Example 4 (15 orders) from RSPM

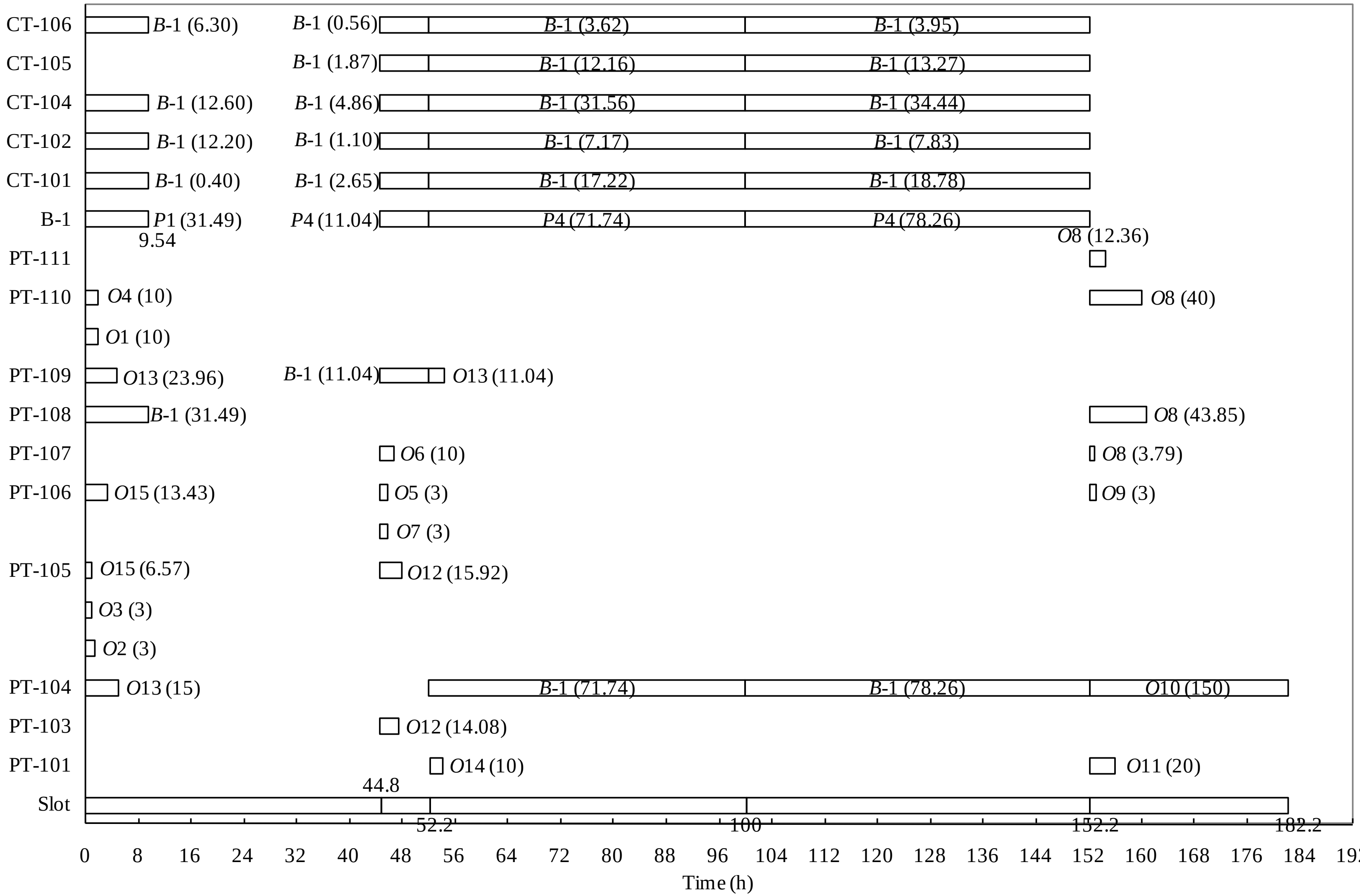

Figure 5.11 Optimal schedule for Example 5 (15 orders) from RMPM

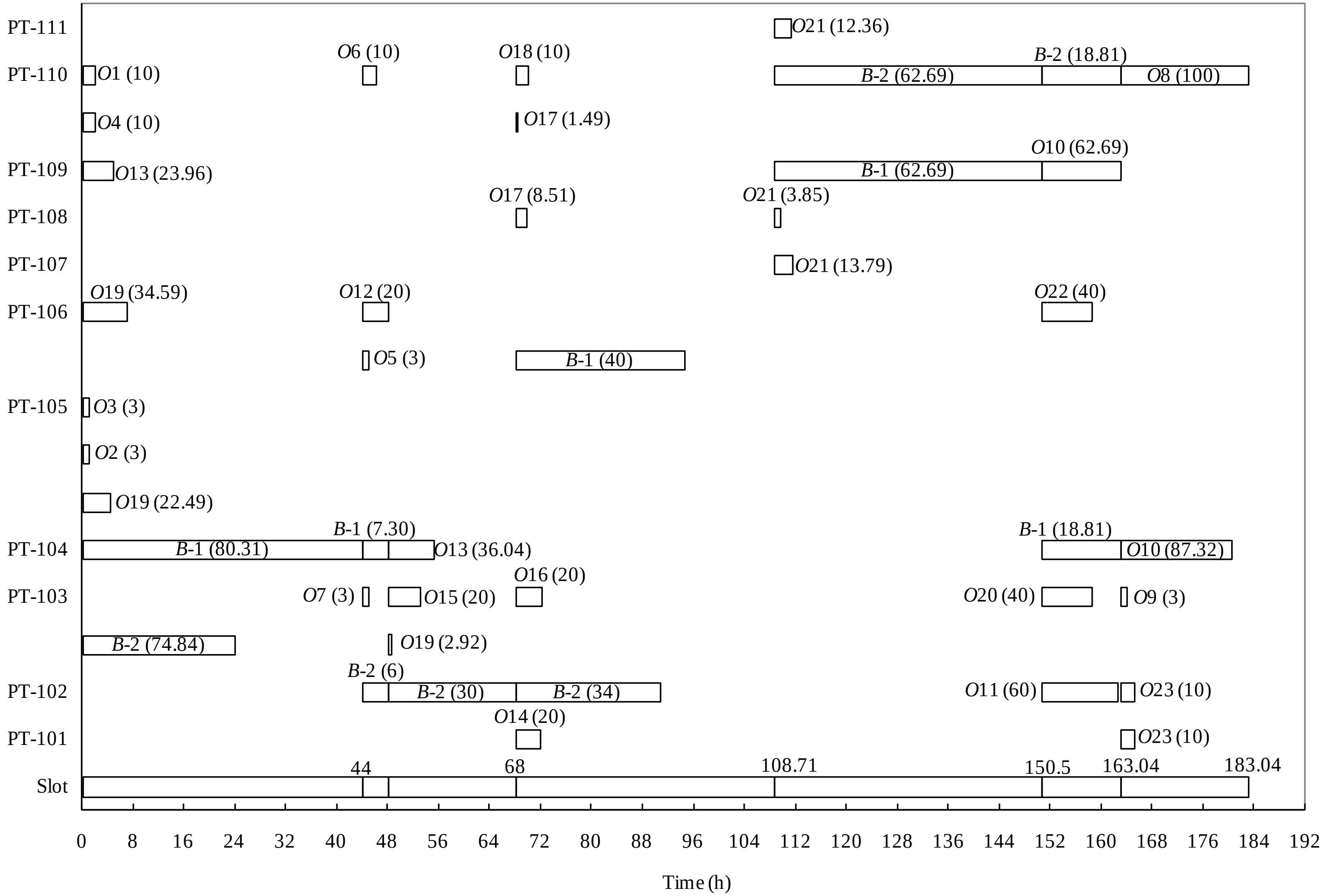

Figure 5.12a Order delivery schedule for Example 9 (23 orders) from RSPM

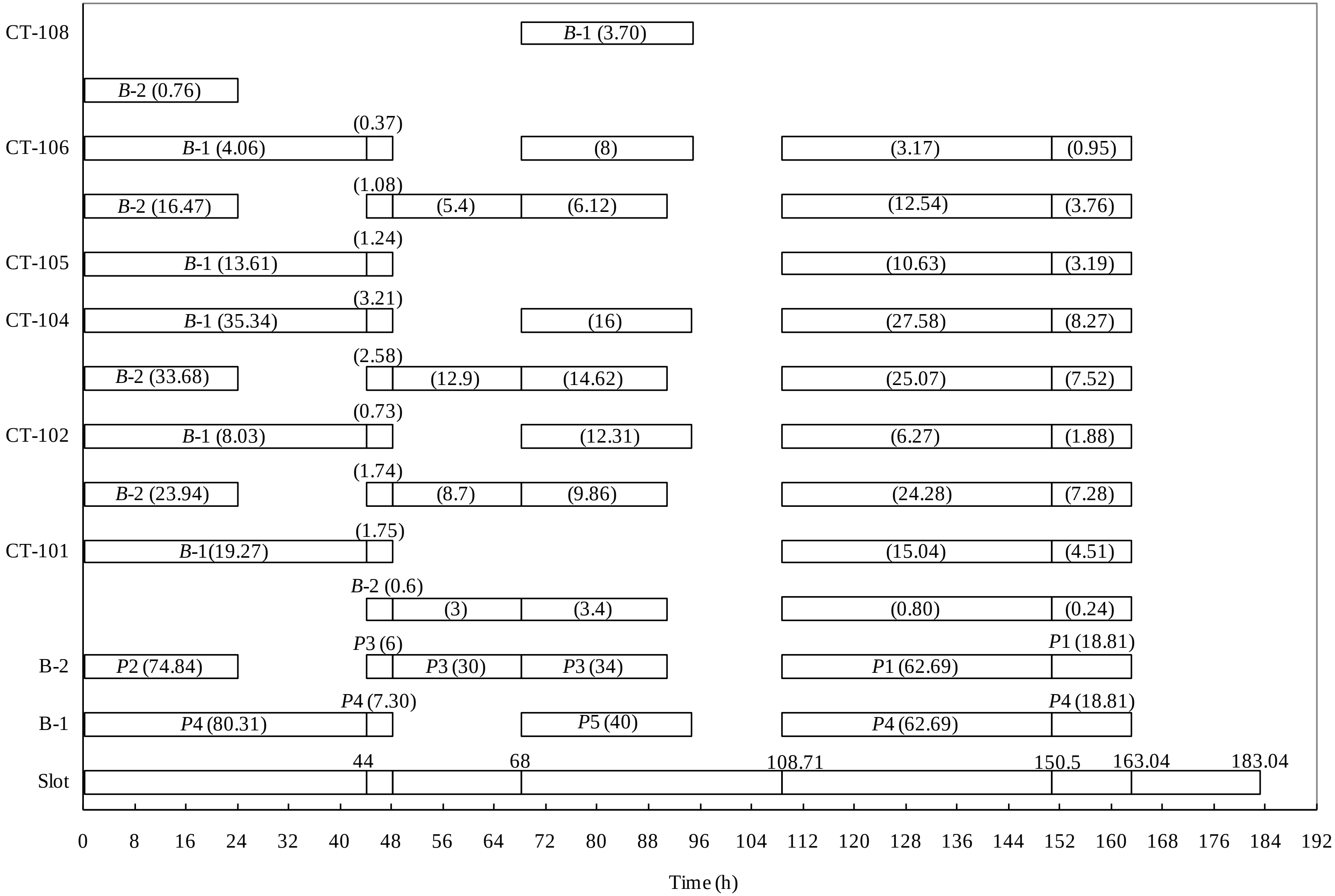

Figure 5.12b Blending schedule for Example 9 (23 orders) from RSPM

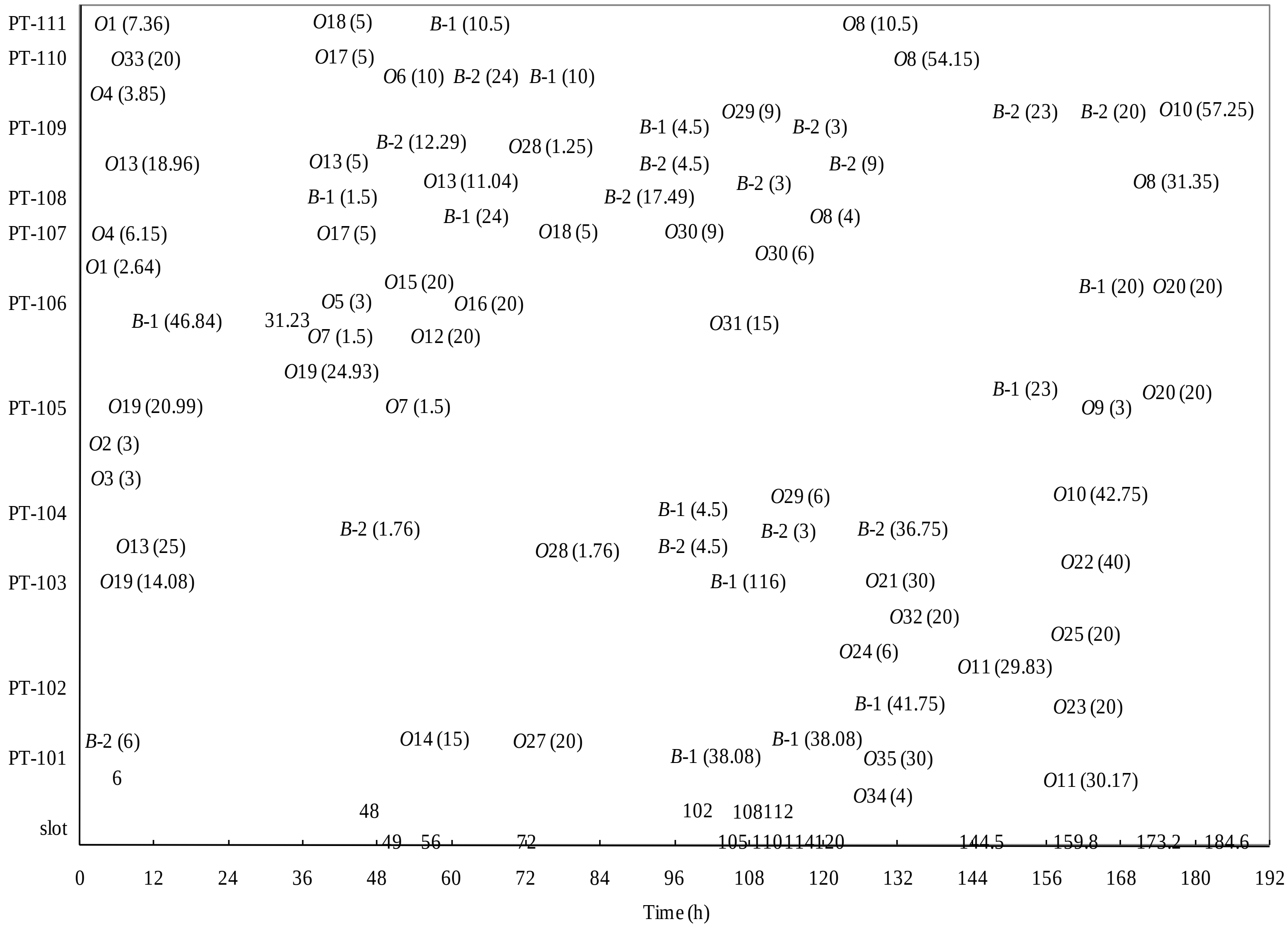

Figure 5.13a Order delivery schedule for Example 12 (35 orders) from RSPM with intermittent delivery of O13 by PT-109

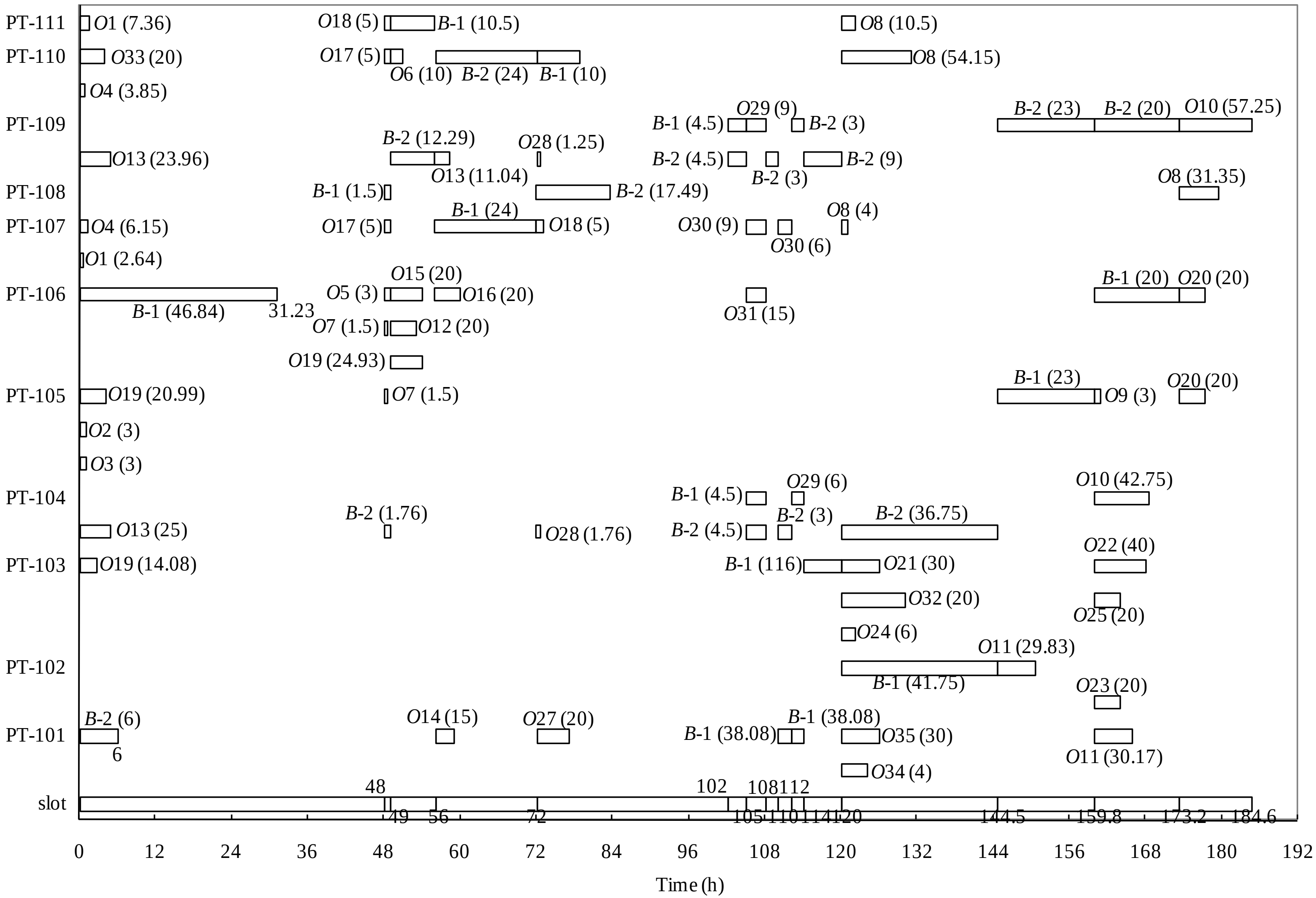

Figure 5.13b The delivery schedule of Figure 5.13a revised by the proposed algorithm where PT-109 delivers O13 continuously

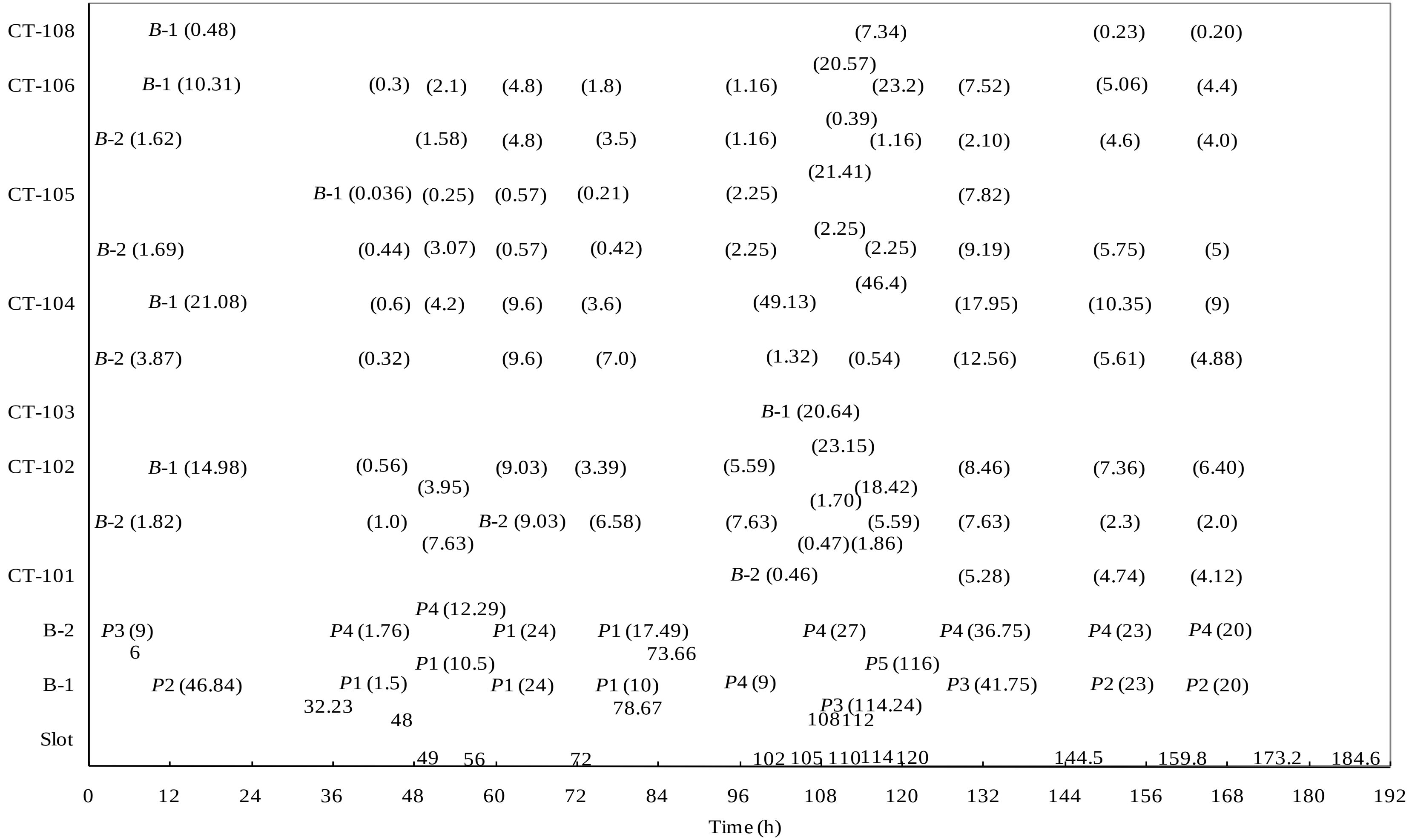

Figure 5.13c Blending schedule for Example 12 (35 orders) from RSPM

3) Tanks deliver multiple orders simultaneously in Figures 5.10-5.13. For instance, PT-106 delivering O2, O3, and O15 during slot 1 in Figure 5.10, PT-110 delivering O1 and O4 during slot 1 in Figure 5.11, PT-103 delivering O15, and O19 during slot 3 in Figure 5.12, and PT-106 delivering O12, O15, and O19 during slot 3 in Figure 5.13b.

4) Multiple tanks deliver a single order simultaneously in Figures 5.10-5.13. For instance, PT-105 and PT-106 delivering O12 during slot 2 in Figure 5.10, PT-107 and PT-108 delivering O8 during slot 5 in Figure 5.11, PT-107 and PT-108 delivering O21 during slot 5 in Figure 5.12, and PT-105 and PT-106 delivering O20 during slot 16 in Figure 5.13b.

5) Multiple blenders (B-1 and B-2) process the same product (P1) simultaneously during slots 4-5 in Figure 5.13c. They again process P4 during slots 6-7.

6) PT-109 delivers O13 intermittently in Figure 5.13a, which the proposed adjustment procedure makes continuous as shown in Figure 5.13b.

5.8 MINLP Formulation

Recall that forcing the blending rate to be constant during a run makes the formulation nonlinear and nonconvex. The proposed adjustment procedure obviated the need to solve MINLPs. To show the effectiveness of the proposed procedure, DICOPT/GAMS and BARON/GAMS are used to solve the nonlinear, nonconvex formulations derived as follows.

Variable F_{bk} ($F_b^L \le F_{bk} \le F_b^U$) is defined as the blending rate of blender b in slot k and impose the following constraints to maintain a single blending rate for each run.

Table 5.11 Solution statistics of various algorithms/codes for SPM for Examples 1-14

Ex	Order	Algorithm	Discrete Variables	Continuous Variables	Constraints	Total CPU Time (s)	Cost (k$)
1	5	DICOPT	100	274	1105	4.8	5149.73
		BARON	100	274	1105	14400*	5169.73
		Ours	100	269	1098	0.83	5149.73
2	10	DICOPT	252	603	2854	151	4169.81
		BARON	252	603	2854	14400*	3658.11
		Ours	252	598	2847	48.6	3658.11
3	12	DICOPT	315	638	3206	1020	3159.12
		BARON	315	638	3206	14400*	3159.12
		Ours	315	633	3199	178	3159.12
4	15	DICOPT	372	728	3830	13431	4556.67
		BARON	372	728	3830	14400*	5974.47
		Ours	372	723	3823	545	4556.67
5	15	DICOPT	372	728	3842	758	4556.67
		BARON	372	728	3842	14400*	4585.88
		Ours	372	723	3835	252	4556.67
6	18	DICOPT	576	956	5447	14400*	5714.02
		BARON	576	956	5447	14400*	5430.09
		Ours	576	950	5438	10813	5213.88
7	20	DICOPT	775	1204	7330	14400*	8491.50
		BARON	775	1204	7330	14400*	N/A
		Ours	775	1197	7319	10822	8100.35
8	20	DICOPT	523	969	6270	14400*	9716.29
		BARON	523	969	6270	14400*	8775.54
		Ours	523	959	6256	10802	8100.35
9	23	DICOPT	1079	1689	12582	14400*	12202.74
		BARON	1079	1689	12582	14400*	13869.70
		Ours	1079	1673	12556	14400*	10781.40
10	25	DICOPT	1503	2229	17066	14400*	N/A
		BARON	1503	2229	17066	14400*	14257.47
		Ours	1503	2209	17032	14400	11327.20
11	30	DICOPT	2328	3242	25521	46800*	N/A
		BARON	2328	3242	25521	46800*	N/A
		Ours	2328	3216	25475	40065	13300.30
12	35	DICOPT	3851	4992	40542	118800*	N/A
		BARON	3851	4992	40542	118800*	N/A
		Ours	3851	4958	40480	118800*	15367.18
13	40	DICOPT	4142	5681	50226	118800*	17814.64
		BARON	4142	5681	50226	118800*	N/A
		Ours	4142	5630	50133	118800*	16239.39
14	45	DICOPT	4830	6493	57756	118800*	N/A
		BARON	4830	6493	57756	118800*	N/A
		Ours	4830	6439	57657	118800*	19780.33

Note: CPU time limit for MIP of DICOPT is set at 10800 s for Exs 1-10, 36000 s for Ex 11, and 108000 s for Exs 12-14
CPU time limit for SPM of ours is set at 10800 s for Exs 1-10, 36000 s for Ex 11, and 108000 s for Exs 12-14
CPU time limit for RSPM of ours is set at 3600 s for Exs 1-10, and 10800 s for Exs 11-14
Total CPU time limit of DICOPT, BARON, and ours is set at 14400 s for Exs 1-10, 46800 s for Ex 11, and 118800 s for Exs 12-14
* Reached total CPU time limit N/A: No feasible solution

Table 5.12 Solution statistics of various algorithms/codes for MPM for Examples 1-14

Ex	Order	Algorithm	Discrete Variables	Continuous Variables	Constraints	Total CPU Time (s)	Cost (k$)
1	5	DICOPT	130	334	1393	78.7	5149.73
		BARON	130	334	1393	14400*	5149.73
		Ours	130	328	1384	12.6	5149.73
2	10	DICOPT	329	735	3609	798	3658.11
		BARON	329	735	3609	14400*	3678.11
		Ours	329	729	3600	137	3658.11
3	12	DICOPT	411	775	4057	531	3179.12
		BARON	411	775	4057	14400*	3179.12
		Ours	411	769	4048	82.7	3179.12
4	15	DICOPT	486	883	4849	4074	4576.67
		BARON	486	883	4849	14400*	4717.13
		Ours	486	877	4840	800	4556.67
5	15	DICOPT	486	883	4849	440	4556.67
		BARON	486	883	4849	14400*	4576.67
		Ours	486	877	4852	95.6	4556.67
6	18	DICOPT	576	956	5447	14400*	5330.16
		BARON	576	956	5447	14400*	5862.26
		Ours	576	950	5438	10801	5213.88
7	20	DICOPT	1219	1735	11155	14400*	8100.35
		BARON	1219	1735	11155	14400*	10025.57
		Ours	1219	1725	11138	10805	8100.35
8	20	DICOPT	1159	1797	12878	14400*	12495.55
		BARON	1159	1797	12878	14400*	9492.56
		Ours	1159	1779	12848	10821	8329.13
9	23	DICOPT	1423	2127	16282	14400*	N/A
		BARON	1423	2127	16282	14400*	12814.06
		Ours	1423	2107	16248	11092	10636.40
10	25	DICOPT	1685	2458	19006	14400*	N/A
		BARON	1685	2458	19006	14400*	18223.18
		Ours	1685	2436	18968	13215	11649.17
11	30	DICOPT	2328	3242	25521	46800*	N/A
		BARON	2328	3242	25521	46800*	N/A
		Ours	2328	3216	25475	36402	13357.69
12	35	DICOPT	4099	5294	43112	118800*	N/A
		BARON	4099	5294	43112	118800*	N/A
		Ours	4099	5258	43046	118800*	15326.00
13	40	DICOPT	4142	6290	53404	118800*	N/A
		BARON	4142	6290	53404	118800*	N/A
		Ours	4142	5966	53305	109051	16282.08
14	45	DICOPT	4830	6493	57756	118800*	N/A
		BARON	4830	6493	57756	118800*	N/A
		Ours	4830	6439	57657	118800*	19207.48

Note: CPU time limit for MIP of DICOPT is set at 10800 s for Exs 1-10, 36000 s for Ex 11, and 108000 s for Exs 12-14
CPU time limit for MPM of ours is set at 10800 s for Exs 1-10, 36000 s for Ex 11, and 108000 s for Exs 12-14
CPU time limit for RMPM of ours is set at 3600 s for Exs 1-10, and 10800 s for Exs 11-14
Total CPU time limit of DICOPT, BARON, and ours is set at 14400 s for Exs 1-10, 46800 s for Ex 11, and 118800 s for Exs 12-14
* Reached total CPU time limit N/A: No feasible solution

$$Q_{bk} \leq F_{bk} \cdot SL_k \qquad 0 < k \leq K \tag{5.55a}$$

$$Q_{bk} \geq F_{bk} SL_k - M_b(v_{b0k} + xe_{bk}) \qquad 0 < k \leq K \tag{5.55b}$$

$$F_{b(k+1)} \geq F_{bk} - F_b^U xe_{bk} \qquad 0 < k < K \tag{5.56a}$$

$$F_{b(k+1)} \leq F_{bk} + F_b^U xe_{bk} \qquad 0 < k < K \tag{5.56b}$$

Eq. 5.17 is replaced by eqs. 5.55-5.56 in SPM and MPM to get MINLP-SPM and MINLP-MPM. For a fair comparison, the same CPU time is allowed for DICOPT/GAMS and BARON/GAMS, as what did for the proposed algorithm. For Examples 6-14, the time limits had been set for SPM/MPM and RSPM/RMPM. The same limits are also set for the MIPs of DICOPT/GAMS as shown in Tables 5.11-5.12.

