

## **INFORMATION TO USERS**

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

**The quality of this reproduction is dependent upon the quality of the copy submitted.** Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

**UMI**

A Bell & Howell Information Company  
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA  
313/761-4700 800/521-0600



# **MOVE SCHEDULE: A PLANNING TOOL FOR SCHEDULING SPACE USE ON CONSTRUCTION SITES**

by

Pierrette Pierre Zouein

A dissertation submitted in partial fulfillment  
of the requirements for the degree of Doctor of Philosophy  
(Civil and Environmental Engineering)  
in The University of Michigan  
1996

Doctoral Committee:

Associate Professor Iris D. Tommelein, Chair  
Associate Professor William P. Birmingham  
Assistant Professor John G. Everett  
Associate Professor Photios G. Ioannou

**UMI Number: 9624775**

**Copyright 1996 by  
Zouein, Pierrette Pierre**

**All rights reserved.**

---

**UMI Microform 9624775  
Copyright 1996, by UMI Company. All rights reserved.**

**This microform edition is protected against unauthorized  
copying under Title 17, United States Code.**

---

**UMI**  
300 North Zeeb Road  
Ann Arbor, MI 48103

\*



© Copyright by Pierrette Pierre Zouein 1996  
All Rights Reserved

*To my mother  
for her love and encouragement*

## **ACKNOWLEDGMENTS**

There are many people to thank for their role in helping me achieve this dream.

I owe most thanks and appreciation to my advisor, Professor Iris Tommelein, for her expert guidance and support in this research and her painstaking editing of my dissertation. I am also grateful to Professor John Everett for the valued input he had in the validation of MoveSchedule; Professor Photios Ioannou for his insightful suggestions and kind words of advice; and Professor William Birmingham for his review and editing of my dissertation.

I am most grateful to the National Science Foundation (Grant #MSM-9010394), the Barbour Scholarship for oriental women, and the Civil Engineering Department at the University of Michigan, who made this research possible by providing funding.

I also want to thank a special group of people who were always there for me when I needed their help at school and outside school: the graduate students in the construction engineering and management program, who helped me survive my first cold winter semester at Michigan and all following ones; my friends in Ann Arbor, who made these productive years a lot of fun; and the teaching assistants in the technical communication department for their editing assistance.

Special thanks goes to my family. First, my husband Walid who has put his professional career on hold so that I could complete this work, and who, having gone through the same process, was adorably understanding and encouraging: "why don't you finish this dissertation so we can all be happy?" Secondly, my one year old son Joseph-Raif, who, in his own way of being difficult (but cute), challenged me to finish. I wish also to thank my father Pierre who went out of his way to provide me with financial

support during my first year of study at the University of Michigan, my sisters Chantale and Valerie and my brother Elie for their loving encouragement, and my parents-in-law, Emirs Joseph and Raif Abillama, for their support and confidence in me throughout this process.

Most importantly, I owe a large debt of gratitude to my mother Georgette for giving all and for taking care of my son so that I could finish. Her sustained patience, loving encouragement, and understanding in the completion of my doctoral degree, made it all happen. Mom, you were always there for me and I could not have finished this dissertation without your help!!

# ABSTRACT

Chairman: Iris D. Tommelein

Site layout planning is crucial for enhancing the overall efficiency of construction operations and for reducing materials (re)location costs. More importantly, it helps identify construction time periods with potential spatial conflicts due to limited space so that they can be avoided in early project plans. This research develops an integrated scheduler and space planner, MoveSchedule, that adjusts an initial project schedule to comply with site space limitations in the process of accommodating resource space needs as they change over time.

MoveSchedule solves a constrained dynamic layout problem with the objective of minimizing resource transportation and relocation costs. Two-dimensional geometric constraints between relative positions of resources are being considered. Four different space profiles describe variations in resource space needs over time. The timing of these needs is inferred from the activity schedule. The layout construction algorithm that underlies MoveSchedule uses Constraint Satisfaction to find the set of all positions that meet the constraints on resources' positions and Linear Programming to find the optimal positions that minimize resource transportation and relocation costs. The solution is a sequence of layouts that describe changes in resource positions at discrete points in time corresponding to arrival and departure times of resources to and from site.

When no solution is found, i.e., when the resources needed in a given time period cannot be accommodated on site without violating constraints on their positions, MoveSchedule adjusts durations or start dates of activities to lower the total space need over the problematic time period. MoveSchedule establishes a time-space relationship for each activity in the schedule based on alternative resource levels so that tradeoffs between

activity duration and space need on site are possible. The resource levels describe different methods or crew-sizes for performing the activity.

Two example applications of MoveSchedule illustrate other areas of application of the use of MoveSchedule for optimally solving a MINISUM location problem with constraints on acceptable locations of the new facility, and for evaluating and choosing between alternative material delivery schedules. MoveSchedule is validated using realistic project data and its solution is qualitatively assessed by an experienced engineer on that project.

# TABLE OF CONTENTS

<b>DEDICATION.....</b>	ii
<b>ACKNOWLEDGMENTS.....</b>	iii
<b>ABSTRACT .....</b>	v
<b>LIST OF FIGURES.....</b>	xii
<b>LIST OF TABLES.....</b>	xvii
<b>LIST OF APPENDICES.....</b>	xx
<b>CHAPTER</b>	
<b>1 INTRODUCTION .....</b>	1
1.1 Motivation .....	1
1.2 Terminology and Problem Definition.....	4
1.2.1 Dynamic Site Layout Problem	4
1.2.2 Space Scheduling Problem	8
1.2.3 Research Scope and Assumptions	11
1.3 Research Objectives .....	13
1.4 Thesis Guide.....	14
<b>2 SITE LAYOUT AND LIMITED RESOURCE SCHEDULING: INDUSTRY PRACTICE AND LITERATURE REVIEW.....</b>	17
2.1 Site Layout and Scheduling Practice.....	18
2.1.1 State of the Practice	18
2.1.2 Lessons from Site Layout and Scheduling Practice	23

<b>2.2 Layout Models.....</b>	<b>24</b>
2.2.1 Static Layout Planning Models	24
2.2.1.1 Tightly- vs. Loosely-packed Arrangements	28
2.2.1.2 Modeling Assumptions for the Static Layout Problem	29
2.2.1.3 Optimal Methods for Solving the Static Layout Problem	30
2.2.1.4 Heuristic Methods for Solving the Static Layout Problem	32
2.2.1.4.1 Heuristic Construction Methods for Solving the Static Layout Problem	33
2.2.1.4.2 Heuristic Improvement Methods for Solving the Static Layout Problem	36
2.2.2 Layout Planning Models that Account for Time	37
2.2.3 Layout Implementation, Control, and Re-planning	39
<b>2.3 Resource Scheduling Models .....</b>	<b>41</b>
2.3.1 Unlimited Resource Scheduling	41
2.3.1.1 Network Generation	41
2.3.1.2 Construction Method and Resource Selection	42
2.3.1.3 Activity Scheduling	42
2.3.1.4 Schedule Improvement: Time-Cost Tradeoff Algorithms	43
2.3.1.5 Schedule Improvement: Resource Leveling Algorithms	44
2.3.2 Limited Resource Allocation	44
<b>2.4 Space Scheduling Models .....</b>	<b>46</b>
<b>2.5 Modeling Needs.....</b>	<b>49</b>
<b>3 COMPUTATIONAL MODELS FOR INTERACTIVE DYNAMIC LAYOUT PLANNING AND SPACE SCHEDULING .....</b>	<b>51</b>
3.1 MovePlan.....	52
3.1.1 Modeling Assumptions	52
3.1.2 Input	53
3.1.3 Computations	54
3.1.4 Output	56
3.2 MovePlan-ConRes.....	59
3.2.1 Modeling Assumptions	59
3.2.2 Input	60
3.2.3 Conflict Resolution Strategies	60
3.2.4 Computations	62
3.2.5 Output	63
3.3 Summary and Research Directions for MoveSchedule.....	63
<b>4 INCREMENTAL DYNAMIC LAYOUT PROBLEM SOLVER .....</b>	<b>65</b>
4.1 Introduction .....	65
4.2 Underlying Model.....	66
4.2.1 Discretizing Time	66

4.2.2	Resource Dimensions and Positions	67
4.2.3	Resource Preference Measures and Hard Constraints	69
4.2.3.1	Preference Measures: Proximity and Relocation Weights	69
4.2.3.2	Hard Constraints	72
4.2.4	Dynamic Layout Evaluation	74
4.3	Problem Solving Method .....	75
4.3.1	Pruning Positions to Meet Hard Constraints Using Constraint Satisfaction	76
4.3.2	Constraint Satisfaction and Propagation Algorithm (CSPA)	78
4.3.3	Optimal Positions to Meet Preference Measures Using Linear Programming	81
4.3.4	Heuristics for Resource Selection	86
4.3.5	Primary Time Frame Layout Construction Algorithm (PTFLCA)	87
4.4	Dynamic Layout Construction Strategy .....	90
4.5	Example Application.....	90
4.6	Capabilities, Limitations, and Possible Extensions .....	104
4.6.1	Approach	104
4.6.2	Spatial Representation	109
4.6.3	Possible Extensions for Process-Level Planning	110
4.7	Conclusions.....	111
<b>5</b>	<b>TIME-SPACE TRADEOFF STRATEGIES FOR SPACE SCHEDULING .....</b>	<b>112</b>
5.1	Introduction .....	112
5.2	Activity Time-Space Relationship .....	113
5.2.1	Resource Area Profiles	113
5.2.1.1	Profile-A	113
5.2.1.2	Profile-B	115
5.2.1.3	Profile-C	116
5.2.1.4	Profile-D	117
5.2.2	Discretizing Space Requirements	118
5.3	Activity Resource Levels .....	119
5.4	Spatial Conflict Resolver (SCR).....	122
5.4.1	Time-Space Tradeoff Strategies	123
5.4.1.1	Strategy A: Delay Activity	123
5.4.1.2	Strategy B: Lower Resource Level of Activity	124
5.4.1.3	Strategy C: Off-Site Storage	126
5.4.1.4	Strategy D: User Intervention	126
5.4.2	Strategy Selection	126
5.4.2.1	Mode 1: SCR Fully Automated	127
5.4.2.2	Mode 2: SCR User Driven	128

5.5	Scheduler.....	130
5.6	Space Scheduling Algorithm .....	130
5.7	Example Application.....	133
5.7.1	Input	133
5.7.2	Computations	138
5.7.3	Output	151
5.8	Capabilities, Limitations, and Possible Extensions .....	152
5.8.1	Model	152
5.8.2	Approach	155
5.8.3	Schedule-Layout Evaluation	161
5.9	Conclusions.....	163
<b>6</b>	<b>EXAMPLE APPLICATIONS OF MOVECHEDULE .....</b>	<b>164</b>
6.1	Introduction .....	164
6.2	Example Use of MoveSchedule for Solving Constrained Location Problem .....	164
6.2.1	Problem Statement	164
6.2.2	MoveSchedule's Solution	167
6.2.3	Traditional Solution	173
6.2.4	Summary	177
6.3	Evaluating Alternative Material Delivery Schedules .....	177
6.3.1	Modeling Assumptions	178
6.3.2	Input Data in MoveSchedule	183
6.3.3	Schedule and Layout Costs with Scenario I	186
6.3.4	Schedule and Layout Costs with Scenario II	196
6.3.5	Comparison of Scenario I vs. Scenario II and Conclusions	205
<b>7</b>	<b>MOVE_SCHEDULE VALIDATION.....</b>	<b>207</b>
7.1	Introduction .....	207
7.2	Project Description .....	208
7.3	Modeling Project Data in MoveSchedule .....	209
7.4	Space Schedule Construction .....	216
7.4.1	Layout Construction of PTF-0-13	217
7.4.2	Layout Construction of PTF-0-15	220
7.4.3	Layout Construction of PTF-0-22	223
7.4.4	Layout Construction of PTF-22-25	228
7.4.5	Layout Construction of PTF-25-34	232
7.4.6	Layout Construction of PTF-34-44	238
7.4.7	Layout Construction of PTF-44-59	240

7.4.8	Layout Construction of PTF-59-61	244
7.5	Summary and Validation of Results .....	246
7.5.1	Model Flexibility	246
7.5.2	Schedule Feasibility	249
7.5.3	Quality of the Dynamic Layout	250
<b>8</b>	<b>CONCLUSIONS .....</b>	<b>255</b>
8.1	Contributions to Knowledge.....	255
8.2	Capabilities and Limitations of MoveSchedule.....	260
8.3	Directions for Future Research .....	262
<b>APPENDICES</b>	.....	<b>266</b>
<b>BIBLIOGRAPHY</b>	.....	<b>288</b>

# LIST OF FIGURES

## Figure

1.1	Dynamic Layout Planning: A Unidirectional Relationship Between the Schedule and the Layout.....	7
1.2	Space Scheduling: A Bidirectional Interaction Between the Schedule and the Layout.....	10
1.3	Space Scheduling Process .....	11
2.1	Distance Measures .....	26
2.2	Tightly- vs. Loosly-packed Arrangements.....	29
2.3	Problems with Limited Space .....	46
3.1	Screen Dump of MovePlan's Layout Display .....	54
3.2	Screen Dump of MovePlan's Total Resource Histogram .....	57
3.3	Top and Bottom Portions of MovePlan's Total Space Histogram Window ....	58
4.1	Resource Primary Time Frames.....	67
4.2	Bounded Interval Representation .....	68
4.3	Constraint Satisfaction .....	77
4.4	Flow Chart of CSPA .....	80
4.5	Solution Proof.....	84
4.6	Flow Chart of PTFLCA.....	89
4.7	Example Input of Proximity Weights.....	91

4.8	Layout of PTF-0-2.....	92
4.9	SPPs Satisfying In-zone with Site Boundaries.....	93
4.10	SPP <sub>1</sub> and SPP <sub>3</sub> before and after Applying the Minimum Distance Constraint.....	94
4.11	Partial Layout of PTF-2-4 after Positioning R-1.....	95
4.12	Partial Layout of PTF-2-4 after Positioning R-3.....	96
4.13	Partial Layout of PTF-2-4 Showing SPP <sub>4</sub> .....	97
4.14	Graphical Solution for X <sub>4</sub> and Y <sub>4</sub> .....	98
4.15	Partial Layout of PTF-2-4 after Positioning R-4.....	99
4.16	Partial Layout of PTF-2-4 after Positioning R-6.....	100
4.17	Final Layout of PTF-2-4 from First Trial.....	101
4.18	Alternative Layout Solutions for PTF-2-4 .....	103
4.19	Actual vs. PTFLCA Computed Distances for Measuring Relocation Costs ..	108
4.20	Modeling Odd-shaped Resources.....	109
 5.1	 Profile-A .....	114
5.2	Profile-B .....	116
5.3	Profile-C and Profile-D Models.....	117
5.4	Primary Time Frames .....	119
5.5	Total Area Requirement of an Activity for Different Resource Profiles .....	121
5.6	Time-Space Tradeoff Strategies for Solving Spatial Conflict.....	125
5.7	Decision Tree of SCR in Mode 2 .....	129
5.8	Flow Chart of Space Scheduling Algorithm.....	131
5.9	Activity Network for Example Project .....	134
5.10	Site Layout with Long-term Static Resources for Example Project.....	134
5.11	Unconstrained Schedule .....	138
5.12	SPPs of Resources in PTF-0-2 after CSPA .....	140
5.13	Final Layout of PTF-0-2.....	142

5.14	SPPs of Resources in PTF-2-4 before Satisfying Hard Constraints .....	145
5.15	SPPs of Resources in PTF-2-4 after Satisfying Hard Constraints.....	146
5.16	Updated Schedule and PTFs .....	148
5.17	SPPs of Resources in PTF-2-4 after Spatial Conflict and after Satisfying Hard Constraints.....	149
5.18	Final Layout of PTF-2-4 from First Trial.....	151
6.1	Floor Layout of a Manufacturing Facility.....	166
6.2	MoveSchedule's Solution to the Drinking Fountain Problem using Fictitious Departments .....	172
6.3	Location of Drinking Fountain using Contour Lines .....	175
6.4	The Excavated Michigan Stadium.....	178
6.5	Stadium and Trenches as Modeled in MoveSchedule .....	179
6.6	Activity Network for the Pipelaying Job.....	180
6.7	Locations at which Pipes are Unloaded in Scenario I .....	181
6.8	Locations at which Pipes are Unloaded in Scenario II .....	182
6.9	Schedule and PTFs in Scenario I .....	186
6.10	Layout of PTF-0-2 With Scenario I .....	187
6.11	Partial Layout of PTF-2-6 .....	189
6.12	Partial Layout of PTF-2-6 with C-1 Positioned.....	190
6.13	Partial Layout of PTF-2-6 with C-1 to C-6 Positioned.....	191
6.14	Final Layout of PTF-2-6.....	191
6.15	Final Layout of PTF-6-10 .....	194
6.16	Final Layout of PTF-10-14.....	194
6.17	Final Layout of PTF-14-18.....	195
6.18	Final Layout of PTF-18-22.....	195
6.19	Final Layout of PTF-22-26.....	196
6.20	Schedule and PTFs of Scenario II.....	197
6.21	Layout of PTF-0-2 with Scenario II.....	198

6.22	Final Layout of PTF-2-4 With Scenario II.....	200
6.23	Layout of PTF-4-6 With Scenario II .....	203
6.24	Layout of PTF-6-8 With Scenario II .....	203
6.25	Layout of PTF-8-10 With Scenario II.....	204
6.26	Layout of PTF-10-12 With Scenario II .....	204
6.27	Layout of PTF-12-14 With Scenario II .....	205
7.1	Site Layout with Long-term Static Resources .....	209
7.2	Activity Network for Masonry Work.....	214
7.3	Unconstrained Schedule and Primary Time Frames.....	216
7.4	Schedule 1: Adjusted Schedule after First Conflict .....	219
7.5	Schedule 2: Adjusted Activity Schedule after Second Conflict.....	222
7.6	SPPs of Resources at the End of CSPA for PTF-0-22.....	225
7.7	Final Layout of PTF-0-22 .....	228
7.8	Final Layout of PTF-22-25.....	232
7.9	SPPC-8 in PTF-25-34 before Applying the Parallel Constraint with D-1 and the Out-zone Constraint with C-7 .....	234
7.10	Schedule 3: Adjusted Activity Schedule after Third Conflict .....	235
7.11	Final Layout of PTF-25-34.....	238
7.12	Final Layout of PTF-34-44.....	240
7.13	Final Layout of PTF-44-59.....	243
7.14	Final Layout of PTF-59-61.....	245
7.15	Arrangements of Pallets for Different Areas and L/W Ratios.....	247
7.16	Effect of Profile-A Consumption on Relocation Costs.....	254
B.1	East Facade Viewed from J.F. Kennedy Blvd.....	270
B.2	Typical South and East Facade.....	271
B.3	Building Cross Section AA.....	272

B.4	Commercial Level Plan.....	273
B.5	Second Parking Level Plan .....	274

# LIST OF TABLES

## Table:

2.1	Equipment Status Report (Table from Humphreys 91 pp. 479) .....	22
3.1	Weights for Prioritizing Substrategies in ConRes .....	63
4.1	Example Input of Resources.....	91
4.2	Solution Summary.....	99
4.3	Summary of Alternative Layout Solutions with CSPA.....	102
5.1	User-Defined vs. System-Computed Parameters of Area Profiles.....	118
5.2	Input Required in Activity Resource Levels .....	120
5.3	Independent Resources' Data for Example Project .....	135
5.4	Activities and Resource Levels for Example Project.....	136
5.5	Dependent Resource Data for Example Project.....	137
5.6	Proximity Weights in PTF-0-2.....	139
5.7	Hard Constraints in PTF-0-2.....	139
5.8	Summary of Layout Solutions for PTF-0-2 in 5 Trials .....	142
5.9	Proximity Weights in PTF-2-4.....	143
5.10	Hard Constraints in PTF-2-4.....	144
5.11	Summary of Layout Solutions for PTF-2-4 in 5 Trials .....	150

6.1	Departments and Related Data.....	167
6.2	Departments and Related Data as Defined in MoveSchedule .....	169
6.3	Method 1: Proximity Weights with the Drinking Fountain .....	170
6.4	Additional Fictitious Departments and Related Data .....	171
6.5	Method 2: Proximity Weights with the Drinking Fountain .....	171
6.6	Values of Z .....	176
6.7	Input of Independent Resources .....	183
6.8	Activity Resource Levels.....	184
6.9	Dependent Resources Data.....	185
6.10	Positions of Resources in PTF-0-2 as Input to MoveSchedule for Scenario I .....	187
6.11	Proximity Weights and Location Constraints for PTF-2-6 .....	188
6.12	Proximity Weights and Location Constraints for all PTFs with Scenario I ...	192
6.13	Summary of Resource Positions and VFL Values for PTFs in Scenario I....	193
6.14	Positions of Resources in PTF-0-2 as Input to MoveSchedule for Scenario II .....	198
6.15	Proximity Weights and Location Constraints for PTF-2-4 .....	199
6.16	Proximity Weights and Location Constraints for all PTFs with Scenario II..	201
6.17	Summary of Resource Positions and VFL Values for PTFs in Scenario II...	202
7.1	Independent Resource Data Input to MoveSchedule.....	210
7.2	Activities and Resource Levels Input to MoveSchedule.....	212
7.3	Dependent Resource Data Input to MoveSchedule.....	213
7.4	Additional Milestone Activities.....	214
7.5	Additional Dummy Resources.....	215
7.6	Proximity Weights in PTF-0-13 .....	217
7.7	Location Constraints in PTF-0-13 .....	218
7.8	Location Constraints in PTF-0-15 .....	220
7.9	Proximity Weights in PTF-0-15 .....	221

7.10	Location Constraints in PTF-0-22 .....	223
7.11	Proximity Weights in PTF-0-22 .....	224
7.12	Solution Steps in the Layout Construction of PTF-0-22.....	226
7.13	Location Constraints in PTF-22-25.....	229
7.14	Proximity Weights in PTF-22-25.....	229
7.15	Solution Steps in the Layout Construction of PTF-22-25 .....	230
7.16	Proximity Weights in PTF-25-34.....	233
7.17	Location Constraints in PTF-25-34.....	233
7.18	Adjusted Proximity Weights in PTF-25-34 after Third Conflict .....	236
7.19	Solution Steps in the Layout Construction of PTF-25-34 .....	237
7.20	Proximity Weights in PTF-34-44.....	239
7.21	Solution Steps in the Layout Construction of PTF-34-44 .....	239
7.22	Location Constraints in PTF-44-59.....	241
7.23	Proximity Weights in PTF-44-59.....	241
7.24	Solution Steps in the Layout Construction of PTF-44-59 .....	242
7.25	Proximity Weights in PTF-59-61 .....	244
7.26	Solution Steps in the Layout Construction of PTF-59-61 .....	244

## **LIST OF APPENDICES**

### **Appendix:**

- |    |   |     |
|----|---|-----|
| A. | List of Acronyms and Definitions.....               | 267 |
| B. | Project Data for Harvard Square Parking Garage..... | 269 |

# Chapter 1

## INTRODUCTION

### 1.1 Motivation

Planning a site's layout is crucial for enhancing the overall efficiency of construction operations and for reducing materials handling and re-handling costs. Field practitioners have become more aware of this problem as they realize that on some projects up to "\$0.40 per dollar that is spent on site is on materials handling" (Gore 95). Muehlhausen (91) states that the ratio of material handling costs to total project costs varies from 30 to 80% depending on the type of construction and the cost of materials. He further notes that "material handling adds cost to the project but not value", i.e., if materials are handled over long distances, double handled, or handled inefficiently, then costs increase with no increase in the value of the project.

Some field practitioners hold the general view that using *Just-in-Time* deliveries will eliminate the need for on-site storage and minimize materials handling on site. This procurement policy assumes that materials and other resources can be installed in the structure immediately after delivery, thereby eliminating the need for laydown space on site. This policy may be suited to resources that can be readily installed at the time of delivery, such as building components that are prefabricated off-site. Advantages to scheduling Just-in-Time deliveries include reducing the chance that materials will be:

1) damaged by lying around on site, 2) lost or misplaced on site, and 3) re-handled between staging areas.

Scheduling Just-in-Time deliveries, however, is not always possible. For example, bricks or concrete blocks used in masonry construction cannot all be installed at their final location in the structure immediately after delivery and will therefore occupy space on site during masonry construction. Moreover, choosing Just-in-Time deliveries is not always desirable because of uncertainties associated with deliveries in general. Delays in deliveries can slow down construction progress which in turn can be costly. Placing orders well ahead and planning enough float for procurement activities can minimize potential delays but would require storing resources on site for some time before using them in construction.

An alternative policy is to keep a supply of the resources on site ***Just-in-Case*** unforecasted shortages or delays occur so that construction activities are not delayed. Under this policy, multiple deliveries are typically made to keep an adequate supply throughout the performance of an activity (in the extreme, a single delivery of the total quantity of the resource can be brought to site when the resource is first needed). Multiple deliveries can be made on time before the resource is totally depleted or before it reaches a designated minimum safety stock. Under a Just-in-Case policy, resources will occupy space on site and may need to be relocated between staging areas before they are installed in the final structure.

In short, a Just-in-Time or Just-in-Case policy imposes different demands for space on site, but neither policy completely eliminates the need for on-site storage. Hence, planning the site layout is needed, under either policy, to minimize resource handling and re-handling on site.

The simplifications often introduced in planning a site layout reduce construction site layout to the static problem of allocating space to objects of predefined shape and subject to proximity constraints. The solution is a single layout (referred to as ***static layout***) for the entire project duration that includes all permanent and temporary facilities needed throughout construction. This practice ignores the possibility of reusing site space to accommodate different resources at different times, the possibility of relocating resources, and variations in space needs of resources over time. These should be explicitly modeled to minimize resource transportation and re-handling costs.

This practice also ignores the impact that site space limitations may have on construction method selection and on the project schedule, and therefore on the overall project cost.

"Aiming to reduce labor costs and cope with a limited site and time-table, a New Jersey utility and its contractors have successfully "rolled in" a 2000-ton precipitator... from its temporary construction site to its ... home at Trenton area power plant. Off-site construction allowed PSE&G to reduce site foundation work, allowing it to meet a tight schedule and manpower needs ... the roll-in approach saved \$3 million on the \$20-million precipitator job." (ENR 90)

ENR (90) shows that limited space on site did constrain the choice of the construction method used for this precipitator job. Choosing off-site construction saved the space needed for the performance of other activities, thus preventing the delay of these activities and of the overall project completion. Hence, planning the use of site space over time along with construction method selection and activity scheduling is key to constructing a project efficiently.

Site layout and activity scheduling have traditionally been modeled and solved as independent problems. Their interdependency is often ignored at the planning stage and may be dealt with only when construction is underway, at which time problems are expensive to remedy.

This research focuses on two problems that exist in the way the layout and activity scheduling problems are currently modeled. Current models do not consider

1. Temporal data in representing and solving the site layout problem
2. Spatial data in solving the activity scheduling problem

This research proposes an integrated scheduler and space planner, named **MoveSchedule**, that adjusts an initial construction schedule to comply with site space limitations. Adjusting the schedule includes changing start dates or durations of activities and investigating how this will impact the amount of free space on site. Determining the amount of free space on site at any given time involves creating a layout of those resources that exist on site at that time.

## 1.2 Terminology and Problem Definition

This research solves the following two problems: the dynamic site layout problem and the space scheduling problem; the former is a subproblem of the latter.

The *dynamic site layout problem* is that of allocating site space to resources governed by a construction schedule so that their space needs—which can vary over time—are satisfied and so that all constraints on their relative locations are met. The objective in this problem is to minimize overall resource transportation and relocation costs subject to constraints on relative locations of resources. This problem may have a solution or may not. It has a solution if it is possible to find positions on site for all resources that do not violate the problem constraints. It does not have a solution if the available space at a given time is not enough to accommodate the resources needed at that time, without violating the problem constraints. In this case, a spatial conflict is identified but not solved. Note that, the dynamic site layout problem does not modify durations of activities in the schedule nor their scheduled start and finish times. These are considered given.

The *space scheduling problem* is that of scheduling activities while minimizing the project duration subject to space availability constraints. The space scheduling problem is a type of limited resource scheduling problem where the limited resource is space. It involves modifying an initial construction schedule by changing start dates or durations of activities. These changes modify activities' space needs and thus the amount of unoccupied space on site. Determining how much space is free on site at any time necessitates solving a dynamic layout problem. In other words, the space scheduling problem solves spatial conflicts detected in the process of solving the dynamic site layout problem. This is done by modifying the construction schedule while minimizing the increase in project duration.

The following two sections present definitions of terms and variables used in this dissertation to model the two problems. The section after that defines the scope and assumptions of this research.

### 1.2.1 Dynamic Site Layout Problem

The word *resources* is used to denote any of a wide variety of physical or regional objects that occupy space on site. This is the case for obstacles (such as existing buildings or trees), facilities that support construction operations (such as trailers or parking lots), materials, equipment, or simply demarcated areas (such as excavation or laydown areas).

gates, or roads). The general problem of allocating site space to resources is known as *site layout*. Site layout has traditionally been treated in the literature as a static layout problem that ignores many of the problem's dynamic properties, including variations in resource space needs over time, the reallocation of space over time, and the relocation of resources.

The demand for site space over time depends in part on the schedule of production and procurement activities, referred to here as the *schedule*. **Production activities** define a quantity of work to be produced by the direct application of resources (including materials, labor, or equipment) to build a product described in the project plans and specifications. Examples are placing concrete, excavating, installing reinforcing steel, etc.

**Procurement activities** reflect the time it takes to have the resources required by production activities available and ready on site by the time those activities start. Examples of procurement activities are placing purchase orders, obtaining permits and licenses, etc. Variability in the delivery of resources to the site, and the time required to inspect materials for quantity, type, and condition, and to log packaging slip data into a materials management computer system, often makes it necessary to order resources so that they arrive on site some time before the scheduled start date of the production activity that will use them. **Mobilization** is here defined to be the time period between the resource's delivery to site and the scheduled start date of a production activity. **Demobilization** is the time period between the finish date of a production activity and the resource's removal from site. For example, some resources such as tower cranes need assembly (mobilization) and disassembly (demobilization).

Production or procurement activities use resources that may or may not require space on site. **Dependent resources** are associated with one or several such activities and require space on site for a time period related to the activities' start and finish dates. A dependent resource is said to be *active* throughout the duration of any activity it is associated with. Otherwise, it is said to be *idle*, though it likely will continue to occupy space on site. This situation may occur when the resource is associated with non-consecutive activities and remains on site in-between activities.

**Independent** resources occupy space on site but cannot be associated with particular activities. Examples are facilities that support construction operations in general (such as construction management trailers, roads, warehouses, trees, obstacles on site).

The time when a resource arrives or is first needed on site marks a new demand for space. This demand needs to be satisfied by finding a position for the resource given the current and possibly future constraints on space use. The time when a resource is removed from site marks an opportunity for other resources to occupy the freed space. Not only could a new resource brought to the site at that time occupy this freed space; a resource already present on site may be relocated there if the value of it being in the new position outweighs its relocation cost. For example, excavation may require space close to a foundation wall to stockpile excavated soil, while the same space can be used by lumber for formwork fabrication at a later time when the stockpile is gone. The smallest time intervals demarcated by the arrival and departure of resources are termed ***Primary Time Frames*** (PTFs).

The amount of space needed to accommodate a resource depends on the resource's characteristics. Rigid objects (e.g., trailers or equipment) occupy an area at least equal to their footprint dimensions, which tend to be constant over time. Other resources do not have fixed dimensions and can be made to fit in a given space. This is the case of bulk materials (e.g., gravel or sand) or materials that can be divided into smaller components and stacked (e.g., trusses, bricks, or concrete blocks). Physical resources (e.g., cranes) cannot occupy space that is occupied by other physical resources, i.e., they cannot overlap. In contrast, resources designating regions on site (e.g., excavated or laydown areas) only conceptually occupy space and could overlap with others. For example, a laydown yard can occupy the same space that materials or equipment occupy. ***Mobile resources*** (e.g., a pipelayer involved in a pipelaying activity or trucks involved in a hauling activity) do not occupy a single, permanent space on site; rather, they travel along a path which is shared by other resources traveling along the same path.

The space required by a construction material is likely to vary with time as the corresponding activity progresses: materials may be ***consumed*** (e.g., the quantity of engineered backfill material decreases when excavated trenches are backfilled) or may ***build up*** (e.g., formwork accumulates as it is being stripped). The amount of space needed over time to accommodate a resource also depends on the procurement policy which determines the number, the size, and the frequency of deliveries (e.g., a single delivery of the whole quantity of a resource or multiple deliveries of smaller quantities can be made).

Not all locations on site are equally suited for positioning a resource: ***feasible positions*** of a resource are those positions where the resource meets all its location constraints. ***Location constraints*** describe spatial relationships between relative positions

of resources. For example, as previously stated, resources may or may not be able to overlap. Resources that are highly susceptible to environmental conditions (e.g., weather, dust, or noise) should be covered or warehoused. Regulatory and safety requirements may impose explicit constraints on relative positions of resources. For example, gasoline or natural gas containers must be kept a minimum distance away from buildings or oxygen tanks and this distance varies with the gas quantity.

When a resource is brought to site, a position that meets all its location constraints must be found. Satisfying the constraints may force the relocation of other resources to accommodate the incoming resource. Interactions between resources should also be weighed when determining the best position for a resource. These interactions can change over time. For example, a loader and backfill material will interact during backfill placement. They will not interact anymore when the excavation activity is finished, whereupon each resource may be assigned to a different activity. When interactions between resources change with time, it may be favorable to relocate those resources so that they may function better.

The *quality of the overall layout* influences how economically construction activities can be performed. A generally accepted way of constructing quality layouts is to position resources so that transportation costs are kept small. **Transportation costs** are costs associated with travel distance from storage (e.g., laydown or staging areas) to point of use (e.g., fabrication or installation areas). Another way to construct quality layouts is to avoid relocating materials or equipment so that relocation costs are kept small. **Relocation costs** are costs associated with travel distance from one storage area to another.

Given a schedule, constructing the *dynamic site layout* consists of determining 1) when resources need space, 2) how much space they need, and 3) where to position them on site so that constraints on their relative positions are met and so that transportation and relocation costs are minimized (Fig. 1.1). The result is a sequence of layouts for the various PTFs; each layout in the sequence spans a unique time interval and shows the resources' positions and dimensions during that interval.



Figure 1.1  
Dynamic Layout Problem:  
A Unidirectional Relationship Between the Schedule and the Layout

### **1.2.2 Space Scheduling Problem**

While dynamic layout planning helps identify construction time periods with potential spatial conflicts due to site space limitations, it does not resolve these conflicts. Once conflicts have been identified, other methods are needed to resolve them. For instance, one may lower the demand for space over such problematic time periods. This gives rise to two questions:

1. How can the demand for space be quantified?
2. How can spatial conflicts be resolved?

It is hard to quantify space. People may think in terms of areas or volumes when it comes to quantifying space. But these measures do not capture all attributes such as area, shape, height, and volume needed for measuring space needs. This research considers area and shape to quantify the demand for space over a given time period, but does not consider height.

Many actions can be taken to resolve a spatial conflict. They often involve lowering the demand for space over problematic time periods, including:

- Changing the start date of selected activities to shift the space requirement of associated resources either forward or backward in time. Of course, this action may only postpone or advance the occurrence of spatial conflicts. For example, delaying the start date of on-site fabrication of trusses allows delaying the need for erecting a fabrication shop.
- Changing the construction method used for performing selected activities: different methods require different resources and have different space requirements. For example, concrete columns can be cast in place using crane-and-bucket or a concrete pump, or they can be precast and erected on site using a crane. The space required to perform the concrete placement activity and the constraints on where this space can be located are different for each of these methods.
- Lowering the number of resources used for performing selected activities. For example, an excavation activity can be performed with one or two loaders. One loader occupies only half the space of two.
- Changing the order in which activities are executed.
- Using off-site storage to increase the total space available. Off-site storage can be some remote area or a lot adjacent to the site.