Tables 5.11-5.12 show the solution statistics for Examples 1-14. For Example 1, the proposed procedure needs only 0.83 CPU s for the optimal solution of \$ 5149.73K, but DICOPT/GAMS needs 4.8 CPU s. BARON/GAMS obtains a suboptimal solution of \$ 5169.73K after 14400 CPU s. For Example 2, the proposed procedure obtains the optimal solution of \$ 3658.11K within 48.6 CPU s, but BARON/GAMS needs 14400 CPU s. DICOPT/GAMS gets a worse solution of \$ 4169.81K in 151 CPU s. For Examples 3-5, DICOPT/GAMS does get the optimal solutions, but requires an order of magnitude longer solution times compared to the proposed procedure. For instance, DICOPT/GAMS takes 13431 CPU s for Example 4 versus only 545 CPU s for the proposed procedure. Interestingly, BARON/GAMS also reaches the optimal solution for Example 3, but needs 14400 CPU s. For the remaining examples (Examples 6-14), the proposed approach always obtains better solutions than both DICOPT/GAMS and BARON/GAMS within the allocated CPU time. For instance, the proposed approach finds a solution of $ 8100.35K for Example 8, while DICOPT/GAMS obtains $ 9716.29K, and BARON/GAMS gets $ 8775.54K. Moreover, the proposed approach obtains a solution of $ 11327.20K for Example 10, while BARON/GAMS gets $ 14257.47K, and DICOPT/GAMS cannot obtain a feasible solution.

5.9 Summary

A general slot-based continuous-time model was developed for integrated scheduling of gasoline blending and distribution operations in a refinery. The model incorporates many real-life operating features and policies such as multiple parallel non-identical blenders, piecewise constant component input flows and qualities, multi-product orders with multiple delivery dates, multi-purpose product tanks, minimum lengths for blending runs, constant blending rate in a run, common transfer policies, blending and storage transitions, etc. Although the problem is inherently non-convex and nonlinear, an ingenious schedule adjustment procedure that requires only MILP solutions was proposed. On 14 test problems of varying sizes and features, the proposed procedure obtained the same or better solutions than commercial solvers such as DICOPT/GAMS and BARON/GAMS. Furthermore, it needed an order of magnitude shorter solution times than DICOPT/GAMS and BARON/GAMS. Since the developed model is general, it can easily be simplified for assumptions such as identical blenders, one blender per product at a time, etc, which are common in existing work. Much further work is needed, as the developed model still cannot optimally solve truly large problems involving more than 30 orders within reasonable time. In the next Chapter, a novel unit-slot formulation is developed to improve the efficiency.

CHAPTER 6

INTEGRATED BLENDING AND DISTRIBUTION OF GASOLINE USING UNIT SLOTS

6.1 Introduction

As discussed in Chapter 5, gasoline is one of the most profitable products of a refinery and can account for as much as 60-70% of total profit. Optimal scheduling of gasoline blending and distribution operation using advanced techniques of mixed-integer programming can avoid ship demurrage, improve order delivery and customer satisfactions, minimize quality give-aways, reduce transitions and slop generation, exploit low-quality cuts, and reduce inventory costs. Therefore, scheduling of gasoline blending and distribution is very crucial.

In Chapter 5, a general global slot-based continuous time formulation for simultaneous treatment of recipe, blending, scheduling, and distribution was developed. Many real-life operation features such as multiple-purpose product tanks, identical or non-identical blenders in parallel, one blender charging at most one product tank at a time were incorporated. Moreover, the blending rate was imposed to be constant during a run, and a schedule adjustment procedure was developed to avoid solving nonconvex MINLP. However, the proposed model in Chapter 5 still needs large time to obtain a feasible solution with large relative gap especially for larger-size examples because of the low RMIPs and slow improvement of RMIPs. In addition, enough component inventories through the entire scheduling horizon were assumed to avoid more slots and improve the efficiency of their model. However, it is not realistic.

Therefore, in this Chapter, a continuous-time formulation using unit slot is developed to improve the efficiency of simultaneous treatment of recipe, blending, scheduling and distribution. All the real-life operation features of Chapter 5 are incorporated into the new formulation. The assumption of enough component inventories through the entire scheduling horizon is relaxed. The approach of schedule adjustment procedure proposed in Chapter 5 is used to avoid solving nonconvex MINLP problem. Fourteen examples from Chapter 5 are used to evaluate the new formulation and compare with process-slot formulation proposed in Chapter 5.

This Chapter is structured as follows. In the next section, the problem is defined in detail. In section 6.3, decision variables and mathematic formulations (single and multiple periods) are defined in detail. Following that, the schedule adjustment procedure in Chapter 5 is followed and revised to solve the nonlinearity arising from the constant blending rate in a run. After that, the proposed unit-slot based model is compared with process-slot based one in Chapter 5 using fourteen examples.

6.2 Problem Statement

Figure 5.1 shows a gasoline blending and distribution unit (GBDU) in a typical refinery. It involves I component tanks (i = 1, 2, …, I) , B blenders (b = 1, 2, …, B), J product tanks (j = 1, 2, …, J), and some lifting ports. I components (i = 1, 2, …, I) are used to make P possible products (p = 1, 2, …, P). Each component i with known quality or specification is stored in its own dedicated component tank i. If one component is stored in several component tanks, then these tanks can be treated as one tank with no loss of generality. The flow profiles over time of various components into respective component tanks are known a priori.

The components from various component tanks with various property indices such as ON (Octane Number), RVP (Reid Vapor Pressure), etc. are fed to the blenders in some proportions to make various products of desired quality at various times. The blenders are operating in semi-continuous mode and process products one at a time. The products produced from these blenders flow to assigned product tanks that may hold different products over time. The products from product tanks are loaded into vehicles or ships at appropriate times for the delivery of various orders.

In the beginning, *O* orders (*o* = 1, 2, …, *O*) with delivery time windows $[DD_o^L, DD_o^U]$ are needed to fulfill during the coming scheduling horizon [0, *H*]. Each order can be assumed to involve a single product although it may involve multiple products because each multi-product order can be broken into several single-product orders. Any delivery after DD_o^U incurs a demurrage cost (DM_o).

With this, the gasoline blending and distribution problem addressed in this Chapter can be stated as:

Given:

1. A scheduling horizon [0, *H*].
2. *I* components and profiles of their property indices.
3. *I* component tanks, their initial inventories, limits on their holdups, flow profiles of feeds into the tanks, and limits on the flows out of the tanks.
4. *B* blenders, the products that each blender can process, the minimum blend times of these products, and limits on their blending rates.
5. *P* products and specification limits on their property indices.
6. *J* product tanks, the products that each tank can store, limits on their holdups, the products and holdups at time zero, and delivery (lifting) rates for various products.
7. *O* orders, their constituent products, amounts, and delivery time windows.
8. Revenues from product sales, component costs, inventory costs (for components and products), and demurrage costs for orders.

Determine:

1. Which blender feeds which component tank in what times, and at what flow rates.
2. Which product is produced in which blender in what times, and at what rates.
3. Which product tank receives which product from which blender in what times, and at what flow rates.
4. Which product tank delivers which order at what times, and in which amounts.
5. The inventory profiles of various tanks (component and product).

Assuming:

1. Flow rate profile of each component from the upstream process and component quality are piecewise constant.
2. Perfect mixing in each blender.
3. No changeover times between products for both blenders and product tanks.
4. Each order involves only one product.
5. Each order is completed during the scheduling horizon.

Subject to the operating rules:

1. A blender can process at most one product at any time. Once it begins processing a product, it must operate for some minimum time, before it can switch to another product.
2. A blender can feed at most one product tank at any time.
3. A product tank cannot receive and deliver a product simultaneously.

Allowing:

1. A component tank may feed multiple blenders, and multiple component tanks may feed one blender at the same time.
2. A blender may feed multiple product tanks during the scheduling horizon.
3. A product tank may deliver multiple orders at the same time.
4. An order may be delivered by multiple tanks at the same time.

The objective is to minimize the total operating cost including component cost, transition cost in both blenders and product tanks, and demurrage cost.

6.3 Motivation

As discussed before, a global process (synchronous) slot-based continuous-time formulation is developed in Chapter 5 for simultaneous treatment of recipe, blending, scheduling, and distribution. These process (synchronous) slots are common or shared for all component tanks, blenders, and product tanks. In other words, when a component tank feeds blenders, a blender feeds a product tank, and a product tank delivers orders in a slot *k*, these feeds and deliveries must start at the same time or time $T_{(k-1)}$. Consider the example (Figure 6.1). Figure 6.1 gives a schedule with six process (synchronous) slots. Blender B-101 withdraws components from component tanks CT-101 and CT-103 to produce P1 and transfers P1 to product tank PT-101 during slots 1 and 2. B-102 consumes components from CT-102 and CT-104 to produce P2, and feeds PT-102 during slots 2 and 3. PT-101 delivers orders O1, O2, and O3 in slots 4-6. Further investigation of Figure 6.2 indicates that blenders B-101 and B-102 do not need to start processing products P1 and P2 at the same time, since they withdraw components from different component tanks and feed different product tanks. Moreover, tank PT-101 can deliver O1-O3 in one slot with different start times because a product tank can deliver an order at any time within order delivery window. By this analysis, a schedule (Figure 6.2) is obtained with 2 slots on CT101-CT-104, B-101, B-102, PT-101 and PT-102. These slots are distinct and independent for each component tank, blender, and product tank. For instance, B-101 starts to withdraw components from CT-101 and CT-103 at 0 in slot 1, while B-102 starts at 4 h with the same index (slot 1). With the reduction of the number of slots and the number of binary variables, the computational time can be reduced and the efficiency of the model can be improved simultaneously.

Therefore, it motivates to develop a new continuous-time formulation for this gasoline blending and distribution problem using unit (asynchronous) slot. Some rigorous constraints are developed to ensure correct inventory profiles of any component tank through the entire scheduling horizon. Thus, the assumption of sufficient components through the entire scheduling horizon in Chapter 5 can be relaxed.

6.4 MILP Formulation

The scheduling horizon $[0, H]$ is divided into K ($k = 1, 2, \ldots, K$) contiguous slots (Figure 6.3) and denote the time before the horizon start by slot zero ($k = 0$). The component tanks, blenders, and product tanks are defined as various units. Each unit has K slots, which are not synchronized (Lim & Karimi, 2003) across the units. In other words, they are unit slots (Liu & Karimi, 2008), where the start/end times and slot lengths of a given slot k need not be the same across all units. Let T_{qk} ($k = 0, 1, 2, \ldots, K$; $T_{q0} \geq 0$, $T_{qK} \leq H$) denote the end time of slot k on unit q, where q becomes i for a component tank, becomes b for a blender, and becomes j for a product tank. Slot k on unit q starts at $T_{q(k-1)}$ and ends at T_{qk}. Since the slots are asynchronous, T_{ik}, T_{bk}, and T_{jk} may vary with units. Thus,

$$T_{qk} \geq T_{q(k-1)} \qquad 1 \leq k \leq K \tag{6.1}$$

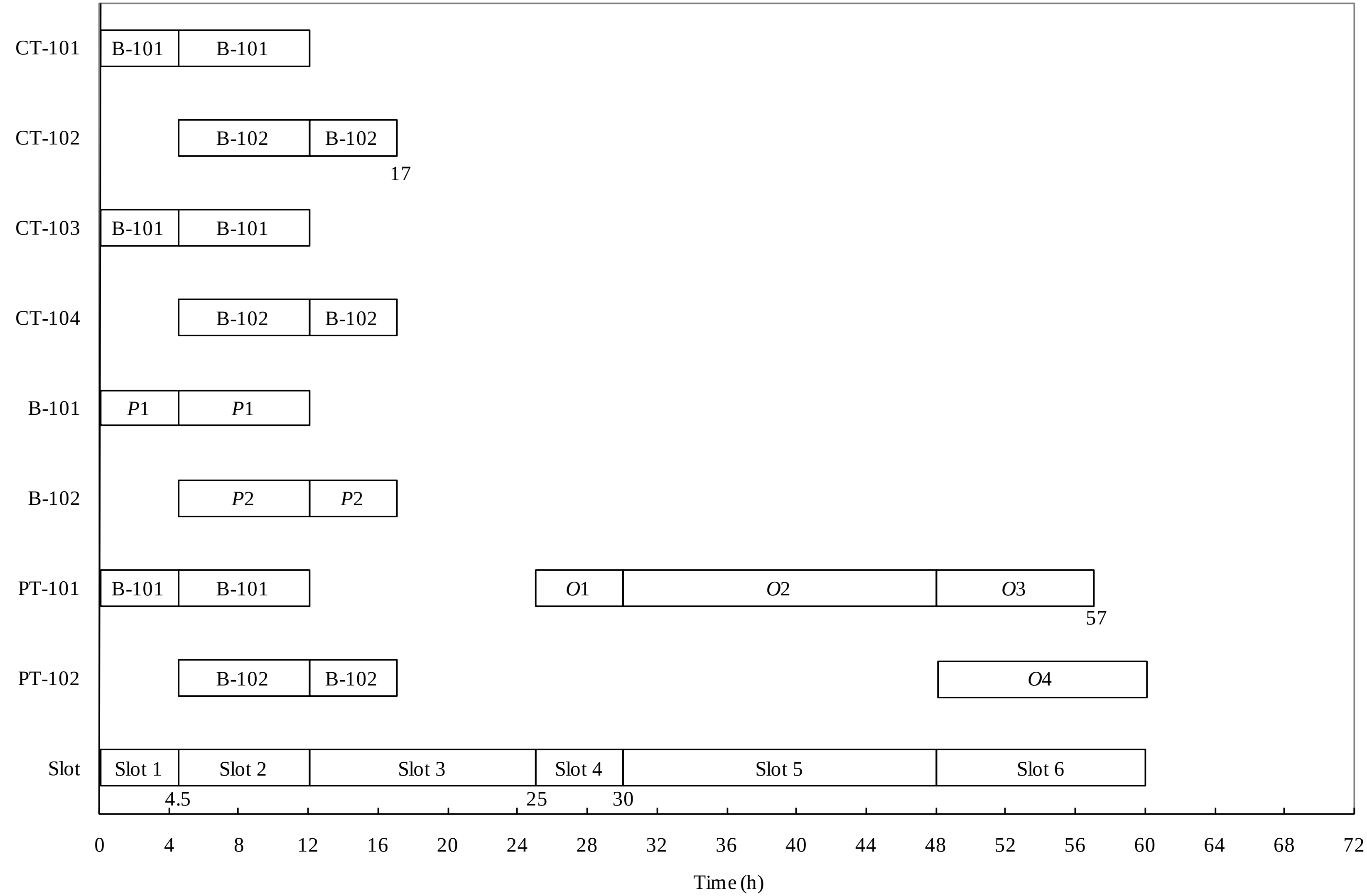

Figure 6.1 A schedule using process slots

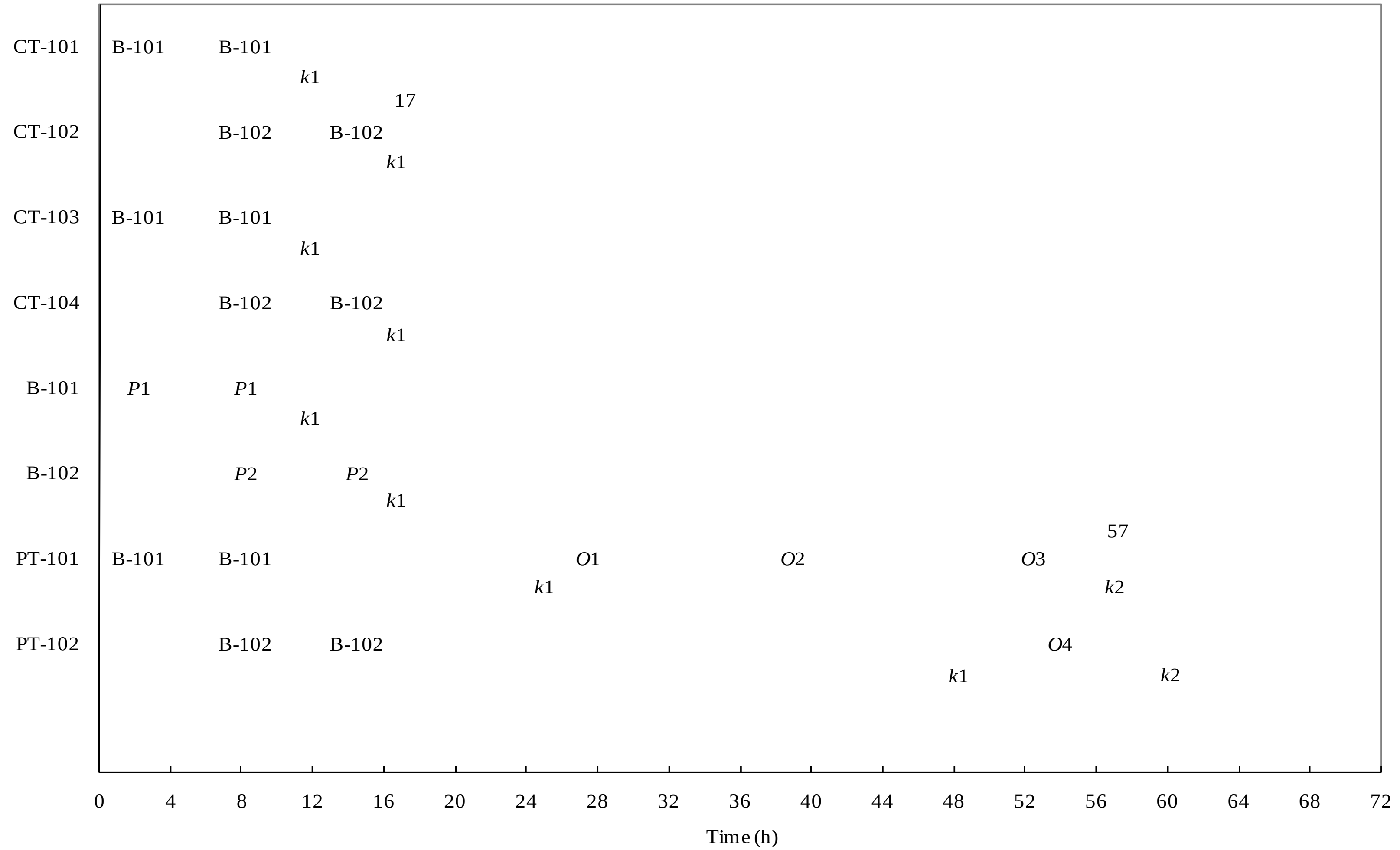

Figure 6.2 The schedule using unit slots for Figure 6.1

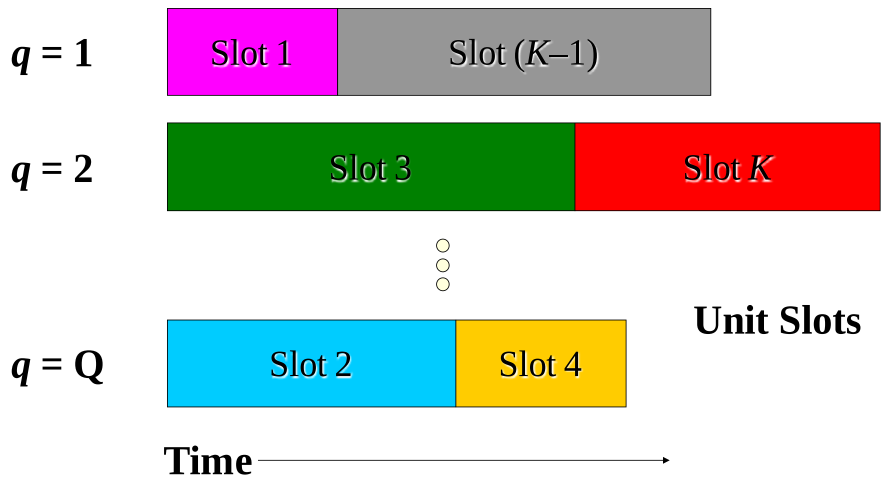

Figure 6.3 Schematic of unit slots design

In this Chapter, the following sets are used.

BP = {(*b*, *p*) | blender *b* can process product *p*}

BJ = {(*b*, *j*) | blender *b* can feed product tank *j*}

JP = {(*j*, *p*) | product tank *j* can hold product *p*}

JO = {(*j*, *o*) | product tank *j* may deliver order *o*}

OP = {(*o*, *p*) | order *o* is for product *p*}

The real operation (blending, storing, delivering, etc.) on a unit is assumed to always begin at the start of a slot, but may end at any time within the slot. In other words, the idle time, if any, is always towards the end of the slot. Unless otherwise indicated, an index takes all its legitimate values in all the expressions or constraints in the proposed formulation.

6.4.1 Blending and Storage

At any time, a blender must be either running or idle. When running, it must be connected to a product tank. If idle, then it is connected to a dummy product tank (j = 0). Thus, there are J real product tanks (j = 1, 2, …, J) and one dummy (j = 0) product tank. When a blender b is processing, it must use some components and send a product at the same time. Suppose a blender b is consuming components in its own slot k, whereas a component tank i is feeding this blender b in its own slot k'. Three scenarios are possible: $k' < k$, $k' = k$, and $k' > k$. For $k' < k$, additional slots on component tank i can be simply introduced to make $k' = k$. For $k' > k$, the same can be done on blender b. In other words, with no loss of generality, if a blender b is consuming components from a component tank i at any time, then the unit slots corresponding to that time on both blender b and component tank i must have the same index. The same holds true for a blender and a product tank, and a product tank and an order delivery. Thus, the following binary variables are defined to model the blending and storage operations in the GBDU.

$$y_{ibk} = \begin{cases} 1 & \text{If component tank } i \text{ feeds blender } b \text{ during slot } k \\ 0 & \text{Otherwise} \end{cases} \qquad 0 \le k \le K$$

$$v_{bjk} = \begin{cases} 1 & \text{If blender } b \text{ feeds product tank } j \text{ during slot } k \\ 0 & \text{Otherwise} \end{cases}$$

$$(b, j) \in \boldsymbol{BJ},\ 0 \le j \le J,\ 0 \le k \le K$$

$$u_{jpk} = \begin{cases} 1 & \text{If product tank } j \text{ holds product } p \text{ during slot } k \\ 0 & \text{Otherwise} \end{cases}$$

$$(j, p) \in \boldsymbol{JP},\ 0 < j \le J,\ 0 \le k \le K$$

$$x_{bpk} = \begin{cases} 1 & \text{If blender } b \text{ processes product } p \text{ during slot } k \\ 0 & \text{Otherwise} \end{cases} \qquad (b, p) \in \boldsymbol{BP},\ 0 \le k \le K$$

$$xe_{bk} = \begin{cases} 1 & \text{If blender } b \text{ ends its current run during slot } k \\ 0 & \text{Otherwise} \end{cases} \qquad 0 \le k < K$$

$$ue_{jk} = \begin{cases} 1 & \text{If product tank } j \text{ switches products at the end of slot } k \\ 0 & \text{Otherwise} \end{cases} \qquad 1 \le j \le J,\ 0 \le k < K$$

Note that even though the same slot index is used for all units in the above, the slots refer to appropriate units. The same set of variables as Chapter 5 is used, and v_{bjk} (j = 1, 2, …, J) is treated as binary, v_{b0k} as 0-1 as continuous, and x_{bpk} is proved to be binary automatically and hence can be treated as 0-1 continuous. Note that proper values must be assigned for x_{bp0} based on the product that blender b was processing before time zero and u_{jp0} based on what was inside tank j before time zero. The following constraints were developed in Chapter 5 for some of the above variables, which is also used in this formulation.

$$\sum_{j=0}^{J} v_{bjk} = 1 \qquad (b, j) \in \boldsymbol{BJ},\ 1 \le k \le K \tag{6.2}$$

$$\sum_{p=1}^{P} u_{jpk} = 1 \qquad (j, p) \in \boldsymbol{JP},\ 0 < j \le J,\ 1 \le k \le K \tag{6.3}$$

$$\sum_{p=1}^{P} x_{bpk} + v_{b0k} = 1 \qquad (b, p) \in \boldsymbol{BP},\ 1 \le k \le K \tag{6.4}$$

$$x_{bpk} \ge u_{jpk} + v_{bjk} - 1$$

$$(b, p) \in \boldsymbol{BP},\ (b, j) \in \boldsymbol{BJ},\ (j, p) \in \boldsymbol{JP},\ 1 \le j \le J,\ 1 \le k \le K \tag{6.5a}$$

$$u_{jpk} \ge x_{bpk} + v_{bjk} - 1$$

$$(b, p) \in \boldsymbol{BP},\ (b, j) \in \boldsymbol{BJ},\ (j, p) \in \boldsymbol{JP},\ 1 \le j \le J,\ 1 \le k \le K \tag{6.5b}$$

$$xe_{bk} \ge x_{bpk} - x_{bp(k+1)} \qquad (b, p) \in \boldsymbol{BP},\ 0 \le k < K \tag{6.6a}$$

$$xe_{bk} \ge x_{bp(k+1)} - x_{bpk} \qquad (b, p) \in \boldsymbol{BP},\ 0 \le k < K \tag{6.6b}$$

$$xe_{bk} + x_{bpk} + x_{bp(k+1)} \le 2 \qquad (b, p) \in \boldsymbol{BP},\ 0 \le k < K \tag{6.7}$$

$$ue_{jk} \ge u_{jpk} - u_{jp(k+1)} \qquad (j, p) \in \boldsymbol{JP},\ 0 \le k < K,\ 1 \le j \le J \tag{6.8a}$$

$$ue_{jk} \ge u_{jp(k+1)} - u_{jpk} \qquad (j, p) \in \boldsymbol{JP},\ 0 \le k < K,\ 1 \le j \le J \tag{6.8b}$$

$$\sum_{b=1}^{B} \sum_{k=1}^{K-1} xe_{bk} \ge NP - B \tag{6.9}$$

where, NP is the number of distinct products that must be processed by blenders during the scheduling horizon. Since a penalty will be imposed for product changeovers in the objective, it is no need to force $ue_{jk} = 0$.

If blender b is idle during slot k, then component tank i cannot be feeding b during k. Similarly, if it is not idle, then at least one component tank must be feeding it.

$$v_{b0k} + y_{ibk} \le 1 \qquad 1 \le k \le K \qquad (6.10a)$$

$$v_{b0k} + \sum_{i=1}^{I} y_{ibk} \ge 1 \qquad 1 \le k \le K \qquad (6.10b)$$

6.4.2 Run Lengths and Product Quality

Since a blending operation is allowed to end at anytime during a slot, the actual blending length in a slot must be known. Let BL_{bk} be the time for which blender b processes real products ($p > 0$) in slot k. BL_{bk} was not needed to use in Chapter 5, as process slots were used. Now, This BL_{bk} must equal slot length, unless the current run is ending during slot k. In other words,

$$BL_{bk} \le T_{bk} - T_{b(k-1)} \qquad 1 \le k \le K \qquad (6.11a)$$

$$BL_{bk} + H \cdot (v_{b0k} + ue_{bk}) \ge T_{bk} - T_{b(k-1)} \qquad 1 \le k \le K \qquad (6.11b)$$

Clearly, if blender b is idle during slot k, then the blending length must be zero.

$$BL_{bk} \le H \cdot (1 - v_{b0k}) \qquad 1 \le k \le K \qquad (6.12)$$

Next, as done in Chapter 5, RL_{bk} ($0 \le k \le K$) is defined as the length of the current run on blender b at the end of slot k, if the run does not end during slot k, and zero otherwise. In other words,

$$RL_{bk} = \begin{cases} \text{Current run length} & \text{If the current run of blender } b \text{ does not end during slot } k \\ 0 & \text{Otherwise} \end{cases}$$

Thus, $RL_{b0} = 0$, if a run has ended at time zero, otherwise it is the current run length at time zero. To compute RL_{bk},

$$RL_{bk} \le RL_{b(k-1)} + BL_{bk} \qquad 1 \le k \le K \qquad (6.13)$$

$$RL_{bk} \le H(1 - xe_{bk}) \qquad 0 \le k \le K \qquad (6.14)$$

Then, to ensure a minimum length (RL_{bp}^{L}) for each blend run,

$$RL_{b(k-1)} + BL_{bk} + RL_{b}^{L}(1 - xe_{bk}) \geq \sum_{p=1}^{P} RL_{bp}^{L} x_{bpk} \qquad (b, p) \in \boldsymbol{BP},\ 1 \leq k < K \qquad (6.15a)$$

$$RL_{b(k-1)} + BL_{bk} \geq \sum_{p=1}^{P} RL_{bp}^{L} x_{bpk} \qquad (b, p) \in \boldsymbol{BP},\ k = K \qquad (6.15b)$$

where, $RL_{b}^{L} = \max_{p}\left(RL_{bp}^{L}\right)$

Blending needs components and must give on-spec products. Let q_{ibk} be the volume that component tank i feeds blender b during slot k, G_{bjk} be the volume that blender b feeds product tank j during slot k and Q_{bk} as the volume processed in blender b during slot k. The volume processed in blender b in slot k is given by,

$$Q_{bk} = \sum_{i=1}^{I} q_{ibk} = \sum_{j \in \boldsymbol{BJ}}^{J} G_{bjk} \qquad 1 \leq k \leq K \qquad (6.16a,b)$$

If component tank i (blender b) does not feed blender b (product tank j) in slot k, then q_{ibk} (G_{bjk}) must be zero.

$$q_{ibk} \leq M_{b} \cdot y_{ibk} \qquad 1 \leq k \leq K \qquad (6.17)$$

$$G_{bjk} \leq M_{b} \cdot v_{bjk} \qquad 1 \leq k \leq K \qquad (6.18)$$

where, $M_{b} = \min\left\{H \cdot F_{b}^{U}, \max_{j}\left(VP_{j}^{U}\right)\right\}$ and VP_{j}^{U} is the capacity of tank j.

Lastly, Q_{bk} must respect the maximum (F_{b}^{U}) and minimum (F_{b}^{L}) processing rates of blender b.

$$F_{b}^{L} \cdot BL_{bk} \leq Q_{bk} \leq F_{b}^{U} \cdot BL_{bk} \qquad 1 \leq k \leq K \qquad (6.19a,b)$$

As noted in Chapter 5, eq. 6.19 allows the blending rate to vary from slot to slot during a run, which is not real practice. Since enforcing this makes the formulation nonlinear and nonconvex, the procedure in Chapter 5 is used to deal with it.

For product quality, twelve most commonly used gasoline indices and their additive bases were identified in Chapter 5. The following constraints were proposed to ensure on-spec products.

$$\sum_{i=1}^{I} q_{ibk}\theta_{is} \geq Q_{bk}\theta_{ps}^{L} - M_b \cdot \left\{\theta_{ps}^{L} - \min_p\left(\theta_{ps}^{L}\right)\right\} \cdot (1 - x_{bpk}) \qquad (b, p) \in \boldsymbol{BP},\ 1 \leq k \leq K \qquad (6.20a)$$

$$\sum_{i=1}^{I} q_{ibk}\theta_{is} \leq Q_{bk}\theta_{ps}^{U} + M_b \cdot \left\{\max_p\left(\theta_{ps}^{U}\right) - \theta_{ps}^{U}\right\} \cdot (1 - x_{bpk}) \qquad (b, p) \in \boldsymbol{BP},\ 1 \leq k \leq K \qquad (6.20b)$$

$$\sum_{i=1}^{I} q_{ibk}\rho_i\theta_{is} \geq \left(\sum_{i=1}^{I} q_{ibk}\rho_i\right)\theta_{ps}^{L} - M_b \cdot \left\{\theta_{ps}^{L} - \min_p\left(\theta_{ps}^{L}\right)\right\} \cdot \rho_{\max} \cdot (1 - x_{bpk})$$

$$(b, p) \in \boldsymbol{BP},\ 1 \leq k \leq K \qquad (6.21a)$$

$$\sum_{i=1}^{I} q_{ibk}\rho_i\theta_{is} \leq \left(\sum_{i=1}^{I} q_{ibk}\rho_i\right)\theta_{ps}^{U} + M_b \cdot \left\{\max_p\left(\theta_{ps}^{U}\right) - \theta_{ps}^{U}\right\} \cdot \rho_{\max} \cdot (1 - x_{bpk})$$

$$(b, p) \in \boldsymbol{BP},\ 1 \leq k \leq K \qquad (6.21b)$$

$$Q_{bk}r_{pi}^{L} - M_b \cdot \left\{r_{pi}^{L} - \min_p\left(r_{pi}^{L}\right)\right\}(1 - x_{bpk}) \leq q_{ibk} \leq Q_{bk}r_{pi}^{U} + M_b \cdot \left\{\max_p\left(r_{pi}^{U}\right) - r_{pi}^{U}\right\}(1 - x_{bpk})$$

$$(b, p) \in \boldsymbol{BP},\ 1 \leq k \leq K \quad (6.22a,b)$$

where, θ_{is} is the known blending index for a property *s* of component *i*, ρ_i is the density of component *i*, [$\theta_{ps}^{L}, \theta_{ps}^{U}$] are the desired limits on property *s* of product *p*, and ρ_{max} is the maximum possible density among all products, eq. 6.20 is for volume-based indices, eq. 6.21 for weight-based indices, and the volume fraction of component *i* in product *p* satisfies limits [r_{pi}^{L}, r_{pi}^{U}].

6.4.3 Order Delivery

For modeling order delivery operations, one binary variable (z_{jok}) was defined in Chapter 5.

$$z_{jok} = \begin{cases} 1 & \text{If product tank } j \text{ delivers order } o \text{ during slot } k \\ 0 & \text{Otherwise} \end{cases}$$

$$(j, o) \in \boldsymbol{JO},\ 0 < j \leq J,\ 1 \leq k \leq K$$

Related to the above, the following constraints were presented in Chapter 5, which remain the same in the model as well.

$$\sum_{j=1}^{J}\sum_{k=1}^{K} z_{jok} \geq 1 \qquad (j, o) \in \boldsymbol{JO} \qquad (6.23)$$

$$v_{bjk} + z_{jok} \leq 1 \qquad (b, j) \in \boldsymbol{BJ},\ (j, o) \in \boldsymbol{JO},\ 0 < j \leq J,\ 1 \leq k \leq K \qquad (6.24)$$

$$z_{jok} \leq u_{jp(k-1)} \qquad (p, j) \in \boldsymbol{PJ},\ (j, o) \in \boldsymbol{JO},\ (o, p) \in \boldsymbol{OP},\ 1 \leq j \leq J,\ 1 \leq k \leq K \qquad (6.25)$$

$$z_{jok} \leq u_{jpk} \qquad (p, j) \in \boldsymbol{PJ},\ (j, o) \in \boldsymbol{JO},\ (o, p) \in \boldsymbol{OP},\ 1 \leq j \leq J,\ 1 \leq k \leq K \qquad (6.26)$$

$$z_{jo(k+1)} + ue_{jk} \leq 1 \qquad (j, o) \in \boldsymbol{JO},\ 1 \leq j \leq J,\ 0 \leq k < K \qquad (6.27)$$

$$DQ_{jok} \leq TQ_o \cdot z_{jok} \qquad (j, o) \in \boldsymbol{JO},\ 1 \leq j \leq J,\ 1 \leq k \leq K \qquad (6.28)$$

$$\sum_{o \in \boldsymbol{JO}} DQ_{jok} \leq DR_j^U [T_{jk} - T_{j(k-1)}] \qquad 1 \leq j \leq J,\ 1 \leq k \leq K \qquad (6.29)$$

$$\sum_{j=1}^{J}\sum_{k=1}^{K} DQ_{jok} = TQ_o \qquad (j, o) \in \boldsymbol{JO} \qquad (6.30)$$

where, DQ_{jok} denotes the volume of order *o* delivered by tank *j* during slot *k*, TQ_o is the required amount for order *o*, and DR_j^U is the maximum cumulative rate for all orders.

While a product tank is allowed to deliver multiple orders during a slot as done in Chapter 5, in this case, the delivery can start and end at any time during the slot. For instance, PT-101 delivers O1 at $t = 2$ and O2 at $t = 4$, both in slot *k*, while slot *k* on PT-101 starts at $t = 1$ and ends at $t = 6$. Because process slots were used in Chapter 5, the deliveries began at the same time in the slot. To know the times precisely, ts_{jok} is defined to denote the time at which the delivery of order *o* by product tank *j* begins in slot *k*. Because the delivery rate (DR_{jo}) of order *o* by product tank *j* is constant, the end time for this delivery is given by $ts_{jok} + DQ_{jok}/DR_{jo}$. This delivery is enforced to be entirely within slot *k* on unit *j*.

$$ts_{jok} \geq T_{j(k-1)} \qquad (j, o) \in \boldsymbol{JO},\ 1 \leq j \leq J,\ 1 \leq k \leq K \qquad (6.31a)$$

$$ts_{jok} + \frac{DQ_{jok}}{DR_{jo}} \le T_{jk} \qquad (j, o) \in \boldsymbol{JO},\ 1 \le j \le J,\ 1 \le k \le K \qquad (6.31b)$$

Eq. 6.31 allows order delivery to be intermittent from a tank, which is not realistic. This issue is discussed later. In addition to eq. 6.31, a delivery does not begin before its own time window [DD_o^L, DD_o^U].

$$ts_{jok} \ge DD_o^L z_{jok} \qquad (j, o) \in \boldsymbol{JO},\ 1 \le j \le J,\ 1 \le k \le K \qquad (6.32)$$

If it ends after DD_o^U , then a delivery delay is given by,

$$d_o \ge ts_{jok} + \frac{DQ_{jok}}{DR_{jo}} - DD_o^U - H(1 - z_{jok}) \qquad (j, o) \in \boldsymbol{JO},\ 1 \le j \le J,\ 1 \le k \le K \qquad (6.33)$$

Finally, the delivery delay (d_o) must be bounded by $(H - DD_o^U)$.

6.4.4 Slot Timings on Component Tanks

When using unit slots in the presence of shared resources such as inventories, the main challenge is to relate the timings of different units that share the same resource. The flow in/out of a resource must be ordered chronologically, so that a correct resource profile can be got. To this end, two facts are recalled. First, a blending activity always starts at the beginning of a slot, but may end before a slot ends. Second, a component tank has a constant inflow of material from upstream units.

Consider the example in Figure 6.4, where CT-101 has a capacity of 1.4 kbbl and feeds B-101 only. The feed rate to CT-101 from upstream units is 1.5 kbbl/s, whereas the consumption rate by B-101 is 3.5 kbbl/s. If BT-101 withdraws from CT-101 during [4, 5] h, then the inventory levels are 0.0 at 3 h, 1.0 kbbl at 6 h, and 1.5 kbbl at 4 h, and –0.5 kbbl at 5 h. Clearly, the inventory levels at 3 h, 4 h, 5 h, and 6 h must be computed and checked. However, this cannot be done by using [3, 6] h as one slot on CT-101 and [4, 5] h as another on B-101. The duration [3, 6] h must be broken up into several slots. First [3, 4] h can be merged into a preceding slot. A slot must begin on both B-101 and CT-101 at 4 h. Finally, [4, 6] h can be considered a single slot on both B-101 and CT-101, however the inventory level at 5 h must be checked using a continuous variable.

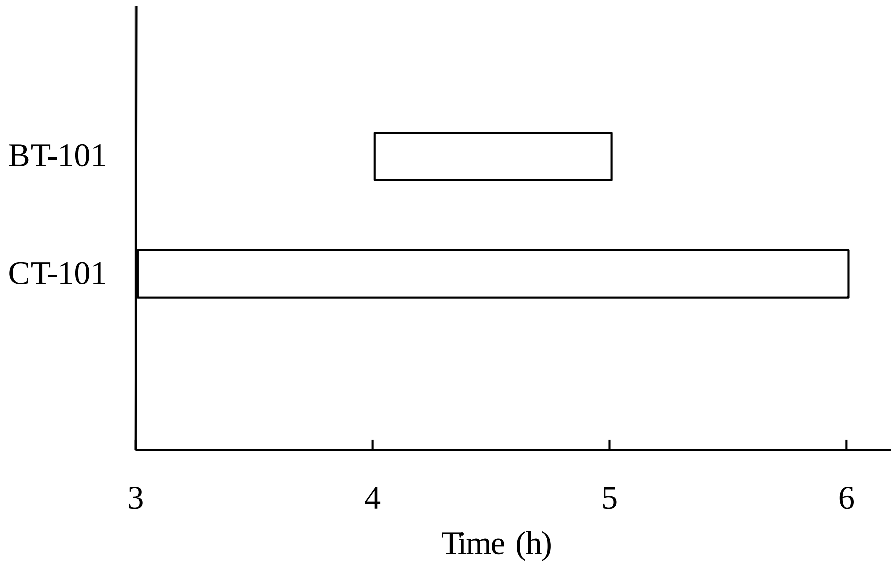

Figure 6.4 An example for inventory violation of a component tank

A clear implication of the above discussion is that whenever a blender withdraws from a component tank during a slot, then the tank must have a matching slot with the same index and the same start time. This is ensured by using the following two constraints.

$$T_{bk} \geq T_{ik} - H \cdot [1 - y_{ib(k+1)}] \qquad 0 \leq k < K \qquad (6.34a)$$

$$T_{bk} \leq T_{ik} + H \cdot [1 - y_{ib(k+1)}] \qquad 0 \leq k < K \qquad (6.34b)$$

For checking the inventory at the end of a blend run, an intermediate point t_{ik} between $T_{i(k-1)}$ and T_{ik} is defined. Then, it demands that the end of blend run must match with this point on the tank by using the following.

$$T_{b(k-1)} + BL_{bk} \leq t_{ik} + H \cdot (1 - y_{ibk}) \qquad 1 \leq k \leq K \qquad (6.35a)$$

$$T_{b(k-1)} + BL_{bk} \geq t_{ik} - H \cdot (1 - y_{ibk}) \qquad 1 \leq k \leq K \qquad (6.35b)$$

Note that eqs. 6.34 and 6.35 ensure that $t_{ik} = T_{ik}$, whenever multiple blenders are withdrawing from the tank during a single slot, and $t_{ik} \le T_{ik}$, whenever only one blender is withdrawing from the tank.

6.4.5 Slot Timings on Product Tanks

The situation with product tanks is much simpler than that with component tanks, because product tanks cannot receive and deliver at the same time. So inflow and outflow can never occur in the same slot. Therefore, if a blender feeds tank *j* during slot *k*, then the start (end) of a slot *k* on a product tank *j* must precede (succeed) the start (end) of slot *k* on the blender.

$$T_{jk} \le T_{bk} + H \cdot [1 - v_{bj(k+1)}] \qquad (b, j) \in \boldsymbol{BJ},\ 1 \le j \le J,\ 0 \le k < K \quad (6.36a)$$

$$T_{jk} \ge T_{bk} + BL_{bk} - H \cdot [1 - v_{bj(k+1)}] \qquad (b, j) \in \boldsymbol{BJ},\ 1 \le j \le J,\ 0 \le k < K \quad (6.36b)$$

6.4.6 Inventory Balance

The timing constraints of sections 6.4.4 and 6.4.5 enable to write the following inventory balances for component and product tanks.

$$VC_{ik} = VC_{i(k-1)} + F_i[T_{ik} - T_{i(k-1)}] - \sum_{b=1}^{B} q_{ibk} \qquad 1 \le k \le K \qquad (6.37)$$

$$VP_{jk} = VP_{j(k-1)} + \sum_{b=1}^{B} G_{bjk} - \sum_{o=1}^{O} DQ_{jok} \qquad (j, o) \in \boldsymbol{JO},\ 1 \le k \le K \qquad (6.38)$$

where, F_i is the constant feed rate to tank *i* from upstream units, VC_{ik} ($VC_i^L \le VC_{ik} \le VC_i^U$) is the inventory of component *i* at T_{ik}, VP_{jk} ($0 \le VP_{jk} \le VP_j^U$) is the holdup in product tank *j* at T_{jk}. Note that the inflow of components into each component tank has been neglected after the schedule is over. In other words, eq. 6.37 does not compute the inventory at the end of the horizon, if the schedule ends before *H*. Furthermore, Eq. 6.37 also does not compute the inventory level at t_{ik}, which must be monitored as well. Therefore,

$$VC_i^L \le VC_{i(k-1)} + F_i[t_{ik} - T_{i(k-1)}] - \sum_{b=1}^{B} q_{ibk} \le V_i^U \qquad 1 \le k \le K \qquad (6.39a,b)$$

Lastly, a product transition cannot occur on a product tank *j*, unless the tank holdup at T_{jk} is zero. Thus,

$$VP_{jk} \le VP_j^U (1 - ue_{jk}) \qquad 1 \le k < K \qquad (6.40)$$

6.4.7 Scheduling Objective

The total operating cost of GBDU from Chapter 5 is given by,

$$\text{TC} = \sum_{i=1}^{I}\sum_{b=1}^{B}\sum_{k=1}^{K} c_i \cdot q_{ibk} + \sum_{b=1}^{B}\sum_{k=1}^{K-1} CB_b \cdot xe_{bk} + \sum_{j=1}^{J}\sum_{k=1}^{K-1} CT_j \cdot ue_{jk} + \sum_{o=1}^{O} DM_o \cdot d_o \qquad (6.41)$$

where, c_i is the price ($ per unit volume) of component *i*, CB_b is the cost ($ per occurrence) of transition on blender *b*, CT_j is the cost ($ per occurrence) of transition in product tank *j*, and DM_o is the demurrage cost ($ per unit time) of order *o*.

This completes the unit-slot-based single period model (SPM) for scheduling blending operations, which comprises eqs. (6.1-6.41). It has two main limitations. First, it allows the blending rates to vary from slot to slot and order delivery to be discontinuous, which are both undesirable in practice. Second, it assumes constant feed rates for components, or it is a single-period formulation. The second limitation or the extension to multiple periods is readily addressed as done in Chapter 5, so modified equations are simply stated. The adjustment procedure in Chapter 5 needs a slight modification to address the first limitation, which is presented as follows.

6.5 Multi-period Extension

Following the approach from Chapter 5, eqs. 6.20-6.21 are modified to ensure on-spec products change for the piecewise-constant profiles of the multi-period scenario.

$$\sum_{i=1}^{I} q_{ibk}\theta_{ist} \ge Q_{bk}\theta_{ps}^L - M_b \cdot \left\{\theta_{ps}^L - \min_p\left(\theta_{ps}^L\right)\right\} \cdot (1 - x_{bpk})$$

$$(b, p) \in \boldsymbol{BP},\ (t, k) \in \boldsymbol{TK},\ 0 < k \le K \qquad (6.42a)$$

$$\sum_{i=1}^{I} q_{ibk}\theta_{ist} \le Q_{bk}\theta_{ps}^{U} + M_b \cdot \left\{\max_p\left(\theta_{ps}^{U}\right) - \theta_{ps}^{U}\right\}\cdot(1-x_{bpk})$$

$$(b, p) \in \boldsymbol{BP},\ (t, k) \in \boldsymbol{TK},\ 0 < k \le K \qquad (6.42\text{b})$$

$$\sum_{i=1}^{I} q_{ibk}\rho_{it}\theta_{ist} \ge \left(\sum_{i=1}^{I} q_{ibk}\rho_{it}\right)\theta_{ps}^{L} - M_b \cdot \left\{\theta_{ps}^{L} - \min_p\left(\theta_{ps}^{L}\right)\right\}\cdot \rho_t^{\max}\cdot(1-x_{bpk})$$

$$(b, p) \in \boldsymbol{BP},\ (t, k) \in \boldsymbol{TK},\ 0 < k \le K \qquad (6.43\text{a})$$

$$\sum_{i=1}^{I} q_{ibk}\rho_{it}\theta_{ist} \le \left(\sum_{i=1}^{I} q_{ibk}\rho_{it}\right)\theta_{ps}^{U} + M_b \cdot \left\{\max_p\left(\theta_{ps}^{U}\right) - \theta_{ps}^{U}\right\}\cdot \rho_t^{\max}\cdot(1-x_{bpk})$$

$$(b, p) \in \boldsymbol{BP},\ (t, k) \in \boldsymbol{TK},\ 0 < k \le K \qquad (6.43\text{b})$$

where, $\boldsymbol{TK} = \{(t, k) \mid \text{slot } k \text{ is in period } t\}$, θ_{ist} denotes the known blending index for a property s of component i during period t, ρ_{it} be the density of component i during period t, and $\rho_t^{\max}$ be the maximum possible density among all products during period t.

Furthermore, eqs. 6.37 and 6.39 change as follows:

$$V_{ik} = V_{i(k-1)} + \sum_{t\in \boldsymbol{TK}} F_{it}[T_{ik} - T_{i(k-1)}] - \sum_{b=1}^{B} q_{ibk} \qquad 1 \le k \le K \qquad (6.44)$$

$$V_i^{L} \le V_{i(k-1)} + \sum_{t\in \boldsymbol{TK}} F_{it}[t_{ik} - T_{i(k-1)}] - \sum_{b=1}^{B} q_{ibk} \le V_i^{U} \qquad 1 \le k \le K \qquad (6.45\text{a,b})$$

where, F_{it} is the flow rate of component i in period t. With this, the multi-period model (MPM) comprises eqs. 6.1-6.19, 6.22-6.36, 6.38, and 6.40-6.45.

6.6 Schedule Adjustment

Recall that a schedule adjustment procedure is necessary to correct for the slot-to-slot variability of blend rates. The main change required in the procedure of Chapter 5 is to handle blend length, which was not present in their formulation. Based on an optimal solution from the basic model, the correct values for run lengths (CRL_{bk}), volumes (CCQ_{bk}), and total volume (TCQ_{bk}) processed by a blender in a run are computed as follows.

$CRL_{bk} = CCQ_{bk} = 0$ if $xe_{bk} = 1$ (6.46a,b)

$CRL_{bk} = CRL_{b(k-1)} + [BL_{bk}]$ if $xe_{bk} = 0$ (6.47a)

$CCQ_{bk} = CCQ_{b(k-1)} + [Q_{bk}]$ if $xe_{bk} = 0$ (6.47b)

$TCQ_{bk} = 0$ if $xe_{bk} = 0$ (6.48a)

$TCQ_{bk} = CCQ_{b(k-1)} + [Q_{bk}]$ if $xe_{bk} = 1$ (6.48b)

Where, $[BL_{bk}]$ and $[Q_{bk}]$ are the values of BL_{bk}, and Q_{bk}, repectively. Using the above parameters, the blending rate (R_{bk}) for each blending run at the slot where it ends is given by,

$$R_{bk} = \frac{CCQ_{b(k-1)} + Q_{bk}}{CRL_{b(k-1)} + BL_{bk}} \text{ for } k \text{ with } xe_{bk} = 1 \,\&\, v_{b0k} = 0 \qquad (6.49)$$

Then, R_{bk} is set for all slots within each run to be the above value. Thus, if a run spans slots 3-6 inclusive, then $R_{bk} = R_{b6}$ for k = 3-5.

Now, to obtain a realistic schedule with the constant blend rates computed above, x_{bpk}, xe_{bk}, and v_{b0k} are fixed and the procedure of Chapter 5 is followed. This gives the flowing reduced formulation, where $[x_{bpk}]$, $[xe_{bk}]$, and $[v_{b0k}]$ respectively denote the optimal values of x_{bpk}, xe_{bk}, and v_{b0k} obtained from the full model.

Eqs. 6.11-6.12 become:

$BL_{bk} = 0$ for (b, k) with $[v_{b0k}] = 1$ (6.50a)

$BL_{bk} = T_{bk} - T_{b(k-1)}$ for (b, k) with $[xe_{bk}] = [v_{b0k}] = 0$ (6.50b)

$BL_{bk} \le T_{bk} - T_{b(k-1)}$ for (b, k) with $[xe_{bk}] = 1$ & $[v_{b0k}] = 0$ (6.50c)

Eqs. 6.13-6.15 become:

$RL_{bk} \le RL_{b(k-1)} + BL_{bk}$ for (b, k) with $[xe_{bk}] = [v_{b0k}] = 0$ (6.51)

$RL_{bk} = 0$ for (b, k) with $[v_{b0k}] = 1$ or $[xe_{bk}] = 1$ & $[v_{b0k}] = 0$ (6.52)

$$RL_{b(k-1)} + BL_{bk} + RL_b^L(1 - xe_{bk}) \ge \sum_{p=1}^{P} RL_{bp}^L x_{bpk}$$

for (b, k) with $[xe_{bk}] = 1$ & $[v_{b0k}] = 0$, $(b, p) \in \boldsymbol{BP}$, $1 \le k < K$ (6.53a)

$$RL_{b(k-1)} + BL_{bk} \ge \sum_{p=1}^{P} RL_{bp}^{L} x_{bpk} \qquad \text{for } (b, k) \text{ with } [v_{b0k}] = 0,\ (b, p) \in \boldsymbol{BP},\ k = K \quad (6.53\text{b})$$

Eq. 6.19 becomes:

$$Q_{bk} = 0 \qquad \text{for } (b, k) \text{ with } [v_{b0k}] = 1 \quad (6.54\text{a})$$

$$Q_{bk} = R_{bk} \cdot BL_{bk} \qquad \text{for } (b, k) \text{ with } [v_{b0k}] = 0 \quad (6.54\text{b})$$

Fixing the values of x_{bpk}, xe_{bk}, and v_{b0k}, eqs. 6.2, 6.5, 6.10, 6.20-6.22, and 6.41 become:

$$\sum_{j=1}^{J} v_{bjk} = 1 - [v_{b0k}] \qquad (b, j) \in \boldsymbol{BJ},\ 0 < j \le J,\ 0 < k \le K \quad (6.55)$$

$$[x_{bpk}] \ge u_{jpk} + v_{bjk} - 1 \quad (b, p) \in \boldsymbol{BP},\ (b, j) \in \boldsymbol{BJ},\ (p, j) \in \boldsymbol{PJ},\ 0 < j \le J,\ 0 < k \le K \quad (6.56\text{a})$$

$$u_{jpk} \ge [x_{bpk}] + v_{bjk} - 1 \quad (b, p) \in \boldsymbol{BP},\ (b, j) \in \boldsymbol{BJ},\ (p, j) \in \boldsymbol{PJ},\ 0 < j \le J,\ 0 < k \le K \quad (6.56\text{b})$$

$$y_{ibk} \le 1 - [v_{b0k}] \qquad \text{for } (b, k) \text{ with } [v_{b0k}] = 1,\ 1 \le k \le K \quad (6.57\text{a})$$

$$1 - [v_{b0k}] \le \sum_{i=1}^{I} y_{ibk} \qquad \text{for } (b, k) \text{ with } [v_{b0k}] = 0,\ 1 \le k \le K \quad (6.57\text{b})$$

$$\sum_{i=1}^{I} q_{ibk}\theta_{is} \ge Q_{bk}\theta_{ps}^{L} - M_b \cdot \left\{\theta_{ps}^{L} - \min_{p}\left(\theta_{ps}^{L}\right)\right\} \cdot (1 - [x_{bpk}])$$

for (b, k) with $[v_{b0k}] = 0$, $(b, p) \in \boldsymbol{BP}$, $0 < k \le K$ (6.58a)

$$\sum_{i=1}^{I} q_{ibk}\theta_{is} \le Q_{bk}\theta_{ps}^{U} + M_b \cdot \left\{\max_{p}\left(\theta_{ps}^{U}\right) - \theta_{ps}^{U}\right\} \cdot (1 - [x_{bpk}])$$

for (b, k) with $[v_{b0k}] = 0$, $(b, p) \in \boldsymbol{BP}$, $0 < k \le K$ (6.58b)

$$\sum_{i=1}^{I} q_{ibk}\rho_i\theta_{is} \ge \left(\sum_{i=1}^{I} q_{ibk}\rho_i\right)\theta_{ps}^{L} - M_b \cdot \left\{\theta_{ps}^{L} - \min_{p}\left(\theta_{ps}^{L}\right)\right\} \cdot \rho_{\max} \cdot (1 - [x_{bpk}])$$

for (b, k) with $[v_{b0k}] = 0$, $(b, p) \in \boldsymbol{BP}$, $0 < k \le K$ (6.59a)

$$\sum_{i=1}^{I} q_{ibk}\rho_i\theta_{is} \le \left(\sum_{i=1}^{I} q_{ibk}\rho_i\right)\theta_{ps}^{U} + M_b \cdot \left\{\max_{p}\left(\theta_{ps}^{U}\right) - \theta_{ps}^{U}\right\} \cdot \rho_{\max} \cdot (1 - [x_{bpk}])$$

for (b, k) with $[v_{b0k}] = 0$, $(b, p) \in \boldsymbol{BP}$, $0 < k \le K$ (6.59b)

$$Q_{bk}r_{pi}^{L} - M_b \cdot \left\{ r_{pi}^{L} - \min_p \left(r_{pi}^{L} \right) \right\} (1-[x_{bpk}]) \le q_{ibk} \le Q_{bk}r_{pi}^{U} + M_b \cdot \left\{ \max_p \left(r_{pi}^{U} \right) - r_{pi}^{U} \right\} (1-[x_{bpk}])$$

for (b, k) with $[v_{b0k}] = 0$, $(b, p) \in \boldsymbol{BP}$, $0 < k \le K$ (6.60a,b)

$$\text{TC} = \sum_{i=1}^{I}\sum_{b=1}^{B}\sum_{k=1}^{K} c_i \cdot q_{ibk} + \sum_{b=1}^{B}\sum_{k=1}^{K-1} CB_b \cdot [xe_{bk}] + \sum_{j=1}^{J}\sum_{k=1}^{K-1} CT_j \cdot ue_{jk} + \sum_{o=1}^{O} DM_o \cdot d_o \qquad (6.61)$$

The revised model (RSPM) comprises eqs. 6.1, 6.3, 6.8, 6.16-6.18, 6.23-6.40, and 6.50-6.61, whose optimal solution will have constant blend rates, it will ensure minimum run length and satisfy the limits on the processing rates of the blenders.

For RMPM, eqs. 6.42-6.43 become:

$$\sum_{i=1}^{I} q_{ibk}\theta_{ist} \ge Q_{bk}\theta_{ps}^{L} - M_b \cdot \left\{ \theta_{ps}^{L} - \min_p \left(\theta_{ps}^{L} \right) \right\} \cdot (1-[x_{bpk}])$$

for (b, k) with $[v_{b0k}] = 0$, $(b, p) \in \boldsymbol{BP}$, $0 < k \le K$ (6.62a)

$$\sum_{i=1}^{I} q_{ibk}\theta_{ist} \le Q_{bk}\theta_{ps}^{U} + M_b \cdot \left\{ \max_p \left(\theta_{ps}^{U} \right) - \theta_{ps}^{U} \right\} \cdot (1-[x_{bpk}])$$

for (b, k) with $[v_{b0k}] = 0$, $(b, p) \in \boldsymbol{BP}$, $(t, k) \in \boldsymbol{TK}$, $0 < k \le K$ (6.62b)

$$\sum_{i=1}^{I} q_{ibk}\rho_{it}\theta_{ist} \ge \left(\sum_{i=1}^{I} q_{ibk}\rho_{it} \right) \theta_{ps}^{L} - M_b \cdot \left\{ \theta_{ps}^{L} - \min_p \left(\theta_{ps}^{L} \right) \right\} \cdot \rho_t^{\max} \cdot (1-[x_{bpk}])$$

for (b, k) with $[v_{b0k}] = 0$, $(b, p) \in \boldsymbol{BP}$, $(t, k) \in \boldsymbol{TK}$, $0 < k \le K$ (6.63a)

$$\sum_{i=1}^{I} q_{ibk}\rho_{it}\theta_{ist} \le \left(\sum_{i=1}^{I} q_{ibk}\rho_{it} \right) \theta_{ps}^{U} + M_b \cdot \left\{ \max_p \left(\theta_{ps}^{U} \right) - \theta_{ps}^{U} \right\} \cdot \rho_t^{\max} \cdot (1-[x_{bpk}])$$

for (b, k) with $[v_{b0k}] = 0$, $(b, p) \in \boldsymbol{BP}$, $(t, k) \in \boldsymbol{TK}$, $0 < k \le K$ (6.63b)

Then, RMPM comprises eqs. 6.1, 6.3, 6.8, 6.16-6.18, 6.23-6.36, 6.38, 6.40, 6.44-6.45, 6.50-6.57, and 6.60-6.63.