- Reducing a resource's mobilization and demobilization times to free up the space that it occupies while idle.
- Changing procurement policy. For example, the delivery of brick can be staged throughout a masonry activity: not all brick is needed at the start of this activity.

Each of these actions may affect the timing of other activities in the schedule. In turn, this may affect how much space is needed and when.

Such actions will not prevent all conflicts: some resources simply may not fit in the space available on site. Even when actions are taken during project planning to alleviate anticipated space conflicts, they will not necessarily prevent spatial conflicts from occurring during project execution. Uncertainty about durations of activities and delivery times of resources affects how much space is needed and when. Actions similar to those mentioned above can often be taken to update or adjust the schedule during project execution to solve real-time spatial conflicts.

Whether the aforementioned actions affect the schedule or not, they all come at a cost. Their cost should be weighed along with their impact on the schedule in choosing the best action. For example, choosing multiple deliveries instead of a single one may be more expensive because of additional packaging and shipping costs. Renting off-site storage adds a cost to move resources from their storage location to the site.

Space scheduling consists of 1) identifying time periods with spatial conflicts due to site space limitations by constructing the dynamic layout, 2) applying actions that lower the demand for space over problematic time periods by adjusting the schedule, and 3) reassessing the feasibility of the schedule and layout (Fig. 1.2). The result is a feasible schedule and dynamic site layout.

Space scheduling is a limited resource scheduling problem with the limited resource being space. Solving this problem is an iterative process, as it relates to determining how much site space an activity needs and how much site space is available for its performance. For example, how much site space an activity needs depends on the available site space, as this limits the method and resources that can be used for its performance. Likewise, how much space is available on site for its performance depends on the space requirement of other activities taking place at the same time.



Figure 1.2  
Space Scheduling:  
A Bidirectional Interaction Between the Schedule and the Layout

Fig. 1.3 illustrates where dynamic layout planning fits in the space scheduling process. It shows how temporal project data, represented by a schedule of four activities and associated resources (represented by the patterned rectangles inside the activity boxes), is used in constructing the dynamic layout. The start and finish dates of the activities mark time periods during which specific sets of resources are needed on site. For example, in the time period 0 to 4, two activities take place. They require resources, namely lumber, backfill material, and a crane (represented by the dotted, cross-hatched, and solid black rectangles respectively) for their performance. These resources require space on site, as shown at the left of Fig. 1.3, equal to the areas of the rectangles that represent them. Constructing the layout for the time period 0-4 then consists of finding feasible and economical positions for these resources on site. A layout of these resources (layout-0-4) is shown in the lower right-hand corner of Fig. 1.3, which also includes layouts for subsequent time periods.

This figure also shows how constructing the dynamic layout helps identify a spatial conflict in the time period 10-12. In positioning the resources needed during that time period, it becomes clear that the concrete batch plant (represented by the gray shaded rectangle below the layout-10-12) cannot be positioned on site without overlapping with other resources needed in this time period. This spatial conflict can be solved by relocating other resources, delaying the activity requiring the use of the concrete batch plant, or changing the construction method to one using ready-mix concrete, for example. These changes may affect the schedule as illustrated by the left and right arrows pointing upward. Hence, this figure shows that dynamic layout planning is a subproblem of the space scheduling problem.

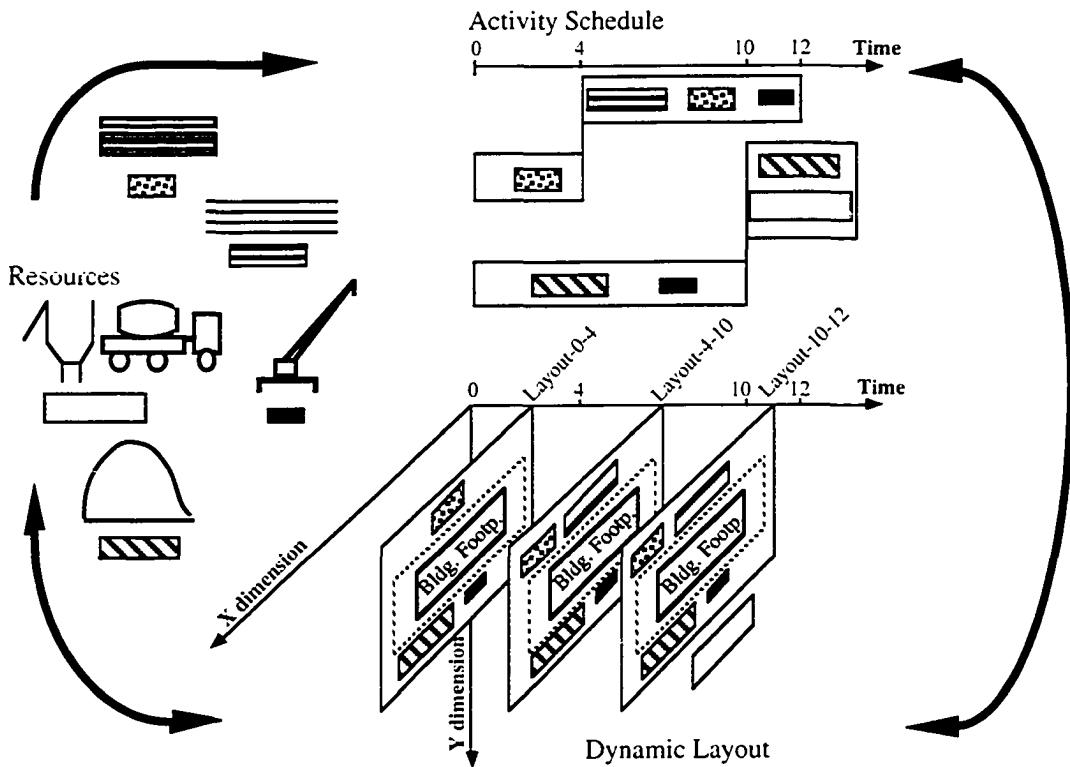


Figure 1.3  
Space Scheduling Process

### 1.2.3 Research Scope and Assumptions

The primary application area of this research is construction, but clearly, as long as the assumptions made in the various models are satisfied, the proposed methods may be applied to any problem domain.

The space scheduling problem is difficult to solve optimally. In fact, even solving subproblems of it is difficult (as discussed later in Chapter 2). For example, constructing the static layout for a given set of resources has been formulated as a Quadratic Assignment Problem (QAP) or a Mixed Integer Program, and both formulations are known to be NP-complete, i.e., optimal solutions can be computed only for small or greatly restricted problems (Kusiak 87). Constructing the dynamic layout for a set of resources tied to a schedule has been formulated as a Dynamic Program with a QAP to solve at each stage (Rosenblatt 86). Scheduling activities with resource capacity constraints is exceedingly difficult for projects of even modest size. Allocating limited resources to activities and scheduling activities with the objective of minimizing the project duration has been

formulated as a Mixed-Integer Program which also requires exponential time to solve (Wiest and Levy 67). Consequently, it does not seem possible to obtain a globally optimal solution to the space scheduling problem.

Therefore, the following simplifications were made to reduce the problem to a manageable size:

**1 All resources (equipment, materials, labor, time, and money) with the exception of space exist in unlimited quantities.**

To better focus on the impact that site space limitations have on the schedule, the problem has been limited to address spatial constraints only and no other resource capacity constraints.

**2 Activity precedence constraints are fixed.**

For the same reason, activity precedence constraints are fixed with no mandatory constraints on start and finish dates of activities.

**3 Resources are laid out in a continuous 2-dimensional orthogonal space.**

Space is modeled as a 2-dimensional variable. To keep the representation simple, the space considered is limited to an orthogonal space where layout objects are represented by rectangles with 0° and 90° orientations only.

**4 The continuous movement of mobile resources is not modeled.**

The continuous movement of mobile resources is not modeled in this dissertation as it involves planning trajectories or paths along which the resources travel. Instead, this research focuses on modeling the dynamics of staging areas and relatively short-term laydown space used for the temporary storage of materials, stationary equipment (e.g., concrete batch plant or mobile cranes with outriggers), prefabrication shops, and work areas.

## **5 Only layout-related costs are modeled.**

This research focuses on the tradeoffs between time and space: therefore, only layout costs, namely resource transportation and relocation costs, are considered to guide the layout of resources on site.

## **6 The merits and shortcomings of the proposed problem solving method are qualitatively assessed.**

Given the complexity of the problem, this research does not aim at finding a globally optimal solution to it. This research uses heuristics to find feasible and satisfactory solutions. A solution is feasible if all resources can be positioned to meet their constraints. Whether a solution is satisfactory depends on the subjective judgment of a human expert.

### **1.3 Research Objectives**

The main goal of this research is to develop a systematic approach for project managers and field engineers to construct quality dynamic site layouts for a given construction schedule that meets a given project duration. This schedule essentially provides constraints on arrival and departure times of resources. In the case where no feasible solution is found, a second goal of this research is to develop a systematic approach to adjust the schedule by changing durations and start dates of activities so that a feasible dynamic layout is found. To achieve these two goals, we established the following tasks:

#### **1 Build a model that ties spatial and temporal data related to resources and activities in a schedule.**

The model should articulate resource space needs and variations in their space needs over time. The model should characterize space needs of activities in a schedule based on the methods and resources used for their performance. It should also represent interactions between resources and constraints on their locations, and account for variations in these interactions and constraints over time.

**2 Develop interactive computational mechanisms to aid people in planning dynamic layouts.**

Interactive computational mechanisms should assist users in constructing dynamic layouts by keeping track of changes in resource positions over time.

**3 Develop a heuristic construction algorithm to solve a constrained dynamic layout problem.**

The heuristic algorithm should seek solutions that minimize resource transportation and relocation costs subject to location constraints on relative positions of resources and changing space needs of resources over time.

**4 Develop a heuristic improvement algorithm to solve the space scheduling problem.**

The heuristic algorithm should seek solutions that minimize the project duration subject to 2-dimensional space availability constraints. The approach should investigate time-space tradeoff strategies that modify the schedule to vary the demand for space at a given time. Heuristics for applying these strategies so that the project duration is minimized should also be investigated.

**5 Test and validate the developed methods.**

Automation of the developed methods is needed to test and validate these methods. Automation is also desired because it provides the user with a tool to generate and evaluate alternative solutions, and it allows us to experiment with the limitations and capabilities of the methods to recommend possible future improvements.

## **1.4 Thesis Guide**

Chapter 2 begins with an overview of site layout field practice. It then reviews the literature in two problem areas: site layout and limited resource scheduling. Section 2.2 reviews previous efforts to solve the layout problem in various domains. It discusses the capabilities and limitations of existing layout models in addressing the dynamic characteristics of construction site layout. Section 2.3 reviews models for limited resource scheduling and focuses on their applicability in modeling space as the limited resource. Section 2.4 reviews recent attempts at incorporating spatial data in characterizing activities

in a schedule. Section 2.5 situates this research in the broad spectrum of the reviewed literature and identifies the modeling needs for MoveSchedule, i.e., the model implemented as part of this research.

Chapter 3 presents interactive dynamic layout planning and space scheduling tools developed to explore several of the modeling needs.

Chapter 4 describes the first module in MoveSchedule: the dynamic layout construction module which solves a constrained dynamic layout problem. This chapter presents the modeling assumptions and choices pertaining to this module, namely the choice of variables to represent transportation and relocation costs of resources and location constraints on their relative positions, the choice of spatial representation, and the choice of an evaluation function. Sections 4.3.2 and 4.3.3 present an algorithm for finding the optimal position of a resource in the layout. Sections 4.3.4 and 4.4 describe heuristics for applying this algorithm in constructing the dynamic layout. An example illustrates the applicability of the proposed methods. Chapter 4 concludes with a detailed discussion of the capabilities and limitations of the proposed methods.

Chapter 5 describes the remaining two modules in MoveSchedule: the Spatial Conflict Resolver which solves spatial conflicts as they arise in the process of creating the dynamic layout, and the scheduler to keep the schedule updated to reflect the changes made by the Spatial Conflict Resolver. This chapter presents their underlying model, namely the resource area profiles and a time-space relationship for each activity in the schedule based on minimum, normal, and maximum resource levels. Next, time-space tradeoff strategies and heuristics for applying these strategies are presented. A flow chart of the algorithm that integrates all three modules of MoveSchedule to automate the space scheduling process is presented. An example illustrates the use of the proposed strategies. Chapter 5 concludes with a discussion of the capabilities and limitations of the overall system.

Chapter 6 presents two example applications of MoveSchedule. The first example applies the algorithm presented in Chapter 4 to optimally solve a single-facility location problem with constraints on acceptable locations of the facility. The second example illustrates the use of the model presented in Chapter 5 to choose between alternative material delivery schedules.

Chapter 7 concerns the validation of MoveSchedule. It details how MoveSchedule is used to model a real project and how it adjusts the project's schedule to comply with site

space limitations. The last section evaluates the results according to model flexibility, schedule feasibility, and quality of the dynamic layout.

Chapter 8 concludes the dissertation by stating this research's contributions to knowledge, summarizing the capabilities and limitations of the proposed model and system, and suggesting directions for future research.

# **Chapter 2**

## **SITE LAYOUT AND LIMITED RESOURCE SCHEDULING: INDUSTRY PRACTICE AND LITERATURE REVIEW**

Section 2.1 reviews site layout and scheduling practice and draws lessons from them. Because space scheduling is so intertwined with site layout, materials management, activity scheduling and resource scheduling, the remainder of this chapter reviews the literature in these problem areas. Section 2.2 reviews layout models. Static layout models are reviewed first because this research builds on some concepts and methods used in these models. Layout models that account for time and related work on layout implementation, control, and re-planning are reviewed next. Section 2.3 reviews both unlimited and limited resource scheduling models and focuses on their applicability to modeling space as a limited resource. Section 2.4 reviews recent attempts at incorporating spatial data in characterizing activities in a schedule. The last section situates this research in the broad spectrum of the reviewed literature and identifies the modeling needs for MoveSchedule.

## **2.1 Site Layout and Scheduling Practice**

### **2.1.1 State of the Practice**

This section summarizes the state of site layout and scheduling practice based on the experience gained in conducting this research and based on the findings of others (e.g., Tommelein 89, Riley 94, Muehlhausen 91) who have interviewed field practitioners to learn about how they lay out construction sites and how they handle spatial interference problems between resources on a construction site.

Site layout planning consists of locating temporary storage for materials, equipment, and facilities that support construction operations within the site boundaries so that these resources can be accessible or functional during construction.

Field practitioners tend to reason about layout changes over time in a sequential manner paralleling construction time. Early layouts get more attention and are developed in greater detail than later layouts, because adequate data for creating a detailed layout is available only for the immediate future, and uncertainty about the construction schedule and layout data increases for the more distant future. However, major and critical factors that affect the layout in the long-term future are taken into account early on, when the first layout is constructed, so that major future work can be adequately performed.

In constructing "early layouts", field practitioners manually mark up a single site drawing to include all major resources needed on site throughout the duration of the project. They draw on knowledge of years of experience in determining positions of resources on site. Because of human cognitive limitations, however, they cannot possibly keep track of all factors that could affect the selection, location, and interactions of all resources to be positioned. Therefore, field practitioners tend to use an *early commitment* approach in constructing the layout. This means that resources are positioned one at a time and, once positioned, each resource remains at that position throughout its presence on site. Consequently, it may restrict possible positions of other resources to be positioned.

The outcome of this approach is a single drawing, that, in reality, consists of an overlay of several drawings, each showing information regarding resources that are on site for different phases of construction and that may be relocated from one phase to the next. For example, the drawing may be a marked-up blueprint of the site arrangement that shows

the permanent facilities at project completion. The marks may show major excavations needed for installation of building foundations, roadwork, and so on. They may also show how areas, excavated at one time and backfilled later, are used as material laydown areas during another phase of construction. This drawing is rarely updated as construction progresses, despite the fact that it is used throughout the project.

Field practitioners have more than one reason for using a single drawing to construct the site layout:

- They want to have all information regarding space use on site available on one document.
- Unanticipated and unpredictable events inherent to construction practice, and uncertainties associated with the timing of construction operations, lead practitioners to hold the general view that detailed spatial planning is impractical. They therefore consider a single drawing that shows the major resources needed for construction to be good enough.
- Many changes take place over time on a construction site. This means that it could require (too) much (manual) work for practitioners to update different site-layout drawings to keep track of all resources, especially those that move around on site. A single drawing seemingly would require less work to update.

However, a single layout drawing to show space use over time is difficult to use if not misleading. Any person who is to interpret this drawing needs to imagine how the layout will evolve over time and thus must have good spatial visualization skills and thorough knowledge of the construction schedule. Using a single drawing to show how resources are relocated between staging areas and how space needs of resources vary over time, if at all possible, is even more misleading.

It is common practice to rely on a field engineer's common sense to ensure that resources are provided adequate space on site as their space needs vary with construction progress. The engineer can influence to some extent the amount of space needed to accommodate some resources and the length of time they will exist on site by carefully choosing procurement policies (e.g., frequency of deliveries and the quantities of each delivery) and provided that the home-office purchasing group has not already done so. For the most part, however, a resource's space needs are dictated by its nature, and the purpose for its existence on site (e.g., common earth used by a backfilling activity will decrease in space need; in contrast it will increase in space need when it is generated by an excavation

activity). Accordingly, space needs of construction resources and their variation with construction progress can be roughly categorized as follows:

- A. *Decreasing space need as construction progresses.* This can describe the space need of a divisible resource (such as bricks or pipes) that is consumed in some fabrication, erection, or installation activity. A decreasing space need can be the result of a procurement policy that brings to site the total quantity of a resource up-front when the resource is first needed. Such a policy is typical if the shortage cost is high, large deliveries are cost effective, and space availability is adequate to accommodate the resource early.
- B. *Seemingly constant space need as construction progresses.* This can describe the space need of a divisible resource (such as fabricated steel or pipes) that is consumed and replenished at such a rate that the space it occupies on site appears to be constant over time. A constant space need can be the result of a Just-in-Case procurement policy that keeps an adequate supply of the resource for the duration it is needed in the production activity. This policy is typical if the cost of stocking or handling the resource on site is high, if the supplier cannot deliver the entire order at once but delivers it in stages, or when site space is limited.
- C. *Constant space need that is a function of construction progress.* This can describe the space need of an indivisible resource with fixed dimensions that is neither consumed nor generated by a construction activity, but, rather, can be reused by different activities. An example resource is construction equipment such as a concrete batch plant.
- D. *Constant space need that is not a function of construction progress.* This can describe the space need of an indivisible resource that is not used by any specific construction activity. Example resources are trailers, parking facilities, and toilets.
- E. *Increasing space need as construction progresses.* This can describe the space need of a divisible resource (such as earth, forms, or debris) that is generated by some excavation, stripping, or demolition activity. An increasing space need can be the result of accumulating the resource on site before removing it.

It is also common practice to rely on the field engineer's ability to keep track of the numerous resources on site over specific time periods and to remedy spatial interference problems as they arise during construction. At that time, however, it is expensive to remedy such problems, especially when these might have been avoided by planning ahead the site layout over time.

When faced with a problem on site, field engineers look for a solution to the spatial conflict that will not jeopardize contractual requirements. Under a general contracting agreement, these are primarily the overall project duration and the total project cost. For some types of construction projects (e.g., highway construction), exceeding the project due date signifies an increase in cost, as contracts contain some type of liquidated damages clause where the contractor is assessed a given amount of money for each day construction continues beyond the stated completion date (Seibert and Evans 91). Contractors thus have an incentive to plan their work carefully, especially when they are rewarded for finishing the project earlier than the specified due date. This planning includes creating the site layout and studying materials handling.

Field practitioners work with many different plans and schedules. One such schedule may show production activities and some milestone activities to mark the delivery to site of key resources. Another schedule (also known as the delivery schedule, procurement schedule, or materials and equipment status report) may show dates related to ordering, and planned and actual delivery of some critical resources. Table 2.1 shows an example equipment status report. These reports are crucial for maintaining continuous progress on production activities and for meeting schedule and budgetary constraints. Consumption of resources, and therefore the space they occupy on site, depends on delivery schedules and the rate at which the corresponding production activities are being performed.

In the discussion so far, field practice was reviewed from the standpoint of the person responsible for the layout on site on a day-to-day basis. From the standpoint of the scheduler, i.e., the person responsible for generating the schedule site space data is not considered with much detail. This may be so because:

- Field data is not available at the time the schedule is developed and only rough space estimates can be used. An exception to this is rigging studies done to determine the required crane capacities.
- The scheduler does not know what spatial data to look for and how it may impact the schedule.
- There are no available methods for scheduling activities subject to site spatial constraints to assist the scheduler in this process.

Table 2.1  
Equipment Status Report (Table from Humphreys 91 pp. 479)

MATERIALS MANAGEMENT										
MINOR EQUIPMENT SUMMARY										
ENGR ITEM NO.	DESCRIPTION	ORDER NO.	VENDOR	ORIGINAL PROMISE	LATEST PROMISE	FIELD NEED	RCVD. DATE	SET DATE	COMPLETE DATE	
CATEGORY: INSTRUMENTATION										
IN FE-090-005	ORIFICE PLATE & FLANGE	P61024	DANIEL IND.	01DEC79	15DEC79V					
IN FI-090-005	FLOW INDICATOR	P71000	DUPONT INST.	15DEC79	15DEC79S	15DEC79	16DEC79			
CATEGORY: ELECTRICAL SPECIALTY										
ES 090-13-011-5	600 V SAFETY SWITCH	P62653	NUNN ELECTRIC	14MAR80	14MAR80S	14MAR80	20MAR80			
ES 090-13-011-6	JUNCTION BOX	P62750	NUNN ELECTRIC	30JUN80	30JUN80E	01JUL80				

The scheduler works from the office in generating the schedule and draws information from many sources to come up with a reasonable target schedule. Such sources include the project blueprints used to determine the activities and their logical sequencing, and the detailed estimates of work quantities and construction methods needed for performing activities and setting their durations. The schedule is subsequently tailored to account for budget and duration constraints. There are several ways in which the scheduler can systematically build the schedule and adapt it to meet time and budget constraints, but space constraints are often overlooked except in some extreme situations. For example, in restoration or rehabilitation projects, site access constraints and constraints imposed by current operations must be considered when developing the schedule. These constraints may require altering the existing structure temporarily to provide ports of entry, or installing supplemental elevators or material handling systems.

### **2.1.2 Lessons from Site Layout and Scheduling Practice**

To address the needs of field practitioners, this research develops a methodology to plan the site layout over time, so that potential spatial problems can be detected before they arise and solutions can be identified. The proposed approach for constructing the dynamic layout is sequential and parallels construction time. This approach reflects the availability of layout data and it reflects a logical way to reason about layout changes over time.

The problem-solving method should address the cognitive limitations of the human planner in deciding about a "good position" for a resource by determining positions that improve resource flow on site. For example, Muehlhausen (91) includes in his definition of tasks for improving material flow on a construction site the following:

- minimizing travel distance from material storage to material point of use
- eliminating duplicate handling of material.

The first task corresponds to minimizing transportation costs as defined in this research (see Chapter 1). The second task involves minimizing the number of moves and the distance over which material is relocated between storage areas. The latter corresponds to minimizing relocation costs as defined in this research (see Chapter 1). The problem-solving method therefore will minimize transportation and relocation costs in constructing the dynamic layout. It will also allow space reuse and the relocation of resources in determining resource positions.

The outcome of this method will replace the single layout drawing by a sequence of layouts, each spanning a unique time period and showing the resources at their positions during that period.

To address the needs of the scheduler, this research develops a model that characterizes the spatial needs of resources and activities in the schedule and presents a methodology for adjusting the schedule while minimizing the project duration subject to space availability constraints.

## 2.2 Layout Models

The subject of layout cuts across several disciplines. For example, electrical engineers lay out micro chips on an electronic board, industrial engineers lay out departments within a production facility and machines within a department, and construction engineers lay out resources on a construction site.

The *facility layout problem*, in the area of production facilities, is by far the most widely studied static layout problem (European 92). The literature on facility layout is therefore reviewed in addition to the literature on construction site layout. Both static and dynamic formulations of the problem are considered. Because the literature on the dynamic layout problem builds on concepts and variables used in static layout models, static layout models are reviewed first.

### 2.2.1 Static Layout Planning Models

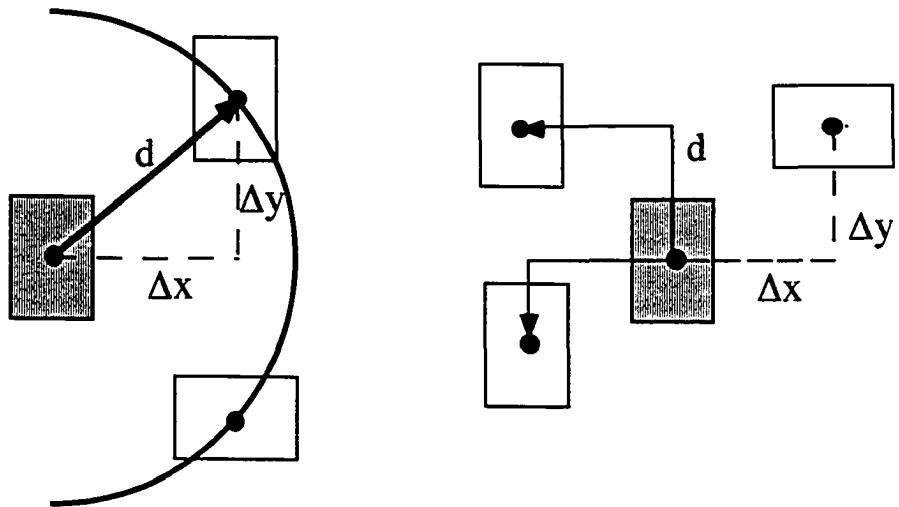
A *static layout*, by definition, does not model any aspect of time; each object in the layout occupies a single position and will remain at that position for the duration of the layout. For example, the static layout of a construction site is a single site arrangement drawing that includes all major facilities needed on site throughout the duration of construction. The static layout of a floor within a manufacturing facility, known as the *block layout*, is a single arrangement of all departments on that floor.

The *static layout problem* is defined as the task of arranging a set of objects that interact with each other, so as to optimize a certain objective without violating a set of constraints. These constraints include *area constraints* (e.g., layout object has a 12 ft<sup>2</sup> area), *overlap constraints* (e.g., layout objects should not overlap with each other),

*shape constraints* (e.g., layout object is a rectangle with an aspect ratio of 5/2), and *location constraints* (e.g., layout objects should be closer than a certain distance from one another, or should be adjacent to each other). Some people (e.g., Tommelein 89, Cheng 92) include in the static layout problem the task of determining the size and shape of the objects to be laid out.

*Interactions between layout objects* are often translated into proximity requirements and can be specified in a qualitative or a quantitative manner. Interactions may be described qualitatively, for example, using the AEIOUX *closeness relationships* proposed by Muther (73). The letters in AEIOUX stand respectively for: **A**solutely necessary, **E**specially important, **I**mportant, **O**rdinary closeness OK, **U**nimportant, and **U**ndesirable or eXcluded. These relationships rate the importance of locating two layout objects close to one another. For example, if two layout objects have to be adjacent in the layout, the value A describes their level of interaction. Interactions may be quantitatively described to measure the flow of materials or some other flow between layout objects. The *flow* can be expressed as the number of trips made, or the quantity of materials moved between two layout objects for a given time period. Determining equivalent flow values for a given problem is necessary when different items are moved between pairs of layout objects. More details on this subject are found in Tompkins and White (84).

The distance measures used to assess closeness between layout objects are an important element in formulating an analytical model. When a distance is measured along a straight-line path connecting two points, it is referred to as *Euclidean distance* (see Fig. 2.1 (a)). When a distance is measured along a path that is parallel to a set of perpendicular (orthogonal) axes, it is referred to as *rectilinear* or *Manhattan distance* (Fig. 2.1 (b)) because of its use in measuring distances within a city like New York, in which all streets are orthogonal. Distances can also be measured between facing edges of objects (Fig. 2.1 (c) and (d)).



$$d = (\Delta x^2 + \Delta y^2)^{1/2}$$

(a) Euclidean distance between centroids      (b) Rectilinear distance between centroids

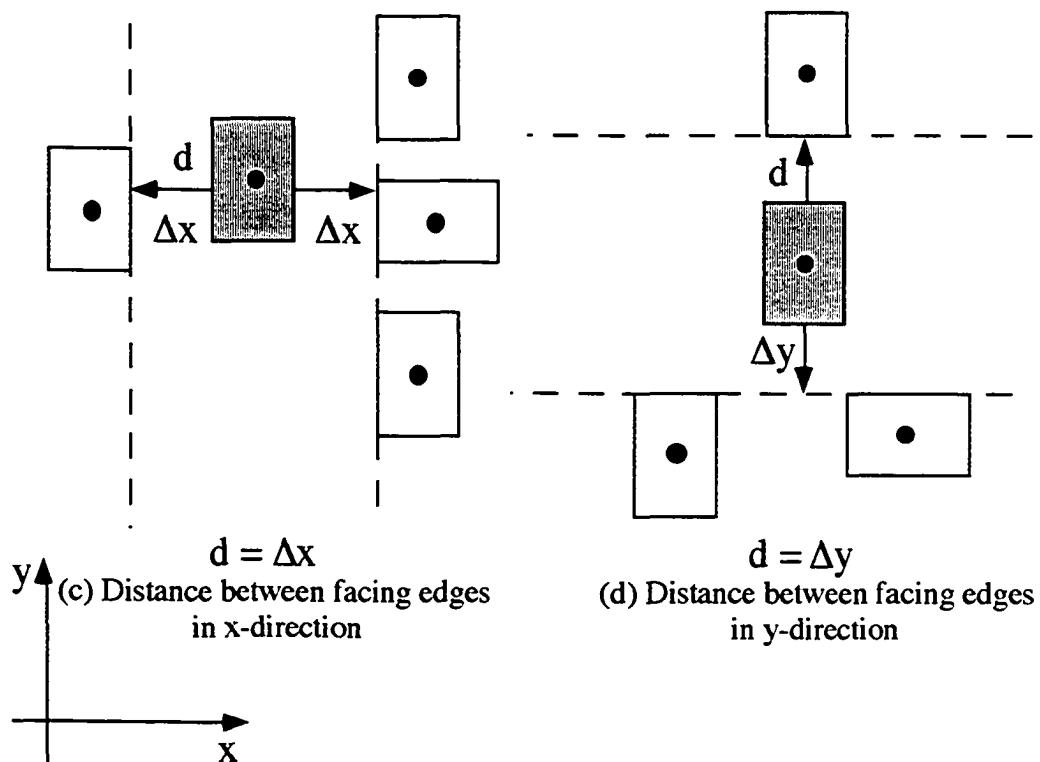


Figure 2.1  
Distance Measures

Commonly-used objective functions to evaluate static layouts are the following:

- 1) Maximize adjacency between the layout objects. This objective function is known as a ***weighted adjacency-based objective function***. It is expressed as the sum of the products of the flow between any two layout objects and a binary variable which is positive if the two layout objects are adjacent:

$$\text{Max} \sum_{i=1}^N \sum_{j=1}^N f_{ij} z_{ij}$$

Where

- i, j : layout objects
- N : total number of layout objects
- $f_{ij}$  : flow between objects i and j over a given time period, or, alternatively, values from a closeness relationship rating
- $z_{ij}$  : 1 if objects i and j are adjacent and 0 otherwise

- 2) Minimize the weighted distance between the layout objects. This objective is known as a ***weighted distance-based objective function***. It is expressed as the sum of the product of the distance between the centroids of each pair of layout objects, the flow between them, and the cost per unit flow per unit distance:

$$\text{Min} \sum_{i=1}^N \sum_{j=1}^N f_{ij} c_{ij} d_{ij}$$

Where

- i, j : layout objects
- N : total number of layout objects
- $f_{ij}$  : flow between objects i and j over a given time period, or, alternatively, values from a closeness relationship rating
- $c_{ij}$  : cost of moving one unit of flow over one unit of distance from i to j
- $d_{ij}$  : rectilinear distance between the centroids of objects i and j

- 3) Meet all constraints. In this case, the objective function is a single True or False variable. It is True if all constraints are satisfied and False otherwise. This objective is known as ***Constraint Satisfaction*** and is used in some problems for which generating a feasible solution is considered a satisfactory result.

In the weighted distance-based objective function, distance is measured using a rectilinear metric, which is chosen over a Euclidean metric to keep the objective function linear. This assumption is acceptable when the flow must follow the aisle structure of a production facility, which is often orthogonal, or when the flow follows construction roads laid out on an orthogonal grid. It is also assumed that travel occurs between the centroids of layout objects. To realistically model flow and distances traveled, some researchers (e.g., Montreuil and Ratliff 88) consider alternative (i.e., non-centroid) locations for the input/output (I/O) point for each layout object. The difficulty in these models is being able to determine positions of I/O points while concurrently developing the layout (Montreuil et al. 89, 90).

The aforementioned objectives are all single-criterion objectives. Recent developments focus on characterizing and evaluating the layout solution by multiple criteria decision making. Malakooti (87) uses an interactive approach where the program assesses the utility of the decision maker (user). The decision maker is presented with a limited number of alternative layouts to elicit paired comparisons of their strengths and weaknesses.

### 2.2.1.1 Tightly- vs. Loosely-packed Arrangements

A layout problem can be modeled as a tightly- or a loosely-packed arrangement of layout objects. This distinction is explained with examples. An example of a *tightly-packed arrangement* is the block layout of a manufacturing facility. A block layout defines the relative locations of departments (blocks) and the departmental boundaries with no aisle spaces or empty spaces between departments (Fig. 2.2 (a)). An example of a *loosely-packed arrangement* is the layout of machines within a department of a production facility where empty spaces may separate the machines to meet access or clearance requirements (Fig. 2.2 (b)). Another example of a loosely-packed arrangement is the layout of construction facilities and resources on a construction site.

The distinction between tightly- and loosely-packed arrangements is based on the level of abstraction at which the layout is conceived. At the detailed level of abstraction, the layout of a manufacturing facility is loosely packed. At the block level of abstraction, a construction site may be subdivided into areas that are assigned to each of several subcontractors so that each contractor can lay down their area independently of the others.

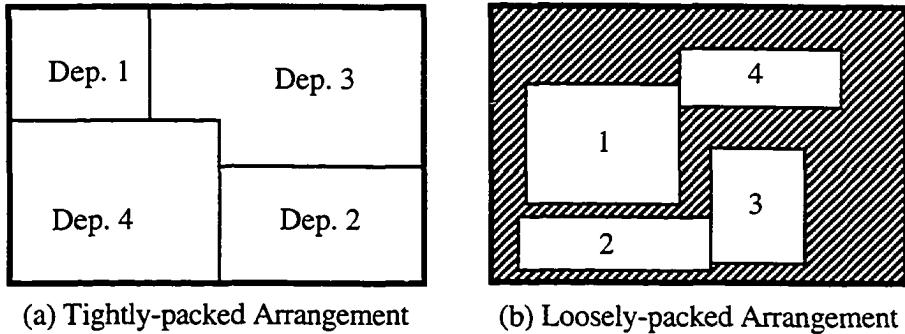


Figure 2.2  
Tightly- vs. Loosely-packed Arrangements

When solving a layout problem, one normally starts by creating an abstract layout before creating a detailed layout. At the abstract level, the layout problem is more tractable as the number of layout objects and interactions between them is smaller. The layout of each abstract object can then be addressed as a layout subproblem. This approach is effective when there are no major interactions between the layout objects of different subproblems. This is typically assumed in designing the floor layout of a production facility, where the block layout is constructed, before the layout of machines within each department is detailed. This approach is not always desirable, however, as going from a block to a detailed layout may not always result in a feasible detailed layout, even though a feasible solution exists. Furthermore, this approach is not always applicable, especially when one cannot abstract groups of objects (blocks) into functional units.