As noted in Chapter 5, the schedule from RSPM or RMPM may still show intermittent delivery of orders. When the delivery is over contiguous slots, then this can be easily revised by simply delivering at a constant rate until the entire order, which is distributed over contiguous slots, is fully delivered. Figure 6.5 shows the complete schedule adjustment procedure.

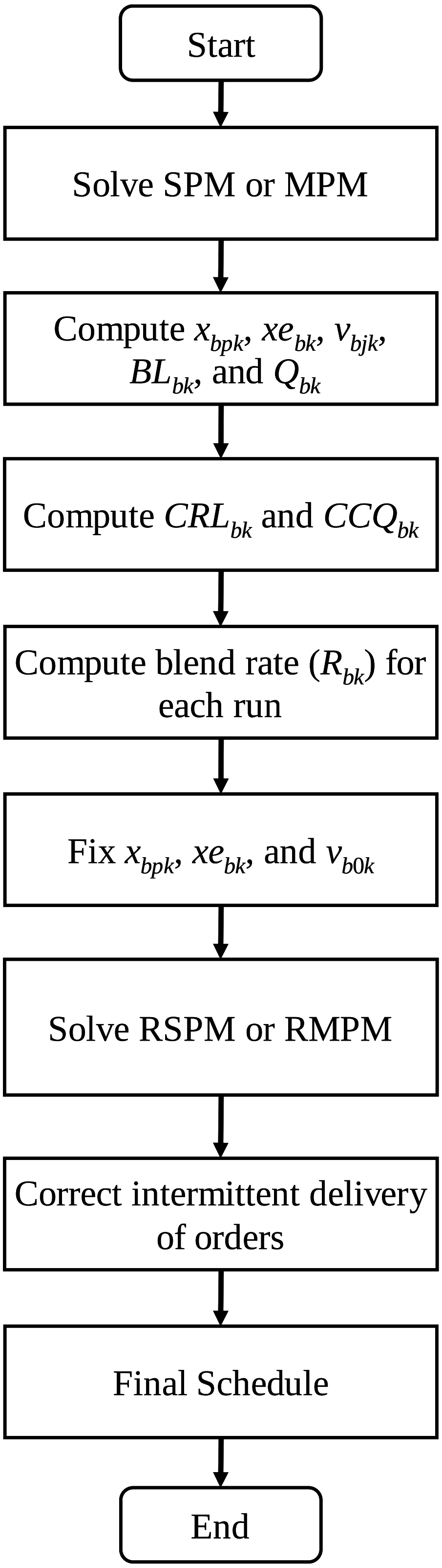

Figure 6.5 Flowchart for the schedule adjustment procedure

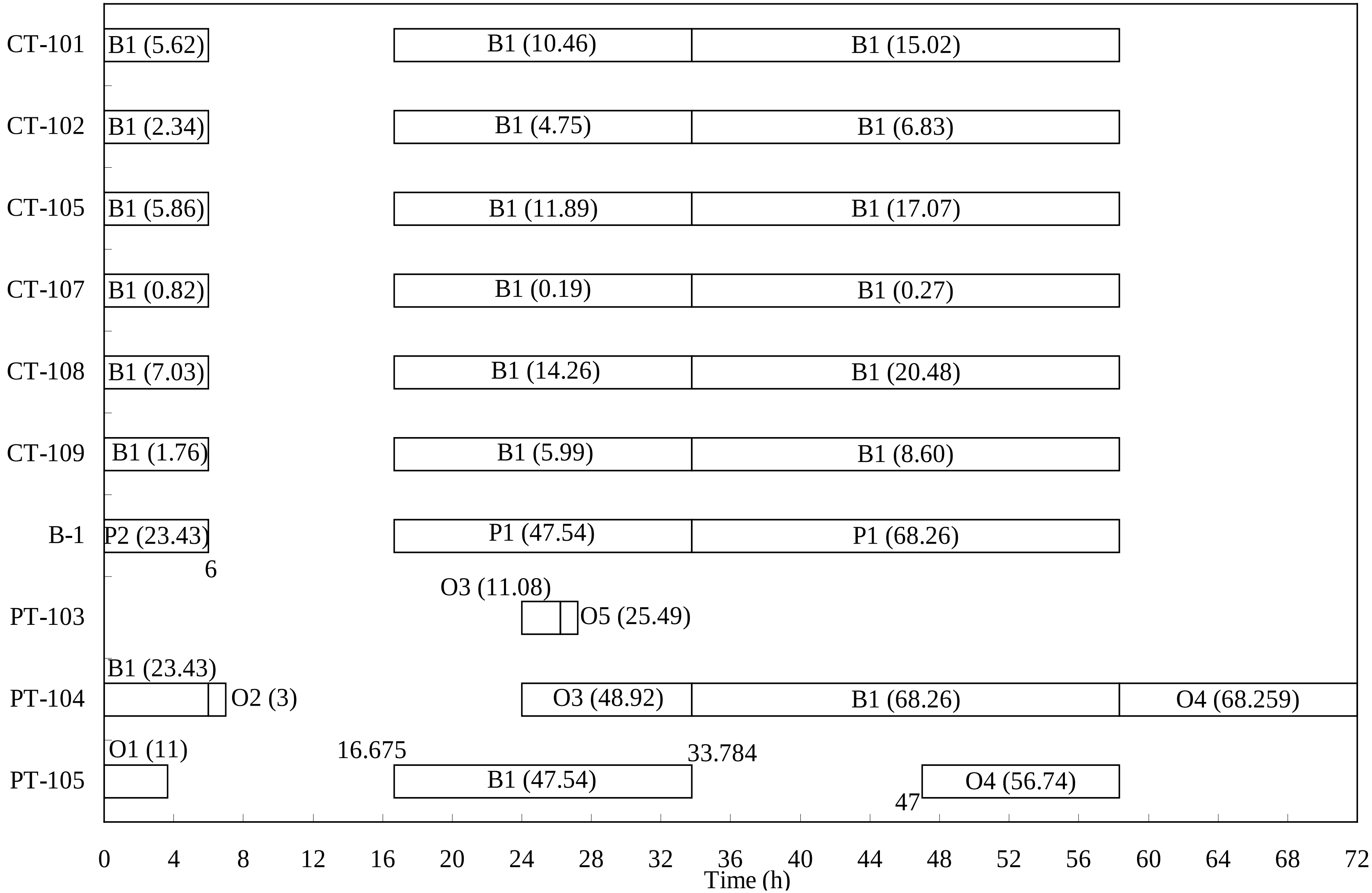

Figure 6.6 Optimal schedule for Example 1 (5 orders) from RSPM

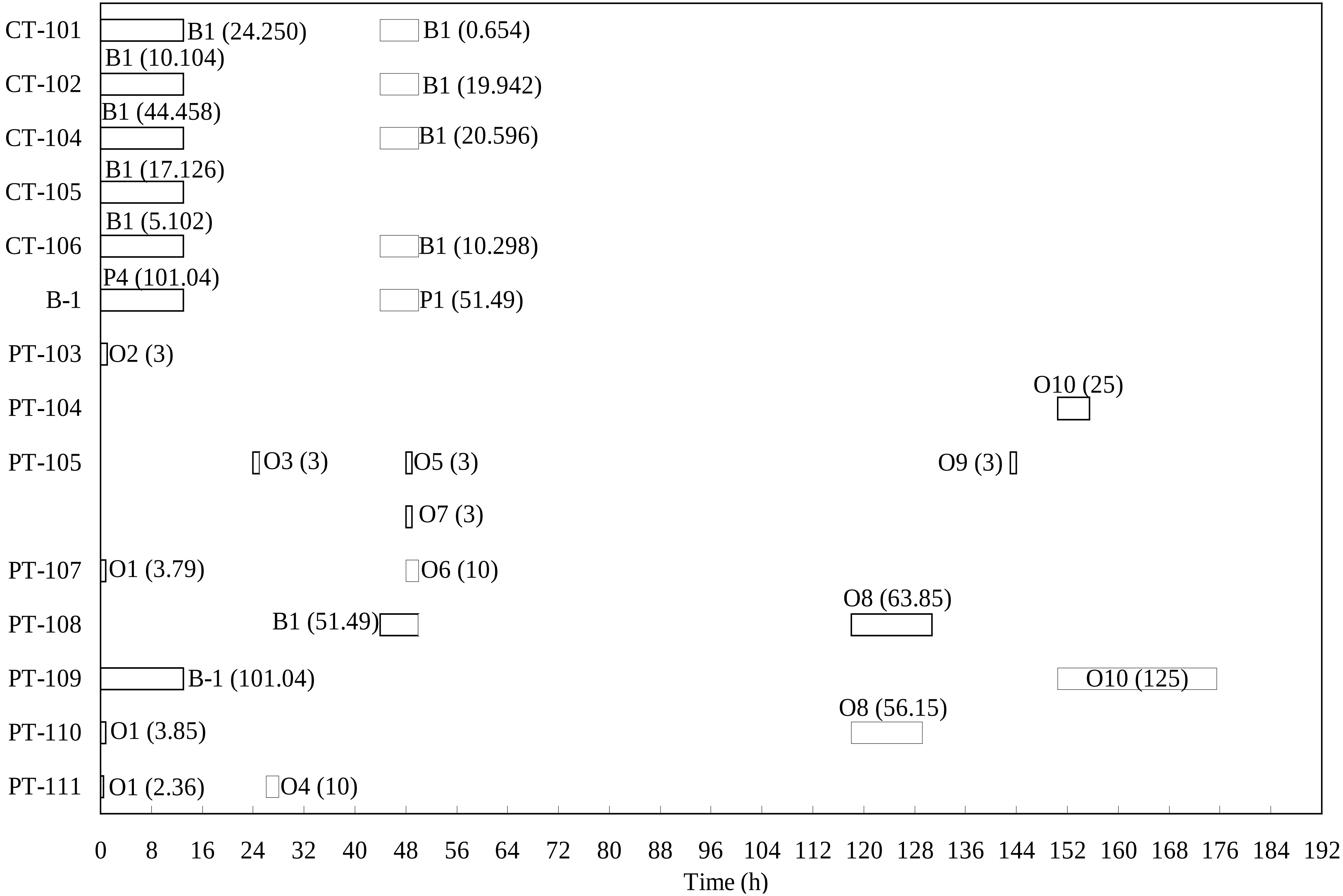

Figure 6.7 Optimal schedule for Example 2 (10 orders) from RSPM

Table 6.1a Period, slots, and feed flow rates to component tanks for Examples 1-8

Ex	Period	Period Duration	Slot	Feed Flow Rate to Component Tank (kbbl/h) CT-101	CT-102	CT-103	CT-104	CT-105	CT-106	CT-107	CT-108	CT-109
1	1	40	1-2	1.2	0.8	1.2	1.2	0.5	0.8	0.0	0.0	1.0
	2	32	3-4	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
2	1	60	1-2	1.2	0.8	1.2	1.2	0.5	0.8	0.0	0.0	1.0
	2	132	3	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
3-4	1	100	1-2	1.2	0.8	1.2	1.2	0.5	0.8	0.0	0.0	1.0
	2	92	3	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
5	1	100	1-3	1.2	0.8	1.2	1.2	0.5	0.8	0.0	0.0	1.0
	2	92	4	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
6	1	120	1-2	1.2	0.8	1.2	1.2	0.7	0.8	0.0	0.0	1.0
	2	72	3-4	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
7	1	80	1-3	1.2	0.8	1.2	1.2	0.7	0.8	0.0	0.0	1.0
	2	70	4-5	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
	3	42	6-7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
8	1	80	1-2	1.2	0.8	1.2	1.2	0.5	0.8	0.0	0.0	1.0
	2	70	3-4	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
	3	42	5-6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Table 6.1b Period, slots, and feed flow rates to component tanks for Examples 9-14

9	1	80	1-2	1.0	0.5	1.0	1.0	0.5	0.5	0.0	0.0	1.0
	2	70	3-5	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
	3	42	6-7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
10	1	80	1-2	1.0	0.5	1.0	1.0	0.8	0.5	0.0	0.0	1.0
	2	60	3-5	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
	3	52	6-7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
11	1	80	1-2	1.0	0.5	1.0	1.0	0.7	0.5	0.0	0.0	1.0
	2	60	3-5	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
	3	52	6-7	0.5	0.5	0.5	0.5	0.5	0.5	0.0	0.0	0.5
12	1	50	1-3	1.0	0.5	1.0	1.0	0.8	0.5	0.0	0.0	1.0
	2	50	4-6	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
	3	50	7-9	0.5	0.5	0.5	0.5	0.5	0.5	0.0	0.0	0.5
	4	42	10-12	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
13	1	50	1-3	1.0	0.5	1.0	1.0	0.7	0.5	0.5	0.5	1.0
	2	50	4-7	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
	3	50	8-10	0.5	0.5	0.5	0.5	0.5	0.5	0.0	0.0	0.5
	4	42	11-13	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
14	1	50	1-2	1.0	0.5	1.0	1.0	0.7	0.5	0.5	0.5	1.0
	2	50	3-6	0.8	0.6	0.6	0.8	0.5	0.6	0.5	0.5	0.0
	3	50	7-10	0.5	0.5	0.5	0.5	0.5	0.5	0.0	0.0	0.5
	4	42	11-13	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Now, the proposed unit slot formulation is illustrated in detail and compared with process-slot model in Chapter 5 using Examples 1 and 2 from Chapter 5. Then, the proposed model is further evaluated using additional twelve larger examples from Chapter 5.

6.7 Examples 1-2

Example 1 involves five orders, three products, nine component tanks, one blender, five product tanks, and one product property, i. e. Octane number. Example 2 involves 10 orders, nine component tanks, one blender, eleven product tanks, and one product property, i.e. Octane number. Tables 5.2-5.7 and 6.1 give the data for Examples 1 and 2. The single-period model (SPM) is first considered for these two examples. They are solved on Dell precision PWS690 (Intel® Xeon^R 5160 @ CPU 3.00 GHZ, 16.0 GB memory) running Windows XP using GAMS 22.2 with solver CPLEX 10.0.1.

The optimal solution for Example 1 from RSPM of $ 5149.73K is obtained with 4 slots in 1.94 CPU seconds. The optimal schedule from RSPM is illustrated in Figure 6.6. However, the optimal solution for this Example from RSPM in Chapter 5 is obtained with 4 slots in 0.83 CPU s. Thus, unit slot formulation needs a bit more time than process slot model for this example because the number of slots is not reduced. For Example 2, the optimal solution from RSPM of $ 3658.11K is obtained with 3 slots in 3.01 CPU s. Figure 6.7 shows the optimal schedule. However, 4 slots and 48.3 CPU seconds are needed for process-slot model in Chapter 5 to obtain the optimal solution. In other words, unit-slot formulation reduces the number of slots by 1 and obtains the optimal solution with less computational time.

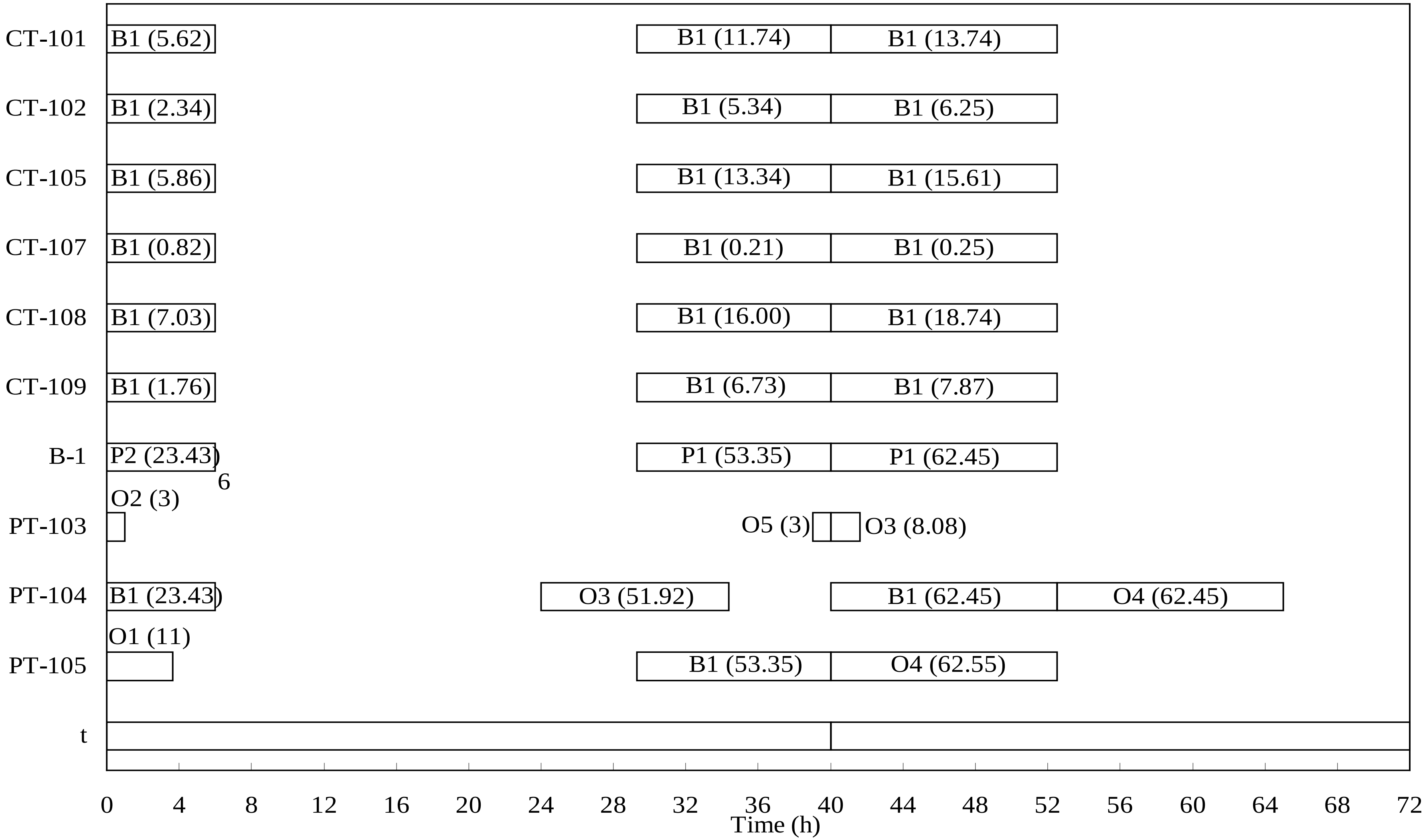

Figure 6.8 Optimal schedule for Example 1 (5 orders) from RMPM

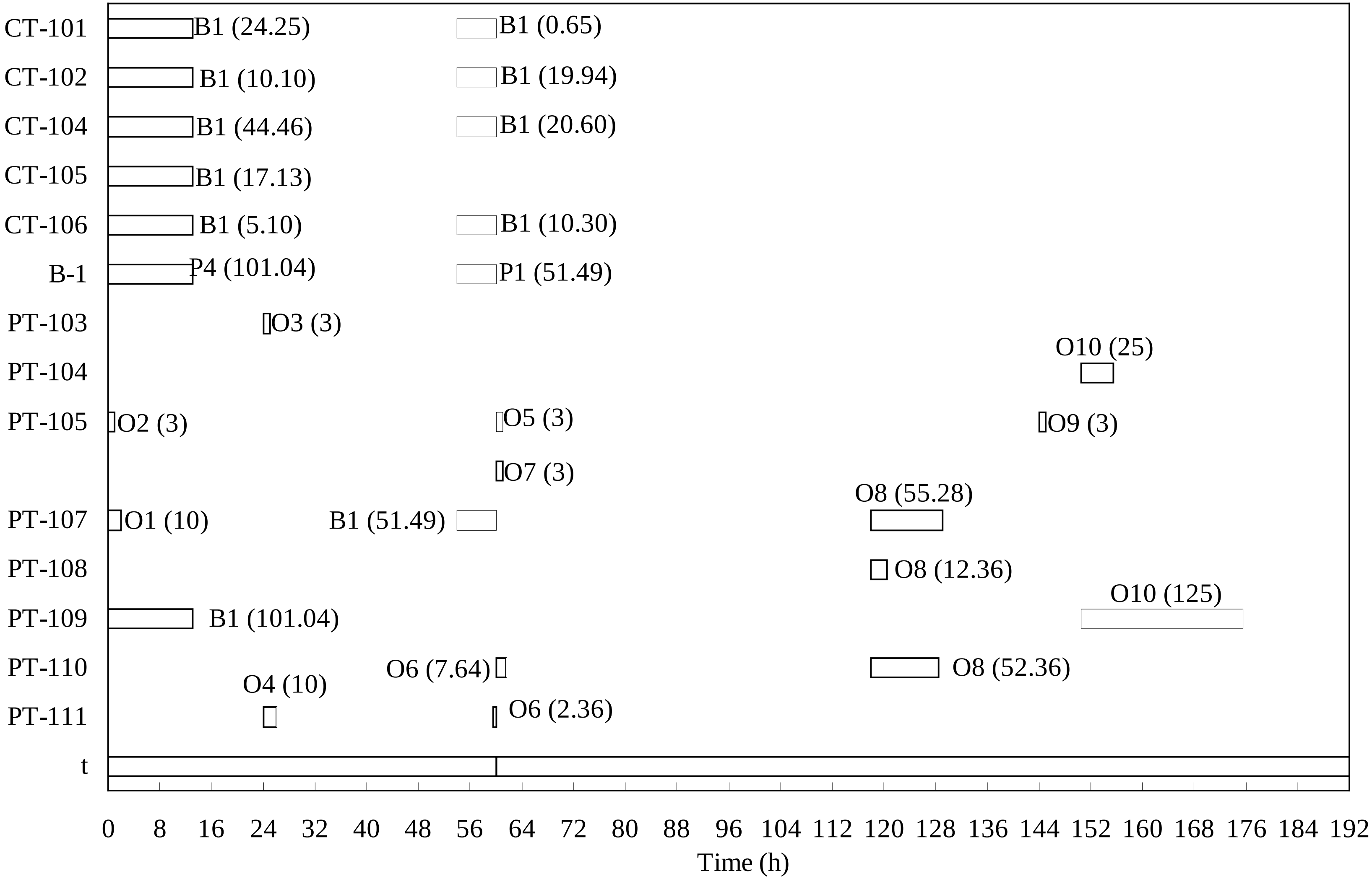

Figure 6.9 Optimal schedule for Example 2 (10 orders) from RMPM

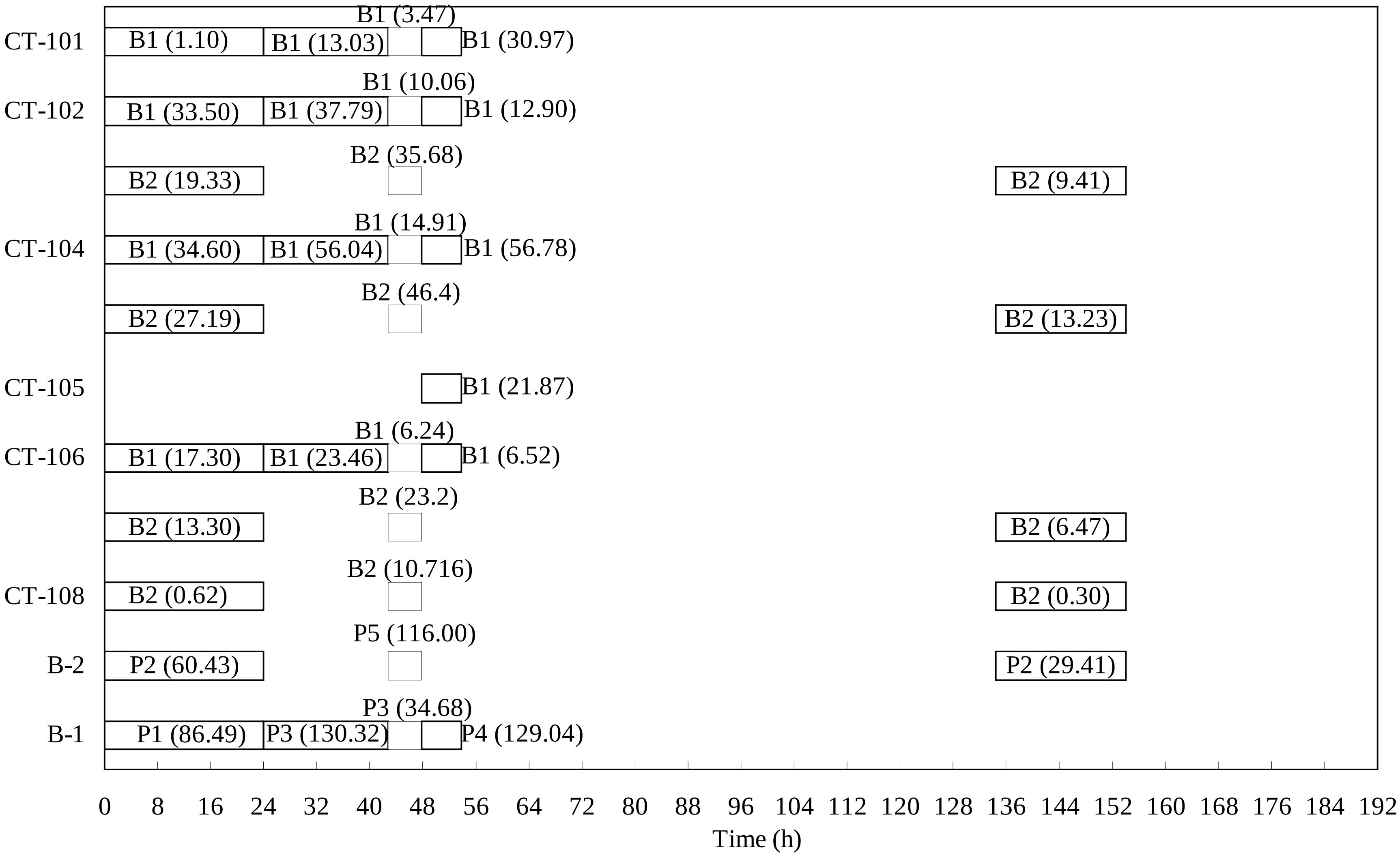

Figure 6.10a Blending schedule for Example 12 (35 orders) from RSPM

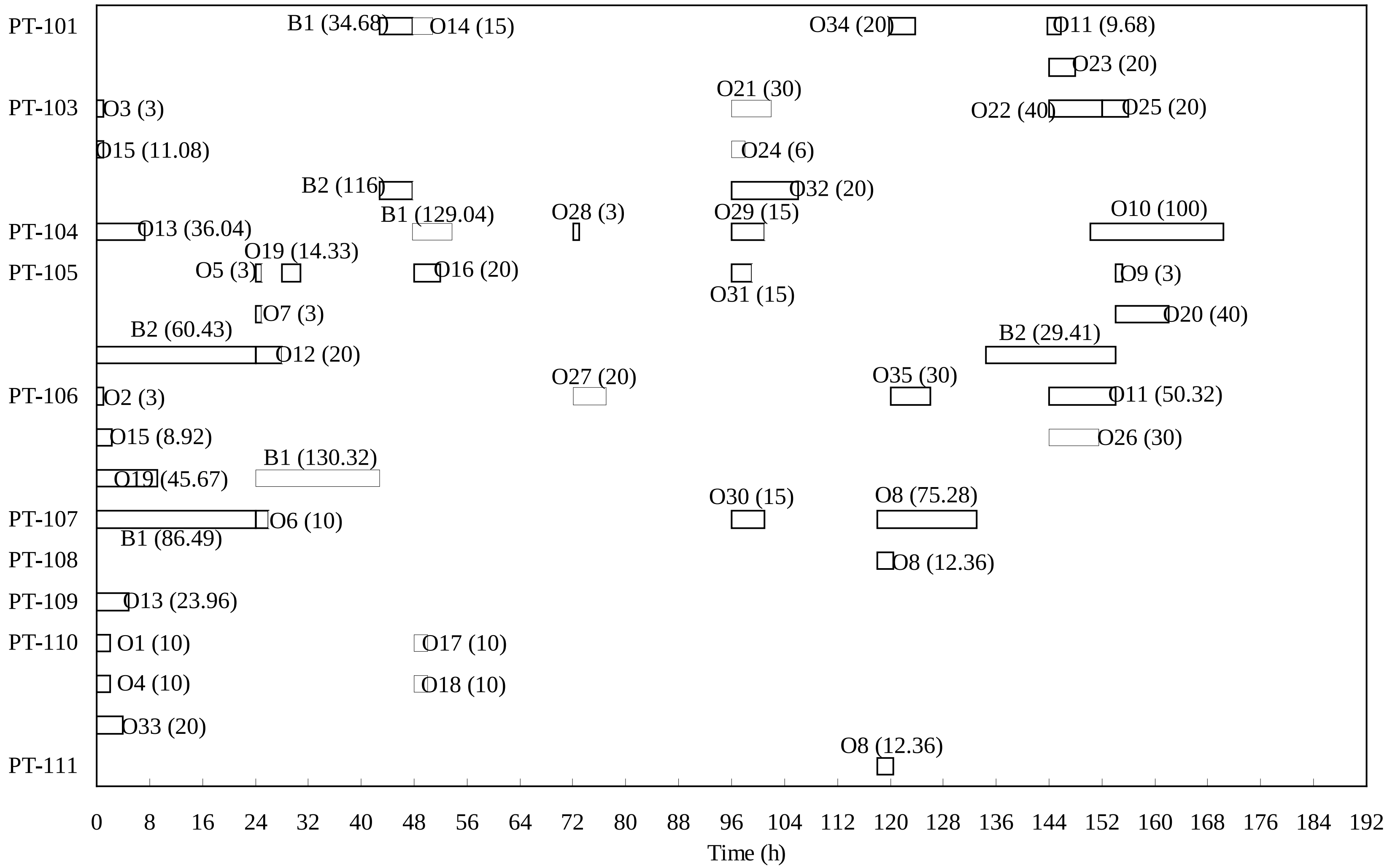

Figure 6.10b Order delivery schedule for Example 12 (35 orders) from RSPM

Now two periods for the feed rate to component tanks are assumed for Example 1. While the first period is from zero to 40 h, the other period is from 40 h to the scheduling horizon. Figure 5.8 shows the feed rate profile to component tanks. Two slots are assigned to the first period and two slots to the other period. The optimal solution of $ 5149.73K is obtained within 1.37 CPU seconds. However, the optimal solution of $ 5149.73K was obtained within 12.6 CPU s with 5 slots (three slots for the first period, and two slots to the other period) using process-slot model in Chapter 5. The main reason is that one slot is reduced using unit-slot model proposed in this Chapter. The optimal schedule using unit-slot model is presented in Figure 6.8. For Example 2, the optimal solution of $ 3658.11K for Example 2 is obtained within 0.62 CPU s using unit-slot model, while 137 CPU s using process-slot (Table 6.3). This is because two slots are reduced using unit-slot model.