### 2.2.1.2 Modeling Assumptions for the Static Layout Problem

Layout models—whether tightly- or loosely-packed—make assumptions regarding the representation of layout objects and the representation of space.

Layout objects in a block layout typically have a non-specified shape but a fixed area requirement. As an exception, BLOCPLAN (Donaghey 86) restricts their shape to be rectangular. Some models (Montreuil et al. 90, Bozer et al. 94) attempt to control layout objects' shape by using shape factors. A **shape factor** is defined as the normalized ratio of the layout object's perimeter over its area. The normalizing factor is computed as the ratio of the object's perimeter-over-area for the ideal shape, which is often assumed to be a square (e.g., Bozer et al. 94). Shape factors do not allow one to directly and explicitly specify the shape of a layout object, since a shape other than the ideal one may have a shape factor that is close to 1.

While a flexible-shape assumption is acceptable at the block level of abstraction, it may be limiting when a detailed layout is constructed. Indeed, it may be impossible to fit or efficiently arrange equipment of known dimensions within an irregularly-shaped department. This is the case in construction, whenever the layout objects represent specific resources (such as equipment or trailers) with known dimensions and not just contractor laydown areas.

Most tightly-packed arrangement layout models use a square grid representation of space. They model each layout object as a composition of unit squares. For example, a layout object area is composed of several elements in a matrix, each of which represents a unit area and the sum of which equals the layout object area. This representation requires that the unit grid size be defined for each problem so that layout objects can be represented in multiples of it. New approaches for modeling tightly-packed arrangements call for the use of "the much richer continuous space representation" (Montreuil 90).

Many loosely-packed arrangements use a continuous 2-dimensional representation of space. The area to be laid out and the layout objects are represented either by convex polygons (Eastman 75, Hamiani 87) (a polygon is composed of a set of sides, each side is defined by two unique end points and is part of only one polygon), or simply by rectangles (Tommelein 89, Van Camp et al. 92, Montreuil and Venkatadri 91). A continuous space representation makes it easy to meet any constraint on relative locations of layout objects.

### **2.2.1.3 Optimal Methods for Solving the Static Layout Problem**

The methods used for solving the static layout problem are closely tailored to suit the spatial representation used. Those using a grid-square representation consider layout to be an assignment problem where layout objects are assigned to well-defined spaces (a space is a set of unit squares) of the space to be laid out. Those using a continuous space representation consider the layout problem to be a 2-dimensional bin-packing problem.

When all layout objects are of equal area with no restrictions on their shape, the layout problem can be formulated as a Quadratic Assignment Problem (QAP). The layout space is divided into locations to which the layout objects will get assigned. These locations should all be equal in shape and size to the layout objects. The quadratic nature of the problem arises because the cost of assigning a layout object to a location is dependent on the locations of the other layout objects. Sahni and Gonzalez (76) showed that this QAP is

NP-complete and optimal solutions can be computed only for small or greatly restricted problems.

When layout objects are of unequal area, the problem can still be formulated as a QAP by dividing every layout object into smaller sub-objects such that all layout objects can be expressed as a collection of equal-area sub-objects. The sub-objects of a layout object are then grouped by defining artificially high interactions between them. Meller (92) proves that the QAP formulation, in this case, produces suboptimal solutions and is computationally taxing because the transformation substantially increases the number of sub-objects to consider. Kusiak (87) gives more information on QAP-based formulations of the layout problem.

Problems that cannot be modeled as QAPs include those with fixed-shape layout objects. Layout problems that consist of fitting fixed-shape rectangles (with or without equal areas) in a given space, are similar to bin-packing problems (Krause et al. 87, Ramanan 89, Leung 89, Foster and Vohra 89). The bin-packing problem is to place each of a given set of predefined objects in the minimal number of bins. Objects can vary in number and size, and constrain each other (e.g., by their geometry, in that they cannot overlap, or by their weight, etc.). The bin-packing problem is also NP-complete. There exist effective heuristics to solve simple bin-packing problems (Garey and Johnson 79), but these do not apply if location constraints are added to the problem.

Montreuil et al. (90) propose a Mixed-Integer Program (MIP) formulation for optimally solving the static layout problem. The objective is to minimize weighted rectilinear distances between well-defined input and output points of two layout objects. The problem is constrained by lower and upper bounds on the length and width of each layout object to control its shape. A constraint on the perimeter of each layout object is added in an attempt to limit its area. Integer variables are used to model non-overlap constraints between layout objects, but this formulation yields a large number of integer variables. Montreuil et al. do not suggest a method for solving the resulting large MIP model and traditional MIP solvers are not sufficient to solve even small problems with six objects. Additional non-linear constraints can be added to fix the areas of layout objects, but in doing so, this formulation becomes a non-linear MIP, which is even harder to solve.

When the size of layout objects is very small compared to the available layout space or the distances separating them, they can be ignored, and the problem becomes a *location problem*. The body of literature on location problems is not reviewed here. For

completeness however, a few location problems applied to construction are mentioned. Rodriguez-Ramos (82) and Rodriguez-Ramos and Francis (83) solve for the optimal location of a crane on a construction site. Warszawski (73b) and Warszawski and Peer (73) consider the problem of locating storage facilities and construction equipment with the objective of minimizing transportation, maintenance, and installation costs. They further differentiate between three different types of transportation costs based on the material handling means: linear, stationary crane, and mobile crane (Warszawski 73a). Warszawski and Peer do not solve the generally-stated problem but consider simple cases thereof and suggest sub-optimal methods for solving them.

#### 2.2.1.4 Heuristic Methods for Solving the Static Layout Problem

Without optimal procedures to solve the static layout problem including layout objects with unequal area and fixed shape, heuristic approaches are needed.

One such approach is based on graph theory. This approach has roots in architecture and is currently researched for constructing block layouts for manufacturing facilities. In this approach (Montreuil et al. 87), the area and shape of the layout objects are initially ignored. Instead, layout objects are represented by nodes. The objective is to determine which layout objects should be adjacent, where adjacency is represented by an arc connecting the two corresponding nodes. The nodes and the arcs connecting them form a graph. Heuristics are used to transform the graph into a 2-dimensional layout (Montreuil and Ratliff 89). The main difficulty in this approach is obtaining a planar graph (i.e., the graph can be drawn in the plane so that no two arcs cross) so that the existence of a layout can be guaranteed. This approach has not been used for construction site layout.

Other heuristic approaches can be classified into construction and improvement algorithms. ***Construction algorithms*** construct a layout by starting with an empty layout space and gradually introducing layout objects to it. ***Improvement algorithms*** begin with an initial layout that includes all layout objects, alter the layout (e.g., by swapping two objects of equal size and shape), and then apply certain evaluation criteria to gauge the degree of improvement.

#### **2.2.1.4.1 Heuristic Construction Methods for Solving the Static Layout Problem**

A construction algorithm typically iterates through the following steps until all layout objects are positioned: 1) select a layout object for placement and 2) place the layout object according to a placement criterion. The selection and placement criteria vary for each algorithm.

There exist many algorithms for constructing a block layout, where the spatial representation used is a grid representation. Because of their limited application to the site layout problem, only those most widely used are reviewed here.

ALDEP (Seehof and Evans 67) randomly selects the first layout object and assigns it to the upper left corner of the layout. The next layout object selected for assignment is the one that has a relationship with the first layout object that is greater than or equal to a user-specified closeness-rating. If no departments have a relationship at least equal to the minimum acceptable closeness rating specified by the user then ALDEP randomly selects the next department. ALDEP uses a sweep method to place objects. This method fills the grid of the layout following a serpentine pattern with a specified band width. ALDEP uses an adjacency-based objective function to evaluate the constructed layout and repeats the layout construction until user satisfaction. Different solutions are possible because the first layout object is randomly selected. ALDEP allows layout objects to be fixed at a certain position but does not prevent layout objects from being split around such objects; this is not acceptable for most layout applications.

CORELAP (Lee and Moore 67), like ALDEP, determines a sequence of layout objects to be positioned in the layout according to interactions between them. It uses a different method, though, for placing objects in the layout, in that it tries to maximize adjacencies. CORELAP attempts to produce the single best layout and no randomness is introduced to produce alternatives. It uses a distance-based objective function to evaluate the constructed layout. CORELAP does not provide the flexibility of fixing layout objects at given positions and often generates layouts with irregular layout object shapes.

BLOCPLAN (Donaghey 86) uses more than one measure to evaluate the layouts (adjacency-based, flow distance-based, and closeness distance-based measures). BLOCPLAN divides the layout into three horizontal tiers with three zones per tier and two sides to a zone. Thus, the total number of layout objects that can be represented given this

spatial representation is 18. This is a major limitation of BLOCPLAN. In addition, BLOCPLAN does not allow layout object positions to be fixed.

The above layout construction programs do not guarantee to find a solution if one exists. This is so because they use an early-commitment approach in constructing the layout and do not provide any routine for backtracking to previous partial solutions in order to avoid reaching a dead end.

While these algorithms may generate satisfactory layouts for tightly-packed arrangement problems, they are not suited to solve loosely-packed arrangement problems for the following reasons:

- They assume that layout objects are of flexible shape and thus can be distorted to fit in any space that has enough area. At best, some may control the shape of layout objects but would not fix their shape.
- Some justify splitting layout objects around fixed-position objects so that the layout is tightly packed.
- They do not accommodate relative constraints on objects' positions (e.g., constraints due to clearance requirements) which may require objects to be at a specified distance from each other to be functional. At most, some algorithms constrain the locations of selected objects to be in a some portion of the site space.

In contrast to the above procedural algorithms, there are artificial intelligence (AI) based systems that use domain-specific knowledge from field experts to construct layout solutions. Those in favor of this approach believe that the knowledge of experts combined with computer power would augment human decision making and outperform computer implementation of general heuristic principles. The main limitation of such systems, however, lies in the difficulty of encoding the expert knowledge.

One such knowledge-based system used for constructing loosely-packed arrangements is Consite (Hamiani 87). It uses a "plan-generate-and-test" method to construct the layout. This method, first, plans by ordering the list of layout objects beforehand in the same way the expert would enter them into the layout. Second, it generates possible positions for the selected object. Then, it identifies applicable location constraints and tests the possible positions for feasibility. The final position of the object is selected from the set of positions that meet the constraints. Consite, unlike SightPlan (described next), keeps track of both feasible and infeasible positions for backtracking in

case dead ends are reached. No explanation was given in the literature on how final positions get selected nor on how the backtracking mechanism works.

SightPlan (Tommelein 89, Tommelein et al. 91, 92) is another knowledge-based system for constructing loosely-packed arrangements. It lays out temporary facilities on construction sites. Its knowledge is represented in 1) the definition of objects in the layout, 2) 2-dimensional constraints between the objects, and 3) the heuristics of the expert embedded in strategies that are used to decide on which layout object to position next and where to position it, or on which detailed or abstract layout to work next. SightPlan uses the constructive assembly method to construct a layout. Its expert strategy is to position objects one at a time without backtracking and for that it is an early commitment strategy. Alternatively, SightPlan can apply a least commitment or a postponed commitment strategy. These strategies favor alternative layouts to be maintained implicitly before a commitment to individual resource positions is made. All SightPlan strategies are implemented using a bounded-interval representation to keep track of all possible positions of each layout object during problem solving and the GS2D (Confrey and Daube 88) Constraint Satisfaction engine. The constraint engine receives the initial sets of two objects' possible positions and a constraint on those objects. It returns the sets of positions that satisfy the constraint. "This capability of maintaining all least-committed positions makes systems like SightPlan, PROTEAN (Hayes-Roth et al. 86), and LOOS (Flemming et al. 88) stand out over all other space planning programs" (Tommelein et al. 91). SightPlan's strategies, however, lack an evaluation function for pruning partial and alternative layout solutions. This is a considerable shortcoming, especially when the number of alternative solutions that can be generated by a least- or post-commitment approach is large. Any feasible solution (i.e., a solution that satisfies the constraints) is considered satisfactory. Like the aforementioned heuristic construction algorithms, SightPlan lacks mechanisms to backtrack to previous partial solutions when it cannot position the current layout object and thus fails to find a solution to the layout problem where one may exist.

AI approaches bring flexibility to modeling the layout objects and the layout process. First, information regarding the size and shape of individual layout objects is specified in a declarative way (a layout object can be defined as a circle or rectangle of a given size, etc.) whereas in most procedural algorithms shape can be controlled to some extent only. Second, interactions between objects are articulated and specified individually, as constraints (e.g., the program user can introduce any type of constraints like distance or zoning constraints). These constraints can be used to limit the feasible set of the resources' positions. Third, domain or problem-specific knowledge can be included in the problem-

solving method (e.g., to guide the resource selection process, to determine which constraints to choose, and when to apply them to reduce the solution space).

#### **2.2.1.4.2 Heuristic Improvement Methods for Solving the Static Layout Problem**

An improvement procedure starts with an initial layout and iterates through the following steps until no improvement is recorded: 1) generate an alternate layout by modifying the current one, 2) evaluate the alternate layout according to an evaluation criterion, 3) if the alternate layout scores better than the old one, make it the new incumbent and repeat the procedure. This approach of gradually improving an existing layout is called "hill-climbing" and the iterations normally end at a local optimum. Which optimum is reached depends on the initial layout and how alternates are generated. A possible way to avoid local optima is to allow for an occasional (e.g., randomized) increase for a minimization (decrease for a maximization) problem in the value of the objective function as is done in simulated annealing algorithms. It is only recently that simulated annealing was applied on layout problems (Meller 92, Yeh 95).

This dissertation uses a construction method for generating site layouts. For completeness, however, a few of the heuristic improvement methods are reviewed here. For a complete review the reader is referred to Kusiak and Heragu (87) and European (92).

CRAFT (Armour and Buffa 63, Buffa et al. 64) is probably the best known implementation of an improvement procedure. CRAFT uses a weighted distance-based objective function, where distances are rectilinear distances between centroids of layout objects. CRAFT begins by determining the cost of a user-provided layout. It then evaluates all possible two-way or three-way location exchanges of facilities which are either adjacent or equal in area. These exchanges yield densely-packed layouts in which all objects are bound to fit because CRAFT does not control the shape of layout objects. It selects the location exchange which results in the greatest cost reduction. CRAFT iterates until there is no location exchange which results in a lesser cost than that of the current layout. It thus uses hill-climbing to reach a local optimum because CRAFT exchanges only two or three facilities that are equal in area or adjacent.

Many routines have been developed to overcome some of CRAFT's weaknesses. For instance, MULTIPLE (Bozer et al. 94, Meller 92) is tailored to improve the layouts of multi-floor facilities but it applies to single floors as well. Similar to CRAFT, its objective function is distance-based and the improvement is done through layout object exchanges.

MULTIPLE, however, does not restrict exchanges to equal area or adjacent objects; all layout objects are considered for possible exchange. This is made possible through the use of spacefilling curves to represent the layouts (similar in concept to the sweep band used in ALDEP). A spacefilling curve is a continuous function that visits all the grid squares on a floor exactly once. Each layout is generated and stored as a unique sequence of layout objects and using exactly one space filling curve throughout the entire problem-solving process. Exchanging two layout objects is then made simple by changing the sequence of layout objects. MULTIPLE allows some control over layout object shapes during the exchange routine.

The CRAFT improvement method has been applied to construction site layout by Rodriguez-Ramos (82) and Yeh (95). As in CRAFT, Rodriguez-Ramos starts with a preliminary layout and uses a rectilinear distance-based objective function to evaluate layout alternatives. His method is similar to CRAFT but with one modification. Before exchanging layout objects pair-wise, he identifies the dominating layout object in the layout, which is the object that has the highest cost between it and all other layout objects. He uses an optimal single facility location algorithm to find the best location for that dominating layout object in the layout, and swaps locations between the dominating object and the object at that best location.

Yeh (95) formulates the construction site layout problem as a discrete allocation problem where a set of predetermined facilities should be assigned to a set of predetermined sites. He uses an annealed neural network approach that merges features from simulated annealing models and a Hopfield neural network to improve an initial layout. An initial layout is generated by randomly assigning facilities to sites. This is possible because facilities and sites are of equal area and shape. Improvement is done by having layout objects exchange sites. Yeh's formulation of the layout problem is limiting as it requires sites to be predefined, and objects and sites to have an equal area and shape.

### 2.2.2 Layout Planning Models that Account for Time

Static layout algorithms must be extended in order to be able to solve the dynamic layout problem. Extensions are required to account for variable object dimensions, object relocation, timing of space needs, and space reusability.

Few layout models account for time when constructing the layout. Rosenblatt (86) introduced the *dynamic facility layout problem* (DFLP) in the area of production

facilities. He formulated his model based on the quadratic assignment problem (QAP) in which the objective is to minimize the total cost, including both material handling and rearrangement costs. Three assumptions in his model limit its applicability to the construction site layout problem: 1) rearrangement costs are independent of the distance separating the positions of layout objects in two subsequent layouts, 2) all layout objects are of equal area and shape, and 3) a layout object's area does not change over time. He used Dynamic Programming to solve a problem of six departments and five periods. The major shortcoming of this approach is that it requires solving successive QAPs within a Dynamic Program (DP). Given that DP relies on pure enumeration and QAPs are difficult to solve, this approach is not particularly suitable for complex problems of the kind addressed in this dissertation. The efficiency of this problem-solving method relies heavily on the availability of tight bounds, as all the possible states (alternative layouts) may not need to be considered for all stages (time periods).

Current research in DFLP focuses on developing lower bounds for the dynamic layout problem (Urban 92) by taking the sum of the workflow costs from the optimal static problem of each period. Even when such bounds are available, this method still requires solving many QAPs. Kouvelis and Kiran (91) used a similar DP approach for solving the DFLP but chose a stochastic single-period (static) layout model for representing changes in interactions and areas of layout objects over time.

Montreuil and Venkatadri (91) used Linear Programming to compute intermediate, optimal positions for the staged growth of facilities on a manufacturing floor. Their method equally applies to a staged decline of facilities. This model relies on user entry of initial and final layouts, and solves for resource dimensions and positions within the spatial boundaries assigned to each facility. Thus, it does not allow space to be reused nor facilities to be relocated over time. Furthermore, this model cannot be used in the context of construction, where growth and decline of facilities (resources) do not all occur in parallel but are activity dependent.

The aforementioned DFLP models use non-overlap constraints between facilities and possibly bounds on the resources' shape and area. However, they do not incorporate other considerations that are necessary for modeling the construction site layout problem, such as allowing layout objects to have fixed shapes and be subject to location constraints.

Some AI-based approaches do account for such considerations but fail to address other aspects of dynamic layout planning. Tommelein's (Tommelein 89, Tommelein et al.

91) Expert-Time Strategy for SightPlan creates a sequence of layouts for the duration of construction of a project, given layout objects, the period for which they are on site, and location constraints between them. SightPlan assumes that consecutive layouts are independent and that layout objects may thus be relocated at no cost from one layout to the next.

Choi (94) reviewed the layout knowledge in SightPlan and LOOS (Flemming 88). He used LOOS' representation to model a layout problem that changes over time, similar to the one that had been addressed in SightPlan.

Cheng (92) used a knowledge-based approach to model the spatial requirements of temporary facilities (TFs) on a construction site with some consideration to the timing of their presence on site. He employed a geographical information system (GIS) linked to a relational database and developed an expert system named Arcsite. In the process of generating potential sites for positioning a heuristically-selected TF, Arcsite checks if the previously-positioned TFs have a departure date earlier than the arrival date of the selected TF to be positioned. If so, then its space can be reused as a potential site for the new TF. The layout generated by Arcsite is a single arrangement where facilities overlap, showing that different facilities will occupy the same space at different times. Relocation of resources is not modeled in Arcsite.

The aforementioned models reflect changes in the layout at discrete points in time and they therefore plan the layout for different stages. In construction, some resources continuously change positions on site. Examples are construction materials flowing between supply and demand centers on site and equipment involved in activities, like hauling, excavating, compacting, backfilling, pipelaying, etc. Some models (e.g., Beliveau et al. 93, Kunigahalli and Russell 94) address the motion of equipment on site and are concerned with planning paths and trajectories of mobile resources on site. The Path-Finder (Morad et al. 92) model's objective is to find a collision-free path for moving an object between two points subject to motion constraints on the handling equipment used to move the object. Path-Finder can be used to predict areas of interference prior to construction start.

### **2.2.3 Layout Implementation, Control, and Re-planning**

A good site layout plan over time may reduce the effort spent during project execution on control and re-planning. Research in the area of site layout implementation, control and re-

planning focuses mainly on two issues: 1) improving material delivery and 2) tracking where resources are on site so that deviations from the plan can be measured and the layout plan can be updated.

Improving material delivery can reduce costly labor delays, idle equipment time in the field, and delays in the overall construction schedule (Bell and Stukhart 86). Bell and Stukhart (87) claim that vendors were meeting promised delivery times and avoided delays when contractors worked closely with them in exploring alternative financing, improved transportation methods, etc., and—not surprisingly—when a personal commitment was obtained from the vendor's company president with respect to timely delivery of quality materials.

Bjornsson (86) developed a mathematical model to address how much, from whom, and when to procure material. Bernold and Salim (93) acknowledged the need for taking the sequencing of construction work into consideration when deciding on material deliveries, including how material gets packaged, bundled, or grouped. Their CAD-based model addresses these issues for fabricated reinforced rebar.

To identify and track resources (in particular materials and small tools on site), research by Bell and McCullouch (88) and Rasdorf and Herbert (90a, 90b) investigated the use of bar coding in construction. More and more construction materials are being bar-coded today. Lundberg and Beliveau (89) combined bar-code technology and a real-time positioning system that accurately locates objects in an open environment, with CAD technology for graphically representing layout object positions on a construction site.

The need to monitor deviations from a site layout plan and updating the plan was pointed out by Tommelein and Zouein (93b). On-going research by Tommelein (94a, 94b) focuses on integrating MovePlan (presented in Chapter 3) with a commercial surveying tool, named CAPSY™, for controlling laydown and materials handling on site. CAPSY™ uses bar-coded beacons and laser technology to measure and record the X and Y coordinates of resources on site. These positions can then be downloaded to MovePlan so that actual positions can be compared to planned ones and corrective actions can be taken if needs be.

## 2.3 Resource Scheduling Models

Space scheduling, as defined earlier, involves adapting the schedule to comply with site space limitations. Hence, space scheduling is a limited resource scheduling problem where space is the limited resource. This section therefore reviews the capabilities and limitations of traditional resource scheduling models.

### 2.3.1 Unlimited Resource Scheduling

*Resource scheduling* includes generating the network of construction activities, selecting methods and resources for performing the activities, and computing start and finish dates of activities. Some aspects of the constructed schedule can subsequently be improved, for example to minimize total project cost or to meet a targeted project completion date. *Unlimited resource scheduling* is based on the assumption that if a resource is needed by a construction activity, then that resource can be provided.

#### 2.3.1.1 Network Generation

Network generation involves:

- breaking down the project into well-defined activities
- identifying precedence relationships among activities and linking activities together accordingly to form the network.

Generating a network of activities is a complex heuristic process that draws on the planner's personal experience and knowledge of construction methods, limits on resource availability, and technical considerations, etc.

In the late '80s, construction project planners using AI programming techniques automated the network generation process. Some (e.g., GHOST (Navinchandra and Logcher 88), SIPEC (Kartam and Levitt 90), CONSTRUCTION PLANEX (Hendrickson et al. 87)) take as input the design elements of the project and generate the activity network by defining an activity for each element and setting up precedence among activities. Others (e.g., CasePlan (Dzeng and Tommelein 94)) reuse parts of previously developed networks by drawing similarities between the current project and cases that describe previous projects and their schedules.

### **2.3.1.2 Construction Method and Resource Selection**

The duration of an activity depends on the type and quantity of work involved in it and on the selected construction method. The selection of a construction method usually reveals several possibilities for resource use. In deciding on the duration of an activity, the experienced scheduler draws on knowledge of:

- labor, equipment, and materials that will be used or consumed by each activity in the project
- construction methods coupled to the above resources
- time and money
- construction site conditions including site space availability
- contractual arrangements including human organizational structures, procurement policies, etc.

Very few systems exist to automate the method selection process, though some research has characterized activity durations in terms of resource availability, construction method selection, and space restrictions on construction sites. CONSTRUCTION PLANEX is one of very few systems that considered resource selection when generating the network, but it did not consider site space or work area availability in this process. Systems that consider spatial data along with method and resource selection have been or are currently being developed and are the subject of section 2.4.

### **2.3.1.3 Activity Scheduling**

Activity scheduling involves:

- computing start and finish dates of activities
- computing activities' floats
- determining the project duration.

The most commonly used scheduling algorithms in the construction industry are those in CPM and PERT (referred to as forward and backward pass algorithms (Harris 78)). These algorithms require that the network of activities be defined and durations be assigned as an input to the algorithms. Durations are deterministic in CPM and probabilistic in PERT. Given this input, they produce early and late activity starts and finishes, float values, etc., and they identify the critical (longest) path that defines the project duration. There are many network-based scheduling systems (e.g., Primavera

(92)) that implement forward and backward pass algorithms and that provide the user with the display of activities and their precedence in different ways.

Forward and backward pass algorithms used in CPM and PERT assume unlimited resource availability. This assumption is not appropriate to come up with realistic schedules for the majority of projects. To better reflect reality, the schedule should be adjusted to account for limited resource capacity constraints. Adjustments may affect start and finish dates of activities, the choice of construction method, or the durations of activities.

#### **2.3.1.4 Schedule Improvement: Time-Cost Tradeoff Algorithms**

Initial schedules generated under the unlimited resource assumption can be improved to meet time or budget constraints. Time-cost tradeoff algorithms are schedule improvement algorithms that vary durations of activities to meet a target project duration while minimizing the project's total direct cost. A basic input to any time-cost tradeoff algorithm is a relationship between the duration of an activity and the cost of performing it. From the time-cost relationship of each activity, normal and crash points are determined. The normal point corresponds to minimum cost and normal duration. The activity duration can be increased beyond the normal point but this will increase its cost. The crash point corresponds to minimum duration and crash cost. The activity cost can be increased beyond the crash point but this will not shorten the activity duration.

Optimal time-cost tradeoff algorithms are based on Linear Program, network flow, or Integer Program formulations. A Linear Program formulation is possible if the time-cost relationship of the activity can be modeled as a convex piecewise-linear continuous function. The objective is to minimize the project's direct costs subject to the following constraints: 1) the duration of each activity is bound by its normal and crash duration, 2) the difference between the start and finish dates of each activity should be equal to the duration of that activity, and 3) the finish date of the last activity(ies) in the network should be less than or equal to a desired project duration. The desired project duration could be dictated, for example, by the construction contract, or by other obligations of the contractor, or by the weather.

Heuristic algorithms with a similar objective produce solutions close to optimal. One such algorithm is Fondhal's (Harris 78) which uses myopic strategies to vary the project duration and cost. One strategy starts with an initial schedule that includes all

activities at their normal duration and minimum cost. It shortens the project duration at a minimum increase in project cost by shortening the duration of critical activities with the smallest cost slope. The inverse strategy starts with an initial schedule that includes all activities at their shortest duration and crash cost. It lengthens the project duration at a maximum decrease in project cost by lengthening first non-critical then critical activities with maximum cost slope. The inverse strategy is useful if project costs should be minimized at the least possible increase in project duration.

#### **2.3.1.5 Schedule Improvement: Resource Leveling Algorithms**

Resource leveling algorithms attempt to reduce peak requirements and smooth out period-to-period fluctuations in resource assignment within a constraint on project duration. Typical resources considered are equipment, money, skilled labor, etc.

Optimal solutions for the resource leveling problem are based on Mixed Integer Program formulations (Easa 89, Shah et al. 93, Seibert and Evans 91). Such formulations are NP-complete and optimal solutions are reached for small sized construction projects only. Heuristic algorithms are therefore needed.

A well-known heuristic algorithm is the Minimum Moment Algorithm (Harris 78). This algorithm builds on the assumption that the durations of the activities in the network are set and will not be altered in the course of applying the algorithm (only their start date will vary). The objective in this algorithm is to minimize daily fluctuations in resource use subject to a fixed project duration. As a proxy to this objective, the algorithm minimizes the moment of the resource histogram around the line that shows zero resources. To accomplish this objective, the algorithm starts from an early-start schedule and shifts non-critical activities forward in time by an amount less than or equal to their free float, starting at the last sequence step and proceeding backward. At each iteration, the shift(s) that yields the maximum reduction in the histogram moment is selected.

#### **2.3.2 Limited Resource Allocation**

As opposed to schedule improvement algorithms, limited resource allocation algorithms are schedule construction algorithms, i.e., they do not have a solution that satisfies resource limits until the very end. Limited resource allocation algorithms check for resource availability before scheduling an activity to start at any given time. Their objective is to find

the shortest-duration schedule consistent with the specified resource limits. Typical resources modeled are skilled labor or specialized equipment.

There exist optimization methods as well as heuristic methods for solving the limited resource allocation problem that go back in time to the late '60s and early '70s (e.g., Wiest 64, Davies 73, and Paulson 73). Scheduling projects with limited resources is a problem that mathematicians refer to as a "large combinatorial problem". Various approaches have been formulated to solve the problem optimally, including Integer Programming (e.g., Wiest and Levy (77)), branch-and-bound (Gavish and Pirkul 91), and Dynamic Programming, but none of these is computationally tractable for any real-life problem size (i.e., schedules including hundreds if not thousands of activities).

Wiest (64) proposed a "basic" heuristic for approaching the resource allocation problem. This heuristic works under the assumptions that the network of activities is given, activity durations are set and remain unchanged, and activities cannot be interrupted. The heuristic iterates through the following steps:

- start on the first day and schedule all activities possible while satisfying the resource daily limit. Do the same for the second day, and so on.
- when several activities compete for the same resource, give preference to the one with the least float.
- reschedule noncritical activities, if possible, to free resources for scheduling critical activities.

Many resource allocation programs built on this basic heuristic. SPAR-1 (Wiest and Levy 77) relaxes the assumption on activity durations by introducing the concept of crew sizes. Each activity can be performed with one of three crew sizes: a maximum, a normal, and a minimum crew size respectively. These crew sizes set the duration of the activity to a crash, a normal, or a maximum duration respectively. Critical activities are assigned a maximum crew-size if sufficient resources are available so as to keep the project duration to a minimum. Otherwise, a normal, or if need be a minimum crew size is used. If insufficient resources are available to schedule critical activities, the program tries to borrow the resources from non-critical activities. If all attempts fail to provide the necessary resources to schedule the activity at its early start date, the activity start is delayed to the subsequent period. SPAR-1's approach, though simple, is powerful because it combines the two strategies used in schedule improvement algorithms which are: 1) to delay activities and 2) to vary resource assignments. However, SPAR-1 unrealistically assumes that crew sizes are limited to a single resource type (e.g., masons).

## 2.4 Space Scheduling Models

The resource scheduling models of section 2.3 do not consider space as an explicit variable. Consequently, they do not reason about space as a limiting resource in the generation of the plan or schedule. Unlike other resources such as equipment and labor, which can be modeled as scalar variables, space is not a scalar variable. Checking for space availability at any time requires checking for fit, location, and accessibility, in addition to checking for total area or volume availability. For example, Fig. 2.3 illustrates the case of an object (shown left as a dotted rectangle) that cannot fit within the site boundaries without overlapping with other objects, although there is enough total open space available to match its area requirement. It also illustrates the case of a laydown area that can be positioned without overlapping with existing objects, but that is too far away from the access gate to be acceptable. Checking for space availability therefore requires solving a layout problem and determining the solution's feasibility.

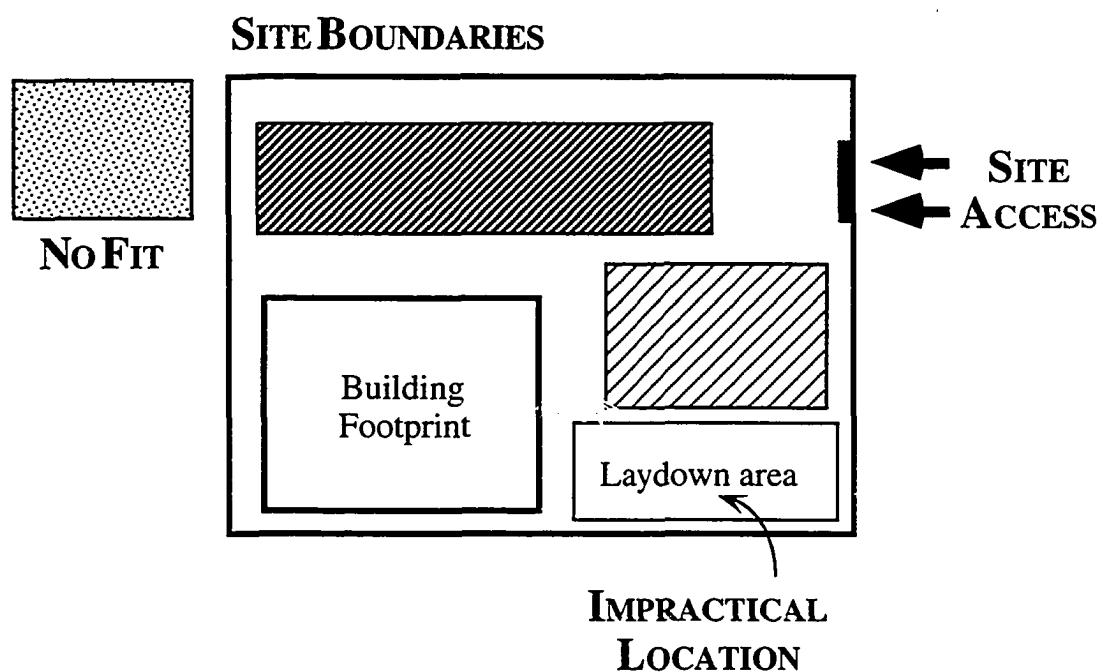


Figure 2.3  
Problems with Limited Space

Few models for space scheduling exist, but research is under way to tie schedule and spatial data together during planning. Some researchers focus on the work space required by construction activities in closed spaces or enclosures (e.g., inside a building). Smith (87) defined the components of an activity's work space as equal to the physical

space of the resource plus two additional spaces: one that surrounds the physical space and that is needed for maneuvering, and another one that surrounds this space and that can be shared with other activities/resources and/or be used momentarily by mobile resources that access the resource. He also rationalizes the variation in the space needs of an activity over its duration and proposes three profile types to model these variations. Type I indicates a constant space need over the activity duration (e.g., construction activities in open air or a machine controlled operation in a factory). Type II indicates a growing space need over the activity duration which can be represented by a hyperbolic, parabolic, exponential, or linear function (e.g., build-up of operations such as the growth of a wall in a room which reduces a portion of the available open space in an enclosure). Type III indicates a decreasing space need as the activity proceeds to completion which can be represented by an inverse hyperbolic, parabolic, or linear function (e.g., manual excavation of a basement, the dismantling of scaffolding). Smith did not provide cases where the different functions are needed to model growth or decline of space needs. Smith's model also did not include a systematic approach for constructing space-based schedules while considering time-space tradeoffs.

Riley (94) described why construction activities need space, and studied the relationship between the required space and the sequence planning process of work areas. Riley also developed a general check-list that can be followed by the layout planner to identify congested work areas and resolve spatial conflicts.

An early model that acknowledged the need for assessing site space availability while scheduling resources is by Mawdesley et al. (88) and by Sirajuddin and Mawdesley (89). This model identifies project activities and their sequencing from blueprints and design drawings. Activities are scheduled opportunistically, i.e., whenever their precedence and resource availability permit. Resources are manually selected and assigned to activities. The space needed by each resource on site is checked for availability before it is scheduled. It is unclear from these two references if checking for space availability involves positioning the resource on the site. The model does not characterize space needs of resources over time or interactions among resources needed to find suitable positions for them. It is also left unspecified how the model handles cases of limited space, i.e., what kind of modifications might be made to the schedule when not enough space is available to accommodate a resource. According to Mawdesley, a considerable amount of work is still needed before his model "would take into account temporary site layout".

KNOW-PLAN (Morad 90, Morad and Beliveau 91) is a knowledge-based planning system that integrates AI with CAD technology to visually simulate the process of constructing a facility on the screen of a dedicated graphics workstation. The sequencing of activities is determined based on information extracted from knowledge bases that capture construction planners' expertise, and spatial relationships among various components of the project that are extracted directly from the 3-dimensional CAD drawings of the facility. Although KNOW-PLAN incorporates spatial data regarding the facility to be built and reasons about this data in the generation of the activity network, the system does not consider the layout of resources involved in the construction process over time. Rather, it focuses on the spatial configuration of the facility in its final state and how this may limit the sequencing of the construction work.

Thabet and Beliveau (93) use CAD drawings of the facility to be constructed to quantify space availability in a process work area. They 1) identify the structure of the work block (i.e., a volume inside the facility where work is to be performed) by requesting manual user input, and 2) quantify the physical space of each work block. This second step is automated using a 3-dimensional processor that evaluates the net available space in each block by deducting the space consumed by elements located inside the boundary of the block from the gross volume of the block. The outcome is an ASCII file describing the work block and the available space. This data can be subsequently used by a scheduling algorithm to determine if the activity can be scheduled based on its spatial requirement and the available work block space.

Thabet and Beliveau (94a, 94b, 94c) incorporate this technique in SCaRC (Space-Constrained and Resource-Constrained), a prototype knowledge-based scheduling system that acknowledges work space as a scheduling constraint. The system classifies the activities to be scheduled to determine the amount of work space they require for their execution on each floor. The system then compares available space (previously determined from partitioning work space into work blocks) to the space required by the activity. Like other limited resource scheduling systems, SCaRC uses a forward pass to delay activities, reduce the production rate of the crew performing it, or interrupt activities (i.e., discontinue their performance) when not enough space is available to accommodate them. The main application of SCaRC is multi-story construction. SCaRC limits the layout space to the area/volume available on each floor and does not address the temporary storage of resources on site, before they are brought to their final position.

## 2.5 Modeling Needs

This chapter reviewed existing models in the three general problem areas of layout planning, resource scheduling, and space scheduling. This section outlines their applicability and limitations to modeling the dynamic site layout and space scheduling problems defined in Chapter 1, and identifies the modeling needs for this research.

Among the reviewed layout models, those that can be used to model the construction site layout problem do not take into account the following variables:

- material flow and how it changes over time
- relocation of resources
- variations in space needs of resources over time.

However, as discussed in Chapter 1, these issues are important and should be considered in order to realistically model the construction site layout problem. Hence, research in this area needs a comprehensive dynamic layout model that explicitly represents the above variables and a layout evaluation function that is comprised of costs attributed to material flow and the relocation of resources.

Existing limited resource scheduling and schedule improvement models construct or improve a schedule to comply with limited resources provided that these resources can be represented using scalar variables, i.e., provided that they are additive. Hence, these models cannot consider space as a limited resource as space is not scalar. To be able to consider space, schedule improvement models need to characterize the spatial requirements of resources and activities over time so that time-space tradeoffs are possible.

The strategies (such as delay activity, change resources, etc.) used in schedule improvement models to vary the total number of resources required at any time can be successfully used to vary the total space required at any time. Therefore, this research uses an improvement method and strategies similar to those used in schedule improvement models to adjust an initial schedule to comply with limited space. These strategies, however, need to be augmented with methods that assess space availability at any given time so that time-space tradeoffs can be evaluated. Determining space availability at any given time necessitates creating a layout of those resources at that time.

The space scheduling models reviewed in section 2.4 assess space availability and needs while scheduling activities. Their focus, however, is on planning the space needs of activities inside the building (e.g., to store, fabricate, and assemble materials or building

components and perform work). The dynamics of space needs inside the building are different from those of staging areas and long- and medium-term laydown space used for temporary storage of materials and stationary equipment on site which are the focus of this research. The differences between the two problem domains impose different modeling needs as highlighted below:

- Resources occupy space inside the building over short time periods as they get consumed and installed in the final structure. The fast turnover in space use inside the building may warrant a continuous monitoring of changes in space needs over time. In contrast, resources occupy laydown space on site over relatively longer time periods while awaiting relocation to staging areas or use by production activities. Hence, changes in space availability and space needs of resources on site over time need only be checked at selected points in time, possibly sequenced to parallel construction progress.
- The building space is a confined space that is limited by the volume of the building components and a story's height. This requires a 3-dimensional space representation to determine the volume of available work space and to check for spatial interference. In contrast, space on a construction site is an open space with practically no vertical interference (except for electrical poles and wires). Hence, a 2-dimensional space representation is enough to model space needs of resources and horizontal interference between them in the form of geometric constraints between their relative positions.
- Distances traveled by resources inside the building are small if compared to distances traveled by resources in between staging or work areas on a construction site. This may explain why costs related to travel distances (such as transportation and relocation costs) were ignored by the existing space scheduling models. These costs can be substantial for a large site and therefore are considered by this research in modeling the site layout problem.

These needs are addressed in several models developed as part of this doctoral research and described in Chapters 3, 4, and 5.

# **Chapter 3**

## **COMPUTATIONAL MODELS FOR INTERACTIVE DYNAMIC LAYOUT PLANNING AND SPACE SCHEDULING**

This chapter presents two computational models for interactive dynamic layout planning and space scheduling: MovePlan and MovePlan-ConRes. The MovePlan model takes as input an activity schedule augmented with spatial data, namely dimensions of resources required to perform the activities, and provides its user with graphical tools to assist them in constructing layouts that change over time. The MovePlan-ConRes model is the loose integration of MovePlan with a reactive scheduler that makes changes to activities in a schedule to remedy spatial conflicts as detected by the MovePlan user during layout construction.

This chapter is organized as follows: section 3.1 presents MovePlan and section 3.2 presents MovePlan-ConRes. Section 3.3 concludes with the research directions set forth in the development of MoveSchedule, which is the subject of the remainder of this dissertation. Note that MoveSchedule was developed independently of MovePlan and MovePlan-ConRes and is subject to different modeling assumptions.

## 3.1 MovePlan

The objectives set forth for the MovePlan system (Tommelein and Zouein 92, 93a, Zouein and Tommelein 92, Tommelein 94c) are to take into account spatial data related to resources used by activities in a schedule during early project planning, and to facilitate the construction of layouts that change over time. The system had to run on a microcomputer, so that it could be made widely available on sites, and had to be user friendly so that no special training would be needed to use it.

MovePlan is an interactive computer program that helps its users address the dynamic layout planning task. To this end, MovePlan takes as input a sequence of activities (including durations and precedences), resources associated with activities and their dimensions, and lets the user choose time frames over which to create layouts. For each such time frame, MovePlan provides 2-dimensional templates of the resources present in it. Resources with known positions are automatically displayed at their positions by MovePlan. Those with no position (yet) must be positioned by the user. The advantage that MovePlan has to offer, over manually creating these layouts that change over time, is that it tracks resource positions over time and maintains consistency between layouts. These support mechanisms are detailed in section 3.1.3.

### 3.1.1 Modeling Assumptions

A discrete representation is used to model time. MovePlan users can slice the project duration into time intervals of any length and create a layout to cover each time interval. By convention, a layout can depict each resource in one and only one position, i.e., it is assumed that the resource is stationary for the duration of a single layout. The length of each time interval can be made small enough to simulate a continuous change in resource positions.

All independent resources have user-defined fixed positions (referred to here as *fixed resources*). Thus, resources that move about the site (i.e., that do not have a user-defined fixed position on site) must be associated with at least one activity so that MovePlan will provide them as templates to be dragged around and positioned by the user during layout construction. Users may have to introduce one or several dummy activities to obtain the needed functionality.

An activity can use zero, one, or multiple resources. A resource can belong to one or multiple activities. There is no limit to the number of concurrent activities that can use a single resource.

Resources, including the project site, the fixed resources, and the dependent resources, are modeled as 2-dimensional templates with one of three shapes: rectangles, rectangles with rounded corners, or ovals.

All resources have predefined and fixed dimensions, i.e., the length and width of each resource are specified by the user before any layout is constructed and cannot be changed after the first layout has been saved.

The dimensions of a dependent resource do not vary over the activity duration. However, this variation can be modeled by dividing the activity into a sequence of shorter activities and the resource into a discrete set of different resources, each of known dimensions and associated with a different activity. Similarly, the dimensions of a fixed resource do not vary over the duration of its existence on site. The change in the resource dimensions, in this case, can be modeled by a discrete set of different resources where each has known dimensions and exists on site for a unique and different time period.

Dependent resources are on site from the early start to the early finish of each activity they are associated with. If a resource remains on site in-between two activities, a dummy activity must be introduced to represent continuity of presence. Similarly, if the resource exists on site before the activity starts and after it finishes, then additional activities can be added to immediately precede and succeed the activity using that resource.

### **3.1.2 Input**

MovePlan has a menu-driven interface to facilitate data entry. Alternatively, MovePlan can read input data from an external ASCII file. Input to MovePlan includes:

- site dimensions
- description, dimensions, shape, color (resources can be color-coded), and position of each fixed resource and the time period over which it exists on site
- description and duration of each activity
- precedence relationships between activities
- description, dimensions, shape, and color of each resource associated with an activity.

### 3.1.3 Computations

Activities are entered one at a time by the user. Upon each new entry, MovePlan updates the schedule: it computes early (and late) start and finish times, and total float values for each activity defined thus far. These computations can be delayed until all activities have been entered into the system. Resources are also entered one at a time and linked to one or several activities by the user. MovePlan schedules all activities at their earliest start. However, MovePlan can read an external data file where start dates of activities have been set which can be different from the earliest start. External data is read at face value, without verification.

MovePlan assists the user in creating a layout for any user-selected time frame (e.g., a time frame with high space demand, the first two startup months of a project, etc.) by deriving from the activity schedule the fixed and dependent resources that are on site for that time frame. For that time frame, MovePlan displays the available site space and fixed resources in a view with scaled site dimensions (see the right-side of Fig. 3.1). The fixed resources are represented to scale at their known position on site (e.g., resources 7, 11, and 12 in Fig. 3.1). The dependent resources are represented to scale as templates in a scrolling view to the left of the site space (see resource 4 at the left-side of Fig. 3.1).

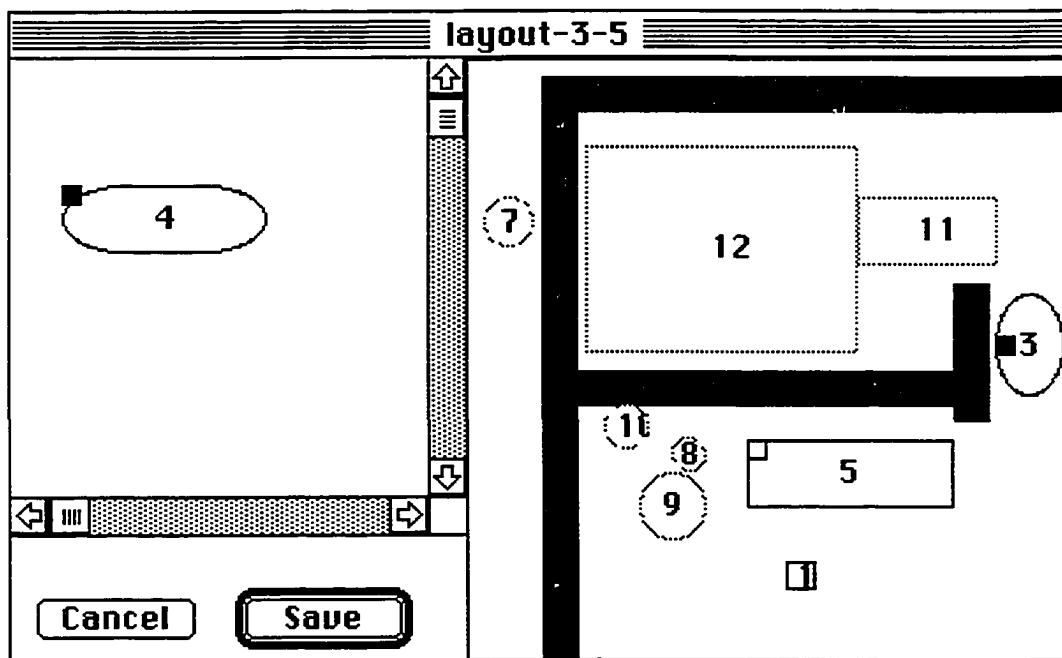


Figure 3.1  
Screen Dump of MovePlan's Layout Display  
(Black areas are fixed resources color-coded black.)

MovePlan uses the following conventions to represent resources. All resources are represented as scaled 2-dimensional templates. A dependent resource is drawn with solid black lines, while a fixed resource is drawn with gray lines. A resource whose position can be changed by the user is tagged with a box. A hollow box indicates that the resource has a position in a previously constructed layout but can be repositioned (e.g., resource 5). A black box indicates that the resource has no other position yet and needs to be positioned (e.g., resources 3 and 4). A resource whose position cannot be changed in the current layout is drawn with no box in its upper left corner (e.g., resource 1).

It is the user's responsibility to position the resources on the site space and thus to create acceptable layouts. The user can position the left-side resources by dragging them into the area representing the site space. Resources can be rotated by 90°, clockwise and counterclockwise, to change their orientation. The user can also relocate these resources until a user-satisfying position is found. Resources can be "unpositioned" (i.e., removed from the layout) by dragging them back into the left template, provided that the current layout has not yet been saved. Resources are "mousable" to display their name, position, dimensions, and the activity they are associated with.

MovePlan helps its users develop dynamic layouts that are forcibly consistent (i.e., a resource can have only one position at any one time) by tracking resource positions over time. Upon the creation of a new layout, MovePlan checks resource positions in previously constructed layouts to maintain consistency among all layouts. It does so by checking if the resource has a position in a time-overlapping layout (i.e., a layout that spans a time frame that overlaps partially or fully with the current time frame). If this is the case, then MovePlan automatically displays the resource at that position on site. The user can change the resource position if the time frame of the previous layout is entirely covered by the time frame of the current one. Each such resource is tagged with a black box. Otherwise, i.e., if the time frame of the previous layout extends beyond that of the current layout, MovePlan locks the resource at that position and shows no box thus preventing the user from changing this position in the current layout. For example, if the current layout spans the time frame 5 to 9 and the resource had a position in a previously constructed layout that spans the time frame 7 to 8, then MovePlan will display it at that position and it will allow the user to change that position. In contrast, if the previously constructed layout spans the time frame 8 to 12, then MovePlan will lock the resource at its previous position.

If the resource has a position in non-overlapping layouts, then MovePlan will display it in the left view along with the other resources that have not yet been positioned in

any layout. The resource, however, will be tagged with a hollow box (as opposed to a black box) to inform the user that it has a position in some previous or later time frames. By clicking on the resource, the user causes MovePlan to move the resource from the left view and put it at its previous position (see resource 5 in Fig. 3.1). If the resource has different positions in these time frames, then MovePlan chooses the position closest in time to the current time frame. For example, if the current layout spans a time frame 5 to 6 and the resource has one position in the layout of time frame 2 to 4 and another position in the layout of time frame 9 to 12, then MovePlan selects the position from time frame 2 to 4. These consistency checks greatly reduce the work that a person has to do when constructing time-overlapping layouts.

MovePlan allows the user to construct sequential layouts, where time frames are selected in chronological order, or hierarchical layouts, where time frames are selected according to some hierarchy. In the latter case, for example, users can select time frames in decreasing order of total site area, or select time frames of longer duration and decompose them into several of shorter duration to gradually add details to the corresponding layouts. The construction of hierarchical layouts is possible because MovePlan allows the user to construct partial layouts: a layout can be created for a chosen time interval where not all resources are positioned on site. This last feature gives the user the flexibility of going back and forth between layouts before finalizing any. Examples using MovePlan for constructing hierarchical and sequential layouts are presented in Tommelein and Zouein (93).

### 3.1.4 Output

MovePlan provides users with many graphical displays to assist them in the construction of dynamic layouts. Among these displays are the time-scaled schedule, individual and total resource histograms, the total space histogram, and the layouts playback. The time-scaled schedule is an activity-on-node display showing precedences between activities. An individual resource's histogram shows that resource's use over time.

The total resource histogram displays all resources used in a project over time where each resource takes up a unit height. In Fig. 3.2, for example, resource 2 (labeled 2 on the histogram) is set up with a height of 1 on the ordinate of the histogram indicating a unit resource that is needed in the time frame 0 to 2. Each number displayed on the histogram refer to a resource number. Note that the black areas shown in that figure depict resources whose color is black. The total number of resources is one indicator of site

congestion, so the total resource histogram can be used to identify periods of congested sites. Guided by the total resource histogram, the user can select time frames for which to construct layouts, e.g., in decreasing order of total number of resources needed on site. In the case shown in Fig. 3.2, the user can select time frame 3 to 6 to lay out first because it has the highest number of resources (a total of 13 resources) needed on site.

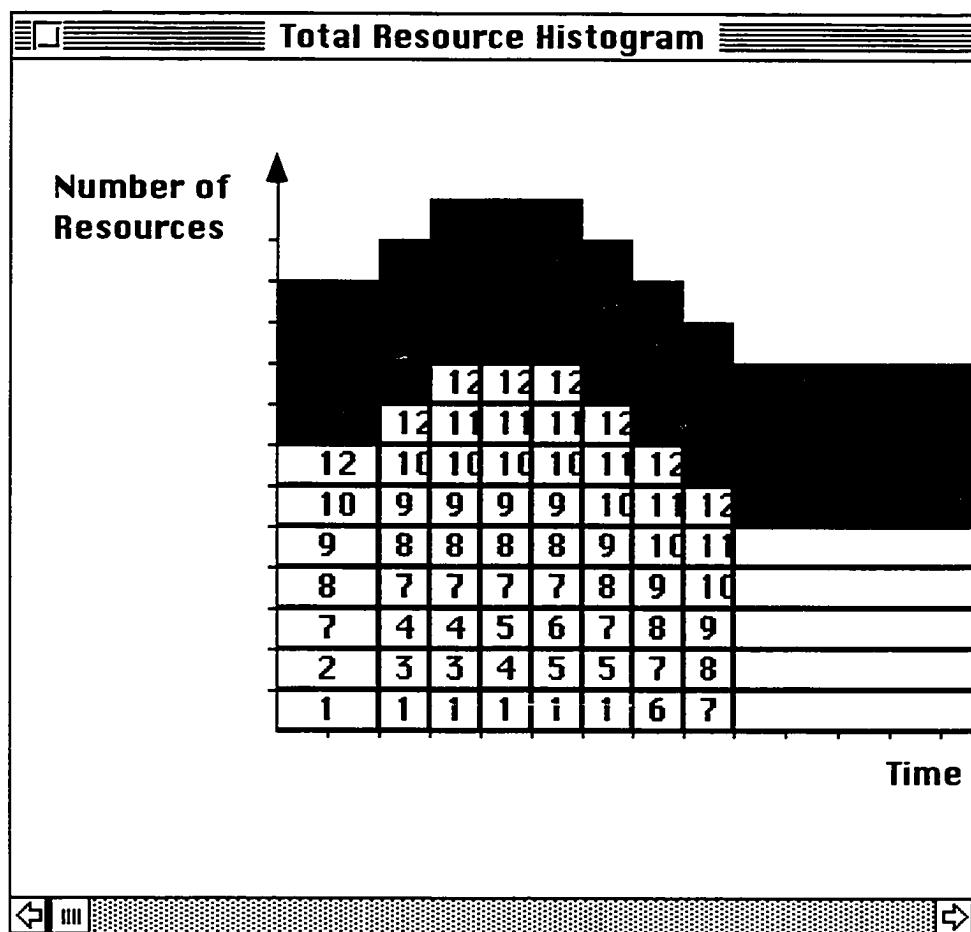


Figure 3.2  
Screen Dump of MovePlan's Total Resource Histogram

Another indicator of site congestion is total space need, so the total space histogram can also be used to identify periods of congested sites. The total space histogram displays the sum of the areas needed by all resources on site over time, where each resource takes up a height proportional to its area. Fig. 3.3 shows the top and bottom portions of a space histogram window displayed by MovePlan.

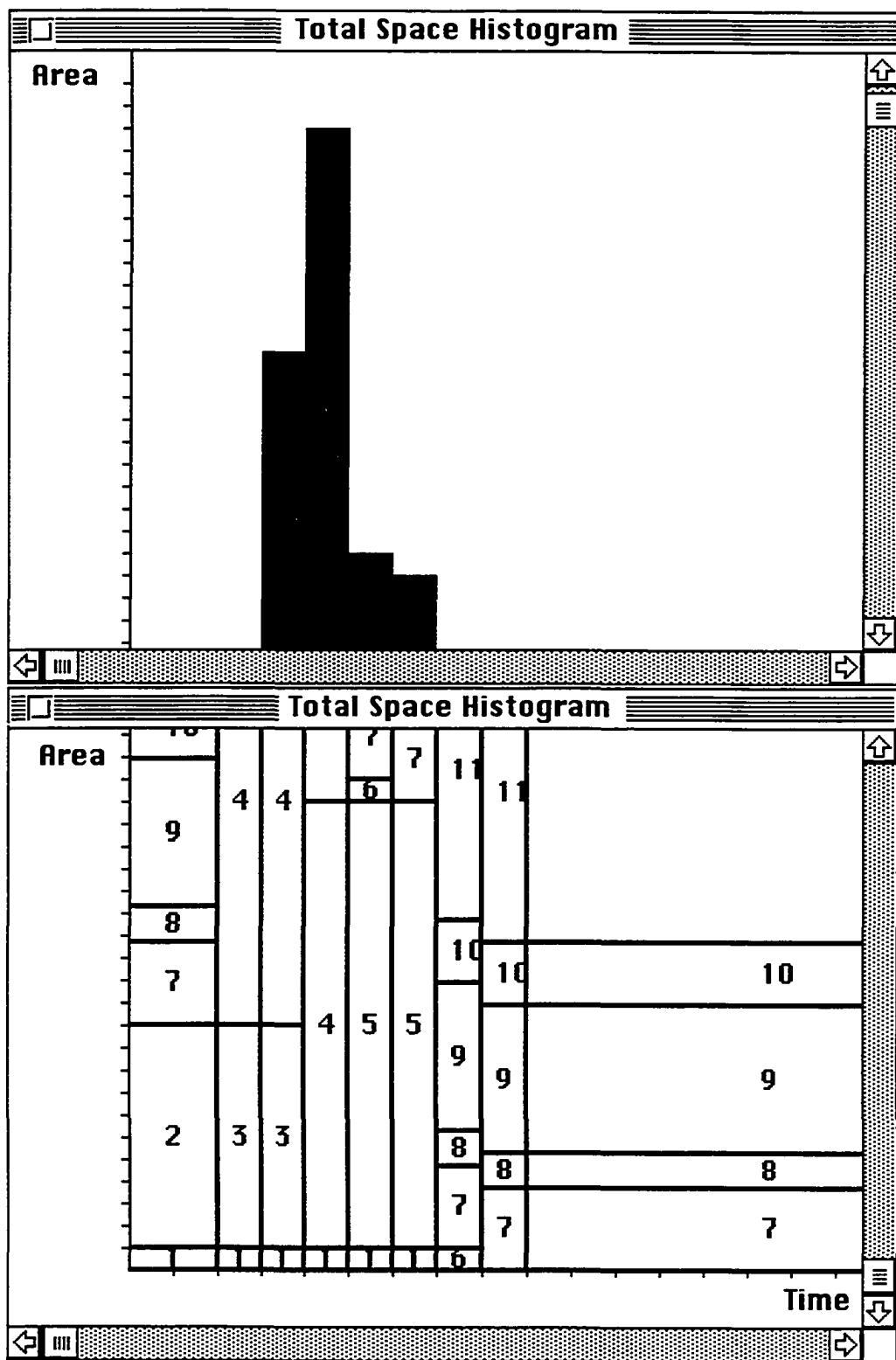


Figure 3.3  
Top and Bottom Portions of MovePlan's Total Space Histogram Window

In this histogram, for example, resource 2 has an area of 10 and is set up with a height of 10 on the ordinate of the histogram corresponding to the time frame 0 to 2. Note that the black areas shown in the top portion of the histogram in Fig. 3.3 depict the areas of resources whose color is black. The total space histogram, much like the total resource histogram, can guide users in their selection of layout time frames. For example, in the case shown in Fig. 3.3, the user can select time frame 4 to 5 to lay out first because it has the highest area requirement.

The layouts playback display simulates how the site layout changes over time by playing back the layouts created by the user in a time-lapse mode. The length of time over which each layout is displayed is either proportional to the duration of the layout time frame or some user-specified time period.

## 3.2 MovePlan-ConRes

MovePlan, as a stand-alone application, does not allow changes to the schedule when spatial conflicts (e.g., the site is too congested, or there is not enough space to accommodate all resources) are detected by the user. To investigate how spatial conflicts can be remedied by changing the schedule, a system named MovePlan-ConRes (Tommelein et al. 93) was developed. The system loosely integrates MovePlan with a reactive scheduler named the Conflict Resolver (ConRes). ConRes is called reactive because it modifies the schedule only when it is opportune to do so, i.e., when a space conflict has been pointed out by the system user. ConRes is comprised of a CPM scheduling algorithm and strategies to resolve a spatial conflict by altering the activity schedule. ConRes and MovePlan run independently on separate computers and exchange data over a network using a generic FileTalk communication package.

### 3.2.1 Modeling Assumptions

When MovePlan is used in conjunction with ConRes, the user is limited in the selection of time frames to what have been termed Primary Time Frames (PTFs) in section 1.2.1. By definition, a PTF includes only resources that coexist on site, i.e., those resources that share the site space during that time interval. Restricting time frames to PTFs makes sure that the spatial conflict is because the user could not create a feasible arrangement including those resources.

The following assumptions govern the ConRes model:

- resources, with the exception of space, exist in unlimited supplies.
- activities are scheduled at their earliest start date but can be delayed, i.e., there are no hard constraints that force activities to start or finish at some mandatory date.
- activities cannot be interrupted.
- activities are performed at a constant resource rate, i.e., the number or type of resources associated with an activity cannot change from one time frame to the next.
- no changes can be made to the start date of an activity or to the number of resources assigned to it once it has started.
- the construction method or the type of resources assigned to an activity cannot be changed. ConRes can change only the number of resources assigned to an activity, provided they are of the same type (e.g., an activity may use 1 or 2 loaders).
- the duration of an activity varies linearly with the number of resources assigned to it (e.g., if an activity that uses 2 loaders has a duration of 1 day, then its duration using 1 loader is 2 days).

### **3.2.2 Input**

ConRes takes as initial input a set of activities with their durations and associated resources. When the user detects a spatial conflict, he/she must provide additional input to ConRes including the problematic resources, the start and end of the problematic time frame where conflict occurred.

### **3.2.3 Conflict Resolution Strategies**

ConRes assumes that the MovePlan user has selected a PTF and identified the resources responsible for the conflict. It uses two main strategies that vary the schedule to remove the user-specified resources from the problematic time frame. The first strategy (1) is to delay one or several activities. The second strategy (2) is to decrease the number of resources of the same type assigned to an activity.

These two strategies specialize into six substrategies (1.1 to 1.4 are substrategies of 1, and 2.1 and 2.2 are substrategies of 2) that either minimize project delay or minimize resource idle time while removing the problematic resources from the given PTF.

Resource idle time is computed as the sum of idle times with no consideration for the number of hirings and firings of the resource. These substrategies are the following:

1.1 *Delay a single activity while minimizing project delay*

This strategy collects the activities that start in the given PTF and use the user-specified resources. From this collection, it selects the activity that minimizes the increase in project duration if delayed to start at the next PTF. If more than one activity ties, it applies the tie breaking rule described at the end of this paragraph. This substrategy may not free up space in the chosen time frame if the delayed activity uses resources that are shared by other on-going or starting activities in this PTF.

1.2 *Delay a single activity while minimizing the increase in resource idle time*

This substrategy is similar to 1.1, except that it selects the activity that minimizes the increase in resource idle time. This substrategy also may not free up space in the chosen time frame for the reason given in 1.1.

1.3 *Delay activities to remove one of the user-specified resources while minimizing project delay*

This strategy collects, for each user-specified resource, the activities using it. For each grouping, it checks if all activities start in the given PTF. If they do not, it discards this grouping. From the remaining groupings, it selects the one whose postponement minimizes the project delay. This substrategy may not be successful in removing any of the user-specified resources from the given PTF if the activities of all groupings do not start in this PTF.

1.4 *Delay activities to remove user-specified resources while minimizing the increase in resource idle time*

This substrategy is similar to 1.3 except that it selects the grouping whose postponement minimizes the increase in resource idle time. The same reservations also apply.

2.1 *Decrease the number of resources while minimizing project delay*

This strategy collects all activities that start in the given time frame and that use the user-specified resources, where each activity uses multiples of the same resource type. From this collection, it reduces the number of resources of the activity that minimizes project delay. This substrategy may be not be successful if the number of resources cannot be reduced or if activities use resources of different types.

2.2 *Decrease the number of resources while maximizing available space*

This substrategy is similar to 2.1 except that it selects the grouping whose postponement maximizes the available site space. The same reservations also apply.

If one of the substrategies of the first strategy results in multiple choices for activities to be modified, the other three are consecutively applied to break the tie. Similarly, if one of the two substrategies specializing the second strategy results in multiple choices for resources to be modified, the other one is applied to break the tie. If no substrategy can break the tie, ConRes selects the first one of its possible choices.

### 3.2.4 Computations

When the user detects a space conflict, he/she identifies the problematic resources and commands MovePlan to send a file to ConRes including these resources, the start and finish of the problematic time frame, and the user preferences for time and money (denoted here as UPT and UPM respectively). The values of UPT and UPM range between [0 100]. These preferences are used to rank the substrategies so that the increase in project duration is minimized if time is preferred or the increase in resource idle time is minimized if money is preferred. When ConRes receives data about a space conflict, it calculates the priority for each substrategy (X) using the priority function given below and the weighting factors in Table 3.1. The 'one activity at a time weight' is used to give priority to the substrategies that return a single activity. The weighting factors shown in Table 3.1 were intended to make the strategy and substrategy selection a dynamic process that can be changed during problem solving by varying these weights. For example, the 'one activity at a time weight' can be changed to zero and this will make the substrategies compete only based on the user preference for time and money. The MovePlan-ConRes project, however, ended at this and there was no analysis done for refining or validating the priority function and these weights.

$$\begin{aligned} \text{Priority function: } \text{Priority (X)} = & \quad \text{UPT} \times \text{Time Preference Weight (X)} \\ & + \text{UPM} \times \text{Money Preference Weight (X)} \\ & + \text{one activity at a time weight (X)} \end{aligned}$$

where X is a substrategy.

ConRes ranks all substrategies in decreasing order of their priority value and selects the first one in the order found. If the selected substrategy does not return an activity or activities to be modified, then ConRes select the next strategy in that order. This process

may result in a schedule change (i.e., an applicable substrategy is found and executed) or failure (i.e., no substrategy is found). If the result is a change in the schedule, then ConRes writes the resulting data to a file to be returned over the network to MovePlan and waits to be called again by MovePlan to solve a new conflict. The file sent to MovePlan includes data regarding the new start and finish dates of activities and the new resources assigned to each. If the result is a failure, then an impasse is reached and ConRes reports it back to MovePlan.

For any selected PTF, ConRes can be called repeatedly by MovePlan to solve spatial conflicts as they are detected by the user during layout construction.

Substrategy	Time Preference Weight	Money Preference Weight	One Activity at a Time Weight
1.1	1	0	100
1.2	0	1	100
1.3	1	0	0
1.4	0	1	0
2.1	1	0	0
2.1	0	1	0

Table 3.1  
Weights for Prioritizing Substrategies in ConRes

### 3.2.5 Output

ConRes output is a data file that contains information regarding the adjusted start and finish dates of activities and resources assigned to these activities if ConRes succeeded in finding a strategy. Otherwise, ConRes returns a file which contains a notification of failure.

## 3.3 Summary and Research Directions for MoveSchedule

MovePlan is a graphical and interactive decision-support tool for interactively constructing dynamic layouts that suit resources' space needs as dictated by an activity schedule. MovePlan supports its users by maintaining consistency between the layouts. Consistency is maintained by tracking resource positions over time and restricting the relocation of those

that would result in inconsistent layouts. It is these features of MovePlan that make it stand out from other generic computer drawing programs.

It is left up to the MovePlan user to assess whether enough space is available to accommodate a resource at any given time. Location constraints between resources and other types of interactions between resources are not explicitly modeled by MovePlan. The user must define what they are and take them into account when creating a layout. Gauging the quality of a dynamic layout, which often depends on such constraints and interactions, is also left to visual inspection by the user. Expressing such constraints and interactions is key to developing a system that can automatically generate and evaluate dynamic layouts. These needs define the research direction taken in the development of MoveSchedule, which is the subject of the remaining chapters of this dissertation.

MovePlan-ConRes is an interactive decision-support tool that addresses the space scheduling problem in a loose setup where two independent programs, MovePlan and ConRes, exchange data regarding the timing of a spatial conflict, the resources that contributed to this conflict, and the adjusted schedule. ConRes uses different strategies to resolve spatial conflicts and is driven by two objectives: minimizing the increase in project duration or resource idle time based on user preference for time and money.

MovePlan-ConRes relies heavily on user interaction in constructing the layouts, identifying conflicts and labeling the cause of the conflicts. The user interaction can be replaced by a fully automated algorithmic approach that solves the space scheduling problem. The advantages and shortcomings of such an approach are investigated in MoveSchedule.

MovePlan and ConRes make many simplifying assumptions regarding resource space use over time, and the type and number of resources assigned to an activity in relation to its duration. MovePlan-ConRes assumes a constant space need for all resources. It also assumes that the duration of an activity varies as a linear function of the number of units of a single resource type. In reality, activity durations depend on the construction method and the type and the number of resources used, and may not vary linearly with a change in method or crew-size. Characterizing resource space needs over time and explicitly characterizing activity durations based on the methods or crew-sizes used for their performance are key to developing a system that realistically models time-space tradeoffs. These modeling needs define another research direction of MoveSchedule.

# Chapter 4

## INCREMENTAL DYNAMIC LAYOUT PROBLEM SOLVER

### 4.1 Introduction

The *dynamic layout construction module* in MoveSchedule solves a constrained dynamic layout problem. This chapter presents this module's underlying model and algorithms. The model uses a continuous 2-dimensional spatial representation to characterize the space requirements and positions of resources on site. Input to the model are resources, their dimensions, when and for how long they will be on site (i.e., their arrival and departure times), interactions among them in the form of proximity preference or transportation costs, their relocation costs, and geometric constraints on their acceptable positions.