6.8 Numerical Evaluation

Tables 5.2-5.6 in Chapter 5 and 6.1 show the data for the other twelve examples (Examples 3-14). These twelve examples have nine components, eleven product tanks and 192-hour (8 days) scheduling horizon. Examples 3-7 have one blender, Examples 8-12 have two, and Examples 13-14 have three blenders. 12-45 orders are used for these examples. As mentioned in Chapter 5, these examples are designed with widely varying structure, size, scale, and complexity, and mimic the real-life industrial scenarios very well. All examples are solved on Dell precision PWS690 (Intel® Xeon^{R} CPU 3.00 GHZ, 16.0 GB memory) running Windows XP using solver CPLEX 10.0.1.

Tables 6.2-6.3 show the solution statistics for Examples 1-14. From Table 6.2, optimal solutions are obtained for Examples 1-6 using unit-slot model with less computational time compared to those using process-slot model. For instance, optimal solution of $ 4556.67K for Example 4 is got within 9.52 CPU s using unit-slot model.

However, it does need 545 CPU s using process-slot model. Most importantly, the solution of \$ 5213.88K for Example 6 is guaranteed to be optimal using unit-slot model, but it fails using process-slot model even after 3 days of CPU time, as reported in Chapter 5. For the remaining examples (Examples 7-14), the optimal solutions are not able to obtain or their optimality cannot be proved. The upper limits (Tables 6.2 and 6.3) on CPU times for these examples are set, as did in Chapter 5. For Example 7, the solution of \$ 8100.35K is obtained within 4868 CPU s using unit-slot model, but process-slot model needs 10822 CPU s. For Examples 9-10, and 12-14, both modes reach the total CPU time limits for SPM and RSPM. However, unit-slot model obtains better solution than process-slot model. For instance, unit-slot model gives \$ 14848.35K for Example 12, while process-slot model gets \$ 15367.18K. For Examples 8 and 11, although unit-slot model obtains better solutions than process-slot model, it needs more computational time. This is because both models reach the time limit for SPM, unit-slot model takes more time to solve RSPM for better solutions. Similar observations can also be made from Table 6.3 for multiple-period scenario. For illustration, the final schedule for Example 12 (35 orders) are given in Figures 6.10a-b.

Table 6.4 gives RMIP values and best possible solutions from SPM for Examples 1-14. Similar to Table 5.10, the best possible solutions, which are the better lower bounds than RMIP values, do not improve much over time for Examples 7-14 with this unit-slot model. For instance, the value of RMIP for Example 11 is \$ 12469.79K, but after 36000 CPU s, it is \$ 12482.22K, improved only 0.10%. Critical improvement is for Examples 1-6. For instance, the value of RMIP for Example 6 is improved from \$ 4409.01K to \$ 5213.88K, improved by 18.26%. Noted that the obtained solution of \$ 5213.88K for Example 6 is proved to be optimal using unit-slot model. However, process-slot model in Chapter 5 cannot prove even after 3 days of CPU time.

Table 6.2 Solution statistics of various models for SPM for Examples 1-14

Ex	Order	Model	Slots	Discrete Variables	Continuous Variables	Constraints	Total CPU Time (s)	Cost (k$)
1	5	Unit-slot	4	136	428	1577	1.94	5149.73
		Process-slot	4	100	269	1098	0.83	5149.73
2	10	Unit-slot	3	202	732	2653	3.01	3658.11
		Process-slot	4	252	598	2847	48.6	3658.11
3	12	Unit-slot	3	246	798	2903	3.56	3159.12
		Process-slot	4	315	633	3199	178	3159.12
4	15	Unit-slot	3	285	924	3431	9.52	4556.67
		Process-slot	4	372	723	3823	545	4556.67
5	15	Unit-slot	3	285	924	3443	2.70	4556.67
		Process-slot	4	372	723	3835	252	4556.67
6	18	Unit-slot	4	476	1319	5174	4045	5213.88
		Process-slot	5	576	950	5438	10813	5213.88
7	20	Unit-slot	5	672	1744	7191	4868	8100.35
		Process-slot	6	775	1197	7319	10822	8100.35
8	20	Unit-slot	4	595	1546	7447	13306	8080.35
		Process-slot	4	523	959	6256	10802	8100.35
9	23	Unit-slot	4	635	1649	8251	14400*	10573.65
		Process-slot	7	1079	1673	12556	14400*	10781.40
10	25	Unit-slot	8	1465	3301	17515	14400*	11306.10
		Process-slot	9	1503	2209	17032	14400*	11327.20
11	30	Unit-slot	8	1644	3727	19475	46800*	13268.58
		Process-slot	12	2328	3216	25475	40065	13300.30
12	35	Unit-slot	9	2237	4992	25901	118800*	14848.35
		Process-slot	16	3851	4958	40480	118800*	15367.18
13	40	Unit-slot	12	3378	7219	42761	118800*	15823.63
		Process-slot	16	4142	5630	50133	118800*	16239.39
14	45	Unit-slot	12	3664	7874	46151	118800*	17888.21
		Process-slot	17	4830	6439	57657	118800*	19780.33

Note: CPU time limit for MIP of DICOPT is set at 10800 s for Exs 1-10, 36000 s for Ex 11, and 108000 s for Exs 12-14
CPU time limit for SPM of ours is set at 10800 s for Exs 1-10, 36000 s for Ex 11, and 108000 s for Exs 12-14
CPU time limit for RSPM of ours is set at 3600 s for Exs 1-10, and 10800 s for Exs 11-14
Total CPU time limit of DICOPT, BARON, and ours is set at 14400 s for Exs 1-10, 46800 s for Ex 11, and 118800 s for Exs 12-14
* Reached total CPU time limit

Table 6.3 Solution statistics of various models for MPM for Examples 1-14

Ex	Order	Model	Slots	Discrete Variables	Continuous Variables	Constraints	Total Time (s)	Cost (k$)
1	5	Unit-slot	4	136	428	1577	1.37	5149.73
		Process-slot	5	130	328	1384	12.6	5149.73
2	10	Unit-slot	3	202	732	2593	0.62	3658.11
		Process-slot	5	329	729	3600	137	3658.11
3	12	Unit-slot	3	246	798	2843	1.56	3159.12
		Process-slot	5	411	769	4048	83	3179.12
4	15	Unit-slot	3	285	924	3323	2.44	4556.67
		Process-slot	5	486	877	4840	800	4556.67
5	15	Unit-slot	4	408	1192	4708	6.94	4556.67
		Process-slot	5	486	877	4852	95.6	4556.67
6	18	Unit-slot	4	476	1319	5174	1216	5213.88
		Process-slot	5	576	950	5438	10801	5213.88
7	20	Unit-slot	7	986	2386	10271	10809	8100.35
		Process-slot	9	1219	1725	11138	10805	8100.35
8	20	Unit-slot	6	949	2242	11275	11074	8120.35
		Process-slot	8	1159	1779	12848	10821	8329.13
9	23	Unit-slot	7	1205	2759	14611	14400*	10627.15
		Process-slot	9	1423	2107	16248	11092	10636.40
10	25	Unit-slot	7	1265	2911	15295	14400*	11391.44
		Process-slot	10	1685	2436	18968	13215	11649.17
11	30	Unit-slot	7	1419	3287	17005	46800*	13302.08
		Process-slot	14	2328	3216	25475	36402	13357.69
12	35	Unit-slot	12	3050	6570	34655	118800*	14829.35
		Process-slot	17	4099	5258	43046	118800*	15326.00
13	40	Unit-slot	13	3677	7798	46357	118800*	15646.15
		Process-slot	17	4142	5966	53305	109051	16282.08
14	45	Unit-slot	13	3989	8505	50033	118800*	17839.71
		Process-slot	17	4830	6439	57657	118800*	19207.48

Note: CPU time limit for MIP of DICOPT is set at 10800 s for Exs 1-10, 36000 s for Ex 11, and 108000 s for Exs 12-14
CPU time limit for MPM of ours is set at 10800 s for Exs 1-10, 36000 s for Ex 11, and 108000 s for Exs 12-14
CPU time limit for RMPM of ours is set at 3600 s for Exs 1-10, and 10800 s for Exs 11-14
Total CPU time limit of DICOPT, BARON, and ours is set at 14400 s for Exs 1-10, 46800 s for Ex 11, and 118800 s for Exs 12-14
* Reached total CPU time limit

Table 6.4 RMIPs and best possible solutions for Examples 1-14 from SPM

Ex	RMIP from SPM (k$)	Best Possible Solution from SPM (k$)	Dev (%)
1	4988.52	5149.73	3.23
2	3570.65	3658.11	2.45
3	3105.08	3159.12	1.74
4	4501.78	4556.67	1.22
5	4501.78	4556.67	1.22
6	4409.01	5213.88	18.26
7	7797.02	7797.02	0.00
8	7777.02	7777.02	0.00
9	10006.69	10016.35	0.10
10	10608.76	10621.19	0.12
11	12469.79	12482.22	0.10
12	13702.46	13715.15	0.09
13	14632.48	14640.66	0.06
14	16493.51	16506.69	0.08

Dev: The relative gap between columns 2 and 3

6.9 MINLP Formulation

Recall that enforcing blending rate in a run to be constant makes the formulation nonlinear and nonconvex. Then, the nonlinear items are:

$$Q_{bk} \le F_{bk} \cdot SL_k \qquad 0 < k \le K \qquad (6.67a)$$

$$Q_{bk} \ge F_{bk} SL_k - M_b (v_{b0k} + xe_{bk}) \qquad 0 < k \le K \qquad (6.67b)$$

$$F_{b(k+1)} \ge F_{bk} - F_b^U xe_{bk} \qquad 0 < k < K \qquad (6.68a)$$

$$F_{b(k+1)} \le F_{bk} + F_b^U xe_{bk} \qquad 0 < k < K \qquad (6.68b)$$

$$F_b^L \le F_{bk} \le F_b^U \qquad (6.69)$$

Eqs. 6.67-6.69 is used to replace eq. 6.17 in the single- and multiple-period formulations to complete the MINLP single- and multiple-period formulations, which are denoted as MINLP-SPM and MINLP-MPM respectively. Since in Chapter 5, it has already shown that the adjustment procedure is better than commercial solves (BARON and DICOPT), it is no need to use BARON and DICOPT to solve the proposed models and compare with the proposed adjustment procedure in this Chapter.

6.10 Summary

In this Chapter, an efficient continuous-time model using unit slots was developed for integrated scheduling of gasoline blending and distribution operations in a refinery. The model incorporated many real-life operating features and policies such as multiple parallel non-identical blenders, piecewise constant component input flows and qualities, multi-product orders with multiple delivery dates, multi-purpose product tanks, minimum lengths for blending runs, constant blending rate in a run, common transfer policies, blending and storage transitions, etc. On 14 test problems of varying sizes and features from Chapter 5, the developed unit-slot based formulation obtained the same or better solutions than the process-slot model proposed in Chapter 5 with fewer binary variables and less computational time. In addition, the proposed model relaxed the assumption of the model in Chapter 5 that components are sufficient through the entire scheduling horizon.

CHAPTER 7

REACTIVE AND ROBUST CRUDE SCHEDULING UNDER UNCERTAINTY

7.1 Introduction

As mentioned in Chapter 3, scheduling of crude oil operations is an important and complex routine task in a refinery. It involves crude oil unloading, tank allocation, storage and blending of crudes, and CDU charging. Optimal crude oil scheduling can increase profits by exploiting cheaper but poor quality crudes, minimizing crude changeovers, avoiding ship demurrage, and managing crude inventory optimally. In Chapter 3, a robust and efficient algorithm has been developed for obtaining optimal schedules for operations without any uncertainty. However, in a practice, some common and frequent uncertainties in refinery operations such as ship arrival delays, demand fluctuation, and equipment malfunction are unavoidable. In the face of these uncertainties, an initial schedule (an optimal schedule obtained using nominal parameter values) may often be suboptimal or even become infeasible. As reviewed In Chapter 2, Neiro and Pinto (2005), and Li et al. (2004, 2005 and 2006) developed models to address refinery planning under demand and economic parameter uncertainties. Arief et al. (2004, 2007a,b) developed heuristic- and model-based rescheduling approach to address disruptions during crude oil scheduling. Cao and Gu

(2006) used chance constrained programming approach to address demand uncertainty in crude oil scheduling. However, their approach may lead to composition discrepancy.

As discussed in Chapter 2, reactive and predictive are the two approaches to disruption management. Reactive approach is used during the actual execution of the plan or schedule, when a disruption has occurred. Arief et al. (2007a) defined three factors for schedule comparison. They are the objective values in terms of difference from the profit of initial schedules, number of rescheduled operations, and computational time. The number of reschedule operations is defined as the number of operation blocks that were in the initial schedule but not in the new schedule, i.e. changes in configuration including parcel-to-tank and tank-to-CDU connections (Arief et al. 2007a). The same configuration with different start or end times is also counted as a reschedule operation. However, an identical configuration with a difference in transfer rates is not. Predictive approach seeks to accommodate possible disruptions while planning or scheduling. In other words, the aim of predictive approach is to produce inherently robust plans or schedules.

In this Chapter, optimization-based reactive approach is first developed to address some disruptions such as demand fluctuation, tank and CDU unavailability, and ship arrival delay and compare with the approach proposed by Arief et al. (2007a). A scenario-based model (predictive approach) is developed to obtain robust schedule for demand uncertainty.

7.2 Problem Statement

Crude oil operations in a typical marine-access refinery (Figure 3.1) comprises offshore facilities for crude unloading such as a single buoy mooring (SBM) or single point mooring (SPM) station, onshore facilities for crude unloading such as one or more jetties, tank farm consisting of crudes storage and/or charging tanks, and processing units such as crude distillation units (CDUs). The unloading facilities supply crude to crude storage tanks via pipelines. The pipeline connecting the SBM/SPM station with crude tanks is called the SBM/SPM line, and it normally has a substantial holdup.

Two types of ships supply crudes to the refinery. Very large crude carriers (VLCCs) or ultra large crude carriers (ULCCs) carry multiple parcels of several crudes and dock at the SBM/SPM station offshore. Small vessels carry single crudes and berth at the jetties. The entire crude oil operation involves unloading and blending crudes from ships into various storage tanks at various times, and charging CDUs from one or more storage tanks at various rates over time. The objective of the scheduling problem is to maximize the gross profit, which is the revenue computed in terms of crude margins minus the operating costs such as demurrage, safety stock penalties, etc. The more detailed information can be referred to Chapter 3.

7.3 Basic Formulation and Algorithm

While several models (Shah, 1996; Lee et al., 1996; Li et al., 2002; Li et al. 2005; Jia et al., 2003; Reddy et al., 2004a,b) exist in the literature for solving varying forms of the crude scheduling problem, one is needed for the development of robust schedule. In Chapter 3, several advantages of the discrete-time model of Reddy et al. (2004b) over the existing models were presented such as continuous-time formulation features, incorporation of some important structural and operations features of a marine-access refinery, 8-h time periods, and consideration of SBM holdup. Moreover, the model of Reddy et al. (2004b) was extended with two refinements in Chapter 3 to control drastic changes in feed rates of CDUs and accommodate nonlinear crude properties. The model of Reddy et al. (2004b) with these two refinements (Chapter 3) forms the basis of reactive or robust schedule development. The detailed explanation for that model can be referred to Reddy et al. (2004b) and Chapter 3.

As done by Reddy et al. (2004b), three primary binary decision variables were used to model parcel-to-SBM/Jetty, tank-to-SBM/jetty, and tank-to-CDU connections.

$$XP_{pt} = \begin{cases} 1 & \text{if parcel } p \text{ is connected for transfer during period } t \\ 0 & \text{otherwise} \end{cases}$$

$$XT_{it} = \begin{cases} 1 & \text{if tank } i \text{ is connected to receive crude during period } t \\ 0 & \text{otherwise} \end{cases}$$

$$Y_{iut} = \begin{cases} 1 & \text{if tank } i \text{ feeds CDU } u \text{ during period } t \\ 0 & \text{otherwise} \end{cases}$$

Recall that the problem of crude oil scheduling is non-convex MINLP, because crude blending in storage tanks results in bilinear items as follows,

$$FCTU_{iuct} = f_{ict} \cdot FTU_{iut} \qquad (i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC} \qquad (7.1)$$

$$VCT_{ict} = f_{ict} \cdot V_{it} \qquad (i, c) \in \boldsymbol{IC} \qquad (7.2)$$

where, f_{ict} denotes the fraction of crude c in tank i at the end of period t, $FCTU_{iuct}$ is the amount of crude c from tank i to CDU u during period t, FTU_{iut} is the total amount of crude from tank i to CDU u during t, VCT_{ict} is the amount of crude c in tank i at the end of period t, and V_{it} is the total amount of crude in tank i at the end of period t. $\boldsymbol{IC} = \{(i, c) |$ tank i can hold crude $c\}$. $\boldsymbol{IU} = \{(i, u) |$ tank i can feed CDU $u\}$.

In Chapter 3, an improved algorithm has been developed to solve this MINLP problem. This algorithm is more robust and efficient than the existing algorithms in the literature (Lee et al, 1996; Li et al., 2002; Reddy et al., 2004a,b; Moro and Pinto, 2004). Therefore, it is used to solve this MINLP problem in this Chapter.

7.4 Reactive Scheduling

In reactive scheduling, the plant operations are scheduled again using the solution algorithm but from the time of disruption informed, not detected because some time may need to inform refinery operators after some disruptions happen like ship arrival delay. This means the plant operations are running based on the initial schedule, which is the schedule without any disruptions, until the disruption informed. The part of the initial schedule before disruption informed is fixed and the remaining part is solved again according to the corresponding type of disruption. Let dt be the period at the beginning of which a disruption is informed. PXP_{pt}, PXT_{it}, PY_{iut}, $PFPT_{pit}$, $PFTU_{iut}$, $PFCTU_{iuct}$, PFU_{ut}, $PVCT_{ict}$, and PV_{it} are the values from the initial schedule. Then, from 0 to $dt - 1$ period, all variables are fixed to that of the initial schedule as follows,

$$XP_{pt} = PXP_{pt} \qquad 0 < t < dt$$

$$XT_{it} = PXT_{it} \qquad 0 < t < dt$$

$$Y_{iut} = PY_{iut} \qquad 0 < t < dt$$

$$FPT_{pit} = PFPT_{pit} \qquad 0 < t < dt$$

$$FTU_{iut} = PFTU_{iut} \qquad 0 < t < dt$$

$$FCTU_{iuct} = PFCTU_{iuct} \qquad 0 < t < dt$$

$$FU_{ut} = PFU_{ut} \qquad 0 < t < dt$$

$$VCT_{ict} = PVCT_{ict} \qquad 0 < t < dt$$

$$V_{it} = PV_{it} \qquad 0 < t < dt$$

Moreover, all constraints before dt period are relaxed as follows,

Parcel-to-SBM Connections

$$XP_{pt} = XP_{p(t-1)} + XF_{pt} - XL_{p(t-1)} \qquad (p, t) \in \boldsymbol{PT}, t > dt \qquad (7.3)$$

$$XP_{pt} \geq XL_{pt} \qquad (p, t) \in \boldsymbol{PT}, t > dt \qquad (7.4)$$

$$\sum_t XF_{pt} = \sum_t XL_{pt} = 1 \qquad (p, t) \in \boldsymbol{PT}, t > dt \qquad (7.5a\text{-}b)$$

$$TF_p = \sum_t (t-1) \cdot XF_{pt} \qquad (p, t) \in \boldsymbol{PT}, t > dt \qquad (7.6)$$

$$TL_p = \sum_t t \cdot XL_{pt} \qquad (p, t) \in \boldsymbol{PT}, t > dt \qquad (7.7)$$

$$\sum_p XP_{pt} \le 2 \qquad (p, t) \in \boldsymbol{PT}, t > dt \qquad (7.8)$$

$$TF_{(p+1)} \ge TL_p - 1 \qquad (7.9)$$

$$TF_p \ge ETA_p \qquad (7.10)$$

SBM-to-Tank Connections

$$\sum_i XI_{it} \le 2 \qquad t > dt \qquad (7.11)$$

$$X_{pit} \ge XP_{pt} + XT_{it} - 1 \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI}, t > dt \qquad (7.12)$$

$$\sum_i X_{pit} \le 2 \cdot XP_{pt} \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI}, t > dt \qquad (7.13)$$

$$\sum_p X_{pit} \le 2 \cdot XT_{it} \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI}, t > dt \qquad (7.14)$$

$$\sum_p \sum_i X_{pit} \le 2 \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI}, t > dt \qquad (7.15)$$

Tank-to-CDU Connections

$$\sum_u Y_{iut} \le 2 \qquad (i, u) \in \boldsymbol{IU}, t > dt \qquad (7.16)$$

$$\sum_i Y_{iut} \le 2 \qquad (i, u) \in \boldsymbol{IU}, t > dt \qquad (7.17)$$

$$2XT_{it} + Y_{iut} + Y_{iu(t+1)} \le 2 \qquad (i, u) \in \boldsymbol{IU}, t > dt - 1 \qquad (7.18)$$

Crude Delivery and Processing

$$FPT_{pi}^{L} X_{pit} \le FPT_{pit} \le FPT_{pi}^{U} X_{pit} \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI}, t > dt \qquad (7.19)$$

$$\sum_p \sum_i \frac{FPT_{pit}}{FPT_{pi}^U} \le 1 \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI}, t > dt \qquad (7.20)$$

$$\sum_{i,t} FPT_{pit} = PS_p \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI} \qquad (7.21)$$

$$FTU_{iut} = \sum_{(i,c)\in IC} FCTU_{iuct} \qquad (i, u) \in \boldsymbol{IU}, t > dt \qquad (7.22)$$

$$Y_{iut} FTU_{iu}^L \le FTU_{iut} \le Y_{iut} FTU_{iu}^U \qquad (i, u) \in \boldsymbol{IU}, t > dt \qquad (7.23)$$

$$FU_{ut} = \sum_{(i,u)\in IU} FTU_{iut} \qquad t > dt \qquad (7.24)$$

$$FU_{ut}^L \le FU_{ut} \le FU_{ut}^U \qquad t > dt \qquad (7.25)$$

$$\gamma_u^L FU_{ut} \le FU_{u(t+1)} \le \gamma_u^U FU_{ut} \qquad t > dt \qquad (7.26a,b)$$

$$FU_{ut} \cdot xc_{cu}^L \le \sum_i FCTU_{iuct} \le FU_{ut} \cdot xc_{cu}^U \qquad (i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC}, t > dt \qquad (7.27)$$

$$xk_{ku}^L FU_{ut} \le \sum_i \sum_c FCTU_{iuct} xk_{kc} \le xk_{ku}^U FU_{ut} \qquad (i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC}, t > dt \qquad (7.28a,b)$$

$$\theta_{ku}^L \left(\sum_i \sum_c FCTU_{iuct} \rho_c \right) \le \sum_i \sum_c FCTU_{iuct} \rho_c \theta_{kc} \le \theta_{ku}^U \left(\sum_i \sum_c FCTU_{iuct} \rho_c \right)$$

$$(i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC}, t > dt \qquad (7.29a,b)$$

$$YY_{iut} \ge Y_{iut} + Y_{iu(t+1)} - 1 \qquad (i, u) \in \boldsymbol{IU}, t > dt - 1 \qquad (7.30a)$$

$$YY_{iut} \le Y_{iu(t+1)} \qquad (i, u) \in \boldsymbol{IU}, d > dt - 1 \qquad (7.30b)$$

$$YY_{iut} \le Y_{iut} \qquad (i, u) \in \boldsymbol{IU}, d > dt \qquad (7.30c)$$

$$CO_{ut} \ge Y_{iut} + Y_{iu(t+1)} - 2YY_{iut} \qquad (i, u) \in \boldsymbol{IU}, d > dt \qquad (7.31)$$

$$M[2 - \sum_i YY_{iut}] + FTU_{iut} \ge FTU_{iu(t+1)} \qquad (i, u) \in \boldsymbol{IU}, d > dt - 1 \qquad (7.32a)$$

$$M[2 - \sum_i YY_{iut}] + FTU_{iu(t+1)} \ge FTU_{iut} \qquad (i, u) \in \boldsymbol{IU}, d > dt - 1 \qquad (7.32b)$$

Crude Inventory

$$VCT_{ict} = VCT_{ic(t-1)} + \sum_{(p,c)\in PC,(p,t)\in PT} FPT_{pit} - \sum_{(i,u)\in IU} FCTU_{iuct} \qquad (i, c)\in \boldsymbol{IC},\ t > dt \qquad (7.33)$$

$$V_{it} = \sum_{(i,c)\in IC} VCT_{ict} \qquad d > dt \qquad (7.34)$$

$$V_i^L \le V_{it} \le V_i^U \qquad (7.35)$$

$$xt_{ic}^L V_{it} \le VCT_{ict} \le xt_{ic}^U V_{it} \qquad (i, c)\in \boldsymbol{IC},\ d > dt \qquad (7.36)$$

$$FCTU_{iuct} = f_{ict} \cdot FTU_{iut} \qquad (i, u) \in \boldsymbol{IU},\ (i, c) \in \boldsymbol{IC},\ t > dt \qquad (7.37)$$

$$VCT_{ict} = f_{ict} \cdot V_{it} \qquad (i, c)\in \boldsymbol{IC},\ t > dt \qquad (7.38)$$

Production Requirements

$$\sum_t FU_{ut} = D_u \qquad (7.39)$$

Demurrage Cost

$$DC_v \ge (TL_p - ETA_p - ETD_v)SWC_v \qquad (p, v) \in \boldsymbol{PV} \qquad (7.40)$$

Stock Penalty

$$SC_t \ge SSP(SS - \sum_t V_{it}) \qquad d > dt \qquad (7.41)$$

Objective Function

$$\text{Profit} = \sum_i \sum_u \sum_c \sum_t FCTU_{iuct} CP_c - \sum_v DC_v - COC \sum_u \sum_t CO_{ut} - \sum_t SC_t \qquad (7.42)$$

The above formulation is proposed for SBM, but can be extended to the case where jetties exist as follows. ***J*** is the number of jetties.

Drop Eqs. 7.9 and 7.11 and modify eqs. 7.8 and 7.15 as follows,

$$\sum_p XP_{pt} \le J \qquad (p, t) \in \boldsymbol{PT},\ t > dt \qquad (7.43)$$

$$\sum_p \sum_i X_{pit} \le 2J \qquad (p, t) \in \boldsymbol{PT},\ (p, i) \in \boldsymbol{PI},\ t > dt \qquad (7.44)$$

Replacing eq. 7.20 by,

$$\sum_i \frac{FPT_{pit}}{FPT_{pi}^U} \le 1 \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI}, t > dt \qquad (7.45)$$

$$\sum_p \frac{FPT_{pit}}{FPT_{pi}^U} \le 1 \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI}, t > dt \qquad (7.46)$$

For both SBM pipeline and Jetties, eq. 7.9 is effective for SBM parcel as follows,

$$TF_{(p+1)} \ge TL_p - 1 \qquad p \in \boldsymbol{SP} \qquad (7.47)$$

Eqs. 7.20, 7.45, and 7.46 are replaced with the following constraints.

$$\sum_i \frac{FPT_{pit}}{FPT_{pi}^U} \le 1 \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI}, t > dt \qquad (7.48)$$

$$\sum_{p \in SP} \frac{FPT_{pit}}{FPT_{pi}^U} + \sum_{p \in JP} \frac{FPT_{pit}}{FPT_{pi}^U} \le 1 \qquad t > dt \qquad (7.49)$$

Where, ***SP*** denotes SBM parcel and ***JP*** denotes jetty parcel.

Thus, the reactive disruption management model (RDM) comprises eqs. 7.3-7.49 with all variables has the same values as the initial schedule before the period at the beginning of which disruption D is informed. It should be noted that although RDM obtains a good profit, the obtained schedule may have many differences to the initial schedule, as illustrated with Example 1. All examples in this Chapter are solved on Dell OPTIPLEX GX 620 using GAMS 22.2 with CPLEX 10.0.1.

7.4.1 Example 1

This example is taken from Arief et al. (2007). The data and the initial schedule (Profit = \$ 1849K) from Arief et al. (2007a) are given in Tables 7.1 and 7.2, respectively. Now, the refinery is informed at the end of period 5 that the ship carrying parcel 7 will arrive at the beginning of period 9, delayed 16 hours (one period is about 8 h) because of bad weather at sea. RDM model is solved to reschedule the operations, in which dt = 5. Optimal profit of \$ 1848.73K is obtained in 20.3 CPU seconds. Table 7.3 gives the obtained schedule. From Tables 7.2 and 7.3, many operation changes (parcel to SBM or jetties connection, SBM or jetties to tank connection, and tank to CDU connection) are needed to accommodate this parcel delay from the initial schedule. For instance, parcel 5 is unloaded to tanks 1 and 6 during period 6 in the initial schedule, while this parcel is unloaded only to tank 6 during this period in the obtained schedule. This also happens for parcel 6 during period 6. From periods 8 to 15, tank 5 is charging CDU 2 in the initial schedule, while it is charging CDU 1 in the proposed schedule. Those differences may cause inconveniency for refiners.