The main algorithm uses a combination of Constraint Satisfaction and Linear Programming to determine positions of resources that 1) satisfy the constraints on their acceptable positions and 2) minimize costs associated with travel distances between interacting resources and the relocation of resources in between staging areas. The output is a sequence of site layouts that covers the duration of the resources' existence on site. During layout construction, changes in the arrival and departure times of resources are not allowed.

This chapter is organized as follows. Section 4.2 presents the modeling assumptions and choices pertaining to this module, namely, the choice of spatial representation and the choice of a dynamic layout evaluation function. Section 4.3 presents a two-stage algorithm for constructing the layout of resources in a given time frame. Section 4.4 describes the strategy used in constructing the layout sequence. Section 4.5 demonstrates the algorithm with an example. Chapter 4 concludes with a discussion of the capabilities, limitations, and possible extensions of the proposed methods.

## 4.2 Underlying Model

This section presents the modeling assumptions and choices pertaining to MoveSchedule's dynamic layout construction module. It discusses the discretization of time, the spatial representation, the variables to represent transportation and relocation costs of resources and constraints on their acceptable positions, and the dynamic layout evaluation function.

### 4.2.1 Discretizing Time

Although construction sites evolve continuously over time, MoveSchedule considers layout changes at discrete points in time only. These points in time are marked by the start and end of a PTF. As defined in Chapter 1, PTFs correspond to the smallest intervals delimited by the appearance and disappearance of resources and mark new demand for space or an opportunity for other resources to claim the freed space.

Given some resources (Resource-1, -2, -3, and -4) and the time intervals over which they exist on site ( $t_1-t_3$ ,  $t_2-t_4$ ,  $t_5-t_7$ ,  $t_6-t_8$ , respectively), Fig. 4.1 shows the corresponding PTFs ( $t_1-t_2$ ,  $t_2-t_3$ ,  $t_3-t_4$ ,  $t_4-t_5$ ,  $t_5-t_6$ ,  $t_6-t_7$ ,  $t_7-t_8$ ).

By definition, PTFs include only resources that coexist on site in that time frame. It is assumed that PTFs mark changes in resource positions, dimensions, and interactions, if applicable. In other words, it is assumed that resources do not change positions, dimensions, and interactions with other resources for the duration of a PTF.

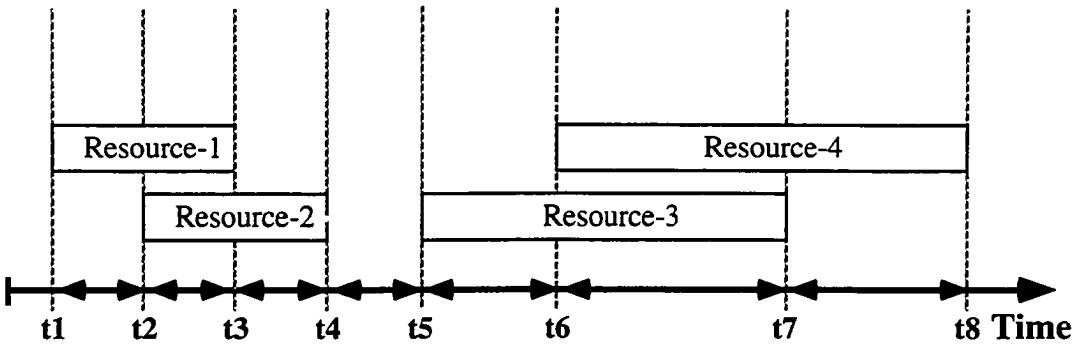


Figure 4.1  
Resource Primary Time Frames

#### 4.2.2 Resource Dimensions and Positions

A continuous 2-dimensional orthogonal grid is chosen for representing the space requirements and positions of resources on site. Resources are modeled as rectangles to be positioned at  $0^\circ$  or  $90^\circ$  orientation. By convention, a resource is said to be at  $0^\circ$  orientation if its long side is parallel to the X-axis, or at  $90^\circ$  if its long side is perpendicular to the X-axis. A resource's dimensions are fixed for the duration of a PTF, but can be different in different PTFs. Note that the dimensions of all resources are given in this problem and are not changed during problem solving.

The position of a resource is uniquely defined by the position of the centroid of the rectangle representing it at either  $0^\circ$  or  $90^\circ$  orientation. Single positions are saved as the x and y coordinates of the centroid in one of the two orientations for each PTF. Multiple positions are saved using a bounded interval representation (Confrey and Daube 88) which describes the *Set of Possible Positions (SPP)* of a resource. The *bounded interval representation* models the SPP of a rectangular object as a disjunction of rectangles. This representation is better explained with an example.

Consider a site space represented by the solid rectangle ( $[0 \text{ Xs}] [0 \text{ Ys}]$ ) and a resource R1 with dimensions ( $d_1 \text{ d}_2$ ) to be positioned within the site boundaries as shown in Fig. 4.2 (a). In this figure, R1 is shown at several of its infinitely many possible positions inside the site space. The SPP of R1 at the  $0^\circ$  orientation is the area defined by the irregularly-dashed rectangle ( $[X_1 \text{ X}_2] [Y_1 \text{ Y}_2]$ ) that confines R1 in its horizontal orientation to ( $[0 \text{ Xs}] [0 \text{ Ys}]$ ). R1's SPP is computed as follows:  $[0 + d_1/2 \text{ Xs} - d_1/2] [0$

$+ d2/2 \text{ Ys} - d2/2]$ . The SPP of R1 at the  $90^\circ$  orientation can be represented similarly. It is computed as  $[0 + d2/2 \text{ Xs} - d2/2] [0 + d1/2 \text{ Ys} - d1/2]$ .

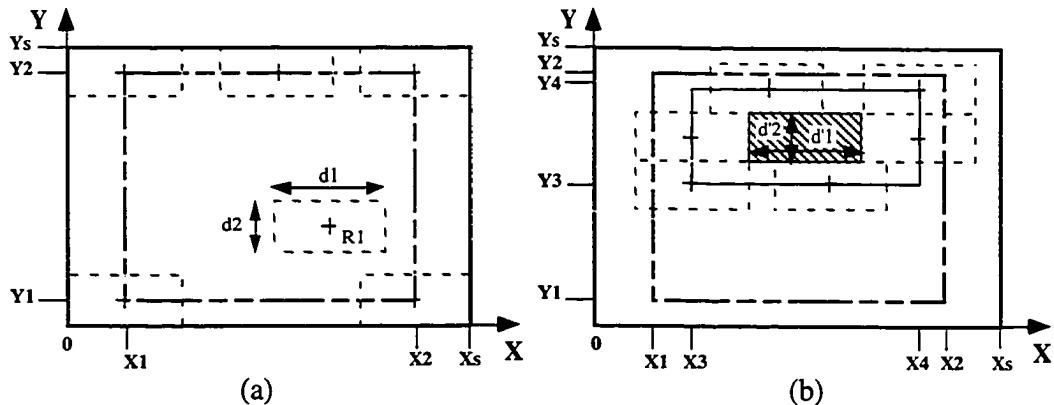


Figure 4.2  
Bounded Interval Representation

If another resource exists on site, such as the one with dimensions  $(d'1, d'2)$ , represented by the cross-hatched rectangle at the position shown in Fig. 4.2 (b), and R1 is not to overlap with it, then R1's SPP at  $0^\circ$  orientation becomes confined to only those points outside the area described by the solid-line rectangle  $([X3 \text{ } X4] \text{ } [Y3 \text{ } Y4])$  and within the rectangle  $([X1 \text{ } X2] \text{ } [Y1 \text{ } Y2])$ . If the centerpoint of R1 is positioned anywhere inside  $([X3 \text{ } X4] \text{ } [Y3 \text{ } Y4])$  then R1 would overlap with the cross-hatched resource. The reduced R1's SPP at  $0^\circ$  is computed by subtracting the area of  $([X3 \text{ } X4] \text{ } [Y3 \text{ } Y4])$  from the area of  $([X1 \text{ } X2] \text{ } [Y1 \text{ } Y2])$ , resulting in the following set of rectangles:  $\{ ([X1 \text{ } X3] \text{ } [Y1 \text{ } Y2]) \cup ([X4 \text{ } X2] \text{ } [Y1 \text{ } Y2]) \cup ([X3 \text{ } X4] \text{ } [Y1 \text{ } Y3]) \cup ([X3 \text{ } X4] \text{ } [Y4 \text{ } Y2]) \}$ . Any point  $(x \text{ } y)$  that belongs to any of these rectangles is a valid position for the centroid R1 at  $0^\circ$  orientation.

The following notation is used to represents the SPP of a resource:

$$\begin{aligned} \text{SPP} = & \{ \{ 0 \{ ([\text{Xmin-1 } \text{Xmax-1}] [\text{Ymin-1 } \text{Ymax-1}]) \dots \\ & ([\text{Xmin-m } \text{Xmax-m}] [\text{Ymin-m } \text{Ymax-m}]) \} \} \\ & \{ 90 \{ ([\text{Xmin-1 } \text{Xmax-1}] [\text{Ymin-1 } \text{Ymax-1}]) \dots \\ & ([\text{Xmin-q } \text{Xmax-q}] [\text{Ymin-q } \text{Ymax-q}]) \} \} \} \end{aligned}$$

where  $m, q \geq 0$  and  $q + m =$  total number of rectangles in SPP.

The first element in the set specifies the set of rectangles within which the centroid of the resource must lie when the resource is at  $0^\circ$  orientation. Similarly, the second element specifies the set of rectangles when the resource is at  $90^\circ$  orientation. More specifically, the first element in each sub-set is the orientation. The second element is the set of rectangles that describe the SPP of the resource at that orientation. Each rectangle is defined between

parentheses with the first two numbers in brackets representing its bounds in the x-direction and the second two those in the y-direction. There could be zero, one, or several such rectangles.

The bounded interval representation enables MoveSchedule to represent a set of infinitely many positions and delay commitment to single resource positions in constructing the layout of a PTF. This representation also proves to be very useful when it comes to selecting a particular position for a resource in the process of constructing the layout of a PTF, because it partitions the feasible space of resource positions into convex sub-regions (individual rectangles). This property makes it possible to formulate and solve the resource positioning problem as a linear program. Note that the SPP of a resource (i.e., the union of rectangles) is not a convex space. However, as shown later, this is not an issue in this problem.

#### **4.2.3 Resource Preference Measures and Hard Constraints**

The construction of the dynamic layout is driven by distance-based preference measures and hard constraints. Both are user input.

##### **4.2.3.1 Preference Measures: Proximity and Relocation Weights**

Preference measures consist of proximity and relocation weights used to gauge the quality of the constructed layouts.

A *proximity weight* applies to a pair of resources and reflects the level of interaction between them based on the rectilinear distance (see Fig. 2.1 (b) in Chapter 2) separating the positions of their respective centroids in a given PTF. These interactions may be in the form of material flow, equipment accessibility, or desired proximity. Proximity weights can have any value greater than or equal to zero. A high value means that the two resources have a high level of interaction and the distance between them should be small. A value of zero indicates that there is no interaction between the two resources and, therefore, the distance separating them is irrelevant. Proximity weights cannot have negative values. A negative value might indicate that two resources should not be close to one another in the layout; this situation is better modeled in MoveSchedule using hard constraints which are treated in the next section.

A proximity weight can be used to denote transportation costs or some closeness desirability rating. It is up to the user to select a value that best describes the level of interaction between any two resources. As a guideline only, the following suggests ways of quantitatively or qualitatively assessing these interactions for a given problem.

*Quantitative measure:* a proximity weight can be used to represent, for example, actual transportation cost per unit time or the amount moved per unit time (flow) between supply and demand areas on site.

A "supply area" is used here to denote the area that the resource currently occupies while a "demand area" denotes the area to which it is being moved. For example, a supply area may be a lumber laydown area, a demand area may be a lumber fabrication shop, and the resource moved between the two may be lumber. The lumber laydown area, the lumber fabrication shop, and lumber are normally referred to in this dissertation as resources. The use of the words supply and demand areas here is intended to facilitate the following illustration on the use of proximity weights.

To model actual transportation costs, a proximity weight may be the unit cost of the handling means used for moving the resource between the supply and the demand area.

To model the flow of resources, the user may wish to establish a measure of flow for the project, that accurately indicates equivalent flow volumes, before assigning any proximity weight value between a pair of resources.

In cases where the resources moved are equivalent with respect to ease of movement then the number of trips per unit time between their respective supply and demand areas can be used as a surrogate measure for flow volumes and thus as the values for proximity weights. For example, consider moving pallets of bricks from their laydown area to the work area next to a wall under construction vs. moving pallets of cement bags from their laydown area to the mortar mixer. Because moving pallets of bricks is as easy as moving pallets of cement bags, the user can use the number of trips per unit time made in moving each resource as the value for the proximity weight. Assume that one trip of cement bags is needed for every four trips of bricks, then the user may set the proximity weight between the bricks' laydown area and the work area to be four times equal to the proximity weight between the cement bags' laydown area and the mortar mixer.

In cases where the resources moved vary in size, weight, value, risk of damage, shape, etc., then the user may wish to establish values for the proximity weights that reflect

the proper relationships among the volume of movements. For example, consider moving drywall gypsum boards from their laydown area to a work area inside the building vs. moving pallets of bricks from their laydown area to a work area next to an exterior wall. Assume that the user can establish a relationship between the two resources based on the relative ease of movement, such as, it is five times easier to move the bricks than to move the gypsum boards; then ease of movement can be used as a surrogate measure of flow. Accordingly, the user may set the proximity weight between the gypsum boards' laydown area and their work area to be five times equal to the proximity weight between the bricks' laydown area and their work area.

***Qualitative measure:*** a proximity weight can express the desirability of having the two resources close to one another in the layout. As such, this weight constitutes a relative measure and should be normalized by the user to reflect the relative magnitude of interactions between pairs of resources.

Proximity weights need to be defined for each PTF because they can vary for any two resources over time. For example, a table-saw assigned to a make-formwork activity, may need to be close to lumber for a time period equal to the duration of the make-formwork activity and thus have a proximity weight defined between it and the needed lumber. The same table-saw may be assigned to another activity that does not overlap in time with the make-formwork activity. If the saw has no interaction with the lumber remaining on site over this second activity's duration, the proximity weight between it and lumber will be zero in this time interval.

A ***relocation weight*** applies to an individual resource and measures the cost of relocating the resource from one layout to the next. In relocating the resource, no consideration is given to the time it takes to move it nor to the availability of the handling means needed for moving it. Relocation weights can have any value greater than or equal to zero. The value zero implies that the resource can easily be relocated, i.e., the cost of relocating it is negligible.

Resources that should not be relocated across layouts are termed ***stationary***; if such a resource ends up at different positions in different layouts, the layout sequence will be considered infeasible. It is up to the user to deem a resource stationary and label it as such when defining it in MoveSchedule. The user should be careful, however, not to overconstrain the problem as in doing so he/she is commanding MoveSchedule to find a

single position for that resource in all layouts and failure to do so will result in a spatial conflict.

#### 4.2.3.2 Hard Constraints

**Hard constraints** are 2-dimensional geometric relationships between relative positions of two resources. MoveSchedule uses hard constraints to prune the SPPs of resources because these constraints must hold for the resources' positions to be acceptable. The 2-dimensional constraints allowed in MoveSchedule between, e.g., R1 with dimensions  $L_1 \times W_1$  and R2 with dimensions  $L_2 \times W_2$ , where  $(X_1, Y_1)$  and  $(X_2, Y_2)$  denote the coordinates of the centroids of R1 and R2 respectively, and C denotes the constraint between them, are as follows:

- **Non-overlap** (also called *out-zone*) **constraints** apply by default between a resource R2 for which a position is to be determined and a zone R1. A **zone** is a resource with a known position in the layout; it could be a static resource or a resource that has already been positioned in the layout. An out-zone constraint is expressed as follows:

$$C = (R2 \text{ } R1 \text{ (out-zone)})$$

where, if R1 is at  $0^\circ$ ,

$$\begin{aligned} \text{or } |X_2 - X_1| &\geq L_1/2 + L_2/2 && ; \text{ for R2 at } 0^\circ \\ |Y_2 - Y_1| &\geq W_1/2 + W_2/2 \end{aligned}$$

or

$$\begin{aligned} \text{or } |X_2 - X_1| &\geq L_1/2 + W_2/2 && ; \text{ for R2 at } 90^\circ \\ |Y_2 - Y_1| &\geq W_1/2 + L_2/2 \end{aligned}$$

Equations for R1 positioned at  $90^\circ$  can be similarly derived. Out-zone constraints prevent resources (such as a crane and a loader) from being assigned to overlapping spaces if they co-exist on site. A default out-zone constraint can be overridden by defining an in-zone constraint between the two resources, i.e., no partial overlap between resources is modeled.

- **In-zone constraints**, in contrast to out-zone constraints, limit the space occupied by a resource (R2) to be within the spatial boundaries of a zone (R1). An in-zone constraint is expressed as follows:

$$C = (R2 \text{ } R1 \text{ (in-zone)})$$

where, if R1 is at  $0^\circ$ ,

$$\begin{cases} |X_2 - X_1| \leq L_1/2 - L_2/2 & ; \text{ for R2 at } 0^\circ \\ |Y_2 - Y_1| \leq W_1/2 - W_2/2 \end{cases}$$

or

$$\begin{cases} |X_2 - X_1| \leq L_1/2 - W_2/2 \\ |Y_2 - Y_1| \leq W_1/2 - L_2/2 \end{cases}; \text{ for } R2 \text{ at } 90^\circ$$

Equations for R1 positioned at  $90^\circ$  can be similarly derived. In-zone constraints limit the space occupied by a resource to some pre-defined area (such as a laydown or fenced-in area).

- **Minimum (Maximum) distance constraints** limit the distance between the facing sides of two resources (R1 and R2) in the x- or y-direction (see Fig. 2.1 (c) and (d)), to be greater (less) than a user-defined value. A minimum distance of V in the x-direction is expressed as follows:

$$C = (R2 \text{ R1 (min x V)}) = (R1 \text{ R2 (min x V)})$$

where

$$|X_2 - X_1| - (L_1/2 + L_2/2) \geq V; \text{ for } R1 \text{ at } 0^\circ \& R2 \text{ at } 0^\circ$$

or  $|X_2 - X_1| - (W_1/2 + L_2/2) \geq V; \text{ for } R1 \text{ at } 90^\circ \& R2 \text{ at } 0^\circ$

or  $|X_2 - X_1| - (L_1/2 + W_2/2) \geq V; \text{ for } R1 \text{ at } 0^\circ \& R2 \text{ at } 90^\circ$

or  $|X_2 - X_1| - (W_1/2 + W_2/2) \geq V; \text{ for } R1 \text{ at } 90^\circ \& R2 \text{ at } 90^\circ$

A minimum distance of V in the y-direction can be expressed similarly by substituting  $X_2$  for  $Y_2$  and  $X_1$  for  $Y_1$  in the above equations. A maximum distance constraint of V in the x-direction can be expressed using the above equation by substituting the  $\geq$  sign for a  $\leq$  sign. Distance constraints may model equipment reach or any general clearance requirements.

- **Orientation constraints** specify that a resource R2 be to the North (South, East, or West) of a reference resource R1. A North-of constraint is expressed as follows:

$$C = (R2 \text{ R1 (North-of)})$$

where

$$Y_2 - Y_1 \geq W_1/2 + W_2/2; \text{ for } R1 \text{ at } 0^\circ \& R2 \text{ at } 0^\circ$$

or  $Y_2 - Y_1 \geq L_1/2 + W_2/2; \text{ for } R1 \text{ at } 90^\circ \& R2 \text{ at } 0^\circ$

or  $Y_2 - Y_1 \geq W_1/2 + L_2/2; \text{ for } R1 \text{ at } 0^\circ \& R2 \text{ at } 90^\circ$

or  $Y_2 - Y_1 \geq L_1/2 + L_2/2; \text{ for } R1 \text{ at } 90^\circ \& R2 \text{ at } 90^\circ$

An orientation constraint may be used to locate access roads, gates, parking, or toilet facilities with respect to the main structure.

- **Parallel (perpendicular) constraints** limit the orientation of a resource R2 to be equal (opposite) to the orientation of a reference zone R1. A parallel constraint is expressed as follows:

$$C = (R1 \text{ R2 (parallel)})$$

where, if R1 is at  $0^\circ$  then R2 is at  $0^\circ$  and

$$\text{or } |X_2 - X_1| \geq L_1/2 + L_2/2$$

$$|Y_2 - Y_1| \geq W_1/2 + W_2/2$$

Equations for R1 positioned at  $90^\circ$  can be similarly derived. A parallel constraint may be used to orient scaffolding to run along the side of a wall. A perpendicular constraint may be used to orient a piece of equipment such as a dumptruck with respect to a fill area.

Hard constraints are defined for each PTF and can be different in different PTFs. For example, a laydown area may be used to store precast structural members throughout the main structure erection stage and store machinery over some other stage of construction. In this case two in-zone constraints need to be defined for different time frames, one between the laydown area and the precast members, and another between the laydown area and the machinery. Each constraint spans a different time interval covering the two stages of construction.

#### 4.2.4 Dynamic Layout Evaluation

The ***Value Function of the Layout (VFL)*** assesses the quality of the dynamic layout. It consists of two components. P measures the sum of costs associated with travel distances between interacting resources for all layouts, each spanning a PTF. R measures the sum of costs associated with relocating resources across layouts. The following equations are used to compute VFL.

$$VFL = P + R \quad (\text{eq. 1})$$

$$P = \sum_t P_t = \sum_t \sum_i \sum_j w_{ij}^t d_{ij}^{tt} \Delta t \quad (\text{eq. 2})$$

$$R = \sum_t R_{t(t-1)} = \sum_t \sum_i w_i^t d_{ii}^{(t-1)t} \quad (\text{eq. 3})$$

Where

$w_{ij}^t$  : Proximity weight between resources i and j in layout t. Layout t denotes a layout that spans the duration of the  $t^{\text{th}}$  PTF

$w_i^t$  : Relocation weight of resource i

$d_{ij}^{tt}$  : Rectilinear distance between centroids of resource i in layout t and resource j in layout t

$d_{ii}^{(t-1)t}$ : Rectilinear distance between centroids of resource  $i$  in layout  $(t - 1)$  and resource  $i$  in layout  $t$

$\Delta_t$  : Length of time over which layout  $t$  extends.

VFL is similar to the weighted distance-based objective function used in solving the static facility layout problem. VFL is adjusted to reflect the nature of the flow in a dynamic environment, namely the possibility of relocating resources over time. First,  $P$  is made into a function not only of distance but also of time. It measures costs that depend on the duration of a PTF. Proximity weights are defined for a single layout and can differ for the same resources in consecutive layouts. In addition, each layout ( $t$ ) is weighed by the time period ( $\Delta_t$ ) it spans: a layout  $t$  will contribute a greater weight to the objective function if it spans a longer  $\Delta_t$ . Second,  $R$  is included to measure the cost associated with relocating resources across layouts.  $R$  measures costs that are a function of distance and not of time.

### 4.3 Problem Solving Method

Solving the dynamic layout problem involves creating a sequence of layouts that span the duration of construction of a project, given resources, the timing of their presence on site, their dimensions, 2-dimensional constraints on their location (hard constraints), transportation weights ( $W_{ij}^t$ ) and relocation weights ( $W'_{ij}$ ). Each layout in the sequence will span a time interval corresponding to a PTF.

The objective is to develop feasible and efficient layouts for all PTFs. The feasibility of an individual layout is established by satisfying the space requirements of each resource unit while ensuring that the hard constraints and the default non-overlap constraints between them are met. The feasibility of layouts over time is ensured by restricting stationary resources from being assigned different positions in different layouts. The efficiency of the layout sequence is obtained by minimizing VFL.

The following section describes the use of Constraint Satisfaction to meet feasibility requirements. Section 4.3.2 presents an algorithm that uses Constraint Satisfaction to determine resources' SPPs that satisfy all hard constraints among them in a PTF. Section 4.3.3 uses Linear Programming to find the positions that minimize VFL for a selected resource in a PTF. Sections 4.3.4 and 4.3.5 integrate the aforementioned methods into a single algorithm for constructing the layout of a PTF.

### 4.3.1 Pruning Positions to Meet Hard Constraints Using Constraint Satisfaction

MoveSchedule's hard constraints are built on top of the GS2D constraint engine (Confrey and Daube 88). GS2D satisfies geometric constraints between rectangles in a 2-dimensional coordinate system using a bounded interval representation. The rectangles are parallel to the X-Y coordinate axes and represent the SPP of resources' centroids at 0° and 90° orientation. As hard constraints are applied to resources, GS2D reads their original SPP and returns a pair of modified SPPs, each resource's SPP pruned to include only those positions consistent with the applied constraint.

Additions to GS2D were necessary to incorporate the orientation constraints (North-of, South-of, etc.) between SPPs of resources. In addition, modifications of the GS2D distance constraints were needed to account for the way distances are measured in MoveSchedule (i.e., between facing edges of rectangles in either the x- or y-direction, see Fig. 2.1 in Chapter 2) as opposed to the minimal orthogonal distance between edges in GS2D (i.e., the minimum of the two distances between edges of rectangles in the x- or y-direction). The modified constraint engine is referred to as MS-GS2D.

The following example illustrates Constraint Satisfaction in the case of two resources, R1 and R2, with a minimum distance constraint C = (R1 R2 (min y 4)). Consider R1 to have dimensions (6 x 4), R2 (8 x 2), and a site space (20 x 12). Fig. 4.3 (a) shows the SPP of R1 and R2 after they have been positioned within the site boundaries but before applying C. In particular, the solid-line rectangles represent the SPP of R1 at 0° and 90° orientations and is noted as:

$$SPP_1 = \{ \{0 \{([3 17] [2 10])\} \} \{90 \{([2 18] [3 9])\} \} \}.$$

Similarly, the dashed line rectangles represent the SPP of R2 at 0° and 90° orientations and is noted as

$$SPP_2 = \{ \{0 \{([4 16] [1 11])\} \} \{90 \{ ([1 19] [4 8])\} \} \}.$$

Fig. 4.3 (b) shows the reduced SPP<sub>1</sub> and SPP<sub>2</sub>, as returned by MS-GS2D, after satisfying the minimum distance constraint. The reduced sets are

$$SPP_1 = \{ \{0 \{([3 17] [2 4]) ([3 17] [8 10])\} \} \{90 \{([2 18] [9 9]) ([2 18] [3 3])\} \} \}$$

$$SPP_2 = \{ \{0 \{([4 16] [1 3]) ([4 16] [9 11])\} \} \{90 nil\} \}.$$

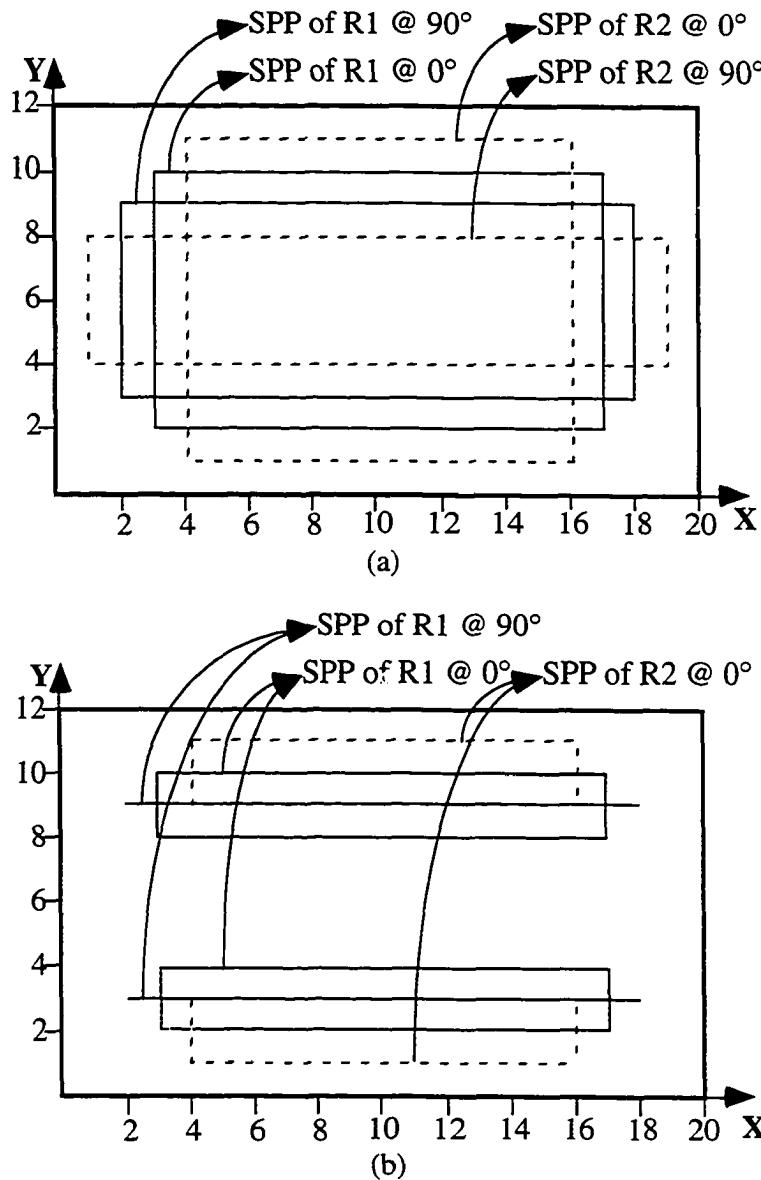


Figure 4.3  
Constraint Satisfaction

This result is interpreted as follows. If R1 is to be positioned at  $90^\circ$ , i.e., its centroid is any point on the solid black lines shown in Fig. 4.3 (b), there exists a feasible position for R2 at  $0^\circ$  that does not violate the minimum distance constraint, and that is a point belonging to the two dashed rectangles in Fig. 4.3 (b). However, if R2 is positioned at  $90^\circ$ , R1 cannot be positioned within the site, whether at  $0^\circ$  or  $90^\circ$ , and still meet the constraint with R2. Therefore, R2 has no feasible positions at  $90^\circ$  orientation for which a feasible position for R1 exists.

This example considers a single constraint between two resources. When a resource is constrained by different resources, constraint propagation becomes necessary after each Constraint Satisfaction to maintain consistency among SPPs of resources. For illustration, consider a small constraint network consisting of constraints C1 and C2, where C1 exists between R1 and R2, and C2 exists between R1 and R3. Let SPP<sub>1</sub>, SPP<sub>2</sub>, and SPP<sub>3</sub> designate SPPs of R1, R2 and R3 respectively. Assume that satisfying C1 pruned SPP<sub>1</sub> and SPP<sub>2</sub> to SPP'<sub>1</sub> and SPP'<sub>2</sub> respectively. To satisfy C2, SPP'<sub>1</sub> and SPP<sub>3</sub> are reduced to SPP"<sub>1</sub> and SPP"<sub>3</sub> respectively. The latter Constraint Satisfaction necessitates the updating of all constraint links that R1 shares with other resources, in this case R2, to reflect the change in SPP'<sub>1</sub>. Indeed,  $\forall (X_2, Y_2) \in SPP'_2, \exists (X_1, Y_1) \in SPP'_1$ , such that (X<sub>1</sub>, Y<sub>1</sub>) and (X<sub>2</sub>, Y<sub>2</sub>) are potential positions of the centerpoints of R1 and R2 respectively that do not violate C1. SPP"<sub>1</sub> is a subset of SPP'<sub>1</sub> and thus may exclude (X<sub>1</sub>, Y<sub>1</sub>), making (X<sub>2</sub>, Y<sub>2</sub>) infeasible. Hence, constraint propagation is needed, which involves updating SPP'<sub>2</sub> to become compatible with SPP"<sub>1</sub>. The algorithm for Constraint Satisfaction and Propagation, used in MoveSchedule, is presented in the following section.

Keeping a set of possible positions for all resources and using Constraint Satisfaction to reduce the set of feasible positions has the advantage of identifying infeasibility early in the problem solving process and before committing resources to a single position. It also prunes the feasible space, so alternative sets of single positions for the resources that do not violate any of the spatial constraints between them in a particular PTF can be generated and evaluated.

#### 4.3.2 Constraint Satisfaction and Propagation Algorithm (CSPA)

CSPA is an algorithm that determines sets of feasible positions for all resources present in a given PTF (PTF-e-f) by satisfying the hard constraints between them. CSPA repeatedly calls MS-GS2D to satisfy selected constraints and propagates the changes in the corresponding resources' SPPs to the SPPs of all other resources in PTF-e-f. Fig. 4.4 shows the flow chart of CSPA.

CSPA goes through the following steps:

1. Let *fixed-set* := {static resources in PTF-e-f} (*static resources*, by definition, have user-defined fixed-positions)  
*remaining-set* := {all other resources in PTF-e-f}.  
 Position resources in *fixed-set*.

2. Initialize the SPP of all resources in remaining-set by satisfying the default in-zone constraint with the site boundaries. If the resulting SPP of any resource is null, go to 9.
3. For each resource  $i$  in remaining-set do the following:
  - a. Let  $in\text{-}zone\text{-}fixed := \{resources in fixed-set that have an in-zone constraint with } i\}$   
 $out\text{-}zone\text{-}fixed := fixed\text{-}set \setminus in\text{-}zone\text{-}fixed$   
 that is, all resources in fixed-set that do not have an in-zone constraint with  $i$ .
  - b. Satisfy the in-zone constraints between  $i$  and resources in  $in\text{-}zone\text{-}fixed$ . If the resulting  $SPP_i = \emptyset$ , go to 9, else update  $SPP_i$ .
  - c. Satisfy the default out-zone constraints between  $i$  and resources in  $out\text{-}zone\text{-}fixed$ . If the resulting  $SPP_i = \emptyset$ , go to 9, else update  $SPP_i$ .
  - d. Satisfy all other hard constraints between  $i$  and resources in  $out\text{-}zone\text{-}fixed$ . If the resulting  $SPP_i = \emptyset$ , go to 9, else update  $SPP_i$ .
4. Let  $hard\text{-}constraints := \{hard constraints that resources in remaining-set share with each other\}$   
 $propagate? := no$ .
5. Let  $C$  be the first constraint in  $hard\text{-}constraints$ ,  $SPP_1$  and  $SPP_2$  the SPPs of the two resources to which  $C$  applies.  
 Satisfy  $C$ . Set  $hard\text{-}constraints := hard\text{-}constraints - \{C\}$ .  
 Let  $SPP'_1$  and  $SPP'_2$  be the modified  $SPP_1$  and  $SPP_2$  respectively as returned by MS-GS2D.
6. a. If  $SPP'_1 = \emptyset$  or  $SPP'_2 = \emptyset$ , then go to 9.  
 b. If  $SPP'_1 \neq SPP_1$  or  $SPP'_2 \neq SPP_2$ , then go to 7, else go to 8.
7. Set  $SPP_1 = SPP'_1$ ,  $SPP_2 = SPP'_2$   
 $propagate? := yes$
8. a. If  $hard\text{-}constraints} = \emptyset$  and  $propagate? = yes$ , then go to 4.  
 b. If  $hard\text{-}constraints} = \emptyset$  and  $propagate? = no$ , then go to 10, else go to 5.
9. The problem is infeasible. No layout for PTF-e-f can be constructed. CSPA terminates unsuccessfully.
10. End. CSPA is successful.

CSPA eliminates resources' positions that will definitely not result in a feasible layout without binding resources to single positions. It may, however, retain more than one acceptable position for each resource in PTF-e-f.

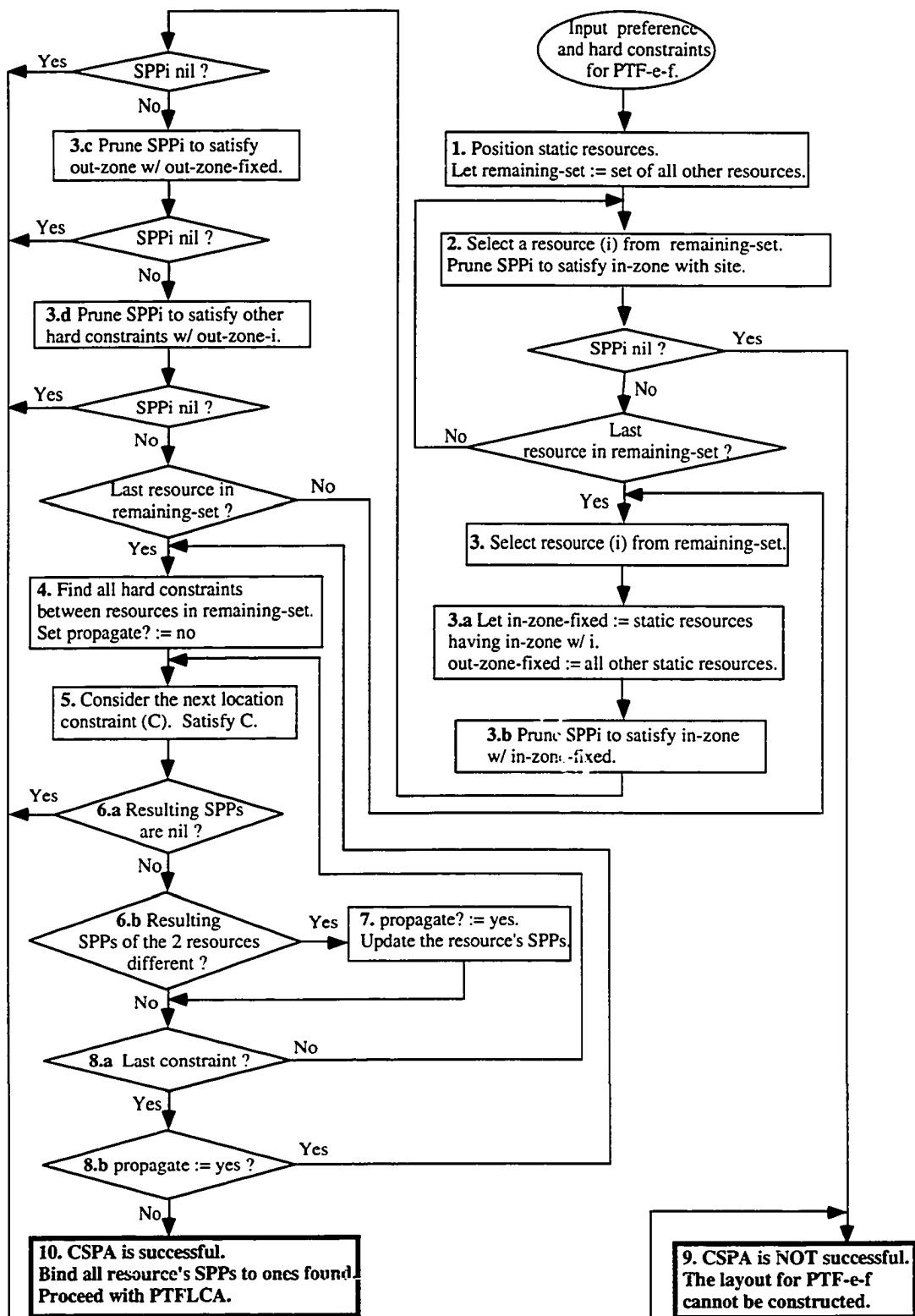


Figure 4.4  
Flow Chart of CSPA

The following sections present a blend of heuristic and optimal methods that start from the SPPs found by CSPA to determine the single best positions for the resources in a given PTF.

### 4.3.3 Optimal Positions to Meet Preference Measures Using Linear Programming

This section solves the isolated static layout problem of finding the optimal position(s) of a resource  $i$  in a layout  $t$  that minimize(s) the relocation cost of  $i$  and proximity costs of  $i$  in  $t$ , where  $i$  could have been positioned in previously constructed layouts. The relocation cost of  $i$  is measured as the product of the relocation weight of  $i$  and the sum of the distances between the position of  $i$  in  $t$  and the positions of  $i$  in previously constructed layouts. These layouts can span time intervals that precede or succeed the time interval that layout  $t$  spans. Proximity costs of  $i$  in  $t$  are measured as the sum of the products of a proximity weight that  $i$  shares with any positioned resource in  $t$  and the distance between  $i$  and that resource weighted over the duration of  $t$ . The position(s) that minimizes relocation and proximity costs of  $i$  should satisfy the hard constraints on the positions of  $i$  in  $t$ .

Let  $t$  be the layout of the  $t^{\text{th}}$  PTF,  $n$  the number of resources that have a known position in  $t$ , and  $i$  the  $(n + 1)$  resource in  $t$  for which a position is sought. Let  $t-1$  and  $t+1$  be the layouts of PTFs preceding and succeeding the  $t^{\text{th}}$  PTF respectively that are closest in time to  $t$  and in which  $i$  has a known position. Let  $SPP_i^t$  be the SPP of  $i$  that satisfies the hard constraints between  $i$  and the  $n$  resources in  $t$ .  $SPP_i^t$  typically comprises the following rectangles:

$$\begin{aligned} SPP_i^t = \{ & \{0 \{([X_{\min-1} X_{\max-1}] [Y_{\min-1} Y_{\max-1}]) \dots \\ & ([X_{\min-m} X_{\max-m}] [Y_{\min-m} Y_{\max-m}]) \dots \} \} \\ & \{90 \{([X_{\min-1} X_{\max-1}] [Y_{\min-1} Y_{\max-1}]) \dots \\ & ([X_{\min-q} X_{\max-q}] [Y_{\min-q} Y_{\max-q}]) \} \} \}. \end{aligned}$$

Each rectangle  $([X_{\min-m} X_{\max-m}] [Y_{\min-m} Y_{\max-m}])$  in turn is denoted by:  $([LB_m UB_m] [LB'_m UB'_m])$ . Given that there are  $m$  rectangles at  $0^\circ$  and  $q$  rectangles at  $90^\circ$ , there will be a total of  $m + q = r$  rectangles to be considered.

The aforementioned problem is denoted by  $Q$  and is formulated as follows:

$$Q \left\{ \begin{array}{l} \text{Min} \sum_{j=1}^n W_{ij}^t \Delta_t \left( |X_i^t - X_j^t| + |Y_i^t - Y_j^t| \right) \\ + W_i \left( |X_i^t - X_i^{t-1}| + |Y_i^t - Y_i^{t-1}| \right) + W_i \left( |X_i^t - X_i^{t+1}| + |Y_i^t - Y_i^{t+1}| \right) \quad (\text{eq. 4}) \\ \text{Such that } \begin{cases} LB_1 \leq X_i^t \leq UB_1 \\ LB'_1 \leq Y_i^t \leq UB'_1 \end{cases} \dots \text{ or } \begin{cases} LB_m \leq X_i^t \leq UB_m \\ LB'_m \leq Y_i^t \leq UB'_m \end{cases} \dots \text{ or } \begin{cases} LB_r \leq X_i^t \leq UB_r \\ LB'_r \leq Y_i^t \leq UB'_r \end{cases} \\ \forall 1 \leq m \leq r: X_i^t, Y_i^t, UB_m, UB'_m, LB_m, LB'_m \geq 0 \end{array} \right.$$

where

$r$  : total number of rectangles in  $SPP_i^t$  in both orientations

$X_i^t, Y_i^t$  : x and y coordinates of the centroid of resource  $i$  in layout  $t$

$UB_m, LB_m$  : upper and lower bound on the x - dimension of rectangle  $m$  in  $SPP_i^t$

$UB'_m, LB'_m$  : upper and lower bound on the y - dimension of rectangle  $m$  in  $SPP_i^t$ .

The objective function in  $Q$  (eq. 4) is the variable part of VFL (eq. 1) that remains after removing all constant terms from it, as the  $P$  and  $R$  components of VFL reduce to a constant for all constructed layouts preceding and succeeding  $t$ .

This formulation of  $Q$  has absolute values in the objective function and has  $r$  sets of disjunctive constraints. Solving  $Q$  requires introducing integer variables to eliminate the absolute values and the disjunction from the constraints. To avoid solving an integer program,  $Q$  is reformulated as  $r$  independent but small linear programs corresponding to the  $r$  sets of disjunctive constraints in  $Q$  (i.e., a linear program for each rectangle  $m$  of  $SPP_i^t$ ). By grouping terms in eq. 4, (including the summation for all  $j$  from 1 to  $n$  plus the two additional terms making a total of  $n+2$  terms) each linear program (denoted by  $Q_m$ ) is of the form

$$Q_m \left\{ \begin{array}{l} Z = \text{Min} \sum_{k=1}^{n+2} C_k |X_i^t - h_k| + C'_k |Y_i^t - h'_k| \\ \text{Such that } LB_m \leq X_i^t \leq UB_m \\ LB'_m \leq Y_i^t \leq UB'_m \\ X_i^t, Y_i^t, UB_m, UB'_m, LB_m, LB'_m \geq 0, \end{array} \right. \quad (\text{eq. 5})$$

where  $\forall k: C_k, C'_k, h_k$ , and  $h'_k$  are positive constants.

The optimal solution to  $Q$  is the solution of the  $Q_m$  with the smallest value of  $Z$ . Because the constraints in  $Q_m$  are single variable constraints and the cost coefficients are all positive,  $Q_m$  is separable. Hence, solving  $Q_m$  for the values of  $X_i^t$  and  $Y_i^t$  is equivalent to solving independently the following two linear programs:

$$Q'_m \left\{ \begin{array}{l} Z' = \text{Min} \sum_{k=1}^{n+2} C_k |X_i^t - h_k| \\ \text{Such that } LB_m \leq X_i^t \leq UB_m \end{array} \right. \quad (\text{eq. 6})$$

$$Q''_m \left\{ \begin{array}{l} Z'' = \text{Min} \sum_{k=1}^{n+2} C'_k |Y_i^t - h'_k| \\ \text{Such that } LB'_m \leq Y_i^t \leq UB'_m \end{array} \right. \quad (\text{eq. 7})$$

Let  $\bar{X}_i^t m$  be the optimal solution to  $Q'_m$  with value  $\bar{Z}'_m$ , and let  $\bar{Y}_i^t m$  be the optimal solution to  $Q''_m$  with value  $\bar{Z}''_m$ , then the optimal solution to  $Q_m$  is  $(\bar{X}_i^t m, \bar{Y}_i^t m)$  with value  $\bar{Z}_m = \bar{Z}'_m + \bar{Z}''_m$ . The optimal solution to  $Q$  is the pair  $(\bar{X}_i^t m, \bar{Y}_i^t m)$  with the smallest value of  $\bar{Z}_m$ . In short, the solution to  $Q$  is obtained by solving  $Q'_m$  and  $Q''_m$  independently for each rectangle  $m$  in  $SPP_i^t$ . The total number of LPs to be solved is twice the number of rectangles describing  $SPP_i^t$  in both  $0^\circ$  and  $90^\circ$  orientation.

For computational and implementation purposes, the structure of  $Q'_m$  and  $Q''_m$  is further investigated for quick and easy solution without resorting to a generic LP problem solver and making the proper variable substitution to remove the absolute values from the objective function. Note that  $Q'_m$  and  $Q''_m$  are single variable LPs with positive cost coefficients ( $C_k$  and  $C'_k$ ). The solution of  $Q'_m$  lies at one of the points  $LB$ ,  $UB$ , or  $h_k$ ; and that of  $Q''_m$  at  $LB'$ ,  $UB'$ , or  $h'_k$ . The above claim is demonstrated for  $Q'_m$  in the case

of  $k=3$  below using geometry. The same proof applies for  $Q''m$  and for any  $k > 0$ . Fig. 4.5 shows a plot of the three terms constituting the objective function (eq. 6) for  $h_2 < h_1 < h_3$  and it assumes some specific values for  $C_1, C_2, C_3$ . These terms combine to form the piece-wise linear convex curve as shown.

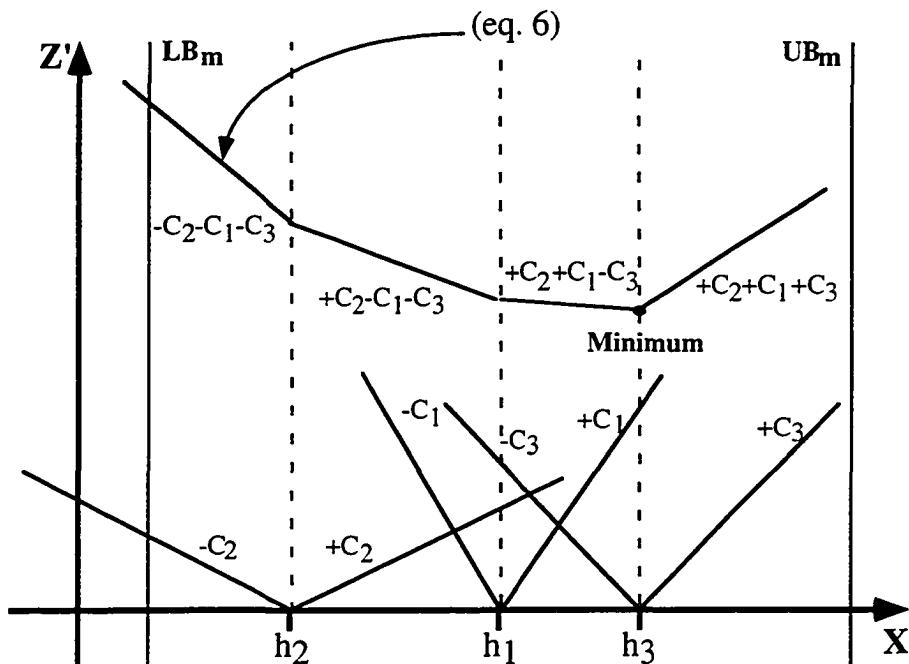


Figure 4.5  
Solution Proof

The solution space is partitioned into regions defined by  $h_1, h_2$ , and  $h_3$  at which a change in slope of eq. 6 is observed. Provided that no upper or lower bound constraints on  $X$  are considered, the value of  $Z'$  is smallest at a single point (because the curve is convex), where the slope changes from negative to positive. This point is referred to as the *unconstrained minimum*. Because the curve is piece-wise linear, a change in slope can only occur at either  $h_1, h_2$ , or  $h_3$ . The single point unconstrained minimum can degenerate to be a line segment when the slope of (eq. 6) is zero in one or more adjacent regions, in which case all points on that line segment represent alternative minima.

The following describes a systematic procedure for finding the unconstrained minimum (minima) of  $Q'm$ . This procedure computes the slope of  $Z'$  in each region as defined by the values of the  $h_k$ s in  $Z'$  and determines at which points the slope of  $Z'$  is zero or changes from negative to positive.

- Arrange the terms in  $Z'$  in increasing value of  $h_k$
- Let  $p$  be the subscript denoting the order of each term in the ordered sequence,  
e.g., the second item in the ordered sequence is denoted by  $C_2 |X_i^t - h_2|$   
 $Sol$  be the set of unconstrained minima.

Procedure:

1. Compute  $S_0 := \sum_{p=1}^{n+1} C_p$
2. If  $S_0 = 0$  then set  $Sol := Sol \cup [-\infty + \infty]$  and go to 9
3. Set  $p := 1$  and  $Sol := \emptyset$
4. Compute  $S_p := 2 C_p + S_{p-1}$
5. If  $S_p > 0$  and  $p \leq n + 2$  then set  $Sol := Sol \cup \{h_p\}$  and go to 9
6. If  $S_p < 0$  and  $p < n + 2$  then set  $p := p + 1$  and go to 4
7. If  $S_p = 0$  and  $p < n + 2$  then alternative optima exist between  $h_p$  and  $h_{p+1}$ .  
Set  $Sol := Sol \cup [h_p h_{p+1}]$ ,  $p := p + 1$ , and go to 4
8. If  $S_p = 0$  and  $p = n + 2$  then set  $Sol := Sol \cup [h_p + \infty]$  and go to 9
9. Stop.

By introducing an upper and lower bound constraint on  $X$ , the feasible solution space will be defined between  $LB_m$  and  $UB_m$ . Depending on the value of  $LB_m$  and  $UB_m$ , the minimum can be  $LB_m$ ,  $UB_m$ ,  $h_1$ ,  $h_2$ , or  $h_3$ . For the case shown in Fig. 4.5, the solution lies at either one of the following points:

- $LB_m$ ; if  $LB_m >$  unconstrained minimum,
- $UB_m$ ; if  $UB_m <$  unconstrained minimum, and otherwise at
- unconstrained minimum (in this case  $h_3$ ).

As a result, one needs to test only the value of  $Z'$  ( $Z''$ ) at these points ( $LB_m$ ;  $UB_m$ ; unconstrained minimum) and choose the point that minimizes it. This formulation results in quick problem solving since the total number of values to be tested is only a multiple of the total number of rectangles describing  $SPP_i^t$  plus a constant. The number of values to be tested is equal to  $4 r + 2$  (i.e.,  $LB$ ,  $UB$ ,  $LB'$ , and  $UB'$  for each rectangle in  $SPP_i^t$  and two unconstrained minima that solve  $Z'$  (eq. 6) and  $Z''$  (eq. 7)).

#### **4.3.4 Heuristics for Resource Selection**

Resources are introduced into the layout one at a time. Their sequencing is based on heuristics. This section reviews different heuristics for resource selection and highlights the heuristic implemented in MoveSchedule.

One way to select a resource is to do it randomly. This has the advantage that local optima, possibly arrived at using any specific heuristic, might be avoided. Other ways are to apply knowledge-based heuristics. One such heuristic is to select resources by decreasing order of their area requirements. This tends to increase the likelihood of coming up with feasible layouts as it usually is easier to fit smaller resources into an existing layout with larger resources already in place than it is to do the opposite. Another heuristic is to select resources by decreasing order of their duration on site. The benefit obtained from processing long term resources first is to guarantee them a good position as they will occupy their position for a longer time period. Yet another heuristic is to prioritize resources based on their relocation cost, so that those with the highest relocation cost are positioned at their preferred position first.

There is no one general-purpose heuristic that best suits any layout problem. What might be a good heuristic for solving one problem may not be so good for another. Though, it may be desirable to make the selection of heuristics a dynamic and real-time process that is based on the specifics of the given problem and on the particular stage of problem solving, doing so is beyond the scope of this research. An objective of this research is to develop a construction-type algorithm that automates dynamic layout construction with simple and general-purpose heuristics.

The heuristic implemented in MoveSchedule favors:

- 1) resources with larger relocation weights. Resources are grouped as being stationary or relocatable. Stationaries are selected first.
- 2) resources with greater interaction with other positioned resources as represented by the sum of their proximity weights.

If more than one resource qualifies (i.e., more than one resource has the same sum of the weights) or if no resource qualifies (i.e., no resource is stationary and no resource is positioned on site) then MoveSchedule offers four tie-breaking rules for the user to choose from:

- 1) select the first resource in the queue
- 2) select a resource at random from the list of candidate resources

- 3) select the resource with the highest relocation weight
- 4) select the resource that has a position in a previously constructed layout.

#### **4.3.5 Primary Time Frame Layout Construction Algorithm (PTFLCA)**

This section presents a heuristic algorithm (PTFLCA) for constructing the layout of an individual PTF (PTF-e-f). Following CSPA, which determines all feasible positions of the resources in PTF-e-f subject to hard constraints among them, PTFLCA starts from these feasible positions to position the resources in PTF-e-f one at a time in the layout. The algorithm selects resources according to the heuristic described in section 4.3.4. PTFLCA pursues a single position for each selected resource before moving on to the next resource. To determine a single position, PTFLCA first updates the resource's SPP (as determined by CSPA) to meet hard constraints with resources that have already been positioned in the layout. Then, it determines the position(s) that minimizes proximity costs with positioned resources and the resource's relocation costs according to the method described in section 4.3.3. When more than one position is found, PTFLCA selects at random a single position and binds the resource at that position in PTF-e-f.

This process of selecting a resource and finding a position for it will yield—provided that all resources can be positioned—a single layout. To generate alternative solutions, PTFLCA must repeat its process, each time starting with a different sequence of resources or with different randomly selected positions. Hence, to construct the layout of PTF-e-f, the following PTFLCA is run X number of trials (X can be set to any number) to generate at most X sets of compatible positions for the resources present in PTF-e-f. These sets are then evaluated using the equation for VFL, and the one that minimizes it is the solution. Fig. 4.6 shows a flow chart of PTFLCA.

The algorithm goes through the following steps to construct the layout of PTF-e-f:

1. Set *counter* := 0; this variable keeps track of the number of sampled sets,  
 $*best\text{-}VFL* := \infty$ ; this variable is the part of VFL pertaining to the layout of PTF-e-f,  
 $*best\text{-}sol* := \emptyset$ ; this variable is the solution of resource positions corresponding to the value of  $*best\text{-}VFL*$ .
2. Set counter := counter + 1  
 $temporary\text{-}set := \{all\ resources\ needed\ in\ PTF\text{-}e\text{-}f\}$ ,

*positioned-set* :=  $\emptyset$ ; this variable is the set of resources that have already been positioned in PTF-e-f

*active-group* :=  $\emptyset$ ; this variable is the set of resources that have not yet been positioned in PTF-e-f.

3. Position static resources in temporary-set. Position previously-positioned stationary resources in temporary-set. Remove these resources from temporary-set and add them to positioned-set.
4. Group the remaining resources in temporary-set into *stationaries* and *relocatables*.
5. If *stationaries* =  $\emptyset$ , then set *active-group* := *relocatables* and *relocatables* :=  $\emptyset$ .  
If *stationaries*  $\neq \emptyset$ , then set *active-group* := *stationaries* and *stationaries* :=  $\emptyset$ .
6. Select from active-group the resource (*i*) that has the highest sum of proximity weights with the resources in positioned-set. Use the default tie breaking rule, unless chosen otherwise by the user, to break ties among candidate resources.
7. Prune SPP<sub>i</sub> (obtained at the end of CSPA) to satisfy in this order: all in-zone constraints with resources in positioned-set, the default out-zone, and all other location constraints with the remaining resources in positioned-set using Constraint Satisfaction (see section 4.3.1).
  - a. If SPP<sub>i</sub> =  $\emptyset$ , then go to 12, else go to 8.
8. Further prune SPP<sub>i</sub> to minimize the increase in VFL ( $\Delta$ VFL) caused by introducing *i* into the layout using the LP formulation of section 4.3.3.
  - a. If the resulting SPP is a singleton, then go to 8.c, else go to 8.b.
  - b. Sample at random a position for *i* from SPP<sub>i</sub>.
  - c. Bind the position of *i* to the single position found.  
Set active-group := active-group - {*i*} and  
positioned-set := positioned-set  $\cup$  {*i*}.
9. a. If active-group  $\neq \emptyset$ , then go to 6.  
b. If relocatables  $\neq \emptyset$ , then go to 5, else go to 10.
10. If VFL for this layout (i.e., with the resource positions found for all resources in PTF-e-f)  $\geq$  \*best-VFL\*, then go to 12, else go to 11.
11. set \*best-VFL\* := value of VFL for this layout and  
\*best-sol\* := {positions found for the resource in PTF-e-f}.
12. a. If counter < X then go to 2.  
b. If \*best-VFL\* =  $\infty$  and \*best-sol\* =  $\emptyset$ , then go to 13, else go to 14.
13. No one set has sampled positions for all resources. PTF-e-f layout construction terminates with no solution found.

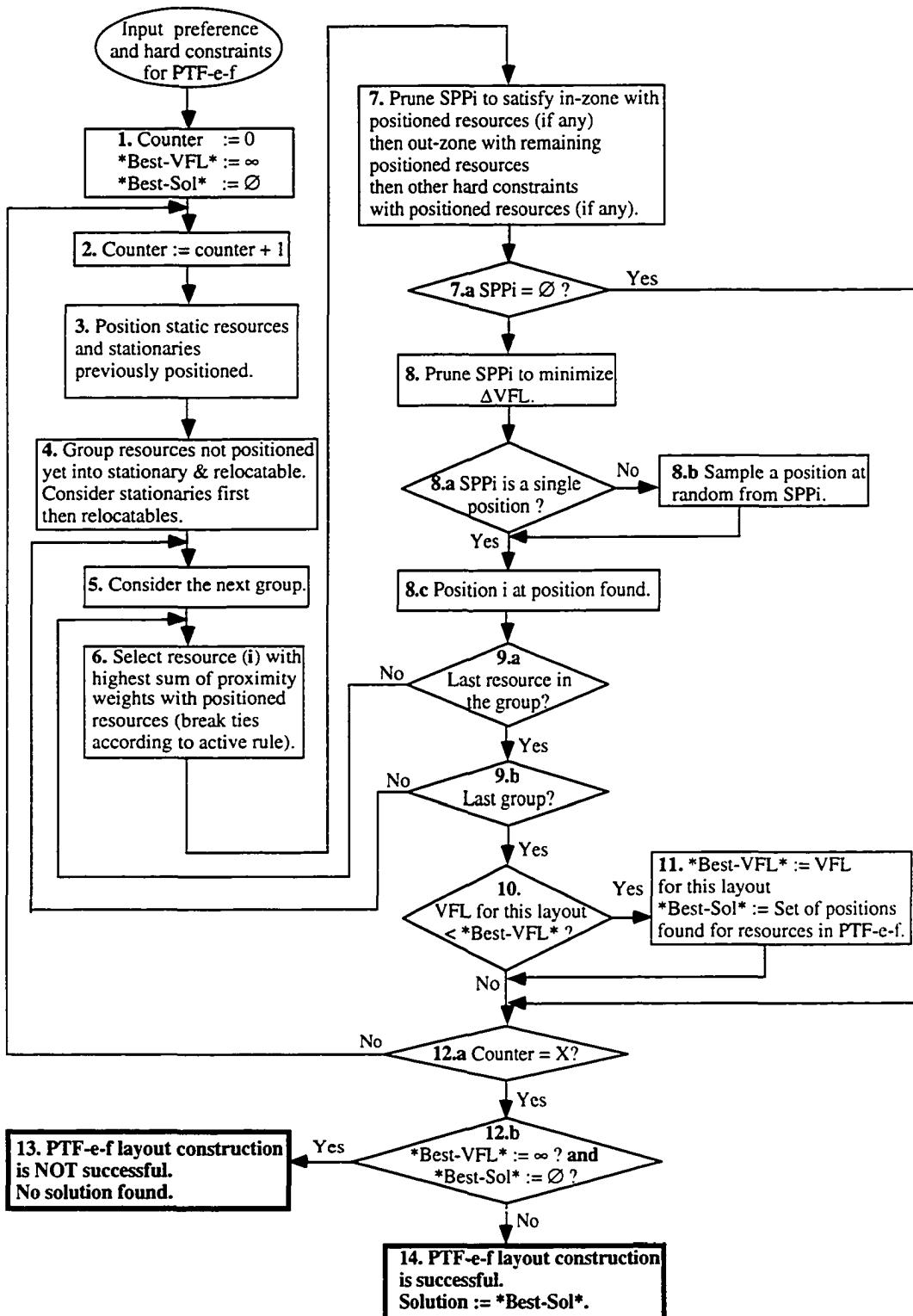


Figure 4.6  
Flow Chart of PTFLCA

14. Bind the positions of the resources in PTF-e-f to the positions in \*best-sol\*.  
PTF-e-f layout construction terminates successfully.

## 4.4 Dynamic Layout Construction Strategy

Section 2.1 rationalized how field practitioners design site layouts over time and concluded that their strategy is worth implementing in MoveSchedule. Hence, MoveSchedule's dynamic layout strategy is to construct the layouts of PTFs in a time sequence paralleling construction time.

The strategy iterates through the following steps until the layouts for all PTFs are constructed:

- Select the PTF that has the earliest start date.
- Construct the layout for this PTF by applying CSPA first, followed by PTFLCA.
- If the layout construction is successful, then save the resources' positions for this PTF and repeat these steps for the subsequent PTF, else stop dynamic layout construction and declare the problem infeasible.

## 4.5 Example Application

The following example illustrates how the layout of a PTF is constructed using CSPA and PTFLCA. The example is made small so that a step-by-step description of the algorithm can be presented. The number of trials in PTFLCA is set to 5 in this example.

Consider a project that takes 4 days and requires 7 resources. The layout of these resources needs to be planned on a 20 x 10 site area. Table 4.1 summarizes the input required for resources. Both R-2 and R-5 have positions defined by the user for the duration of their existence on site; all other resources need to be positioned.

Because proximity weights and hard constraints may vary from one PTF to another, MoveSchedule first identifies the PTFs based on the aforementioned input. Two PTFs are found in this case: PTF-0-2 and PTF-2-4. The resources needed in PTF-0-2 are R-1, R-2, R-4, and R-5; those needed in PTF-2-4 are R-1, R-3, R-4, R-6 and R-7.

The user then inputs the hard constraints and proximity weights. Assume that a minimum distance of 8 in the x-direction between R-3 and R-1 should be maintained at all

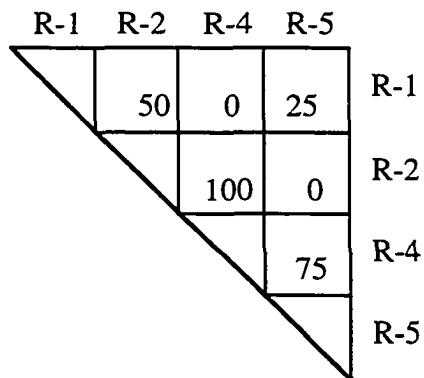
times. The proximity weights for both time frames are shown in Fig. 4.7. The default proximity weight of 0 indicates that there is no interaction between the two resources.

Resource	Dimensions L x W	Time on Site	Relocation Weight	Fixed Position [X, Y, Orientation]
R-1	8 x 8	0 -> 4	75	none
R-2	2 x 1	0 -> 2	0	[16, 8.5, 0]
R-3	2.8 x 2.8	2 -> 4	50	none
R-4	4 x 2	0 -> 4	75	none
R-5	4 x 2	0 -> 2	0	[11, 6, 90]
R-6	4 x 3	2 -> 4	75	none
R-7	4 x 2	2 -> 4	50	none

Table 4.1  
Example Input of Resources

Following the identification of PTFs, MoveSchedule proceeds to construct a layout for each PTF individually. PTFs are processed in chronological order, i.e., the layout for PTF-0-2 is constructed first, followed by that for PTF-2-4. Only the layout construction of PTF-2-4 is detailed in this section, to demonstrate how the layout construction of a PTF can be constrained by the layouts created for preceding PTFs.

**PTF-0-2**



**PTF-2-4**

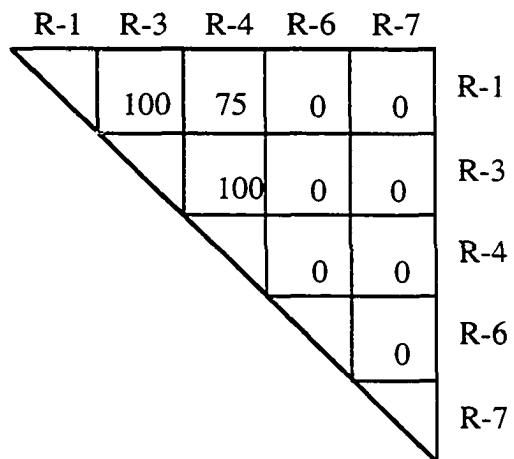


Figure 4.7  
Example Input of Proximity Weights

### Layout construction of PTF-0-2

Assume that the layout construction of PTF-0-2 resulted in the layout shown in Fig. 4.8.

The value of VFL at this stage of the solution is

$$\begin{aligned} P[0-2] = 2 \{ & ; \text{duration of PTF-0-2} \\ 50 (|16 - 16| + |4 - 8.5|) & ; \text{proximity weight x distance between R-2 and R-1} \\ + 25 (|16 - 11| + |4 - 6|) & ; \text{proximity weight x distance between R-1 and R-5} \\ + 100 (|16 - 13| + |8.5 - 9|) & ; \text{proximity weight x distance between R-2 and R-4} \\ + 75 (|13 - 11| + |9 - 6|) \} & ; \text{proximity weight x distance between R-4 and R-5} \\ = 2,250 & \end{aligned}$$

R[], [0-2] is undefined for this layout since there exists no preceding PTF

VFL = P[0-2] = 2,250.

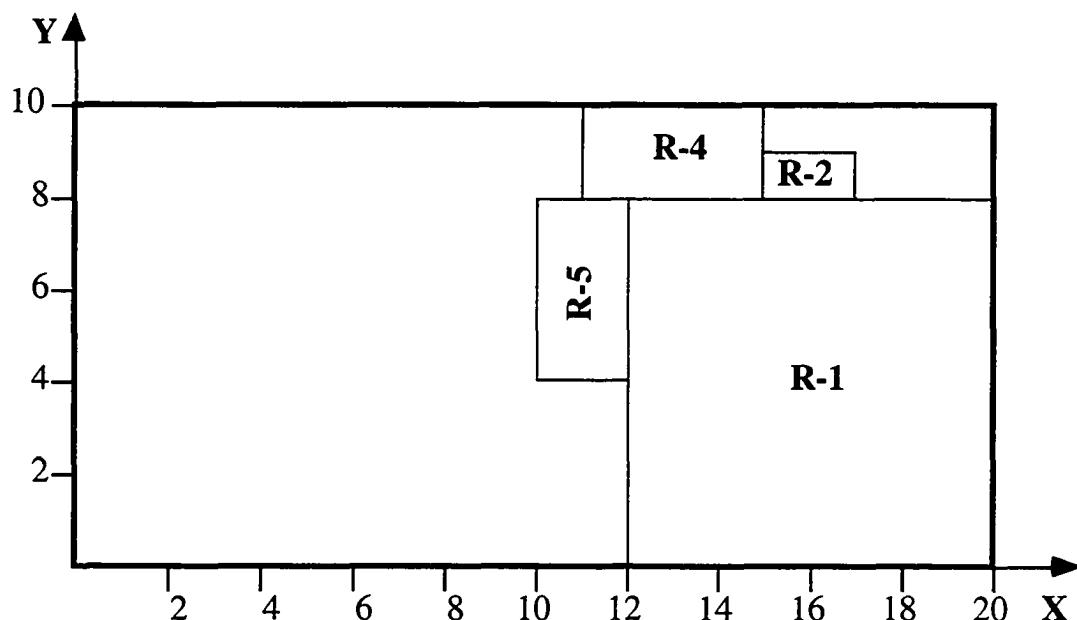


Figure 4.8  
Layout of PTF-0-2

### Layout Construction of PTF-2-4

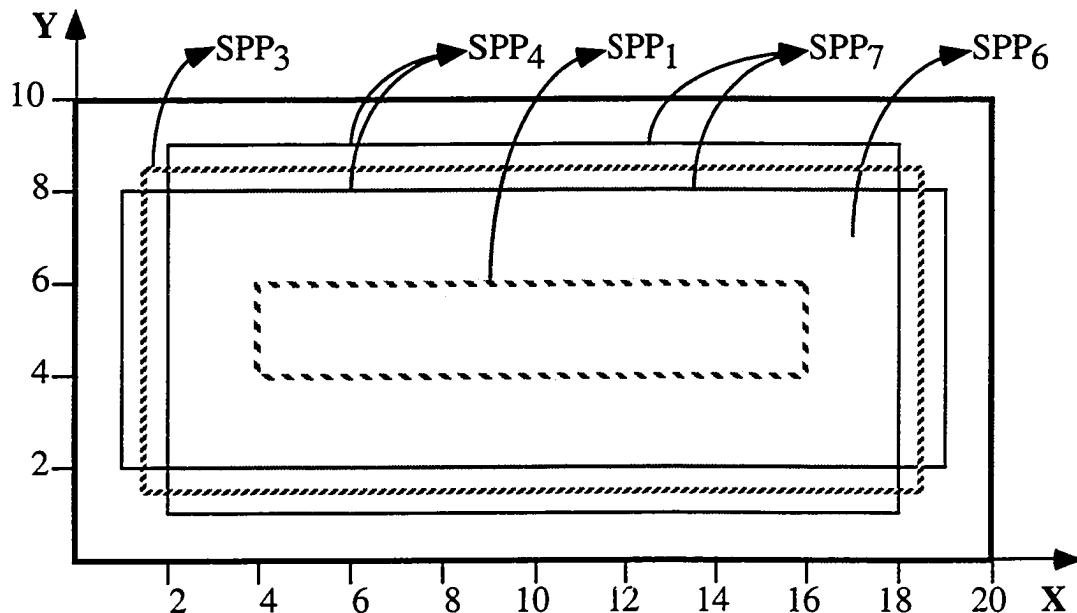
CSPA is called first to initialize resources' SPPs to satisfy the in-zone constraints with the site boundaries. The resulting SPPs are as shown in Fig. 4.9.