To overcome this disadvantage, schedule change is defined in which three additional variables are defined to model schedule changes from the initial schedule and a penalty for these schedule changes is imposed in the objective function. Three 0-1 continuous variables $PPXP_{pt}$, $PPXT_{it}$, and PPY_{iut} are defined and corresponding constraints are developed to model the schedule changes from the initial schedule as follows,

$$PPXP_{pt} \geq XP_{pt} - PXP_{pt} \qquad (p, t) \in \boldsymbol{PT} \tag{7.50a}$$

$$PPXP_{pt} \geq PXP_{pt} - XP_{pt} \qquad (p, t) \in \boldsymbol{PT} \tag{7.50b}$$

$$PPXT_{it} \geq XT_{it} - PXT_{it} \tag{7.51a}$$

$$PPXT_{it} \geq PXT_{it} - XT_{it} \tag{7.51b}$$

$$PPY_{iut} \geq Y_{iut} - PY_{iut} \qquad (i, u) \in \boldsymbol{IU} \tag{7.52a}$$

$$PPY_{iut} \geq PY_{iut} - Y_{iut} \qquad (i, u) \in \boldsymbol{IU} \tag{7.52b}$$

Then, the objective function changes to:

Table 7.1 Data for Example 1

Tanker	Arrival Period	Parcel No: (Crude, Parcel Size kbbl)
VLCC-1	2	1: (C2, 10), 2: (C6, 100), 3: (C1, 100), 4: (C4, 90)
V1-V2	4	5: (C2, 125), 6: (C5, 125)
V3	6	7: (C3, 100)

Tank & CDU	Capacity (kbbl)	Heel (kbbl)	Initial Inventory (kbbl)	Allowable Crude (Class)	Initial Crude Amount (kbbl) C1 or C4	Initial Crude Amount (kbbl) C2 or C5	Initial Crude Amount (kbbl) C3 or C6	Total Demand (kbbl)	Processing Limits (kbbl/period) Min-Max	Property Specification Range (Min-Max)
T1	400	50	250	C1, C2, C3 (1)	100	100	50	-	-	-
T2	400	50	250	C4, C5, C6 (2)	50	100	100	-	-	-
T3	400	50	300	C4, C5, C6 (2)	100	100	100	-	-	-
T4	400	50	350	C4, C5, C6 (2)	100	150	100	-	-	-
T5	400	50	250	C4, C5, C6 (2)	100	75	75	-	-	-
T6	400	50	100	C1, C2, C3 (1)	25	25	50	-	-	-
T7	400	50	100	C1, C2, C3 (1)	50	25	25	-	-	-
T8	400	50	250	C1, C2, C3 (1)	75	75	100	-	-	-
CDU 1	-	-	-	C4, C5, C6 (2)				400	20-50	0.0125-0.0185
CDU 2	-	-	-	C4, C5, C6 (2)				400	20-50	0.0125-0.0175
CDU 3	-	-	-	C1, C2, C3 (1)				400	20-50	0.0040-0.0070

Crude	Property Specification	Margin ($/bbl)
C1	0.005	1.500
C2	0.008	1.750
C3	0.004	1.850
C4	0.015	1.250
C5	0.010	1.450
C6	0.020	1.650

Parcel-Tank flow rate: 10-250 kbbl/period   Tank-CDU flow rate: 0-50 kbbl/period
Demurrage cost: 15 k$/period   Changeover loss: 5 k$/instance
Safety stock penalty: 0.2 $/bbl/period   Desired safety stock: 1200 kbbl

Table 7.2 The initial schedule from Arief et al. (2007a) for Example 1 (Profit = $ 1849K)

	Crude Amount [to CDU No.] (from Vessel No.) in kbbl for Period														
Tank	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	−20.0[3]	−20.0[3]	+10.0(1)	+100(3)	+10.0(5)	+10.0(5)									
2			+90.0(2)	+10.0(2)		−20.0[2]	−20.0[2]	−32.5[1]	−32.5[1]	−32.5[1]	−32.5[1]	−32.5[1]	−32.5[1]	−32.5[1]	−32.5[1]
3					+80.0(4)	+20.0(6)									
4	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−7.50[2]	−7.50[2]	−7.50[2]	−7.50[2]	−7.50[2]	−7.50[2]	−7.50[2]	−7.50[2]
	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]										
5					+10.0(4)	+105(6)		−25.0[2]	−25.0[2]	−25.0[2]	−25.0[2]	−25.0[2]	−25.0[2]	−25.0[2]	−25.0[2]
6						+105(5)	+100(7)		−50.0[3]	−50.0[3]	−50.0[3]	−30.0[3]	−20.0[3]	−20.0[3]	−20.0[3]
7			−20.0[3]	−3.20[3]	−3.20[3]	−3.20[3]	−3.20[3]	−3.20[3]							
8				−16.8[3]	−16.8[3]	−16.8[3]	−16.8[3]	−16.8[3]							

'−' sign represents delivery to [CDU], '+' sign represents receipt from (Parcels)

Table 7.3 Proposed schedule with RDM for Example 1–Parcel 7 delayed to the end of period 7, informed at the end of period 4 (Profit = $ 1848.73K)

	Crude Amount [to CDU No.] (from Vessel No.) in kbbl for Period														
Tank	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	−20.0[3]	−20.0[3]	+10.0(1)	+100(3)	+10.0(5)				+10(7)	+10(7)					
2			+90.0(2)	+10.0(2)		−20.0[2]	−20.0[2]	−50.0[2]	−20.0[2]	−20.0[2]	−30.0[2]	−50.0[2]	−50.0[2]	−20.0[2]	−20.0[2]
3					+80.0(4)										
4	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−50.0[1]	−50.0[1]								
	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]										
5					+10.0(4)	+125(6)		−25.0[1]	−25.0[1]	−25.0[1]	−25.0[1]	−25.0[1]	−25.0[1]	−25.0[1]	−25.0[1]
6						+115(5)				+80(7)		−50.0[3]	−50.0[3]	−30.0[3]	−50.0[3]
7			−20.0[3]	−3.20[3]	−3.20[3]	−3.20[3]	−3.20[3]	−3.20[3]	−3.20[3]	−3.20[3]	−3.20[3]				
8				−16.8[3]	−16.8[3]	−16.8[3]	−16.8[3]	−16.8[3]	−16.8[3]	−16.8[3]	−16.8[3]				

'–' sign represents delivery to [CDU], '+' sign represents receipt from (Parcels)

$$\text{Profit} = \sum_i \sum_u \sum_c \sum_t FCTU_{iuct} CP_c - \sum_v DC_v - COC \sum_u \sum_t CO_{ut} - \sum_t SC_t$$
$$- M \left(\sum_p \sum_t PPXP_{pt} + \sum_i \sum_t PPXT_{it} + \sum_i \sum_u \sum_t PPY_{iut} \right)$$

where, M is a big number. The revised reactive disruption management model (RRDM) comprises eqs. 7.3-7.52 with all variables that have the same values as the initial schedule before period *dt*. The objective of RRDM is to obtain good schedules to accommodate disruptions, while minimizing schedule changes from the initial schedule simultaneously.

Now, the revised reactive disruption management model (RRDM) is used to solve Example 1 for parcel 7 arrival delay. Profit of \$ 1838.58K is obtained in 4.7 CPU s. The obtained schedule is given in Table 7.4. The only differences of Tables 7.4 and 7.2 are: parcel 7 is unloaded to tank 7 in period 10, not to tank 6 in period 7 and tank 7 is charging CDU 3 during periods 14 and 15. Therefore, the obtained schedule with RRDMM has small difference to the initial schedule. The operators can react to the disruptions quickly in real situation without any confusion. The proposed RRDMM model is used to solve other disruptions presented by Arief et al. (2007a). They are:

(2) Tank 4 is unavailable from periods 2 to 4, informed at the end of period 1.

(3) Demand of CDU 3 increases from 400 kbbl to 450 kbbl, informed at the end of period 4.

(4) The VLCC is delayed by three periods, informed at the end of period 1.

(5) Tank 2 becomes unavailable in periods 4-6, and concurrently the demand of CDU 2 increases from 400 kbbl to 440 kbbl, informed at the end of period 2.

Table 7.4 Proposed schedule with RRDM for Example 1–parcel 7 delayed to the end of period 7, informed at the end of period 4 (Profit = $ 1838.58K)

	Crude Amount [to CDU No.] (from Vessel No.) in kbbl for Period														
Tank	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	−20.0[3]	−20.0[3]	+10.0(1)	+100(3)	+10.0(5)	+10(5)									
2			+90.0(2)	+10.0(2)		−20.0[2]	−20.0[2]	−50.0[1]	−20.0[1]	−50.0[1]	−50.0[1]	−30.0[1]	−20.0[1]	−20.0[1]	−20.0[1]
3					+80.0(4)	+10(6)									
4	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]								
	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]			−7.50[2]	−7.50[2]	−7.50[2]	−7.50[2]	−7.50[2]	−7.50[2]	−7.50[2]	−7.50[2]
5					+10.0(4)	+115(6)		−25.0[2]	−25.0[2]	−25.0[2]	−25.0[2]	−25.0[2]	−25.0[2]	−25.0[2]	−25.0[2]
6						+105(5)			−20.0[3]	−20.0[3]	−30.0[3]	−20.0[3]	−50.0[3]	−0.00[3]	−0.00[3]
7			−20.0[3]	−3.20[3]	−3.20[3]	−3.20[3]	−3.20[3]	−3.20[3]		+100(7)				−50.0[3]	−50.0[3]
8				−16.8[3]	−16.8[3]	−16.8[3]	−16.8[3]	−16.8[3]							

'–' sign represents delivery to [CDU], '+' sign represents receipt from (Parcels)

Table 7.5 Proposed schedule with RRDM for Example 1–Tank 4 unavailable from period 2-4, informed at the end of period 1 (Profit = $ 1834.26K)

	Crude Amount [to CDU No.] (from Vessel No.) in kbbl for Period														
Tank	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	−20.0[3]	−20.0[3]	+10.0(1)	+100(3)	+17.5(5)	+17.5(5)									
2			+10.0(2)	+90.0(2)		−50.0[2]	−50.0[2]	−20.0[1]	−20.0[1]	−20.0[1]	−30.0[1]	−20.0[1]	−50.0[1]	−20.0[1]	−20.0[1]
3		−50.0[1]	−50.0[1]	−20.0[1]	+80.0(4)	+115.0(6)									
		−20.0[2]	−20.0[2]	−40.0[2]											
4	−20.0[1]				−20.0[1]	−20.0[1]	−20.0[1]								
	−20.0[2]				−20.0[2]			−22.5[2]	−22.5[2]	−22.5[2]	−22.5[2]	−22.5[2]	−22.5[2]	−22.5[2]	−22.5[2]
5					+10.0(4)	+10.0(6)		−00.0[2]	−00.0[2]	−00.0[2]	−00.0[2]	−00.0[2]	−00.0[2]	−00.0[2]	−00.0[2]
6						+90.0(5)	+100(7)		−50.0[3]	−50.0[3]	−50.0[3]	−30.0[3]	−20.0[3]	−20.0[3]	−20.0[3]
7			−20.0[3]	−00.0[3]	−00.0[3]	−00.0[3]	−00.0[3]	−00.0[3]							
8				−20.0[3]	−20.0[3]	−20.0[3]	−20.0[3]	−20.0[3]							

'–' sign represents delivery to [CDU], '+' sign represents receipt from (Parcels)

Table 7.6 Proposed schedule with RRDM for Example 1–CDU 3 demand increases from 400 to 450, informed at the end of period 4 (Profit = $ 1930.80K)

	Crude Amount [to CDU No.] (from Vessel No.) in kbbl for Period														
Tank	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	−20.0[3]	−20.0[3]	+10.0(1)	+100(3)	+10.0(5)	+10.0(5)									
2			+90.0(2)	+10.0(2)				−50.0[1]	−20.0[1]	−50.0[1]	−20.0[1]	−50.0[1]	−20.0[1]	−20.0[1]	−30.0[1]
						−20.0[2]	−20.0[2]								
3					+80.0(4)	+10.0(6)									
4	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]								
	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]			−7.50[2]	−7.50[2]	−7.50[2]	−7.50[2]	−7.50[2]	−7.50[2]	−7.50[2]	−7.50[2]
5					+10.0(4)	+115(6)		−25.0[2]	−25.0[2]	−25.0[2]	−25.0[2]	−25.0[2]	−25.0[2]	−25.0[2]	−25.0[2]
6						+105(5)	+100(7)		−15.0[3]	−50.0[3]	−50.0[3]	−50.0[3]	−50.0[3]	−20.0[3]	−20.0[3]
7			−20.0[3]	−3.20[3]	−3.20[3]	−3.20[3]	−3.20[3]	−3.20[3]							
8				−16.8[3]	−16.8[3]	−16.8[3]	−16.8[3]	−16.8[3]	−35.0[3]						

'–' sign represents delivery to [CDU], '+' sign represents receipt from (Parcels)

Table 7.7 Proposed schedule with RRDM for Example 1−VLCC delayed by 3 periods, informed at the end of period 1 (Profit = $ 1930.80K)

	Crude Amount [to CDU No.] (from Vessel No.) in kbbl for Period														
Tank	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	−20.0[3]	−20.0[3]			+35(5)										
2								−50.0[1]	−50.0[1]	−50.0[1]	−50.0[1]	−0.0[1]	−0.0[1]	−0.0[1]	−0.0[1]
							−20[2]								
3								+90(4)							
4	−20.0[1]		−50.0[1]	−50.0[1]			−5.0[1]								
	−20.0[2]							−3.75[2]	−3.75[2]	−3.75[2]	−3.75[2]	−3.75[2]	−3.75[2]	−3.75[2]	−3.75[2]
5			−37.5[1]	−37.5[1]		+125(6)		−43.75[2]	−43.75[2]	−43.75[2]	−43.75[2]	−43.75[2]	−43.75[2]	−43.75[2]	−43.75[2]
						+100(2)									
6						+90(5)	+100(7)		−50.0[3]	−50.0[3]	−10.0[3]	−50.0[3]	−0.0[3]	−0.0[3]	−0.0[3]
						+10(1)	+100(3)								
7			−0.0[3]	−0.0[3]	−0.0[3]	−0.0[3]	−3.2[3]	−3.20[3]							
8				−40[3]	−40[3]	−40[3]	−40[3]	−40[3]							

'–' sign represents delivery to [CDU], '+' sign represents receipt from (Parcels)

Table 7.8 Proposed schedule with RRDM for Example 1−Tank 2 unavailable in periods 4-6, and concurrently the demand of CDU 2 increases from 400 kbbl to 440 kbbl, informed at the end of period 2 (Profit = $ 1895.08K)

	Crude Amount [to CDU No.] (from Vessel No.) in kbbl for Period														
Tank	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	−20.0[3]	−20.0[3]	+10.0(1)	+100(3)	+25(5)	+10(5)									
2						−50.0[2]	−20.0[2]	−20.0[1]	−20.0[1]	−20.0[1]	−30.0[1]	−20.0[1]	−20.0[1]	−20.0[1]	−20.0[1]
3			−50.0[1]	−50.0[1]	+80.0(4)	+115.0(6)									
4	−20.0[1]	−20.0[1]			−20.0[1]	−30.0[1]	−50.0[1]								
	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]	−20.0[2]			−00.0[2]	−00.0[2]	−00.0[2]	−00.0[2]	−00.0[2]	−00.0[2]	−00.0[2]	−00.0[2]
5			+100(2)		+10.0(4)	+10.0(6)		−37.5[2]	−37.5[2]	−37.5[2]	−37.5[2]	−37.5[2]	−37.5[2]	−37.5[2]	−37.5[2]
6						+90.0(5)	+100(7)		−50.0[3]	−30.0[3]	−50.0[3]	−20.0[3]	−50.0[3]	−20.0[3]	−20.0[3]
7			−20.0[3]	−00.0[3]	−00.0[3]	−00.0[3]	−00.0[3]	−00.0[3]							
8				−20.0[3]	−20.0[3]	−20.0[3]	−20.0[3]	−20.0[3]							

'–' sign represents delivery to [CDU], '+' sign represents receipt from (Parcels)

Table 7.9 Proposed approach vs. block preservation for Example 1

	Case 1			Case 2			Case 3		
Methods	Profit	Reschedule operations	CPU Time (s)	Profit	Reschedule operations	CPU Time (s)	Profit	Reschedule operations	CPU Time (s)
The Proposed Approach	1839	2	6.2	1834	2	3.4	1931	0	3.8
Block Preservation	1846	4	1	1837	4	1	1935	0	1

	Case 4			Case 5		
Methods	Profit	Reschedule operations	CPU Time (s)	Profit	Reschedule operations	CPU Time (s)
The Proposed Approach	1824	7	9.42	1895	3	6.7
Block Preservation	1847	8	1	1895	4	1

Block Preservation: Arief et al. (2007a)
Case 1: Parcel 7 delayed to the end of period 7, informed at the end of period 4
Case 2: Tank 4 unavailable from periods 2 to 34 informed at the end of period 1
Case 3: CDU 3 demand increases from 400 to 450, informed at the end of period 4
Case 4: VLCC delayed by 3 periods, informed at the end of period 1
Case 5: Tank 2 unavailable from periods 4 to 6 and CDU 2 demand increases from 400 to 440 simultaneously, informed at the end of period 4

Table 7.10 An alternative initial schedule for Example 1

	Crude Amount [to CDU No.] (from Vessel No.) in kbbl for Period														
Tank	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1			+10(1)		+125(5)				–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3]
2						–20[1]	–20[1]	–20[1]							
	–20[2]	-20[2]	–20[2]	–20[2]	–20[2]	–20[2]	–20[2]								
3	–28.75[1]	–28.75[1]	–28.75[1]	–28.75[1]	–28.75[1]	+125(6)									
								–13.28[2]	–13.28[2]	–13.28[2]	–13.28[2]	–13.28[2]	–13.28[2]	–13.28[2]	–13.28[2]
4					+40(4)										
5			+100(2)		+50(4)				–20[1]	–20[1]	–20[1]	-50[1]	-46.25[1]	–20[1]	–20[1]
								–19.22[2]	–19.22[2]	–19.22[2]	–19.22[2]	–19.22[2]	–19.22[2]	–19.22[2]	–19.22[2]
6	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3]
7				+100(3)				+100(7)							
8	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]							

'–' sign represents delivery to [CDU], '+' sign represents receipt from (Parcels)

Tables 7.5-7.8 give the obtained schedules from RRDM for disruptions 2-4, respectively. The performance of the proposed approach (RRDM) and block preservation proposed by Arief et al. (2007a) is shown in Table 7.9. Although the proposed approach obtained worse solution than block preservation, it needs fewer reschedule operations to accommodate those disruptions. For instance, the proposed approach needs 2 reschedule operations for the second disruption, while block preservation needs 4 reschedule operations.

7.4.2 Example 2

This example has the same data with Example 1, but has different initial schedule (Table 7.10). With this initial schedule, five disruptions are addressed as follows.

(1) The refinery is informed at the end of period 4 that the ship with parcel 6 will arrive at the beginning of period 7, delayed 16 hours because of bad weather.

(2) The refinery is informed at the end of period 1 that tank 2 will be unavailable from period 2 to period 3 because of dewatering.

(3) The refinery is informed at the end of period 4 that the demand of CDU 1 increases to 450 kbbl.

(4) The refinery is informed at the end of period 1 that the SBM pipeline will be unavailable from period 2 to period 5.

(5) The refinery is informed at the end of period 2 that tank 3 will be unavailable from period 4 to period 5 and the demand of CDU 2 will decrease to 387.5 kbbl.

Table 7.11 Proposed schedule with RRDM for Example 2–parcel 6 delayed two periods, informed at the end of period 4 (Profit = $ 1833.35K)

	Crude Amount [to CDU No.] (from Vessel No.) in kbbl for Period														
Tank	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1			+10(1)		+125(5)				–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3]
2						–20[1]	–20[1]	–20[1]	+125(6)						–38.96[1]
	–20[2]	–20[2]	–20[2]	–20[2]	–20[2]	–20[2]	–20[2]								
3	–28.75[1]	–28.75[1]	–28.75[1]	–28.75[1]	–28.75[1]										
								–9.66[2]	–9.66[2]	–9.66[2]	–9.66[2]	–9.66[2]	–9.66[2]	–9.66[2]	–9.66[2]
4					+50(4)										
5			+100(2)		+40(4)				–20[1]	–20[1]	–46.25[1]	–20[1]	–20[1]	–20[1]	–11.04[1]
								–22.84[2]	–22.84[2]	–22.84[2]	–22.84[2]	–22.84[2]	–22.84[2]	–22.84[2]	–22.84[2]
6	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3]
7				+100(3)				+100(7)							
8	–19.2[3]	–19.2[3]	–19.2[3]	–19.2[3]	–19.2[3]	–19.2[3]	–19.2[3]	–19.2[3]							

'–' sign represents delivery to [CDU], '+' sign represents receipt from (Parcels)

Table 7.12 Proposed schedule with RRDM for Example 2–Tank 2 unavailable from periods 2 to 3, informed at the end of period 1 (Profit = $ 1833.00K)

	Crude Amount [to CDU No.] (from Vessel No.) in kbbl for Period														
Tank	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1			+10(1)		+125(5)				–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3]
2						-30[1]	-20[1]	-20[1]							
	–20[2]			–20[2]	–20[2]	–50[2]	–20[2]								
3	–28.75[1]	–28.75[1]	–28.75[1]	–28.75[1]	–28.75[1]	+125(6)									
								–5.78[2]	–5.78[2]	–5.78[2]	–5.78[2]	–5.78[2]	–5.78[2]	–5.78[2]	–5.78[2]
4		–50[2]	–50[2]		+80(4)										
5			+100(2)		+10(4)				-36.25[1]	–20[1]	–20[1]	–50[1]	–20[1]	–20[1]	–20[1]
								–15.47[2]	–15.47[2]	–15.47[2]	–15.47[2]	–15.47[2]	–15.47[2]	–15.47[2]	–15.47[2]
6	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3]
7				+100(3)				+100(7)							
8	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]							

'–' sign represents delivery to [CDU], '+' sign represents receipt from (Parcels)

Table 7.13 Proposed schedule with RRDM for Example 2–CDU 1 demand increases from 400 to 450, informed at the end of period 4 (Profit = $ 1910.40K)

	Crude Amount [to CDU No.] (from Vessel No.) in kbbl for Period														
Tank	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1			+10(1)		+125(5)				–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3]
2						–20[1]	–20[1]	–20[1]							
	–20[2]	–20[2]	–20[2]	–20[2]	–20[2]	–20[2]	–20[2]								
3	–28.75[1]	–28.75[1]	–28.75[1]	–28.75[1]	–28.75[1]	+125(6)									
								–20.78[2]	–20.78[2]	–20.78[2]	–20.78[2]	–20.78[2]	–20.78[2]	–20.78[2]	–20.78[2]
4					+50(4)										
5			+100(2)		+40(4)				–20[1]	–36.25[1]	–50[1]	–50[1]	–50[1]	–20[1]	–20[1]
								–11.72[2]	–11.72[2]	–11.72[2]	–11.72[2]	–11.72[2]	–11.72[2]	–11.72[2]	–11.72[2]
6	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3]
7				+100(3)				+100(7)							
8	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]							

'–' sign represents delivery to [CDU], '+' sign represents receipt from (Parcels)

Table 7.14 Proposed schedule with RRDM for Example 2–SBM pipeline unavailable from periods 2 to 5, informed at the end of period 1 (Profit = $ 1831.96K)

	Crude Amount [to CDU No.] (from Vessel No.) in kbbl for Period														
Tank	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1					+115(5)				–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3]
2						-20[1]		–40[1]							
	–20[2]	–20[2]	–20[2]	–20[2]	–20[2]	–20[2]	–20[2]								
3	–28.75[1]	–28.75[1]	–28.75[1]	–28.75[1]	–28.75[1]	+100(2)									
						+125(6)		–32.5[2]	–32.5[2]	–32.5[2]	–32.5[2]	–32.5[2]	–30[2]	–30[2]	-30[2]
4							-50[1]	+90(4)							
5									–20[1]	–20[1]	–20[1]	–20[1]	–20[1]	–26.25[1]	–20[1]
6	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3]
7						+10(1)		+100(3)							
						+10(5)		+100(7)							
8	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]							

'–' sign represents delivery to [CDU], '+' sign represents receipt from (Parcels)

Table 7.15 Proposed schedule with RRDM for Example 2–Tank 3 unavailable from periods 4 to 5 and demand of CDU 2 decreases from 400 to 387.5 simultaneously, informed at the end of period 2

	Crude Amount [to CDU No.] (from Vessel No.) in kbbl for Period														
Tank	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1			+10(1)		+125(5)				–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3]	–28.02[3
2				–20[1]	–20[1]	–20[1]									
	–20[2]	–20[2]	–20[2]	–20[2]	–20[2]	–20[2]	–20[2]								
3	–28.75[1]	–28.75[1]	–20[1]			+125(6)									
								–21.25[2]	–21.25[2]	–21.25[2]	–21.25[2]	–21.25[2]	–21.25[2]	–21.25[2]	–21.25[2
4					+50(4)										
5			+100(2)		+40(4)		–20[1]	–20[1]	–20[1]	–50[1]	–50[1]	–42.5[1]	–20[1]	–20[1]	–20[1]
								–9.69[2]	–9.69[2]	–9.69[2]	–9.69[2]	–9.69[2]	–9.69[2]	–9.69[2]	–9.69[2
6	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–0.77[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3]	–6.26[3
7				+100(3)				+100(7)							
8	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]	–19.23[3]							

'–' sign represents delivery to [CDU], '+' sign represents receipt from (Parcels)

Table 7.16 Proposed approach vs. block preservation for Example 2

	Case 1			Case 2			Case 3		
Methods	Profit	Reschedule operations	CPU Time (s)	Profit	Reschedule operations	CPU Time (s)	Profit	Reschedule operations	CPU Time (s)
The Proposed Approach	1833.348	2	3.217	1832.995	1	2.914	1910.403	0	1.466
Block Preservation	1830.634	2	1	1825.626	4	1	1908.68	0	1

	Case 4			Case 5		
Methods	Profit	Reschedule operations	CPU Time (s)	Profit	Reschedule operations	CPU Time (s)
The Proposed Approach	1831.957	7	5.328	1804.778	3	5.496
Block Preservation	1825.449	7	1	1797.311	6	1

Block Preservation: Arief et al. (2007a)
Case 1: Parcel 6 delayed two periods, informed at the end of period 4
Case 2: Tank 2 unavailable from periods 2 to 3, informed at the end of period 1
Case 3: CDU 1 demand increases from 400 to 450, informed at the end of period 4
Case 4: SBM pipeline unavailable from periods 2 to 5, informed at the end of period 1
Case 5: Tank 3 unavailable from periods 4 to 5 and demand of CDU 2 decreases from 400 to 387.5 simultaneously, informed at the end of period 2

The proposed schedules after the above five disruptions happen are presented in Tables 7.11-7.15. The block preservation (heuristic rescheduling) approach of Arief et al. (2007a) is also used to solve these five disruptions. The performance of the proposed approach and the approach from Arief et al. (2007a) is given in Table 7.16. The proposed approach obtains better solutions with fewer reschedule operations than that of Arief et al. (2007a). However, the computational time does not increase too much. For instance, the proposed approach needs 5.50 CPU s for the fifth disruption compared to 1 s of block preservation of Arief et al. (2007a).

From Examples 1 and 2, the proposed approach needs fewer reschedule operations to address different disruptions than block preservation from Arief et al. (2007a). However, it cannot guarantee better solution than block preservation. Thus, it largely depends on the initial schedule. Therefore, it is very critical to develop methods to obtain robust initial schedule, which is the main objective of the remaining of this Chapter

7.5 Robustness Definition and Evaluation

Gan and Wirth (2004) defined schedule robustness as schedule effectiveness, performance predictability and rescheduling stability as follows,

Schedule effectiveness: This is the objective function of the perturbed schedule, which is the schedule generated after a rescheduling is performed in reaction to a disruption. In other words, this effectiveness means the feasibility of the perturbed schedule.

Performance predictability: This is the deviation of the objective of the perturbed schedule from the initial schedule, which is the schedule for deterministic case (i.e without any disruption).

Rescheduling stability: This is the sum of all changes in a perturbed schedule with respect to the initial schedule.

In this problem, CDU shutdown or demand shortfalls are allowed to ensure feasibilities to keep schedule effectiveness. For rescheduling stability, schedule changes (defined later) including parcel-SBM/Jetty connection, SBM/Jetty-Tank connection and Tank-CDU connection changes are allowed. To measure schedule effectiveness and rescheduling stability, different penalties are imposed and incorporated into the objective function. To measure performance predictability, empirical robustness index (RI) is proposed as follows to denote the objective deviation. Now, the following procedure is proposed to evaluate the robustness of obtained schedules:

1. Simulate S random disruptions.
2. Calculate their corresponding probabilities–P_s based on some assumed distribution.
3. Obtain optimal schedule for each scenario–$Profit_{opt,s}$ by using the algorithm proposed in Chapter 3.
4. Adjust the obtained schedule to address each scenario by using the approach developed in section 7.4 and compute $Rprofit_s$ as follows,

$$\text{Rprofit}_s = \text{Profit}_s - \text{Penalties}_s$$

5. Calculate empirical robustness index for schedules,

$$\mathrm{RI} = 1 - \sum_{s=1}^{S} \mathrm{P}_s \frac{\mathrm{Profit}_{opt,s} - \mathrm{Rprofit}_s}{\mathrm{Profit}_{opt,s}}$$

Next, a scenario-based formulation is developed for demand uncertainty to obtain robust schedule. The obtained schedule is evaluated with the above proposed procedure and compared with the initial schedule.

7.6 Demand Uncertainty

To improve the deterministic schedule, demand uncertainty is considered at the scheduling stage. This is achieved by developing a scenario-based formulation involving different demand scenarios within the expected range of demand variability. In this scenario-based formulation, all binary decision variables (i.e. XP_{pt}, XT_{it}, Y_{iut}) are treated as here and now and other continuous decision variables like FPT_{pit}, FTU_{iut} as wait and see. In other words, only one more index s denoting scenarios is added to those continuous decision variables, while all binary variables are independent of scenarios. The objective is to maximize the average profit over all scenarios with the assumption that all scenarios have the same probability. The mathematical model of the scenario-based formulation is given as follows:

Parcel-to-SBM Connections

$$XP_{pt} = XP_{p(t-1)} + XF_{pt} - XL_{p(t-1)} \qquad (p, t) \in \boldsymbol{PT} \qquad (7.53)$$

$$XP_{pt} \geq XL_{pt} \qquad (p, t) \in \boldsymbol{PT} \qquad (7.54)$$

$$\sum_t XF_{pt} = 1 \qquad (p, t) \in \boldsymbol{PT} \qquad (7.55)$$

$$\sum_t XL_{pt} = 1 \qquad (p, t) \in \boldsymbol{PT} \qquad (7.56)$$

$$TF_p = \sum_t (t-1) \cdot XF_{pt} \qquad (p, t) \in \boldsymbol{PT} \qquad (7.57)$$

$$TL_p = \sum_t t \cdot XL_{pt} \qquad (p, t) \in \boldsymbol{PT} \qquad (7.58)$$

$$\sum_p XP_{pt} \le 2 \qquad (p, t) \in \boldsymbol{PT} \tag{7.59}$$

$$TF_{(p+1)} \ge TL_p - 1 \tag{7.60}$$

$$TF_p \ge ETA_p \tag{7.61}$$

SBM-to-Tank Connections

$$\sum_i XI_{it} \le 2 \tag{7.62}$$

$$X_{pit} \ge XP_{pt} + XT_{it} - 1 \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI} \tag{7.63}$$

$$\sum_i X_{pit} \le 2 \cdot XP_{pt} \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI} \tag{7.64}$$

$$\sum_p X_{pit} \le 2 \cdot XT_{it} \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI} \tag{7.65}$$

$$\sum_p \sum_i X_{pit} \le 2 \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI} \tag{7.66}$$

Tank-to-CDU Connections

$$\sum_u Y_{iut} \le 2 \qquad (i, u) \in \boldsymbol{IU} \tag{7.67}$$

$$\sum_i Y_{iut} \le 2 \qquad (i, u) \in \boldsymbol{IU} \tag{7.68}$$

$$2XT_{it} + Y_{iut} + Y_{iu(t+1)} \le 2 \qquad (i, u) \in \boldsymbol{IU} \tag{7.69}$$

Crude Delivery and Processing

$$FPT_{pi}^L X_{pit} \le FPT_{pits} \le FPT_{pi}^U X_{pit} \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI} \tag{7.70}$$

$$\sum_p \sum_i \frac{FPT_{pits}}{FPT_{pi}^U} \le 1 \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI} \tag{7.71}$$

$$\sum_{i,t} FPT_{pits} = PS_{ps} \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI} \tag{7.72}$$

$$FTU_{iuts} = \sum_{(i,c)\in IC} FCTU_{iucts} \qquad (i, u) \in \boldsymbol{IU} \tag{7.73}$$

$$Y_{iut} FTU_{iu}^L \le FTU_{iuts} \le Y_{iut} FTU_{iu}^U \qquad (i, u) \in \boldsymbol{IU} \tag{7.74}$$

$$FU_{uts} = \sum_{(i,u)\in IU} FTU_{iuts} \tag{7.75}$$

$$FU_{ut}^L \le FU_{uts} \le FU_{ut}^U \tag{7.76}$$

$$\gamma_u^L FU_{uts} \le FU_{u(t+1)s} \le \gamma_u^U FU_{uts} \tag{7.77a,b}$$

$$FU_{uts} \cdot xc_{cu}^{L} \leq \sum_{i} FCTU_{iucts} \leq FU_{uts} \cdot xc_{cu}^{U} \qquad (i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC} \qquad (7.78a,b)$$

$$xk_{ku}^{L} FU_{uts} \leq \sum_{i} \sum_{c} FCTU_{iucts} xk_{kc} \leq xk_{ku}^{U} FU_{uts} \qquad (i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC} \qquad (7.79a,b)$$

$$\theta_{ku}^{L} \left(\sum_{i} \sum_{c} FCTU_{iucts} \rho_{c} \right) \leq \sum_{i} \sum_{c} FCTU_{iucts} \rho_{c} \theta_{kc} \leq \theta_{ku}^{U} \left(\sum_{i} \sum_{c} FCTU_{iucts} \rho_{c} \right)$$

$$(i, u) \in \boldsymbol{IU}, (i, c) \in \boldsymbol{IC} \qquad (7.80a,b)$$

$$YY_{iut} \geq Y_{iut} + Y_{iu(t+1)} - 1 \qquad (i, u) \in \boldsymbol{IU} \qquad (7.81a)$$

$$YY_{iut} \leq Y_{iu(t+1)} \qquad (i, u) \in \boldsymbol{IU} \qquad (7.81b)$$

$$YY_{iut} \leq Y_{iut} \qquad (i, u) \in \boldsymbol{IU} \qquad (7.81c)$$

$$CO_{ut} \geq Y_{iut} + Y_{iu(t+1)} - 2YY_{iut} \qquad (i, u) \in \boldsymbol{IU} \qquad (7.82)$$

$$M[2 - \sum_{i} YY_{iut}] + FTU_{iuts} \geq FTU_{iu(t+1)s} \qquad (i, u) \in \boldsymbol{IU} \qquad (7.83a)$$

$$M[2 - \sum_{i} YY_{iut}] + FTU_{iu(t+1)s} \geq FTU_{iuts} \qquad (i, u) \in \boldsymbol{IU} \qquad (7.83b)$$

Crude Inventory

$$VCT_{icts} = VCT_{ic(t-1)s} + \sum_{(p,c)\in PC,(p,t)\in PT} FPT_{pits} - \sum_{(i,u)\in IU} FCTU_{iucts} \qquad (i, c)\in \boldsymbol{IC} \qquad (7.84)$$

$$V_{its} = \sum_{(i,c)\in IC} VCT_{icts} \qquad (7.85)$$

$$V_{i}^{L} \leq V_{its} \leq V_{i}^{U} \qquad (7.86)$$

$$xt_{ic}^{L} V_{it} \leq VCT_{icts} \leq xt_{ic}^{U} V_{it} \qquad (7.87)$$

$$FCTU_{iucts} = f_{icts} \cdot FTU_{iuts} \qquad (7.88)$$

$$VCT_{icts} = f_{icts} \cdot V_{its} \qquad (7.89)$$

Production Requirements

$$\sum_{t} FU_{uts} = D_{us} \qquad (7.90)$$

Period-period Crude Feed Changes

$$\gamma_{u}^{L} FU_{uts} \leq FU_{u(t+1)s} \leq \gamma_{u}^{U} FU_{uts} \qquad (7.91a,b)$$

Demurrage Cost

$$DC_v \geq (TL_p - ETA_p - ETD_v)SWC_v \qquad (p, v) \in \boldsymbol{PV} \qquad (7.92)$$

Stock Penalty

$$SC_{ts} \geq SSP(SS - \sum_t V_{its}) \qquad (7.93)$$

In order to extend the above formulation to jetties, they made some modifications to the above formulation. ***J*** is the number of jetties.

Drop eqs. 7.60 and 7.62 and modify eqs. 7.59 and 7.66 as follows,

$$\sum_p XP_{pt} \leq J \qquad (p, t) \in \boldsymbol{PT} \qquad (7.94)$$

$$\sum_p \sum_t X_{pit} \leq 2J \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI} \qquad (7.95)$$

Replacing eq. 7.71 by,

$$\sum_i \frac{FPT_{pits}}{FPT_{pi}^U} \leq 1 \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI} \qquad (7.96)$$

$$\sum_p \frac{FPT_{pits}}{FPT_{pi}^U} \leq 1 \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI} \qquad (7.97)$$

For both SBM pipeline and Jetties, eq. 7.60 is effective for SBM parcel as follows,

$$TF_{(p+1)} \geq TL_p - 1 \qquad p \in \boldsymbol{SP} \qquad (7.98)$$

Eqs. 7.71, 7.96 and 7.97 are replaced with the following constraints.

$$\sum_i \frac{FPT_{pits}}{FPT_{pi}^U} \leq 1 \qquad (p, t) \in \boldsymbol{PT}, (p, i) \in \boldsymbol{PI} \qquad (7.99)$$

$$\sum_{p \in SP} \frac{FPT_{pits}}{FPT_{pi}^U} + \sum_{p \in JP} \frac{FPT_{pits}}{FPT_{pi}^U} \leq 1 \qquad (7.100)$$

Where, ***SP*** denotes SBM parcel and ***JP*** denotes jetty parcel.

The objective is:

$$\text{Profit} = \sum_i \sum_u \sum_c \sum_t \sum_s FCTU_{iucts} CP_{cu} - \sum_v DC_v - COC \sum_u \sum_t CO_{ut} - \sum_t \sum_s SC_{ts}$$

Table 7.17 Data for Example 3

Tanker	Arrival Period	Parcel No: (Crude, Parcel Size kbbl)
VLCC-1	2	1: (C2, 10), 2: (C1, 300), 3: (C4, 300), 4: (C3, 340)

Tank & CDU	Capacity (kbbl)	Heel (kbbl)	Initial Inventory (kbbl)	Allowable Crude (Class)	Initial Crude Amount (kbb)		Crude Concentration Range (Min-Max)		Total Demand (kbbl)	Processing Limits (kbbl/period) Min-Max	Property Specification Range (Min-Max)
					C1 or C3	C2 or C4	C1 or C3	C2 or C4			
T1	700	50	300	C1, C2(1)	200	100	0.1-0.9	0.1-0.9	-	-	-
T2	700	50	300	C3, C4(2)	100	200	0.0-1.0	0.0-1.0	-	-	-
T3	900	50	250	C3, C4 (2)	50	250	0.0-1.0	0.0-1.0	-	-	-
T4	700	50	300	C1, C2 (1)	130	170	0.1-0.9	0.1-0.9	-	-	-
CDU 1	-	-	-	C1, C2 (1)	-	-	0.2-0.8	0.2-0.8	400	20-80	0.004-0.006
CDU 2	-	-	-	C3, C4 (2)	-	-	0.0-1.0	0.0-1.0	400	20-80	0.014-0.0158

Crude	Property Specification	Margin ($/bbl)
C1	0.0040	3.000
C2	0.0065	4.500
C3	0.0165	5.000
C4	0.0145	6.000

Parcel-Tank flow rate: 10-400 kbbl/period
Demurrage cost: 100 k$/period
Safety stock penalty: 0.2 $/bbl/period

Tank-CDU flow rate: 0-100 kbbl/period
Changeover loss: 5 k$/instance
Desired safety stock: 1200 kbbl

Table 7.18 An initial schedule for Example 3

	Period								
Tank	1	2	3	4	5	6	7	8	9
1	+10(1) +100(2)					-37.5[1]	-37.5[1]	-37.5[1]	-37.5[1]
2		+300(3)		-33.447[2]	-40.137[2]	-48.164[2]	-57.797[2]	-69.356[2]	-80[2]
3	-20[2]	-23.227[2]	-27.873[2]	+340(4)					
4	-22.995[1]	-27.594[1]	-33.113[1]	-39.736[1]	-47.683[1]	-19.72[1]	-19.72[1]	-19.72[1]	-19.72[1]

'–' sign represents delivery to [CDU], '+' sign represents receipt from (Parcels)

Table 7.19 Proposed robust schedule for Example 3

	Period								
Tank	1	2	3	4	5	6	7	8	9
1	+10(1) +300(2)				−30[1]	−30[1]	−30[1]	−30[1]	−30[1]
2		+300(3)		−33.45[2]	−40.14[2]	−48.16[2]	−57.80[2]	−69.36[2]	−80[2]
3	−20[2]	−23.23[2]	−27.87[2]	+340(4)					
4	−25.42[1]	−30.50[1]	−36.60[1]	−43.93[1]	−22.71[1]	−22.71[1]	−22.71[1]	−22.71[1]	−22.71[1]

'–' sign represents delivery to [CDU], '+' sign represents receipt from (Parcels)

Table 7.20 Computation result for Example 3

Scenario	Demand of CDU 1	Demand of CDU 2	Optimal Profit	Rprofit from Robust Schedule	Rprofit from Initial Schedule
1	250	250	2420.23	2408.83	2408.83
2	250	300	2714.30	2700.50	2700.50
3	250	350	3010.17	2992.03	2992.03
4	250	400	3303.90	3283.46	3283.46
5	250	450	3596.78	3574.78	3574.78
6	250	500	3889.88	3865.97	3865.97
7	250	550	4182.72	4157.05	4157.05
8	300	250	2578.75	2572.36	2572.36
9	300	300	2874.17	2864.02	2864.02
10	300	350	3168.69	3155.56	3155.56
11	300	400	3462.43	3446.99	3446.99
12	300	450	3755.30	3738.30	3738.30
13	300	500	4048.40	4029.50	4029.50
14	300	550	4341.25	4320.57	4320.57
15	350	250	2742.28	2735.88	2735.88
16	350	300	3037.70	3027.54	3027.55
17	350	350	3332.22	3319.08	3319.08
18	350	400	3625.95	3610.51	3610.51
19	350	450	3918.83	3901.82	3901.83
20	350	500	4211.93	4193.02	4193.02
21	350	550	4504.77	4484.09	4484.10
22	400	250	2905.80	2899.41	2899.41
23	400	300	3201.22	3191.07	3191.07
24	400	350	3495.74	3482.61	3482.61
25	400	400	3789.47	3774.04	3774.04
26	400	450	4082.35	4065.35	4065.35
27	400	500	4375.45	4356.55	4356.55
28	400	550	4658.30	4637.62	4637.62
29	450	250	3069.33	3062.93	3062.93
30	450	300	3364.75	3354.59	3354.59
31	450	350	3659.26	3646.13	3646.13
32	450	400	3952.99	3937.56	3937.56
33	450	450	4245.88	4228.87	4228.87
34	450	500	4528.98	4510.07	4510.07
35	450	550	4811.82	4791.14	4791.14
36	500	250	3232.85	3226.46	2851.35
37	500	300	3528.27	3518.12	3143.01
38	500	350	3822.79	3809.66	3434.55
39	500	400	4116.52	4101.09	3725.98
40	500	450	4399.40	4382.40	4016.36
41	500	500	4682.50	4663.60	4297.56
42	500	550	4965.35	4943.85	4578.63
43	550	250	3396.37	3389.98	2601.35
44	550	300	3691.80	3681.64	2893.01
45	550	350	3986.31	3973.18	3184.55
46	550	400	4270.05	4254.61	3475.98
47	550	450	4552.93	4535.92	3766.36
48	550	500	4836.03	4817.12	4047.56
49	550	550	5109.37	5088.69	4328.63

where, $FCTU_{iucts}$ is the amount of crude c from tank i charged to CDU u during period t under scenario s. CP_{cu} is the margin ($ per unit volume) of crude c processed in CDU u. COC denote the cost per changeover. CO_{ut} denotes a changeover of a CDU u during period t. SC_{ts} denotes safety stock penalty for period t under scenario s. Next, a small example is used to illustrate the proposed model.

7.6.1 Example 3

In this example, a refinery has one VLCC with four crudes, four storage tanks, and two CDUs. The scheduling horizon is 9 periods. The nominal demands of CDU 1 and CDU 2 are 400 kbbl and 400 kbbl, respectively. The detailed data is shown in Table 7.17. Table 7.18 gives an initial schedule for this example. Now those two demands vary within [250kbbl, 550kbbl] uniformly. Five scenarios involving the four vertexes and the nominal demand [i.e. (250kbbl, 250kbbl), (250kbbl, 550kbbl), (550kbbl, 250kbbl), (550kbbl, 550kbbl), and (400kbbl, 400kbbl)] are used to obtain the robust schedule, which is shown in Table 7.19. To evaluate the robustness of the obtained schedule and the nominal schedule, 49 scenarios uniformly distributed within the range of demand variability ([250 kbbl, 550 kbbl] for two demands) are produced. If infeasibility happens, crudes are purchased from spot market. Table 7.20 shows the result. From the proposed procedure, RI is 0.956 for the initial schedule, while it is 0.996 for the obtained schedule. It means that the obtained schedule is more robust than the initial schedule. Moreover, it is found that the obtained schedule is feasible over the entire demand uncertainty range, while the initial schedule is infeasible over some part of the uncertainty range. For instance, for the thirty-sixth scenario, the profit from robust schedule is about $ 3226.46K, while it is $ 2851.35K from the initial schedule. If crudes are not purchased from spot market, then it is infeasible for this scenario from the initial schedule. The same situation arises for scenarios 37-49. In other words, it cannot accommodate demand uncertainties for scenarios 36-49 by using the initial schedule, if crudes are not purchased from spot market.

7.7 Summary

In this Chapter, reactive and predictive approaches are used to address uncertainties during crude oil scheduling. Reactive approach is first used to deal with several disruptions and compared with the heuristic approach (block preservation) proposed by Arief et al. (2007a). It is shown that the new schedule obtained with the proposed approach and block preservation (heuristic-based) proposed by Arief et al. (2007a) largely depended on the quality of the initial schedule. Then, schedule robustness was defined and a procedure was proposed to evaluate schedule robustness. A scenario-based model was developed to address demand uncertainty. The result shows that the resulting schedule were superior to the initial schedule. In the future, the proposed schedule evaluation procedure and scenario-based model will be refined further for demand uncertainty and used to address more uncertainties like ship arrival delay.

CHAPTER 8

CONCLUSIONS AND RECOMMENDATIONS

8.1 Conclusions

This thesis addressed three aspects of refinery operations. These are crude oil scheduling, scheduling of gasoline blending and distribution, and disruptions during crude oil operations.

In crude oil scheduling operations, fifteen crude properties that are critical to crude distillation and downstream processing were identified. The practical utility of Reddy et al. (2004b)'s MINLP formulation was enhanced by adding appropriate linear blending correlations for these properties. Based on the detailed analysis of Reddy et al. (2004)'s algorithm, it is found that progressive fixing of some infeasible combinations of binary variables led to infeasibility and the algorithm lacked a mechanism to retract from these infeasible combinations. Therefore, a minimal set of binary variables responsible for infeasibility was identified and a backtracking strategy using an intelligent integer cut to eliminate the infeasibility and revive the algorithm's progress was developed. The robustness of the improved algorithm was evaluated using twenty examples of different sizes and with different real life operation features. Its performance with other three algorithms (DICOPT; Li et al., 2002; Reddy et al., 2004b) was compared. A general-purpose code such as DICOPT failed to solve most problems and was horribly slow in solving the rest. Even the best algorithm of Reddy et al.

(2004b) fails to solve several problems. In contrast, our improved algorithm works on all problems and is much more efficient than the other three algorithms. To further increase solution speed, a partial relaxation method in which we relax the integrality restrictions on the binary variables of limited use was developed. The tests show that the partial relaxation method greatly reduced the computation time and at the same time improved the solution quality for most examples, especially for scheduling problems with horizons as long as 20 days. It gave schedules with profits within 6% of a conservative upper bound. Moreover, it is also much faster (by a factor of nearly five on an average) than an NLP-based algorithm devised by improving the algorithm of Li et al. (2002). In addition, Reddy et al. (2004b)'s formulation was also revised to ensure practically realistic schedules with limited flow rate changes to the CDUs.

Secondly, the improved algorithm above was extended to other types of refineries such as in-land refineries although the revised algorithm is intended for a marine-access refinery. The complete formulation incorporating storage tanks, charging tanks and all other operational features such as SBM/SPM, VLCCs, and Jetties was also developed. The results show that the algorithmic strategy is successfully applicable to other types of refineries such as in-land refineries.

Thirdly, a general slot-based MINLP formulation for an integrated treatment of recipe, specifications, blending, storage, and distribution was developed and many real-life features such as multi-purpose tanks, parallel non-identical blenders, constant rates during blending runs, minimum run lengths, changeovers, linear property indices, piecewise constant profiles for blend component qualities and feed rates, etc. were incorporated. Since commercial MINLP solvers are unsatisfactory for solving this complex MINLP, a novel and efficient procedure that solves successive MILPs instead of an MINLP, and gives excellent solutions was developed. Fourteen examples of varying sizes and features were used to illustrate the superiority and effectiveness of our formulation and solution approach.

Fourthly, an efficient continuous-time formulation using unit slots was developed for integrating blending and distribution of gasoline operation. This formulation incorporated all real-life operation features of process-slot model in Chapter 5 such as multi-purpose tanks, parallel non-identical blenders, constant rates during blending runs, minimum run lengths, changeovers, piecewise constant profiles for blend component qualities and feed rates, etc. By solving the fourteen examples from Chapter 5, it shows that the proposed unit-slot based model obtained the same or better solutions than the process-slot model in the third part with fewer binary variables and less computational time.

Finally, a novel approach was developed for reactive crude oil scheduling and compared with heuristic approach proposed by Arief et al. (2004, 2007a). The results show that the proposed approach needs fewer reschedule operations than that proposed by Arief et al. (2004, 2007a), but could not guarantee better solutions. It largely depended on the initial schedule. Then, schedule robustness was defined based on schedule effectiveness, predictability, and stability and a penalty function was used to measure schedule robustness. To evaluate the robustness of a schedule, a series of random disruptions was simulated, each schedule was adjusted to accommodate each disruption, and the penalty function for these adjustments was calculated. Based on this, a measure for evaluating robustness was defined. A scenario-based model was developed to obtain robust schedule under demand uncertainty. The result shows that the resulting schedule had high robustness than the initial schedule.

8.2 Recommendations

In this work, two sections in refinery supply chain, namely crude oil scheduling and scheduling of production blending and distribution operations, were taken into account. During the development and evaluation of model and algorithm, some key points, gaps can be observed. Combined with those observations, recommendations are presented as follows.

1. In Chapter 3, robust and efficient algorithms were developed for non-convex MINLP crude oil scheduling problem. The evaluations show that this improved algorithm cannot guarantee global optimal solutions because of the conservative lower bounds. Further work is now appearing (Karuppiah et al., 2006 and 2008) and is desirable on global optimization algorithms for solving this difficult scheduling problem. However, Karuppiah et al. (2006 and 2008) can only solve small-size problems with few real operational features. Therefore, development of new algorithms utilizing this problem property to improve the quality of lower bounds is very critical in the future. Some global optimization methods like branch and reduced algorithm, contract and branch algorithm, and Lagrangial method may be used to develop some additional efficient cuts to remove some feasible regions in which global solutions do not occur.
2. In Chapters 5-6, continuous-time MINLP using process and unit slots were developed for scheduling of gasoline blending and distribution operations. Although the developed models and algorithms can be used to solve many examples with many practical operation features and unit-slot model improved the performance of process-slot model, large integrality gaps for SPM or MPM were observed in large-size problems. Some methods such as decomposition method may be useful to reduce computational time for large-size problems in the future.

3. In Chapter 7, a scenario-based formulation was developed to obtain robust schedule for demand fluctuation. One example assuming demands vary uniformly in a certain range was solved to obtain robust schedule. In the future, this scenario-based formulation will be used to obtain robust schedules for stochastic demand following other distributions such as normal distribution. Moreover, many other disruptions such as ship arrival delay, tank unavailability and pipeline malfunction also happen frequently during crude oil scheduling operations. Obtaining robust schedules for those disruptions is still very challenging.
4. As shown in the Introduction, petroleum supply chain ranges from the production of crude oil to the distribution of products to customers. Besides the two sections considered in the work, other parts like intermediate processing scheduling, scheduling of pipeline distribution of multiple petroleum derivatives to customers, and supply and distribution of crudes and multiple products via marine logistics are also challenging. In addition to handling each section separately, the integration of the whole supply chain is substantially challenging because of large problem size. Optimization methods combined with simulation may be very useful for solving this integration problem.

REFERENCES

1. Abumaizar R. J., Svestka J. A., Rescheduling job shops under random disruptions, *International Journal of Production Research*, 35, pp. 2065-2082. 1997.
2. Adhitya, A., Srinivasan, R., Karimi, I. A., A heuristic reactive scheduling strategy for recovering from refinery supply chain disruptions, Presented at AIChE Annual Meeting, Austin, TX, Nov 7-12, 2004.
3. Adhitya, A, Srinivasan, R., Karimi, I. A., Heuristic rescheduling of crude oil operations to manage abnormal supply chain events, *AIChE Journal*, *53(2)*, pp.397-422. 2007a.
4. Adhitya, A, Srinivasan R., Karimi, I. A., A model-based rescheduling framework for managing abnormal supply chain events, *Computers and Chemical Engineering*, 31(5-6), pp. 496-518. 2007b.
5. Akturk M. S., Gorgulu, E., Theory and methodology-Match-up scheduling under a machine breakdown, *European Journal of Operational Research*, 112, pp. 81-97. 1999.
6. Arnold, V. E., Microcomputer Program Converts TBP, ASTM, EFV distillation curves, *Oil & Gas Journal*, 1985, 83(6), Feb 11, 55-62.
7. Aytug, H., Lawley, M. A., Mckay, K., Mohan, S., Uzsoy, R., Executing production schedules in the face of uncertainties: a review and some future directions, *European Journal of Operational Research*, 161, pp. 86-110. 2005.
8. Balas, E., Jeroslow, R., Canonical cuts on the unit hypercube, *SIAM Journal on*

Applied Mathematics, 23(1), pp. 61-69. 1972.

9. Balasubramanian, J., Grossmann, I. E., Scheduling optimization under uncertainty-an alternative approach, *Computers and Chemical Engineering*, 27(4), pp. 469–490. 2003.

10. Balasubramanian, J., Grossmann, I. E., Approximation to multistage stochastic optimization in multiperiod batch plant scheduling under demand uncertainty, *Industrial and Engineering Chemistry Research*, 43(14), pp. 3695–3713. 2004.

11. Ballintjin, K., Optimization in refinery scheduling: modeling and solution, In T. A. Ciriani, & R. C. Leachman, *Optimization in Industry*, 3, pp. 191-199, Wiley: New York. 1993.

12. Bodin, L., Golden, B., Assad, A., Ball, M., Routing and scheduling of vehicles and crews: the state of the art, *Computers and Operations Research*, 10(2), pp. 63-211. 1983.

13. Bodington, C. E., Baker, T. E., A history of mathematical programming in the petroleum industry, *Interfaces*, 20(4), pp. 117-127. 1990.

14. Bodington, C. E., Inventory Management in Blending Optimization: Use of Nonlinear Optimization for Gasoline Blends Planning and Scheduling, *ORSA/TIMS National Meeting*, San Francisco (CA), 1992.

15. Cafaro, D. C., Cerda, J., Optimal scheduling of multiproduct pipeline systems using a non-discrete MILP formulation, *Computers and Chemical Engineering*, 28(10), pp 2053-2068. 2004.

16. Cao C. W., Gu X. S., Chance constrained programming with fuzzy parameters for

refinery crude oil scheduling problem, *Fuzzy Systems and Knowledge discovery*, Proceeding Lecture Notes in Computer Science, 4223, pp. 1010-1019. 2006.

17. Castro, P. M., Barbosa-Povoa, A. P., Matos, H., An improved RTN continuous-time formulation for the short-term scheduling of multipurpose batch plants, *Computers and Chemical Engineering*, 40, pp. 2059-2068. 2001.
18. Castro, P. M., Barbosa-Povoa, A. P., Matos, H. A., Novais, A. Q., Simple continuous-time formulation for short-term scheduling of batch and continuous process, *Industrial and Engineering Chemistry Research*, 43, pp. 105-118. 2004.
19. Cerda, J., Henning, G. P., Grossmann, I. E., A mixed-integer linear programming model for short-term scheduling of single-stage multi-product batch plants with parallel lines, *Industrial and Engineering Chemistry Research*, 36(5), pp.1695-1707. 1997.
20. Chryssolouris, G., Papakostas, N., Mourtzis, D., Refinery short term scheduling with tank farm, inventory and distillation management: An integrated simulation-based approach, *European Journal of Operational Research*, 166, pp. 812-827. 2005.
21. Coxhead, R. E., Integrated planning and scheduling systems for the refining industry. In T. A. Ciriani, & R. C. Leachman, *Optimization in Industry*, 2, pp. 185-199, Wiley: New York. 1994.
22. Decroocq, D., Catalytic Cracking of Heavy Petroleum Fractions, *Editions Technip.*, 1984.
23. Dempster, M. A. H., Pedron, N. H., Medova, E. A., Scott, J. E., Sembos, A.,

Planning logistics operations in the oil industry, *Journal of the Operational Research Society*, 51(11), pp. 1271-1288, 2000.

24. Dewitt, C. W., Lasdon, L. S., Waren, A. D., Brenner, D. A., Melhem, S. A., OMEGA: An improved gasoline blending system for Texaco, *Interfaces*, 19, pp. 85-101. 1989.
25. Doganis, P., Sarimveis, H., Bafas, G., Koufos, D., An MILP model for optimal scheduling of the lubricant production plant, Chemical Engineering Communications, 1992 (8), pp. 1067-1084. 2005.
26. Dorn, J., Case based reactive scheduling, in Artificial Intelligence in Reactive Scheduling, edited by R. Kerr and E. Szelke, pp. 32-50, 1994 (Chapman & Hall: London).
27. Dorn, J., Kerr, R., Thalhammer, G., Reactive scheduling in a fuzzy-temporal framework. Knowledge based reactive scheduling, *IFIP Trans. B (Appl. Tech.)*, B-15, pp. 39-54. 1994.
28. Duran, M. A., Grossmann, I. E., A mixed integer nonlinear programming algorithm for process systems synthesis, *AIChE Journal*, 32(4), pp. 592-606. 1986a
29. Duran, M. A., Grossmann, I. E., An outer approximation algorithm for a class of mixed integer nonlinear programs, *Mathematical Programming*, 36, pp. 307-339. 1986b
30. Erdirik-Dogan, M., Grossmann, I. E., Slot-based formulation for the short-term scheduling of multistage, multiproduct batch plants with sequence-dependent

changeovers, *Industrial and Engineering Chemistry Research*, 47, pp. 1159-1183. 2008.

31. Ferrer-Nadal, S., Capon-Garcia, E., Mendez, C. A., Puigjaner, L., Material transfer operations in batch scheduling. A critical modelling issue, *Industrial and Engineering Chemistry Research*, 47 (20), pp. 7721-7732. 2008.
32. Fieldhouse, M., The pooling problem, Chapter 13. In: ciriani, T. A., Leachman, R. C. (Eds.), *Optimization in Industry*. Wiley, Chichester, 223-230. 1993.
33. Floudas, C. A., Lin, X. X., Continuous-time versus discrete-time approaches for scheduling of chemical processes: a review, *Computers and Chemical Engineering*, *28*, pp. 2109-2129. 2004a.
34. Floudas, C. A., Lin, X. X. Mixed integer linear programming in process scheduling: Modeling, algorithms, and applications, *Annals of Operations Research*, 139, pp. 131-162. 2004b.
35. Furman, K. C., Jia, Z. Y., Ierapetritou, M. G., A robust event-based continuous time formulation for tank transfer scheduling, *Industrial and Engineering Chemistry Research*, 46, pp. 9126-9136. 2007.
36. Gan, H. S., Wirth, A., Generating robust schedules on identical parallel machines: heuristic approaches, ICOTA 6, Ballarat, Austra, Dec 9-11, 2004.
37. Garner, B. J., Ridley, G. J., Application of neural network process models in reactive scheduling. Knowledge based reactive scheduling, *IFIP Trans. B (Appl. Tech.)*, B-15, pp. 19-28. 1994.
38. Geoffrion, A. M., generalized benders decomposition, *Journal of Optimization*

Theory Application, 10(4), pp. 237-260. 1972.

39. Giannelos, N. F., Georgiadis, M. C., A novel event-driven formulation for short-term scheduling of multipurpose continuous processes, *Industrial and Engineering Chemistry Research*, 41, pp. 2431. 2002.

40. Glismann, K., Gruhn, G., Short-term planning of blending processes: scheduling and nonlinear optimization of recipes, *Chemical Engineering Technology*, 24(3), pp. 246-249. 2001a.

41. Glismann, K., Gruhn, G., Short-term scheduling and recipe optimization of blending processes, *Computers and Chemical Engineering*, 25, pp. 627-634. 2001b.

42. Gothe-Lundgren M., Lundgren J.T., Persson J.A., An optimization model for refinery production scheduling, *International Journal of Production Economics*, 78(3), pp. 255-270, 2002.

43. Goyal, V., Ierapetritou, M. G., Computational studies using a novel simplicial approximation based algorithm for MINLP optimization, *Computers and Chemical Engineering*, 28, pp. 1771-1780. 2004.

44. Gupta, S., Karimi, I. A., Scheduling a two-stage multiproduct process with limited product self in intermediate storage, *Industrial and Engineering Chemistry Research*, 42, pp. 490-508. 2003a

45. Gupta, S., Karimi, I. A., An improved MILP formulation for scheduling multi-product, multi-stage batch plants, *Industrial and Engineering Chemistry Research*, 42(11), pp. 2365-2380. 2003b.

46. Henseler, H., Reaktion: a system for event independent reactive scheduling. In: Artificial Intelligence in Reactive Scheduling, London, UK: Chapman & Hall, pp. 19-31. 1994.

47. Herroelen, W., Leus, R., Robust and reactive project scheduling: a review and classification of procedures, International Journal of Production Research, 42, pp. 1599-1620. 2004.

48. Holmberg, K., On the convergence of the cross decomposition, *Mathematical Programming*, 47, pp. 269-296. 1990.

49. Honkomp, S. J., Mockus, L., Reklaitis, G.V., A framework for schedule evaluation with processing uncertainty, *Computers and Chemical Engineering*, 23, pp. 595–609. 1999.