Then, CSPA identifies static and previously-positioned stationary resources. None exist in this PTF; hence, CSPA does not need to check for in-zone and default out-zone

constraints between those and other resources. Last, CSPA identifies all hard (other than in-zone and out-zone) constraints that exist between resources in PTF-2-4. It finds a single minimum distance constraint of 8 in the x-direction between R-1 and R-3. Satisfying the constraint further prunes SPP<sub>1</sub> and SPP<sub>3</sub> to the SPPs shown in Fig. 4.10.

Constraint propagation is not needed in this case because the constraint network consists of a single constraint. Thus, the SPPs of the remaining resources in PTF-2-4 are not affected.

PTFLCA is run next with the above SPPs as initial SPPs for the resources in PTF-2-4. From this stage onward, the following steps will be repeated at each trial, for a total of 5 trials. The computations involved in only one trial are detailed here.



Legend:

```

SPP1 = { {0 {[4.0 16.0] [4.0 6.0])} } {90 {[4.0 16.0] [4.0 6.0])} } }
SPP3 = { {0 {[1.4 18.6] [1.4 8.6])} } {90 {[1.4 18.6] [1.4 8.6])} } }
SPP4 = { {0 {[2.0 18.0] [1.0 9.0])} } {90 {[1.0 19.0] [2.0 8.0])} } }
SPP6 = { {0 {[2.0 18.0] [1.5 8.5])} } {90 {[1.5 18.5] [2.0 8.0])} } }
SPP7 = { {0 {[2.0 18.0] [1.0 9.0])} } {90 {[1.0 19.0] [2.0 8.0])} } }

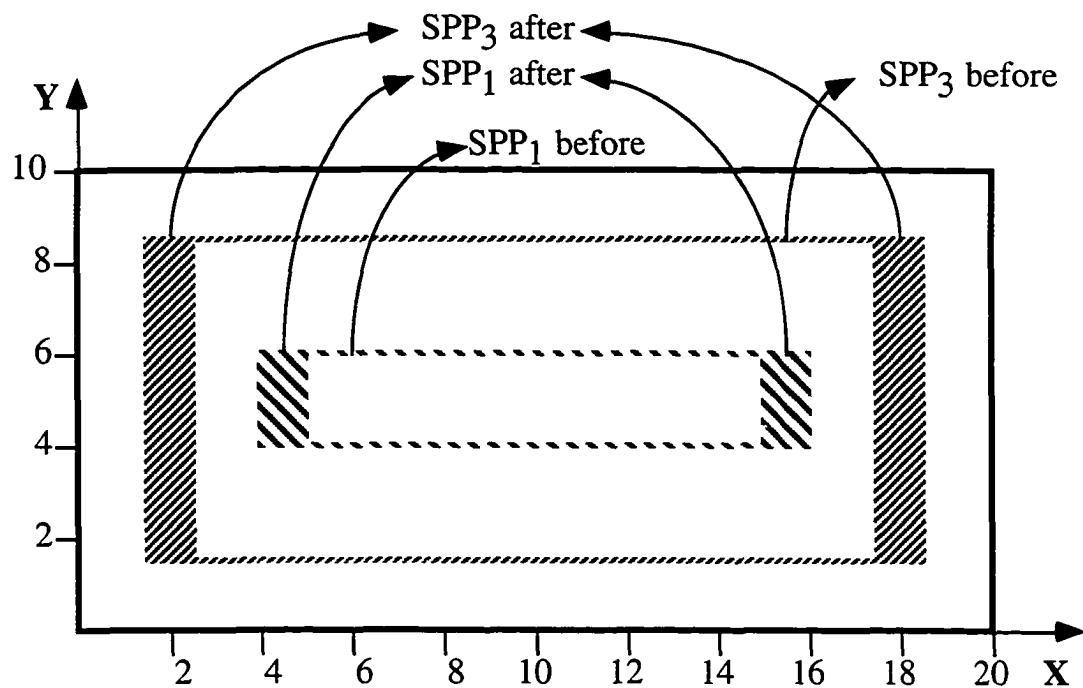
```

Figure 4.9  
SPPs Satisfying In-zone with Site Boundaries

PTFLCA groups the resources in PTF-2-4 into stationaries and others, and positions stationaries first. Since none of the resources in this example are stationary, the

second group consists of all resources in PTF-2-4. The resource that has the highest sum of proximity weights with the positioned resources in PTF-2-4 is selected first. Since none has been positioned yet and there are no static or previously-positioned stationary resources in this frame, PTFLCA selects one by firing the default tie-breaking rule (which is choosing a resource at random).

PTFLCA randomly selects R-1 from the list of candidate resources: {R-1, R-3, R-4, R-6, R-7}. SPP<sub>1</sub>, obtained after Constraint Satisfaction (shown in Fig. 4.10), does not need updating since no resource has been positioned yet in PTF-2-4. The set of positions of R-1 that minimizes proximity costs with positioned resources and R-1's relocation costs are determined next. Positioning R-1 does not affect the value of P since no resource has been positioned yet; it only increases the R component of VFL since R-1 has been positioned in the preceding PTF, PTF-0-2.



Legend:

SPP<sub>1</sub> = { {0 {[4.0 5.2] [4.0 6.0]} {[14.8 16.0] [4.0 6.0]} } }  
{90 {[4.0 5.2] [4.0 6.0]} {[14.8 16.0] [4.0 6.0]} } }

SPP<sub>3</sub> = { {0 {[1.4 2.6] [1.4 8.6]} {[17.4 18.6] [1.4 8.6]} } }  
{90 {[1.4 2.6] [1.4 8.6]} {[17.4 18.6] [1.4 8.6]} } }

Figure 4.10  
SPP<sub>1</sub> and SPP<sub>3</sub> before and after Applying the Minimum Distance Constraint

Thus the increase in VFL is  $\Delta VFL = 75 (|16 - X_1| + |4 - Y_1|)$ . The values of  $X_1$  and  $Y_1$  that minimize  $\Delta VFL$  subject to  $\{X_1, Y_1\} \in SPP_1$  are  $X_1 = 16$  and  $Y_1 = 4$  @ 0 or  $90^\circ$  (see Fig. 4.11) with  $\Delta VFL = 0$  and  $VFL = 2250$ . That is, R-1 should remain in its previous position since that is a feasible position in the current layout as well. When the optimal position has two possible orientations and has coordinates equal to those of the resource's position in a previous layout, PTFLCA maintains the orientation from the previous layout to avoid rotating the resource (in this case, however, it does not make a difference because R-1 is square). PTFLCA does not favor the relocation of a resource unless a change in its level of interaction with other resources justifies its relocation.

The list of candidate resources now reads  $\{R-3, R-4, R-6, R-7\}$ . R-3 is selected next because it has the highest proximity weight  $W_{i1}^{2-4}$  among R-4, R-6 and R-7. PTFLCA first updates  $SPP_3$  (see Fig. 4.10) to reflect the positioning of R-1 by reapplying the minimum distance constraint. This reduces  $SPP_3$  to the cross-hatched rectangle shown in Fig. 4.12.

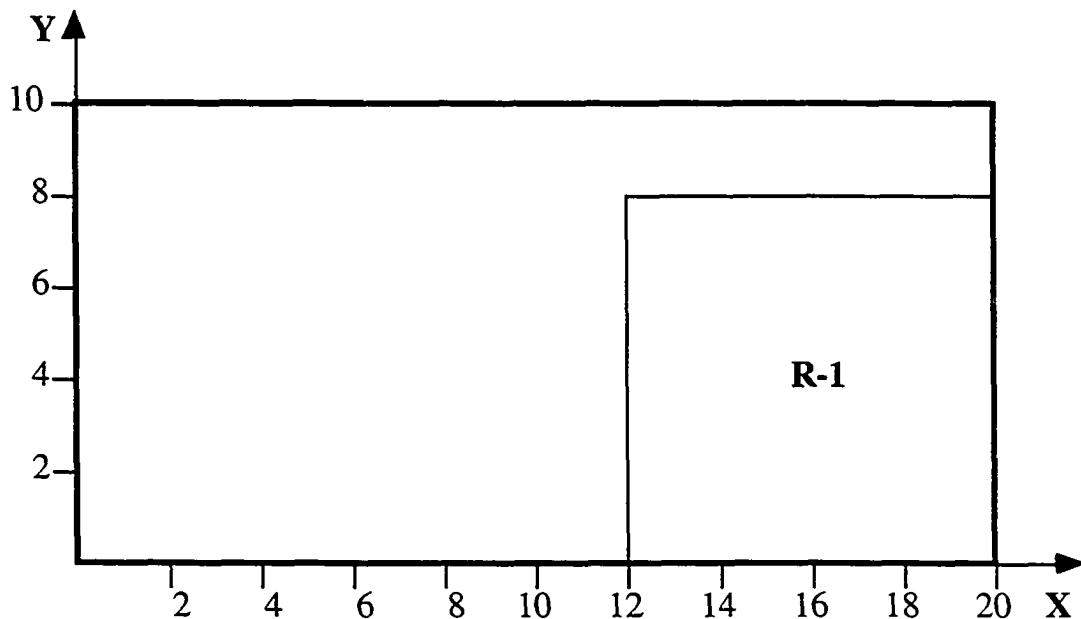


Figure 4.11  
Partial Layout of PTF-2-4 after Positioning R-1

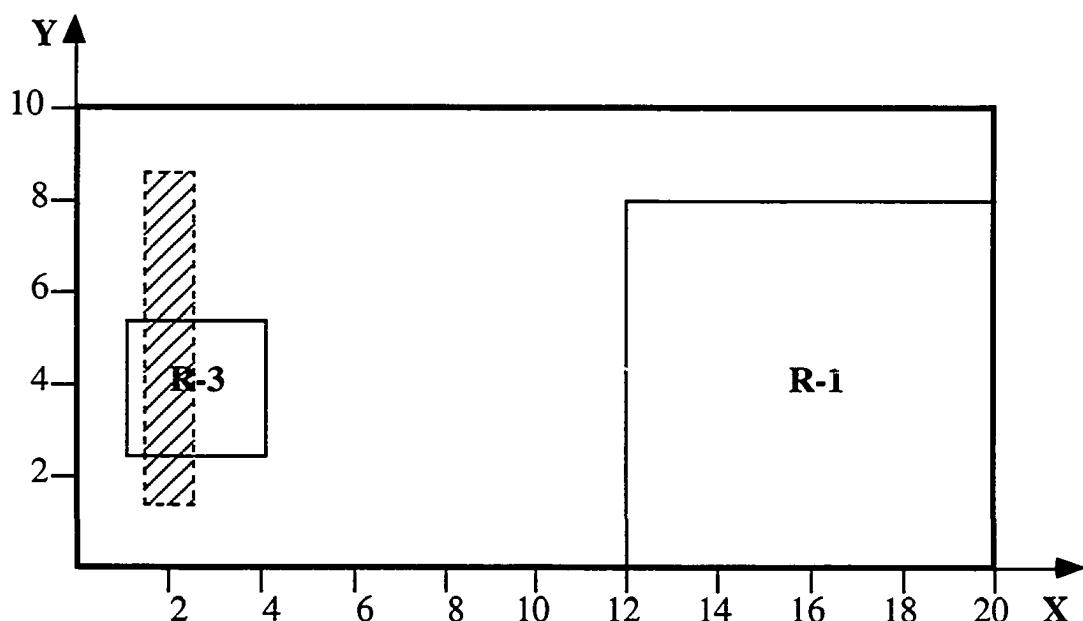
PTFLCA, then, further prunes  $SPP_3$  to minimize the increase in VFL. The problem is formulated as follows:

$$\left\{
 \begin{array}{ll}
 \text{Min } \Delta VFL = 2 & ; \text{duration of PTF-2-4} \\
 \times 100 (|16 - X_3| + |4 - Y_3|) & ; \text{proximity cost between R-3 and R-1} \\
 \text{Such that} & 1.4 \leq X_3 \leq 2.6 \\
 & 1.4 \leq Y_3 \leq 8.6 \\
 & \text{where } X_3, Y_3 \geq 0
 \end{array}
 \right.$$

As demonstrated in section 4.3.3, the above program is separable into two LPs with  $X_3$  and  $Y_3$  as respective variables. Solving the two LPs results in

$X_3 = 2.6$ ,  $Y_3 = 4$  (see the actual position of R-3 in Fig. 4.12), which yields  
 $\Delta VFL = 2,680$ , and  
 $VFL = 2,250 + 2,680 = 4,930$ .

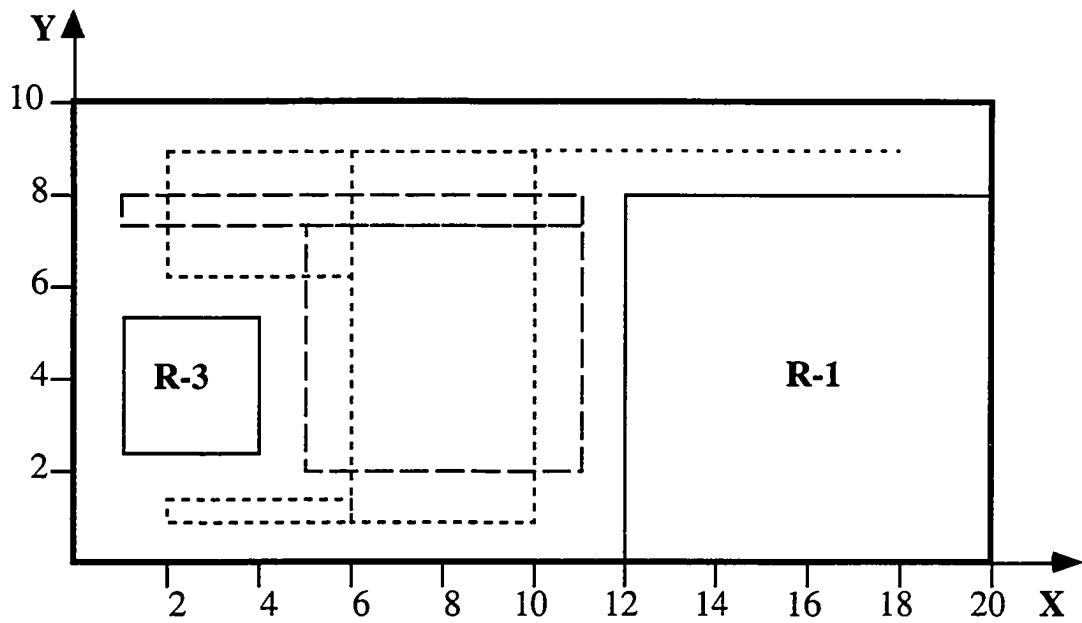
The next resource with the highest sum of proximity weights  $W_{i1}^{2-4} + W_{i3}^{2-4}$  with R-1 and R-3 is R-4, totaling 175. Pruning SPP4 (from Fig. 9) to satisfy the default out-zone constraints with R-1 and R-3 results in the rectangles shown in Fig. 4.13.



Legend:

SPP3 = { {0 { ([1.4 2.6] [1.4 8.6]) }} {90 { ([1.4 2.6] [1.4 8.6]) }} }.

Figure 4.12  
Partial Layout of PTF-2-4 after Positioning R-3



Legend:

[---] SPP4 @ 0°      [—] SPP4 @ 90°

$$\begin{aligned} \text{SPP4} = & \{0 \{ ([2 \ 6] [6.4 \ 9]) ([6 \ 10] [1 \ 9]) ([2 \ 6] [1 \ 1.6]) ([10 \ 18] [9 \ 9])\} \} \\ & \{90 \{ ([1 \ 11] [7.4 \ 8]) ([5 \ 11] [2 \ 7.4])\}\} \end{aligned}$$

Figure 4.13  
Partial Layout of PTF-2-4 Showing SPP4

The problem of minimizing  $\Delta VFL$  to minimize proximity and relocation costs of R-4 can now be formulated as follows:

$$\left\{ \begin{array}{l} \text{Min } \Delta VFL = 2 \{ \quad ; \text{duration of PTF-2-4} \\ \quad 100(|2.6 - X_4| + |4 - Y_4|) \quad ; \text{proximity cost between R-4 and R-3} \\ \quad + 75(|16 - X_4| + |4 - Y_4|) \quad ; \text{proximity cost between R-4 and R-1} \\ \quad + 75(|13 - X_4| + |9 - Y_4|) \quad ; \text{relocation cost of R-4} \\ \\ \text{Such that } \begin{array}{lll} \textcircled{1} \left\{ \begin{array}{l} 2 \leq X_4 \leq 6 \\ 6.4 \leq Y_4 \leq 9 \end{array} \right. & \text{or} & \textcircled{2} \left\{ \begin{array}{l} 6 \leq X_4 \leq 10 \\ 1 \leq Y_4 \leq 9 \end{array} \right. & \text{or} & \textcircled{3} \left\{ \begin{array}{l} 2 \leq X_4 \leq 6 \\ 1 \leq Y_4 \leq 1.6 \end{array} \right. \\ \textcircled{4} \left\{ \begin{array}{l} 10 \leq X_4 \leq 18 \\ 9 \leq Y_4 \leq 9 \end{array} \right. & \text{or} & \textcircled{5} \left\{ \begin{array}{l} 1 \leq X_4 \leq 11 \\ 7.4 \leq Y_4 \leq 8 \end{array} \right. & \text{or} & \textcircled{6} \left\{ \begin{array}{l} 5 \leq X_4 \leq 11 \\ 2 \leq Y_4 \leq 7.4 \end{array} \right. \end{array} \end{array} \right.$$

where the sets of constraints (1), (2), (3), and (4) describe SPP4 at  $0^\circ$  and the sets (5) and (6) describe SPP4 at  $90^\circ$  orientation.

This program is solved for the value of  $X_4$  and  $Y_4$  as follows:

- Find the unconstrained minimum, i.e., the solution that minimizes  $\Delta VFL$  before any constraint is considered, by solving the following equations (see eq. 6 & 7):  

$$\Delta VFL = Z' + Z''$$

$$Z' = 200 |X_4 - 2.6| + 75 |X_4 - 13| + 150 |X_4 - 16|$$

$$Z'' = 200 |Y_4 - 4| + 150 |Y_4 - 4| + 75 |Y_4 - 9|$$

The solution is obtained by solving  $Z'$  and  $Z''$  separately for the value of  $X_4$  and  $Y_4$  using the procedure described in section 4.3.3.  $\Delta VFL$  is minimum at the points where the slope of the objective functions,  $Z'$  and  $Z''$ , becomes positive, i.e., at  $X^* = 13$  (see Fig. 4.14 (a)) and  $Y^* = 4$  (see Fig. 4.14 (b)).

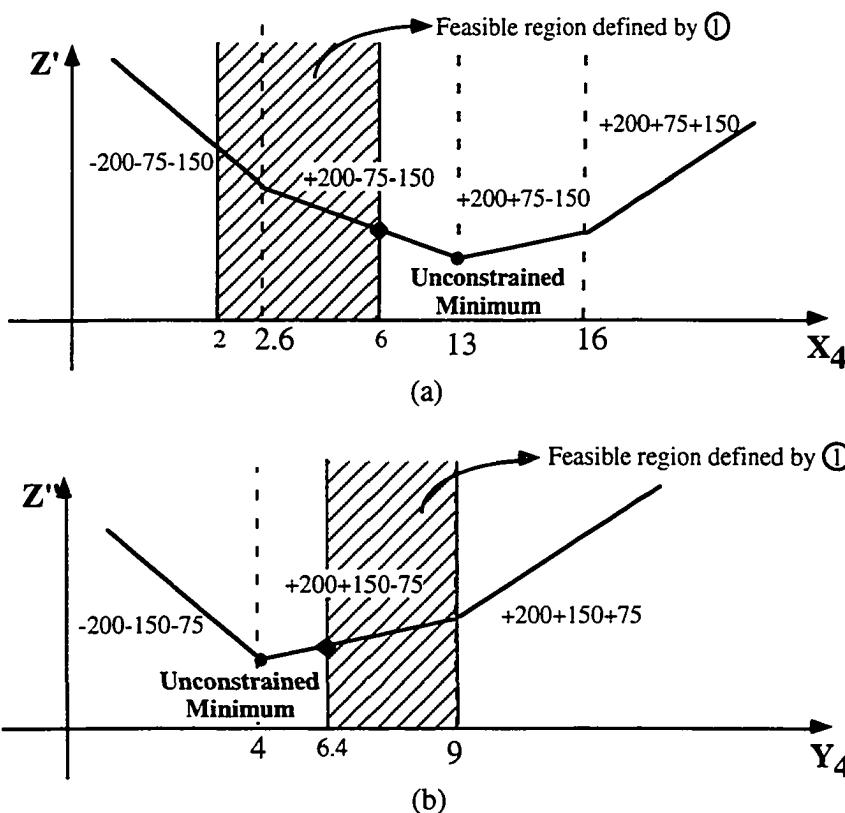


Figure 4.14  
Graphical Solution for  $X_4$  and  $Y_4$

- Find the solution that minimizes  $\Delta VFL$  subject to each set of constraints by comparing the upper and lower bound values on the variables  $X_4$  and  $Y_4$  to the unconstrained minimum ( $X^*$  and  $Y^*$ ). The solution that minimizes  $\Delta VFL$

subject to (1) is  $X_4(1) = 7$  (see Fig. 4.14 (a)) and  $Y_4(1) = 5.5$  (see Fig. 4.14 (b)) with  $\Delta VFL = 3,740$ . Table 4.2 summarizes the solution to  $\Delta VFL$  subject to each of the six sets of constraints.

Constraint Set #	(1)	(2)	(3)	(4)	(5)	(6)
<b>X<sub>4</sub></b>	6	10	6	13	11	<b>11</b>
<b>Y<sub>4</sub></b>	6.4	4	1.6	9	7.4	<b>4</b>
<b>ΔVFL</b>	3,740	2,980	4,100	4,280	3,890	<b>2,955</b>

Table 4.2  
Solution Summary

Hence, the solution that minimizes  $\Delta VFL$  is  $X_4 = 11$  and  $Y_4 = 4$  at  $90^\circ$  orientation (see Fig. 4.15) with  $VFL = 2,250 + 2,680 + 2,955 = 7,855$ .

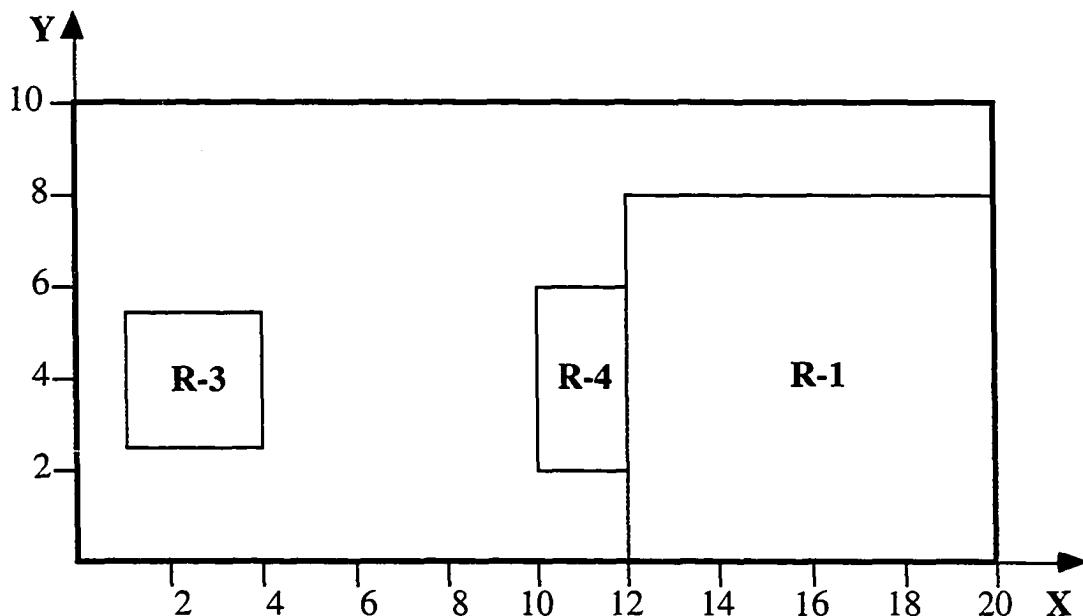


Figure 4.15  
Partial Layout of PTF-2-4 after Positioning R-4

The remaining resources to be positioned are R-6 and R-7. Neither resource has a proximity weight with any of the positioned resources. Thus, both resources are candidates for selection. The default tie breaking rule randomly selects R-6. SPP6 is pruned to satisfy the default out-zone constraints with the positioned resources thus far:

R-1, R-3, and R-4. Since there is no interaction between R-6 and the positioned resources, an instance position for R-6 is sampled at random from its SPP. The sampled position is at  $X_6 = 7.6$  and  $Y_6 = 4.2$  at  $0^\circ$  orientation (see Fig. 4.16).

Similarly, SPP<sub>7</sub> is computed and an instance position for R-7 is sampled at random. It is found at  $X_7 = 9.6$  and  $Y_7 = 8.6$  at  $0^\circ$  (see Fig. 4.17). The value of VFL remains unchanged after positioning R-6 and R-7 because neither one of these resources appeared in the preceding layout and neither one interacts with resources in the current layout. This concludes one cycle of PTF-2-4 layout construction.

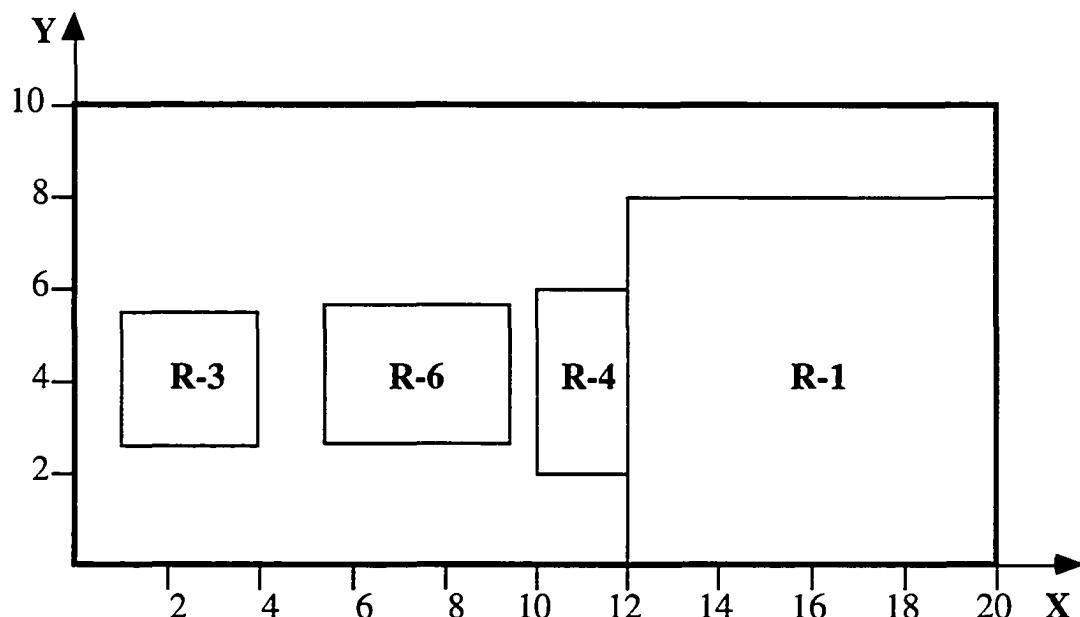


Figure 4.16  
Partial Layout of PTF-2-4 after Positioning R-6

The PTF layout construction algorithm is run a total of 5 trials to generate alternative layout solutions for PTF-2-4. Table 4.3 summarizes the results. The order in which resource positions are presented in the third column of Table 4.3 corresponds to the order in which they are introduced and positioned in the layout. Note that the starting point (i.e., the first resource selected for positioning) and the order in which they are positioned changes in the different trials. This variation is the result of randomly breaking ties among resources and randomly sampling a position for a resource when alternative positions minimize  $\Delta VFL$ .

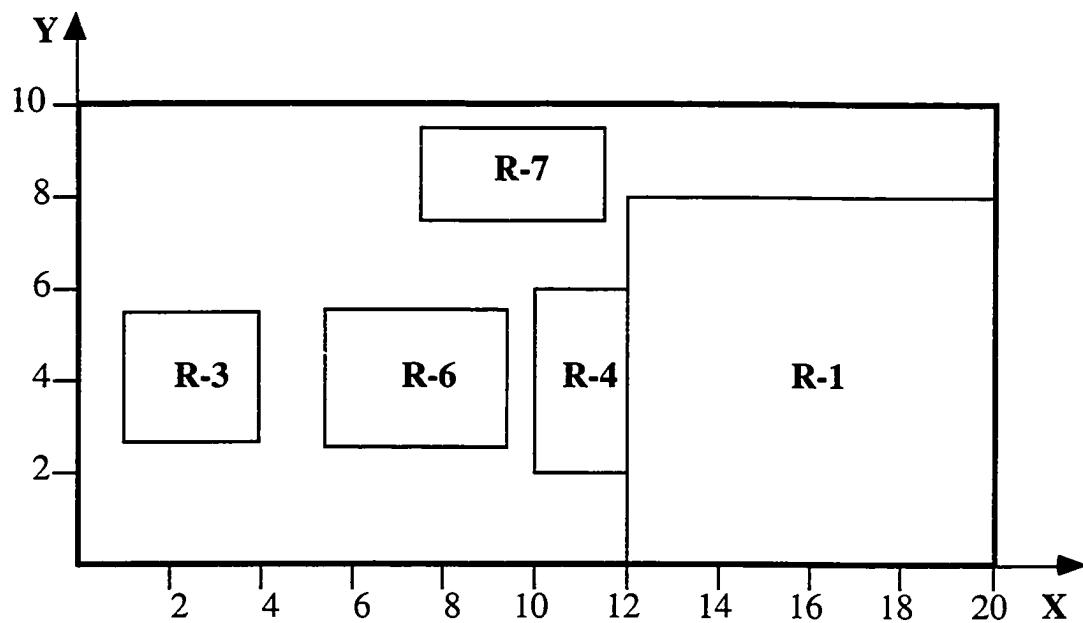


Figure 4.17  
Final Layout of PTF-2-4 from First Trial

The different layouts obtained in the 5 runs are compared based on the value of VFL, and the layout with the smallest value of VFL is chosen. Only 4 out of 5 runs resulted in a successful layout construction. Two alternative layouts were found with equal minimum value for  $VFL = 7,885$  as shown in Fig. 4.18 (a) and (b).

By varying the tie breaking rule, other solutions can be generated. If the second or third tie breaking rule had been chosen, then the sequence in which resources are selected would have been the same in all trials, and variations in the layout solutions would result only from the random sampling of the positions of R-6 and R-7.

Trial #	Success	Summary of Positions	VFL Value
1	Yes	$X_1 = 16, Y_1 = 4 @ 0^\circ$ $X_3 = 2.6, Y_3 = 4 @ 0^\circ$ $X_4 = 11, Y_4 = 4 @ 90^\circ$ $X_6 = 7.6, Y_6 = 4.2 @ 0^\circ$ $X_7 = 9.6, Y_7 = 8.6 @ 0^\circ$	7,885
2	Yes	$X_4 = 13, Y_4 = 9 @ 0^\circ$ $X_3 = 17.4, Y_3 = 8.6 @ 0^\circ$ $X_1 = 4, Y_1 = 6 @ 90^\circ$ $X_6 = 9.5, Y_6 = 6 @ 90^\circ$ $X_7 = 11.6, Y_7 = 1.4 @ 0^\circ$	9,260
3	Yes	$X_3 = 18.2, Y_3 = 1.5 @ 90^\circ$ $X_4 = 18, Y_4 = 3.9 @ 0^\circ$ $X_1 = 4.8, Y_1 = 4 @ 0^\circ$ $X_7 = 10.9, Y_7 = 8.7 @ 0^\circ$ $X_6 = 13.3, Y_6 = 4.6 @ 0^\circ$	9,546.3
4	No	$X_6 = 9.9, Y_6 = 3.3 @ 0^\circ$ $X_4 = 13, Y_4 = 9 @ 0^\circ$ $X_3 = 17.4, Y_3 = 8.6 @ 90^\circ$ <b>No feasible position for R-1</b>	N/A
5	Yes	$X_1 = 16, Y_1 = 4 @ 0^\circ$ $X_3 = 2.6, Y_3 = 4 @ 0^\circ$ $X_4 = 11, Y_4 = 4 @ 90^\circ$ $X_7 = 6.1, Y_7 = 7 @ 90^\circ$ $X_6 = 2.2, Y_6 = 7.7 @ 90^\circ$	7,885

Table 4.3  
Summary of Alternative Layout Solutions with CSPA

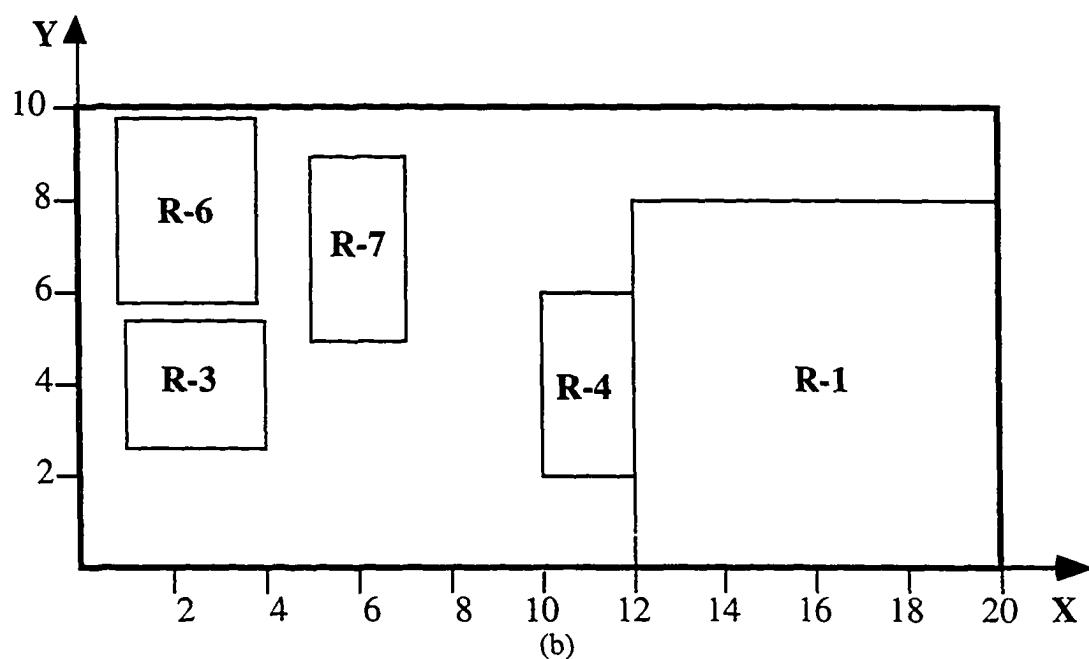
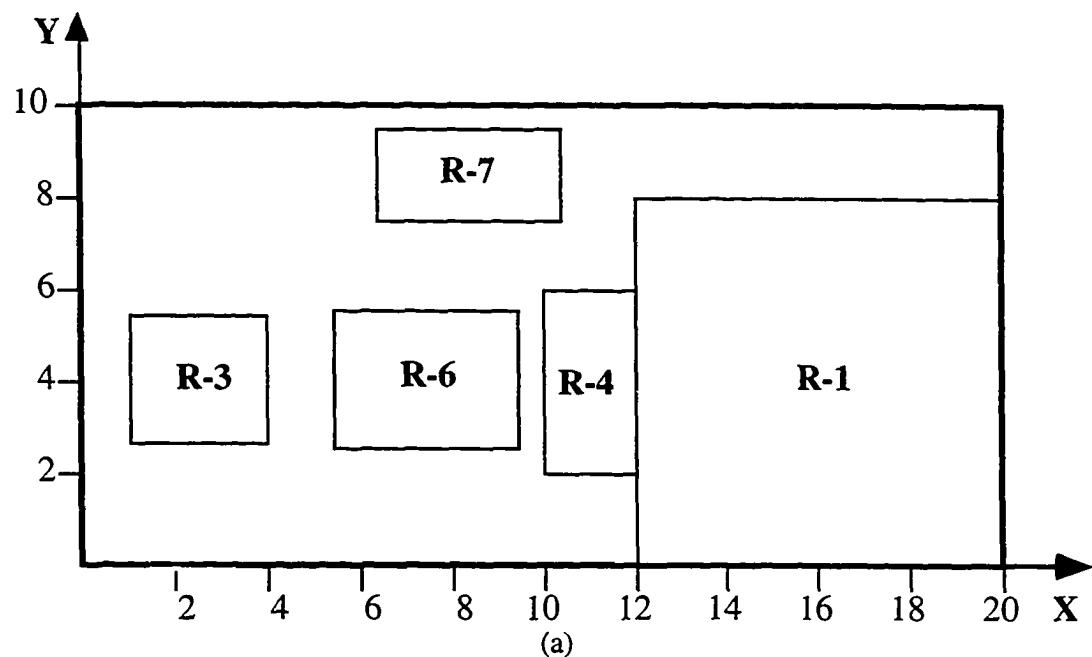


Figure 4.18  
Alternative Layout Solutions for PTF-2-4

## 4.6 Capabilities, Limitations, and Possible Extensions

The following section discusses the capabilities and limitations of the proposed approach and model, namely, the spatial representation underlying MoveSchedule's dynamic layout construction module and suggests possible extensions to this module as future research directions.