50. Hui, C. H., Gupta, A., A novel MILP formulation for short-term scheduling of multistage multiproduct batch plants, *Computers and Chemical Engineering*, 24, pp. 1611-1617. 2000.

51. Hui, C. H., Gupta, A., Meulen, H. A. J., A novel MILP formulation for short-term scheduling of multistage multiproduct batch plants with sequence-dependent constraints, *Computers and Chemical Engineering*, 24, pp. 2705-2717. 2000.

52. Iakovou, E. T., An interactive multiobjective model for the strategic maritime transportation of petroleum products: risk analysis and routing, *Safety Science*, 39, pp.19-29. 2001.

53. Ierapetritou, M. G., Pistikopoulos, E. N., Global optimization for stochastic planning, scheduling and design problems, In I. E. Grossmann (Ed.), Global

optimization in engineering design (pp. 231–287). Kluwer Academic Publishers. 1996.

54. Ierapetritou, M. G., Floudas, C. A., Effective continuous-time formulation for short-term scheduling. Part I. Multipurpose batch processes, *Industrial and Engineering Chemistry Research*, 37, pp. 4341-4359. 1998a.
55. Ierapetritou, M. G., Floudas, C. A., Effective continuous-time formulation for short-term scheduling: 2. continuous and semi-continuous processes, *Industrial and Engineering Chemistry Research*, 37, pp. 4360-4374. 1998b.
56. Ierapetritou, M. G., Hene, T. S., Floudas, C. A., Effective continuous-time formulation for short-term scheduling. 3. Multiple intermediate due dates, *Industrial and Engineering Chemistry Research*, 38, pp. 3446-3461. 1999.
57. Janak, S. L., Lin, X., Floudas, C. A., Enhanced continuous-time unit-specific event-based formulation for short-term scheduling of multipurpose batch processes: Resource constraints and mixed storage policies, *Industrial and Engineering Chemistry Research*, 43, pp. 2516-2533. 2004.
58. Janak S. L., Lin X. X., Floudas, C. A., A new robust optimization approach for scheduling under uncertainty: II. uncertainty with known probability distribution, *Computers and Chemical Engineering*, 31, pp. 171-195. 2007.
59. Jia, Z., Ierapetritou, M. G., Efficient short-term scheduling of refinery operation based on continuous time formulation, *FOCAPO*, pp. 327-330. 2003a.
60. Jia, Z., Ierapetritou, M. G., Mixed-integer linear programming model for gasoline blending and distribution scheduling, *Industrial and Engineering Chemistry*

Research, 42, pp. 825-835. 2003b.

61. Jia, Z., Ierapetritou, M. G., Efficient short-term scheduling of refinery operations based on a continuous time formulation, *Computers and Chemical Engineering*, 28, pp. 1001-1019. 2004.
62. Jia, Z, Ierapetritou, MG, Kelly, JD. Refinery short-term scheduling using continuous time formulation: crude oil operations, *Industrial and Engineering Chemistry Research*, 42, pp. 3085-3097. 2003.
63. Joly M., Pinto, J. M., Scheduling optimization of fuel oil and asphalt production, In: ECCE 2, Montpellier, France. 1999.
64. Joly, M., Moro, L. F. L., Pinto, J. M., Planning and scheduling for petroleum refineries using mathematical programming, *Brazilian Journal of Chemical Engineering*, 19(2), 207-228. 2002.
65. Joly M., Pinto J. M., Mixed-integer programming techniques for the scheduling of fuel oil and asphalt production, *Chemical Engineering Research & Design*, 81(A4), pp. 427-447. 2003.
66. Karimi, I. A., McDonald C. M., Planning and scheduling of parallel semicontinuous processes. 2. short-term scheduling, *Industrial and Engineering Chemistry Research*, 36, pp. 2701-2714. 1997.
67. Karuppiah, R., Furman, K., Grossmann, I. E., Global optimization for scheduling refinery crude oil operations, Presented in INFORMS Annual Meeting, Pittsburgh, PA, USA, Nov. 5-8, 2006.
68. Karuppiah, R., Furman, K. C., Grossmann, I. E., Global optimization for

scheduling refinery crude oil operations, *Computers and Chemical Engineering*, 32, pp. 2745-2766. 2008.

69. Kelley, J. D., Forbes, J. F., Structured approach to storage allocation for improved process controllability, *AIChE Journal*, 44(8), pp. 1832-1840. 1998.

70. Kelley, J. D., Chronological decomposition heuristic for scheduling: a divide and conquer Method, http://www.dashoptimization.com/pdf/cdhmilpv2.pdf. March 2002.

71. Kelley, J. D., Mann, J. L., Crude oil blend scheduling optimization: an application with multi-million dollar benefits-part 1. hydrocarbon processing, 82 (6), pp. 47-51. 2003a.

72. Kelley, J. D., Mann, J. L., Crude oil blend scheduling optimization: an application with multi-million dollar benefits-part 2. hydrocarbon processing, 82 (7), pp. 72-79. 2003b.

73. Kocis, G. R., Grossmann, I. E., Relaxation strategy for the structural optimization of process flow sheets, *Industrial and Engineering Chemistry Research*, 26(9), pp. 1869-1880. 1987.

74. Kocis, G. R., Grossmann, I. E., Global optimization of nonconvex mixed integer nonlinear programming (MINLP) problems in process synthesis, *Industrial and Engineering Chemistry Research*, 27, pp. 1407-1421. 1988.

75. Kondili, E., Pantelides, C. C., Sargent, R. W. H., A general algorithm for short term scheduling of batch operations –I, MILP formulation, *Computers and Chemical Engineering*, 17, pp. 211-227. 1993.

76. Kunnathur, A. S., Sundararaghavan, P. S., Sampath, S., Dynamic rescheduling using a simulation-based expert system, Journal of Manufacturing Technology Management, 15, pp. 199-212. 2004.

77. Lamba, N., Karimi, I. A., Scheduling parallel production lines with resource constraints. 1. model formulation, *Industrial and Engineering Chemistry Research*, 41(4), pp.779-789. 2002a.

78. Lamba, N., Karimi, I. A., Scheduling parallel production lines with resource constraints. 2. decomposition algorithm, *Industrial Engineering Chemistry Research*, 41(4), pp.790-800. 2002b.

79. Lee, H., Pinto, J. M., Grossmann, I. E., Park, S., Mixed-integer linear programming model for refinery short-term scheduling of crude oil unloading with inventory management, *Industrial and Engineering Chemistry Research*, 35(5), pp. 1630-1641. 1996.

80. Lee, K. H., Park, H. I., Lee, I. B., A novel nonuniform discrete time formulation for short-term scheduling of batch and continuous processes, *Industrial and Engineering Chemistry Research*, 40, pp. 4902-4911. 2001.

81. Li, J., Li, W. K., Karimi, I. A., Srinivasan, R., Robust and efficient algorithm for optimizing crude oil operations, Presented in AIChE Annual Meeting, Cincinnati, OH, USA, Oct. 30-Nov. 4, 2005a.

82. Li, J., Karimi, I. A., Srinivasan, R., Robust and partial relaxation algorithms for crude oil scheduling, Presented in the 3rd Sino-Japanese Optimization Meeting in the e-Business Era, Singapore, Oct. 31-Nov. 2, 2005b.

83. Li, J., Karimi, I. A., Srinivasan, R., Robust scheduling of crude oil operations under demand and ship arrival uncertainty, Presented in AIChE Annual Meeting, SanFrancisco, CA, USA, Nov. 12-17, 2006.

84. Li, J., Susarla, N., Karimi, I. A., A novel continuous-time formulation for short-term scheduling of continuous processes, Presented in *AIChE Annual Meeting*, Philadelphia, PA, USA, November 16-21, 2008.

85. Li, W. K., Hui, C. W., Hua, B., Tong, Z., Scheduling crude oil unloading, storage, and processing, *Industrial and Engineering Chemistry Research*, 41, pp. 6723-6734. 2002.

86. Li W. K., Hui, C. W.,, Hua, B., Zhong, X. T., Plant-wide scheduling and marginal value analysis for a refinery, *FOCAPO*, 2003.

87. Li, W., K., Modeling oil refinery for production planning, scheduling and economic analysis, PhD Thesis, HongKong University of Science and Technology, 2004a.

88. Li, W. K., Hui, C. W., Li, P., Li, A. X., Refinery planning under uncertainty, *Industrial and Engineering Chemistry Research*, 43, pp. 6742-6755. 2004b.

89. Li, W. K., Karimi, I. A., Srinivasan, R., Planning under correlated and truncated price and demand uncertainties, Presented at AIChE Annual Meeting, Cincinnati, OH, Oct 30- Nov 4, 2005.

90. Li, W. K., Karimi, I. A., Srinivasan, R., Refinery planning under correlated and truncated price and demand uncertainties, Presented at ESCAPE-16 and PSE-9, Garmisch-Partenkirchen, Germany, Jul 9-13, 2006.

91. Li, Z. K., Ierapetritou, M., Process scheduling under uncertainty: review and challenges, *Computers and chemical Engineering*, 32, pp. 715-727. 2008.
92. Lim, M. F., Karimi, I. A., A slot-based formulation for single-stage multi-product batch plants with multiple orders per product, *Industrial and Engineering Chemistry Research*, 42(9), pp.1914-1924. 2003.
93. Lin, X. X., Floudas, C. A., Design, synthesis and scheduling of multipurpose batch plants via an effective continuous-time formulation, *Computers and Chemical Engineering*, 25, pp. 665-674. 2001.
94. Lin X. X., Chajakis E. D., Floudas C. A., Scheduling of tanker lightering via a novel continuous-time optimization framework, *Industrial and Engineering Chemistry Research*, 42(20), pp. 4441-4451. 2003.
95. Lin X. X., Janak, S. L., Floudas, C. A., A new robust optimization approach for scheduling under uncertainty: I. bounded uncertainty, *Computers and Chemical Engineering*, 28, pp. 1069-1085. 2004.
96. Liu, Y., Karimi, I. A., Novel continuous-time formulations for scheduling multi-stage batch plants with identical parallel units, *Computers and Chemical Engineering*, 31, pp. 1671-1693. 2007a.
97. Liu, Y., Karimi, I. A., Scheduling multistage, multiproduct batch plants with nonidentical parallel units and unlimited intermediate storage, *Chemical Engineering Science*, 62, pp. 1549-1566. 2007b.
98. Liu, Y., Karimi, I. A., Scheduling multistage batch plants with parallel units and no interstage storage, *Computers and Chemical Engineering*, 32, pp. 671-693. 2008.

99. Lundgren, M. G., Lundgren, J. T., Persson, J. A., An optimization model for refinery production scheduling, *International Journal of Production Economics*, 78, pp. 255-270. 2002.

100. Luo, C. P., Rong, G., Hierarchical approach for short-term scheduling in refineries, *Industrial and Engineering Chemistry Research*, 46, pp. 3656-3668. 2007.

101. Magalhaes, M. V., More, L. F. L., Smania, P., Hassimotto, M. K., Pinto, J. M., Abadia, G. J., SIPP-A solution for refinery scheduling, NPRA Computer Conference, Nov. 16-18, San Antonio, TX, 1998.

102. Magalhaes, M. V., Shah, N., Crude oil scheduling, *FOCAPO*, pp. 323-326. 2003.

103. Magatao L., Arruda L. V. R., Neves F., A mixed integer programming approach for scheduling commodities in a pipeline, *Computers and Chemical Engineering*, 28(1-2), pp. 171-185. 2004.

104. Manne, A. S., Scheduling of petroleum operations, Harvard University Press: Cambridge, MA, 1956.

105. Maravelias, C. T., Grossmann, I. E., New general continuous-time state-task network formulation for short-term scheduling of multipurpose batch plants, *Industrial and Engineering Chemistry Research*, 42, pp. 3056-3074. 2003.

106. Mas, R., Pinto, J. M., A mixed-integer optimization strategy for oil supply in distribution complexes, *Optimization and Engineering*, 4, pp. 23-64. 2003.

107. McCormick, G. P., Computability of global solutions to factorable nonconvex programs: part I-convex underestimating problems, *Mathematical Programming*, 10, pp. 146-175. 1976.

108. Mcdonald, C. M., Karimi, I. A., Planning and scheduling of parallel semi-continuous processes: 1. Production planning, *Industrial and Engineering Chemistry Research*, 36, pp. 2691-2700. 1997.

109. Mendez, C. A., Cerda, J., An efficient MILP continuous-time formulation for short-term scheduling of multiproduct continuous facilities, *Computers and Chemical Engineering*, 26, pp. 687-695. 2002.

110. Mendez, C. A., Cerda, J., An MILP framework for short term scheduling of multipurpose batch processes under different operation strategies, *Optimization and Engineering*, 4, 7-22. 2003a.

111. Mendez, C. A., Cerd´a, J., Dynamic scheduling in multiproduct batch plants, *Computers and Chemical Engineering*, 27, pp. 1247–1259. 2003b.

112. Mendez, C. A., Cerda, J., Short-term scheduling of multiple stage batch processes subject to limited finite processes, *Computer-aided Chemical Engineering*, 15B, 984-989. 2004.

113. Mendez, C. A., Cerda J., An MILP framework for batch reactive scheduling with limited discrete resources, *Computers and Chemical Engineering*, 28, pp. 1059-1068. 2004.

114. Mendez, C. A., Cerda, J., Grossmann, I. E., Harjunkoski, I., Fahl, M., State-of-the art review of optimization methods for short-term scheduling of batch processes, *Computers and Chemical Engineering*, 30, pp. 913-946. 2006.

115. Mendez, C. A., Grossmann, I. E., Harjunkoski, I., Kabore, P., A simultaneous optimization approach for off-line blending and scheduling of oil-refinery

operations, *Computers and Chemical Engineering*, 30, pp. 614-634. 2006.

116.Mendez, C. A., Henning, G. P., Cerda, J., Optimal scheduling of batch plants satisfying multiple product orders with different due dates, *Computers and Chemical Engineering*, 24, pp. 2223-2245. 2000.

117.Mendez, C. A., Henning, G. P., Cerda, J., An MILP continuous-time approach to short-term scheduling of resource-constrained multi-stage flowshop batch facilities, *Computers and Chemical Engineering*, 2001, 25, 701-711.

118.Miyashita K., Case Based knowledge acquisition for schedule optimization, *Artificial Intelligence in Engineering.*, 9, pp. 277-287. 1995.

119.Mockus, L., Reklaitis, G. V., Mathematical programming formulation for scheduling of batch operations based on non-uniform time discretization, *Computers and Chemical Engineering*, 21, pp. 1147-1156. 1994.

120.Moro LFL, Process technology in the petroleum refinung industry-current situation and future trends, *Computers and Chemical Engineering*, 27(8-9), pp. 1303-1305. 2003.

121.Moro, L. F. L., Zanin, A. C., Pinto, J. M., A planning model for refinery diesel production, *Computers and Chemical Engineering*, 22(S), pp. S1039-S1042. 1998.

122.Moro, L. F. L., Pinto, J. M., Mixed-integer programming approach for short-term crude oil scheduling, *Industrial and Engineering Chemistry Research*, 43, pp. 85-94. 2004.

123.Neiro, S. M. S., Pinto, J. M., A general modeling framework for the operational planning of petroleum supply chains, *Computers and Chemical Engineering*, 28, pp. 871-896. 2004.

124. Neiro, S., Pinto, J. M., Multiperiod optimization for production planning of petroleum refineries, *Chemical Engineering Communications*, 192 (1), pp. 62-88. 2005.

125. Paolucci, M., Sacile, R., Boccalatte, A., Allocating crude oil supply to port and refinery tanks: a simulation-based decision support system, *Decision Support Systems*, 33, pp. 39-54. 2002.

126. Pelham, R., Pharris, C., Refinery operations and control: a future vision, *Hydrocarbon Processing*, 75(7), pp. 89-94. 1996.

127. Persson J. A., Gothe-Lundgren M., Lundgren J. T., Gendron B., A tabu search heuristic for scheduling the production processes at an oil refinery, *International Journal of Production Research*, 42(3), pp. 445-471. 2004.

128. Pinto, J. M., Grossmann, I. E., A continuous time mixed integer linear programming model for short-term scheduling of multistage batch plants, *Industrial and Engineering Chemistry Research*, 34, pp. 3037-3051. 1995.

129. Pinto, J. M., Joly, M., Mixed-integer programming techniques for the scheduling of fuel oil and asphalt production, *ESCAPE-10*, pp. 1063-1068. 2000.

130. Pinto, J. M., Moro, L. F. L., A mixed integer model for LPG scheduling, *ESCAPE-10*, pp. 1141-1146. 2000.

131. Pinto J. M., Joly, M., Moro, L., Planning and scheduling models for refinery operations, *Computer and Chemical Engineering*, 24, pp. 2259-2276. 2000.

132.Pitty, S. S., Karimi, I. A., Novel MILP models for scheduling permutation flowshops, *Chemical Product and Process Modeling*, 3(1), pp. 1-46. 2008.

133.Qi, J. G., Burns, G. R., Harrison, D. K., The application of the parallel multipopulation genetic algorithms to dynamic job shop scheduling, *Int. J. Adv. Manuf. Technol.*, 16, pp. 609-615. 2000.

134.Quesada, I., Grossmann, I. E., Global optimization of bilinear process networks with multicomponent flows, *Computes and Chemical Engineering*, 19(12), pp. 1219-1242. 1995.

135.Ramage, M. P., The petroleum industry of the 21st century, *FOCAPO*, Snowmass, CO, 1998.

136.Reddy, P. C. P., Karimi, I. A. , Srinivasan, R., A new continuous time formulation for scheduling crude oil operations, *Chemical Engineering Science*, 59(6), pp. 1325-1341. 2004a.

137.Reddy, P. C. P. , Karimi, I. A., Srinivasan, R., A novel solution approach for optimizing scheduling crude oil operations, *AIChE Journal*, 50(6), pp. 1177-1197. 2004b.

138.Rejowski, R. Jr., Pinto, J. M., An MILP formulation for the scheduling of multiproduct pipeline systems, *Brazilian Journal of Chemical Engineering*, 19(4), pp. 467-474. 2002.

139.Rejowski, R. Jr., Pinto, J. M., Scheduling of a Multiproduct Pipeline System, *Computers and Chemical Engineering*, 27, pp. 1229-1246. 2003.

140.Rejowski, R. Jr., Pinto, J. M., Efficient MILP formulations and valid cuts for multiproduct pipeline scheduling, *Computers and Chemical Engineering*, 28(8), pp. 1511-1528. 2004.

141. Rejowski, R. Jr., Pinto, J. M., A novel continuous time representation for the scheduling of pipeline systems with pumping yield rate constraints, *Computers and chemical Engineering*, 32 (4-5), pp. 1042-1066. 2008.

142. Reklaitis, G. V., Overview of scheduling and planning of batch process operations, Presented at NATO Advanced Study Institute-batch process system engineering, Antalya, Turkey, 1992.

143. Rigby, B., Lasdon, L. S., Waren, A. D., The evolution of texaco's blending systems: from OMEGA to StarBlend, *Interfaces*, 25, pp. 64-83. 1995.

144. Rosenberger, J. M., Johnson, E. L., Nemhauser, G. L., Rerouting aircraft for airline recovery, *Transportation Science*, 37, pp. 408-421. 2003.

145. Rovithakis, G. A., Perrakis, S. E., Christodoulou, M. A., Application of a neural network scheduler on a real manufacturing system, *IEEE Trans. Cont. Sys. Tech.*, 9, pp. 261-270. 2001.

146. Sahinidis, N., Optimization under uncertainty: state-of-the-art and opportunities, *Computers and chemical Engineering*, 28, pp. 971-983. 2004.

147. Sasikumar, M., Prakash, P. R., Patil, S. M., Ramani, S., Pipes: a heuristic search model for pipeline schedule generation, *Knowledge-Based System*, 10, pp. 169-175. 1997.

148. Schmidt, G., How to apply fuzzy logic to reactive production scheduling, Knowledge based reactive scheduling, *IFIP Trans. B (Appl. Tech.)*, B-15, pp.

57-66. 1994.

149.Sear, T. N., Logistics planning in the wownstream oil industry, *Journal of Operational Research Society*, 44(1), pp. 9-17. 1993.

150.Shaik, M. A., Janak, S. L., Floudas, C. A., Continuous-time models for short-term scheduling of multipurpose batch plants: A comparative study, *Industrial and Engineering Chemistry Research*, 45, pp. 6190-6209. 2006.

151.Shaik, M. A., Floudas, C. A., Improved unit-specific event-based continuous-time model for short-term scheduling of continuous processes: Rigorous treatment of storage requirements, *Industrial and Engineering Chemistry Research*, 46, pp. 1764-1779. 2007.

152.Shaik, M. A., Floudas, C. A., Unit-specific event-based continuous-time approach for short-term scheduling of batch plants using RTN framework, *Computers and Chemical Engineering*, 32, pp. 260-274. 2008.

153.Shah, N., Pantelides, C. C., Sargent R. W. H., A general algorithm for short-term scheduling of batch operations – 2. Computational Issues, *Computers and Chemical Engineering*, *17*, pp. 229-244. 1993.

154.Shah, N., Mathematical programming techniques for crude oil scheduling. *Computers and Chemical Engineering*, 20(S), pp. S1227-1232. 1996.

155.Shobrys, D. E., White, D. C., Planning, scheduling and control: why cannot they work together, *Computers and Chemical Engineering*, 26, 149-160. 2002.

156.Simon, J. D., Azma, H. M., Exxon experience with large scale linear and nonlinear programming applications, *Computers and Chemical Engineering*, 7(5), 605-614.

1983.

157.Spargg, J.E., Fozzard, G. and Tyler, D. J., Constraint based reactive rescheduling in a stochastic environment, *Proceedings of recent advances in AI Planning, 4th European Conference on Planning, ECP' 97*, pp. 403-413. 1997.

158.Srinivasan, R., Bansal, M., Karimi, I. A., A multi-agent approach to supply chain management in the chemical industry, In *Multiagent-Based Supply Chain Management* Eds: B. Chaib-draa and J. P. Müller, Studies in Computational Intelligence, Vol 28, p419-448. 2006.

159.Subramaniam, V., Raheja A. S., mAOR: A heuristic-based reactive repair mechanism for job shop schedules, *International Journal of Advanced Manufacturing Technology*, 22, pp. 669-680. 2003.

160.Subramaniam, V., Raheja, A. S., Reddy, K. R. B., Reactive repair tool for job shop schedules, *International Journal of production research*, 43 (1), 1-23. 2005.

161.Sundaramoorthy, A., Karimi, I. A., A simpler better slot-based continuous-time formulation for short-term scheduling in multiproduct batch plants, *Chemical Engineering Science*, 60, pp. 2679-2702. 2005.

162.Susarla, N., Li, J., Karimi, I. A., Short-term scheduling of multipurpose batch processes using unit slots, Presented in *AIChE Annual Meeting*, Philadelphia, PA, USA, November 16-21, 2008.

163.Symonds, G. H., Linear programming: the solution of refinery problems, Esso Standard Oil Company: New York, 1955.

164.Vin, J. P., Ierapetritou, M. G., A new approach for efficient rescheduling of multiproduct batch plants, *Industrial and Engineering Chemistry Research*, 39, pp. 4228-4238. 2000.

165. Vin, J. P., Ierapetritou, M. G., Robust short-term scheduling of multiproduct batch plants under demand uncertainty, Industrial and Engineering Chemistry Research, 40, pp. 4543-4554. 2001.

166. Voudouris, V. T., Grossmann, I. E., Mixed integer linear programming reformulations for batch process design with discrete equipment sizes, *Industrial and Engineering Chemistry Research*, 31, pp. 1315-1325. 1992.

167. Wang, J., A fuzzy robust scheduling approach for product development projects, *European Journal of Operational Research*, 152(1), pp. 180-194. 2004.

168. Watkins, R. N., Petroleum Refinery Distillation, 2nd Edition, Gulf Publishing Co., Houston, 1979.

169. Zhang, N., Zhu, X. X., A novel modeling and decomposition strategy for overall refinery optimization, *Computers and Chemical Engineering*, 24, pp. 1543-1548. 2000.

170. Zhang, J., Zhu, X. X., Towler, G. P., A simultaneous optimization strategy for overall integration in refinery planning, *Industrial and Engineering Chemistry Research*, 40, 2640-2653. 2001.

171. Zhao, Y., Decision support system for management of oil pipeline. In L. F. Pau (Ed.), Artificial intelligence in economics and management. Amsterdam: Oxford. 1986.

Appendix A

The model of Jia and Ierapetritou (2003) has two basic problems.

First, the model is infeasible as shown below. Eqs. 16a, 11a, and 11b (denoted as JI-16a, JI11a, etc. here) from their model are:

$$\sum_{j \in J_s} Blnd_{sjn} \geq 6Bflow \qquad \forall s \in \boldsymbol{S}, n \in \boldsymbol{N} \qquad \text{(JI-16a)}$$

$$sv_{sjn} \leq xv_{sn} \leq \sum_{j \in J_s} sv_{sjn} \qquad \forall s \in \boldsymbol{S}, n \in \boldsymbol{N} \qquad \text{(JI-11a)}$$

$$\sum_{s} xv_{sn} \leq 1 \qquad \forall s \in \boldsymbol{S}, n \in \boldsymbol{N} \qquad \text{(JI-11b)}$$

Since *Bflow* is a positive blend rate, product *s* is being produced and must be transferred from the blender to at least one product tank *j* at event point *n*. Therefore, there must be at least one *j* such that,

$$sv_{sjn} \geq 1 \qquad \forall s \in \boldsymbol{S}, n \in \boldsymbol{N} \qquad \text{(A.1)}$$

where sv_{sjn} = 1, if product *s* is produced and transferred to tank *j* at event point *n*.

From eqs. (A.1) and (JI-11a), it can be got xv_{sn} = 1. Hence,

$$\sum_{s} xv_{sn} = N \qquad \forall s \in \boldsymbol{S}, n \in \boldsymbol{N} \qquad \text{(A.2)}$$

where, *N* is the number of products that are needed to process in blenders in the problem. Since $N > 1$ for most problems, $\sum_{s} xv_{sn} > 1$, which contradicts eq. (JI-11b).

Second, it allows a tank to hold multiple products at a time, as shown below. They used the following (eq. 2 in their paper) to force the inventory of product *s* in tank *j* to be zero, if *j* does not hold *s* at event point *n*.

$$V_j^{\min} \cdot y_{sjn} \leq Pst_{sjn} + Blnd_{sjn} \leq V_j^{\max} \cdot y_{sjn} \qquad \forall s \in \boldsymbol{S}, j \in J_s, n \in \boldsymbol{N} \qquad \text{(JI-2)}$$

where, y_{sjn} = 1, if *j* holds *s* at event point *n*. Note that eq. JI-2 cannot guarantee that *j* holds at most one product at an event point.

Appendix B

Prove that eqs. 5.2, 5.4, 5.7, and 5.9 make x_{bpk} binary. Recall that a blender b is either idle or feeding a product tank at any time.

1. If b is idle during slot k, then $v_{b0k} = 1$ and $x_{bpk} = 0$ for $(b, p) \in \boldsymbol{BP}$ from eq. 5.9.

2. If b is not idle during k, then $v_{b0k} = 0$ and $v_{bjk} = 1$ for one j with $(b, j) \in \boldsymbol{BJ}$ and $v_{bj'k} = 0$ for $j' \neq j$ from eq. 5.2. If product tank j holds a product p during slot k, then $u_{jpk} = x_{bpk} = 1$ from eq. 5.7. Eq. 5.9 then forces $x_{bp'k} = 0$ for $p' \neq p$.

Appendix C

Prove that the proposed adjustment procedure ensures that constant blend rate satisfies the limits on the blending rates and the minimum run length at the same time. From eqs. 5.16 and 5.17, we get $0 \le Q_{bk} \le F_b^U SL_k$. Then, using eqs. 5.37 and 5.38, we have $0 \le CCQ_{bk} \le F_b^U CRL_{bk}$ for k with $xe_{bk} = 0$, and $0 \le CCQ_{b(k-1)} + Q_{bk} \le F_b^U [CRL_{b(k-1)} + SL_k]$ for k with $xe_{bk} = 1$. In other words,

$$0 \le \frac{CCQ_{b(k-1)} + Q_{bk}}{CRL_{b(k-1)} + SL_k} \le F_b^U \qquad \text{for } xe_{bk} = 1 \tag{C.1}$$

The above along with eq. 5.40 ensures $F_b^L \le R_{bk} \le F_b^U$ for $xe_{bk} = 1$ & $v_{b0k} = 0$. Thus, the optimal solution from RSPM will satisfy the blend rate limits.

For $xe_{bk} = 1$, we get the following from eqs. 5.20 and 5.39.

$$TCQ_{bk} \ge F_b^L \sum_{p=1}^{P} RL_{bp}^L x_{bpk} \qquad \text{for } xe_{bk} = 1,\ (b, p) \in \boldsymbol{BP},\ 0 < k \le K \tag{C.2}$$

If $F_b^L \ge \dfrac{CCQ_{b(k-1)} + Q_{bk}}{CRL_{b(k-1)} + SL_k}$ for k with $xe_{bk} = 1$ and $v_{b0k} = 0$, then $R_{bk} = F_b^L$ from eq. 5.40, and

$$\frac{TCQ_{bk}}{R_{bk}} = \frac{TCQ_{bk}}{F_b^L} \ge \sum_{p=1}^{P} RL_{bp}^L x_{bpk} \qquad \text{for } xe_{bk} = 1,\ (b, p) \in \boldsymbol{BP},\ 0 < k \le K \tag{C.3}$$

which means that the run length exceeds the minimum.

If $F_b^L \leq \dfrac{CCQ_{b(k-1)} + Q_{bk}}{CRL_{b(k-1)} + SL_k}$ for k with $xe_{bk} = 1 \,\&\, v_{b0k} = 0$, then $R_{bk} = \dfrac{CCQ_{b(k-1)} + Q_{bk}}{CRL_{b(k-1)} + SL_k}$, and

$$\frac{TCQ_{bk}}{R_{bk}} = \frac{CCQ_{b(k-1)} + Q_{bk}}{\dfrac{CCQ_{b(k-1)} + Q_{bk}}{CRL_{b(k-1)} + SL_k}} = CRL_{b(k-1)} + SL_k$$

$$\text{for } xe_{bk} = 1,\ (b, p) \in \boldsymbol{BP},\ 0 < k \leq K \qquad \text{(C.4)}$$

From eqs. 5.12-5.14, 5.37, and 5.38, we know that CRL_{bk} satisfies eq. 5.14, and hence,

$$CRL_{b(k-1)} + SL_k \geq \sum_{p=1}^{P} RL_{bp}^L x_{bpk} \qquad \text{for } xe_{bk} = 1,\ (b, p) \in \boldsymbol{BP},\ 0 < k \leq K \qquad \text{(C.5)}$$

Eqs. C.4 and C.5 show that the run length exceeds the minimum for this case too.