### 4.6.1 Approach

- Accounts for Relocation Costs During Static Layout Construction

PTFLCA is a heuristic algorithm that uses an early-commitment approach to construct the static layout of a given PTF. It positions resources one at a time in the layout and for each resource, it computes a single position that 1) satisfies the hard constraints on its acceptable positions and 2) minimizes proximity costs with other positioned resources in the layout and its relocation cost. To assess relocation costs, PTFLCA considers resource positions in other constructed layouts, which may span time frames that precede or succeed the PTF of the current layout. However, PTFLCA ignores all anticipated layout costs in determining a single position for the resource. That is, it ignores transportation and relocation costs of unpositioned resources in the current layout and of unpositioned resources in layouts of PTFs that have not yet been fully constructed.

- Compensates for Drawbacks of Early Commitment Approach

Random sampling is introduced in PTFLCA to allow variations in the solution. By running the algorithm X number of times, one can generate at most X alternative solutions to the layout problem, as some runs may not result in a feasible solution. Alternative solutions are possible because at each run the starting point can change or the sequence in which resources are selected can change. PTFLCA may not find a solution if one exists, but its probability of success is higher with randomness than without it.

Limitations inherent to an early-commitment strategy may be mitigated by adding backtracking or look-ahead capabilities. One may undo the results of actions that were taken earlier and proceed with different actions, a process known as backtracking. PTFLCA does not have a mechanism that allows it to backtrack to partial layouts and

pursue a different path to the solution. It would benefit from the addition of a backtracking mechanism to vary the order in which resources are selected to enter the layout.

Alternatively, PTFLCA could be augmented with look-ahead capabilities that allow it to assess the "value" of the outcome of an action before committing to it. Several look-ahead capabilities exist:

#### ◊ **Improving PTFLCA's Selection of a Single Resource Position**

The objective function VFL can be extended to include a measure of the desirability of a position or some lower-bound estimate on the expected layout costs of unpositioned resources. For example, in determining the best position of a resource  $i$ , PTFLCA can compute for selected positions of  $i$  the expected layout costs of the next resource  $j$  in the list of resources to be positioned. Selected positions of  $i$  can be the optimal positions of  $i$  that minimize  $\Delta VFL$  and some randomly sampled positions from  $SPP_i$ . PTFLCA would then select the position that minimizes the sum of  $\Delta VFL$  and the expected costs of  $j$ .

#### ◊ **Avoiding the Pursuit of Dead-End Alternatives**

Methods that assess the feasibility of the overall solution at intermediate stages of problem solving are needed. CSPA is one such method: it eliminates dead-end positions at the start of a PTF's layout construction and before PTFLCA positions individual resources. CSPA uses a least-commitment approach for determining sets of feasible positions for all resources in a PTF. It is not repeated after PTFLCA positions a resource, but it could be, so that dead-end alternatives are eliminated at intermediate stages of PTF layout construction. At present, PTFLCA uses Constraint Satisfaction only to update the SPP of each newly selected resource to satisfy out-zone and hard constraints with the positioned resources; it does not update the SPPs of any other resources that remain to be positioned.

- **Constructs Layouts of PTFs Chronologically but Can Relocate Resources**

MoveSchedule's dynamic layout construction strategy handles PTFs on a first-come-first-served basis. It constructs the layouts of PTFs in chronological order and consequently positions resources in the order they are needed on site. This strategy does not bind resources to single positions for the entire time period they are on site but it allows their relocation when other resources with higher priority compete for the same space. By virtue

of being myopic in handling the temporal aspect of the layout problem, this strategy may result in extra handling of resources that might be prevented by positioning higher priority resources first. Again this strategy could be improved by augmenting it, like PTFLCA, with backtracking and look-ahead capabilities.

- **Alternate Dynamic Layout Construction Strategies**

MoveSchedule's dynamic layout construction module can easily be extended to accommodate different dynamic layout construction strategies. Because of time limitations however, alternative strategies were not implemented. The following discussion rationalizes the expected benefits and shortcomings of some of these strategies and shows how the methods presented here can be extended to accommodate them.

#### ◊ **Long Term Resources**

One possible strategy is to rate resources based on their criticality and to position them by decreasing order of criticality using PTFLCA. Criticality may be a function of the size of the resource, or the length of time it is on site, etc.

Assuming the strategy selects the resource with the longest duration on site, it will create a layout that spans that longest time period and position the chosen resource in that layout, then pick another resource that exists on site for the second largest time period, and focus on the time period over which it exists on site to position it next, etc. The outcome of this strategy is a hierarchy of layouts spanning overlapping time periods. This strategy assumes that each resource occupies a single position for the duration it exists on site. Thus this strategy can be used to solve layout problems with a substantial number of stationary resources (such as tower cranes, fabrication shops) for which a single good position should be found on site.

In MoveSchedule, the time period over which a resource exists on site will include one or more PTFs. The dynamic layout strategy can be changed to select one or more PTFs at a time. To this end, the linear program described in section 4.3.3 is generally applicable and can be used to determine the optimal position for any resource that minimizes its relocation cost and proximity costs with positioned resources in the given time frame. PTFLCA, however, should be augmented with control mechanisms so that it can construct partial layouts. Furthermore, Constraint Satisfaction in PTFLCA should be adjusted so that default out-zone constraints would be met between co-existing resources

only, as resources that exist in time frames longer than a single PTF may not all coexist on site.

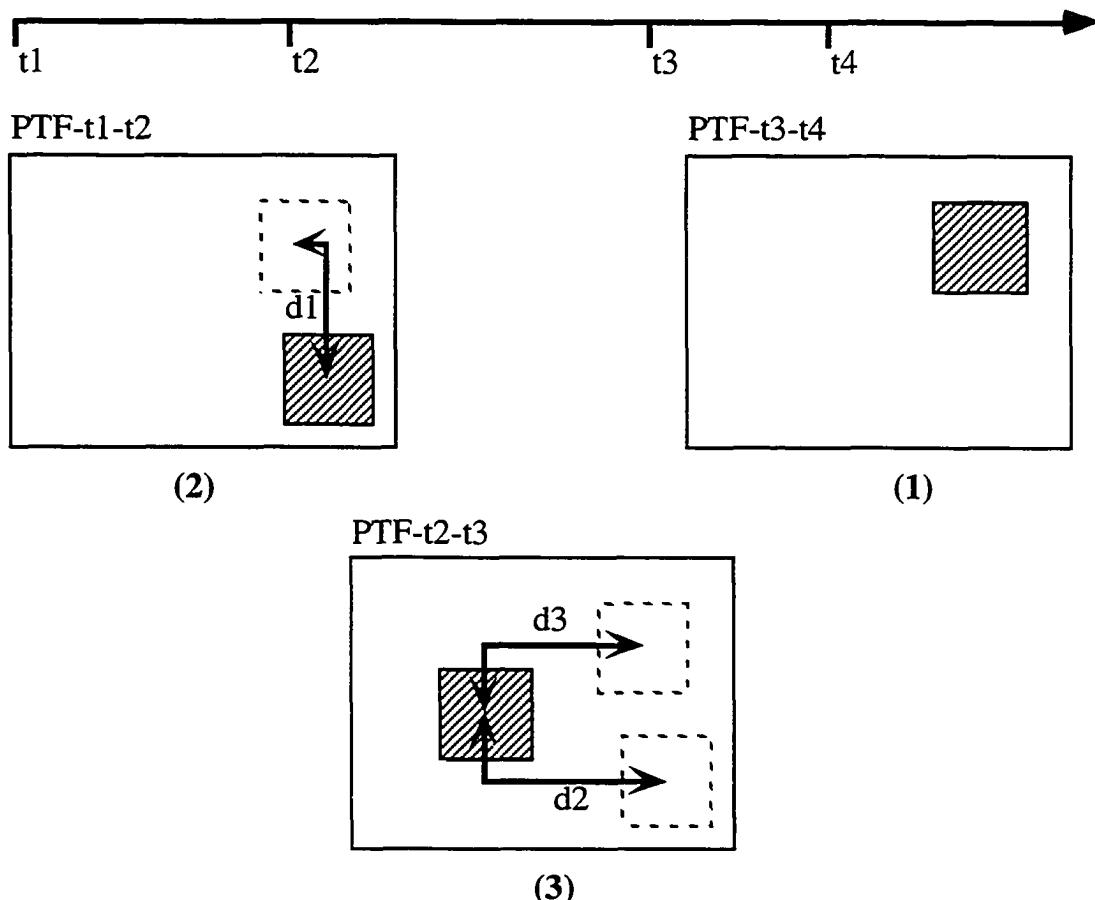
#### ◊ Critical PTFs

Another strategy may rate PTFs, as opposed to resources, based on their criticality and call PTFLCA to construct the layout of PTFs by decreasing order of their criticality. For example, criticality can be a measure of the total area requirement in that PTF. The rationale behind selecting PTFs with the highest area requirement first is that they are more likely than others to be subject to site congestion. Handling them first then allows the identification of infeasible layouts early in the process of dynamic layout construction.

MoveSchedule's dynamic layout strategy can be changed to sequence PTFs in any order so that PTFLCA will construct layouts accordingly. In this case, PTFLCA would need to be adjusted so that it does not overestimate relocation costs by double-counting the distance over which a resource is relocated.

For example, consider the scenario shown in Fig. 4.19 where the layouts of PTF-t3-t4, PTF-t1-t2, and PTF-t2-t3 are constructed in this order. Assume that PTFLCA positioned the cross-hatched resource in PTF-t3-t4 at the position shown in (1) and that PTFLCA determined that it would be cost-effective to relocate it in PTF-t1-t2 to the position shown in (2). Accordingly, PTFLCA increments VFL with the value for proximity costs (not relevant here) and the resource relocation cost which is equal to  $W' \times d_1$  where  $W'$  denotes the resource's relocation weight and  $d_1$  the distance traveled by the resource. Next, in constructing the layout of PTF-t2-t3, PTFLCA finds it again cost-effective to relocate the resource to the position shown in (3). Accordingly, it increments VFL with this resource's relocation cost which is equal to  $W' (d_2 + d_3)$ , where  $d_2$  is the distance between the resource's positions in PTF-t1-t2 and PTF-t2-t3, and  $d_3$  the distance between its positions in PTF-t3-t4 and PTF-t2-t3. In short, PTFLCA would add a total cost of  $W' (d_1 + d_2 + d_3)$  to VFL, while in reality the resource has only been moved a distance equal to  $d_2 + d_3$  and the corresponding relocation cost that should be added to VFL is  $W' (d_2 + d_3)$ .

PTFLCA can be adjusted so that it does not incrementally add relocation costs to VFL but instead adds them at the end of dynamic layout construction by considering the final sequence of layouts.



**Total distance counted by PTFLCA:**  $d_1 + d_2 + d_3$ .  
**Actual distance over which resource is relocated:**  $d_2 + d_3$

**Legend:**

- (#) Order in which the layouts are constructed.
- [Dashed Box] Position of the resource in previously constructed layouts.
- [Shaded Box] Position of the resource in current PTF.
- [Double-headed arrow] Distance that PTFLCA minimizes in each PTF to minimize the relocation cost of the resource.

Figure 4.19  
Actual vs. PTFLCA Computed Distances for Measuring Relocation Costs

◊ **Impact of Alternate Strategies on Overall System**

The above discussion does not address how the overall system, i.e., including the 2 other modules of MoveSchedule, can accommodate different dynamic layout construction

strategies. This issue is addressed after the presentation of the overall system in the following chapter (see section 5.8). We will only point out here that the dynamic layout construction module in MoveSchedule can be easily extended to accommodate different dynamic construction strategies, while the overall system cannot.

#### 4.6.2 Spatial Representation

The limitations of the spatial representation originate from the fact that all resources are modeled as rectangles to be positioned in either  $0^\circ$  or  $90^\circ$  orientation. This forces the SPP of any resource to be a disjunction of rectangles. Resources whose possible positions are delimited by, e.g., circles or non-rectangular polygons, cannot be modeled as such. For example, the set of positions reached by a crane cannot be described by a rectangle.

However, the space requirement of an odd-shaped static resource (such as the site space) can be approximated in MoveSchedule. In Fig. 4.20 (a), the resource (drawn with thick black lines) is modeled as a set of component rectangles where each rectangle corresponds to a separate resource. In Fig. 4.20 (b), it is modeled as the intersection of one large rectangle representing the resource, with many smaller rectangles (shown in stripes) representing dummy resources. The latter case provides a continuous space (the white space in the figure) where other resources can be positioned, by using in-zone constraints to force resources to be within the surrounding rectangle and out-zone constraints to prevent them from overlapping with the smaller rectangles.

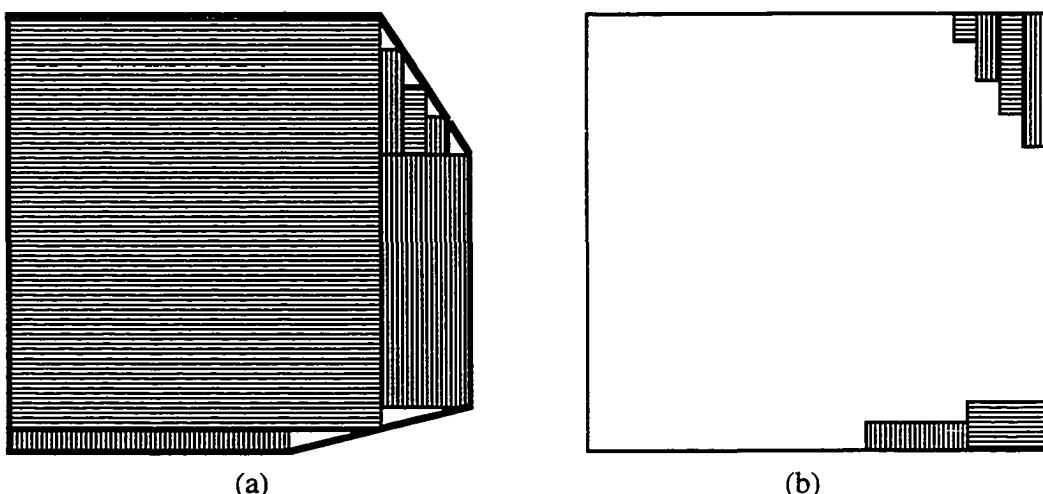


Figure 4.20  
Modeling Odd-shaped Resources

If the resource is not static, the aforementioned approximation will not work because the component rectangles will be positioned independently from one another by PTFLCA and their union may not result in the same shape of the original object.

Another limitation of the spatial representation lies in the method of measuring distances in hard constraints. Distances are measured between facing edges of rectangles as opposed to Euclidean distances measured between points. In the case of a minimum distance constraint, this is a conservative approximation to Euclidean distances and will result in omitting some feasible positions after satisfying the constraint. In the case of a maximum distance constraint, satisfying the constraint will result in possible positions that are farther away from the specified distance in a Euclidean metric.

#### **4.6.3 Possible Extensions for Process-Level Planning**

- **Adding Process-Level Costs: Rotation and Pick-up Costs**

PTFLCA ignores many process-planning details, such as the time it takes to relocate a resource or to rotate it. It also ignores costs that cannot be directly attributed to travel distances. For example, PTFLCA ignores the cost of rotating a resource if this is done while keeping its centroid in place. This cost is assumed to be included in the relocation cost of the resource if PTFLCA relocates its centroid. PTFLCA will keep a previously positioned resource in the same position and orientation only if its optimal SPP, i.e., the SPP that minimizes  $\Delta VFL$ , includes these. PTFLCA also ignores pick-up costs. It assumes that the cost of relocating the resource, for example, over three unit distance, is equivalent to the cost of relocating the resource over a one unit distance repeated three times. In the second case there is an additional cost equal to picking-up and putting-down the resource twice.

- **Planning Trajectories and Positions of Mobile Resources**

The MoveSchedule model, presented here, can model trajectories and intermediate positions of mobile resources (such as trucks) along a path. Intersecting rectangles can model trajectories with odd shapes and mobile resources can be forced to travel on these trajectories by using in-zone and out-zone constraints. PTFLCA can compute intermediate positions of mobile resources provided that the user divides the relevant PTFs into smaller ones and balances proximity weights of the resource with its departure and destination

points so that the resource has an increasingly larger proximity weight with its destination point as time goes by.

## 4.7 Conclusions

This chapter presented the dynamic layout construction module of MoveSchedule. This module solves a constrained dynamic layout problem with 2-dimensional geometric constraints between relative positions of resources. The presented model represents resources as rectangles with fixed or variable dimensions over time. Interactions between resources are modeled using preference measures and hard constraints on their relative positions. These interactions drive the positioning and relocation of resources. To construct the layout of all PTFs, a first-come-first-served strategy is used. A two-stage algorithm for constructing the layout of each PTF was presented. In the first stage, CSPA uses Constraint Satisfaction to determine feasible positions for the resources in a PTF that satisfy the hard constraints on their relative positions. In the second stage, PTFLCA uses Linear Programming to determine optimal positions of a selected resource that minimize its transportation and relocation costs.

# **Chapter 5**

## **TIME-SPACE TRADEOFF STRATEGIES FOR SPACE SCHEDULING**

### **5.1 Introduction**

Planning a dynamic layout has a benefit in that it helps foster economical construction operations. It also helps identify time frames in the schedule where spatial conflicts exist due to site space limitations. When space is limited and the resources needed in a given time frame cannot be accommodated on site without violating constraints on their relative positions, tradeoffs between construction method selection, resource allocation to activities, and space allocation to resources become necessary to generate a feasible schedule and dynamic layout.

This chapter addresses the space scheduling problem and proposes a space-time trade-off model for adjusting activity durations and start dates to decrease the need for space over problematic time frames. The model characterizes resources' space requirements over time. It establishes a time-space relationship for each activity in the schedule based on minimum, normal, and maximum resource levels. The resource levels describe different space requirements for different activity durations so that tradeoffs between activity duration and space requirements on site are possible.

Space-time tradeoff strategies and heuristics for applying these strategies are discussed as part of the computerized implementation of the model, named *Space Conflict Resolver* (SCR). SCR opportunistically changes the schedule when the dynamic layout construction module cannot construct feasible layouts that meet the spatial requirements and constraints of resources. The *scheduler* performs critical path method computations to maintain the schedule as it gets changed by SCR. The integration of the dynamic layout construction module with SCR and the scheduler is presented in this chapter as well.

## 5.2 Activity Time-Space Relationship

### 5.2.1 Resource Area Profiles

MoveSchedule differentiates between dependent and independent resources in modeling their area profiles. Dependent resources have a space requirement that varies with the duration of the associated activity(ies) and they are present on site for a time period corresponding to the associated activity(ies)' start and finish date(s). In contrast, independent resources have a constant space requirement and are present on site for a user-defined time period.

MoveSchedule offers four alternative area profiles for the user to choose from while modeling the area requirement of a resource over time. The area required by the resource at any one time during problem solving is inferred from this area profile. While actual area profiles of resources, presented previously in section 2.1 of Chapter 2, are varied and depend on the nature of the resource (e.g., whether the resource is stackable, indivisible), the project, the nature of the site, and the company's procurement policies (such as resource availability, delivery costs, and suppliers' policies), the four profiles modeled in MoveSchedule are not exhaustive but constitute the basic building blocks using which more exact area profiles can be built or approximated.

#### 5.2.1.1 Profile-A

Profile-A models a resource that depends on a single activity and whose space requirement continuously decreases as the activity progresses. For example, construction materials

(e.g., sand or gravel) can be modeled using Profile-A as they typically get consumed when the corresponding activity (e.g., an erection and installation activity) progresses.

For this profile, it is assumed that:

- The resource's mobilization and demobilization times are zero. The total quantity of the resource is brought to site when the resource is first needed by the activity and there is no excess resource left when the activity finishes.
- The total quantity of the resource needed by the activity is constant and independent of the activity duration.
- The resource consumption rate is linear and proportional to the activity production rate.

Fig. 5.1 (a) illustrates an example scenario where the whole quantity is brought to site when the resource is first needed and is depleted as the activity progresses. The area that the resource occupies on site therefore shrinks over time (see direction of the arrows in Fig. 5.1 (a)). This is a typical scenario when the resource shortage penalty is high, large deliveries are cost effective, and adequate means are available to accommodate the resource early. From a site space management standpoint, this scenario may not be desirable as more space is tied up at the start of the activity.

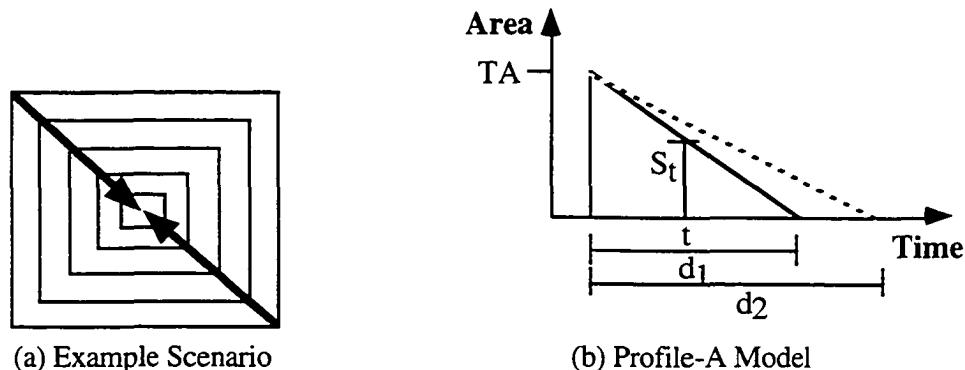


Figure 5.1  
Profile-A

Fig. 5.1 (b) shows how the area of a profile-A resource is modeled in MoveSchedule. The total area (TA) required to accommodate the total quantity of the resource is reserved on site when the associated activity starts. The area  $S_t$  occupied by the resource at any one time  $t$  (note that  $t$  is used here to denote a single point in time and not the duration of a layout) decreases linearly over the activity duration  $d_1$ . The dashed line in Fig. 5.1 (b) shows that when the activity takes longer to complete (duration  $d_2$ ), the

resource consumption rate is slower, thus making the area required at any one time larger except for the start and finish times of the activity.

TA must be defined by the user for each resource. The actual dimensions of the resource at any one time are computed using a user-defined length-to-width (L/W) ratio, which remains constant, and the value of the area derived from the area profile at that time.

### 5.2.1.2 Profile-B

Profile-B models a resource that depends on a single activity and whose area requirement is either constant or fluctuates between a minimum and a maximum level as the activity progresses. For example, construction materials (e.g., sand, bricks) that are replenished on a regular basis during the course of the corresponding activity, or that can be stacked so that they occupy a constant space on site (e.g., roof trusses or plywood) can be modeled using Profile-B.

The assumptions of this model are in part similar to those of Profile-A:

- The resource's mobilization and demobilization times are zero.
- The total quantity of the resource needed by the activity is constant and independent of the activity duration.
- The area required by the resource is smaller when the activity is performed at a slower rate (for example, resource depletion and replenishment counterbalance each other).

Fig. 5.2 (a) shows one scenario where the area occupied by the resource shrinks and expands as the resource is consumed and replenished over time. Fig. 5.2 (b) shows a resource that is stacked so that the occupied area remains constant, although the height increases and decreases over time. These are typical scenarios when space on site is limited, and the cost of stocking or handling the resource is high and multiple deliveries are made to procure the resource.

Fig. 5.2 (c) shows how the area of a Profile-B resource is modeled in MoveSchedule. A constant area  $A_1$  is reserved on site to accommodate the resource at its maximum space requirement for the duration of the activity  $d_1$ . When the activity takes longer to complete (duration  $d_2$ ), the area  $S_t$  required at any one time  $t$  is smaller as indicated by  $A_2$  and the dashed line in Fig. 5.2 (c).

$A_i$ , and therefore  $S_t$  since  $A_i$  and  $S_t$  are equal for this profile, are user-defined for each activity duration  $d_i$  (where  $i = 1, 2, \dots$ ). The actual dimensions of the resource are computed using a user-defined L/W ratio and are constant over the activity duration.

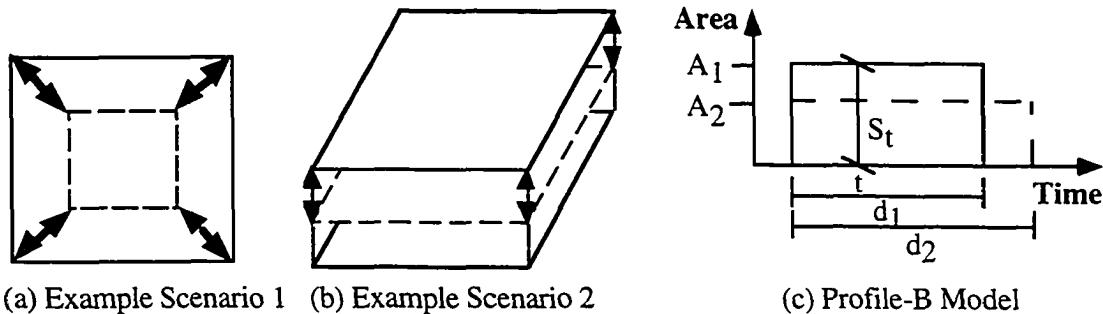


Figure 5.2  
Profile-B

### 5.2.1.3 Profile-C

Profile-C models a resource that depends on one or several activities and whose space requirement is constant over time. For example, a piece of construction equipment that is assigned to more than one activity and that is possibly idle between activities can be modeled using Profile-C. The space required by the equipment is constant as defined by its footprint plus any additional area for maneuvering.

For this profile it is assumed that:

- The resource requires space of fixed shape and area that do not vary over time.
- The resource remains on site if idle, i.e., it is available on site for a time interval (TB) extending from the start of the first activity to the end of the last activity requiring its use.
- The resource can be shared by concurrent activities or can be partially used by an activity.

Fig. 5.3 shows how the area of a Profile-C resource is modeled in MoveSchedule. A rectangle of length L and width W models the shape of the area required to accommodate the resource over TB.

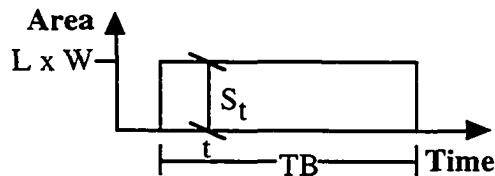


Figure 5.3  
Profile-C and Profile-D Models

$L$  and  $W$  are user-defined for each resource.  $TB$  is computed by MoveSchedule based on the start and finish dates of the activities to which the resource is assigned.

Note that for Profile-C, the resource's mobilization and demobilization times need not be zero and can be defined as separate activities using this resource, in which case the resource will be on site for the duration of these activities as well.

#### 5.2.1.4 Profile-D

Profile-D models an independent resource whose space requirement is constant for the duration it is present on site. For example, facilities that support construction operations (such as trailers, parking lots, or obstacles on site such as trees or existing buildings), or some construction equipment needed to support construction operations in general (such as a tower crane) can be modeled using Profile-D.

For this profile it is assumed that:

- The resource requires space of fixed shape and area that does not vary over time.
- The resource exists on site for a user-defined time interval  $TB$ .

Fig. 5.3 shows how the area of a Profile-D resource is modeled in MoveSchedule. A rectangle of length  $L$  and width  $W$  models the shape of the space required to accommodate the resource over a time interval  $TB$ .

$L$ ,  $W$ , and  $TB$  must be user-defined for each resource. Note that here too, the resource's mobilization or demobilization time need not be zero and can easily be included in the user-defined  $TB$ .

Table 5.1 summarizes the user-defined parameters vs. the computed parameters associated with each profile type.

<b>Profile Type</b>	<b>User-Defined Parameters</b>	<b>System-Computed Parameters</b>
Profile-A	TA, $d_i$ L/W ratio	$S_t$ , $L_t$ , $W_t$ , Start of activity, Finish of activity
Profile-B	$A_i$ , $d_i$ L/W ratio	$S_t = A_i$ , $L_t$ , $W_t$ , Start of activity, Finish of activity
Profile-C	L, W	$S_t = L \times W$ , Start of TB, Finish of TB
Profile-D	L, W,      Start of TB, Finish of TB	$S_t = L \times W$

Table 5.1  
User-Defined vs. System-Computed Parameters of Area Profiles

### 5.2.2 Discretizing Space Requirements

PTFs (defined in Chapter 1) slice the project duration into discrete time intervals based on the arrival and departure of resources to and from the modeled site space. These arrival and departure times are inferred from the CPM schedule for dependent resources and are user-defined for independent resources. Assume that the schedule of some project is as shown in the top portion of Fig. 5.4 where the boxes represent the activities in the schedule and the solid lines between activities represent finish-to-start precedence relationships between them. Assume also that the arrival and departure times of the independent resources in that project are as shown in the bottom portion of Fig. 5.4. Then, the PTFs identified by MoveSchedule for this project are as defined by the vertical dotted lines and marked on the time axis in Fig. 5.4.

PTFs mark changes in resource space requirements, resource positions, and resource constraints. Each PTF defines the duration of a layout. It is assumed that each resource in a layout occupies space corresponding to the maximum area needed by it during that time frame, as inferred from the resource's area profile. For example, if a Profile-A resource is needed in PTF-e-f then an area equal to  $S_e$  (where  $S_e \leq TA$ ) is needed to accommodate the resource for the time interval e to f. For all other profiles the area required is constant and hence no assumption on the timing of the space required is necessary.

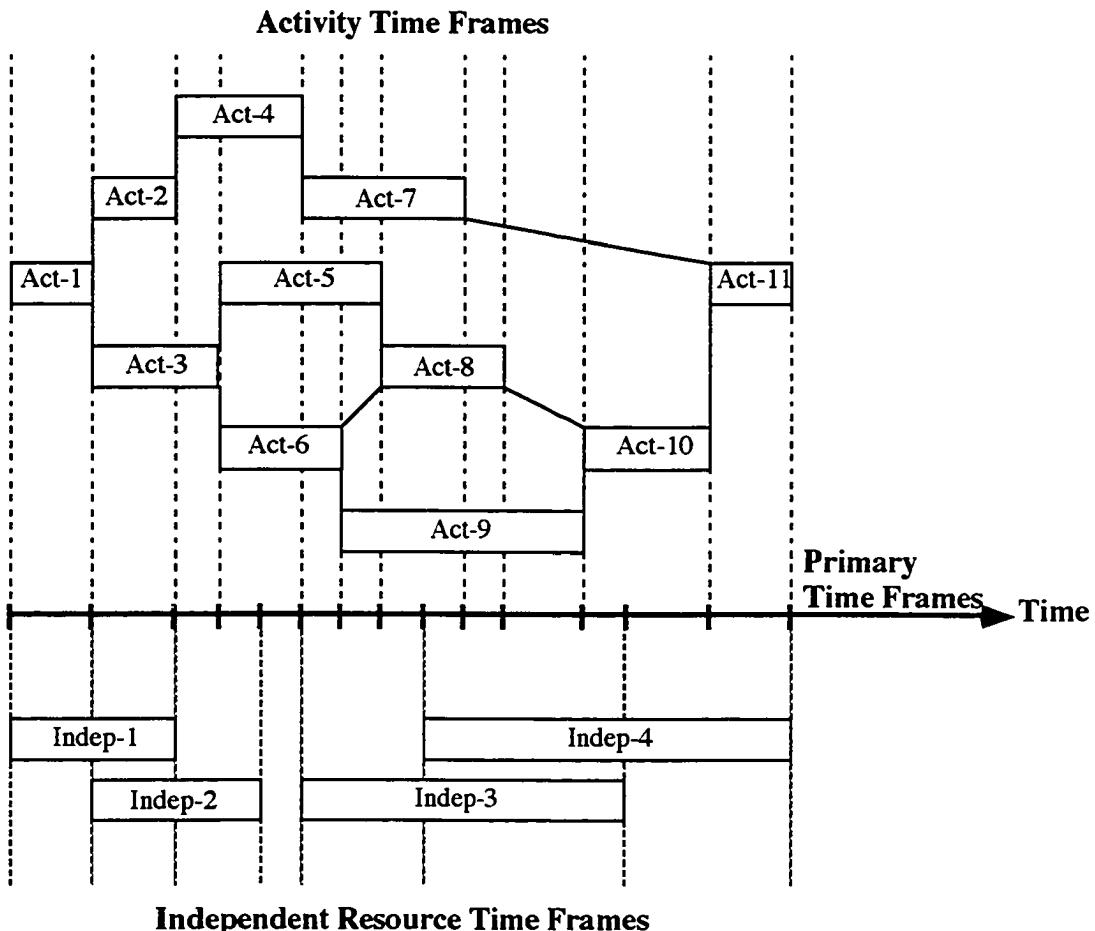


Figure 5.4  
Primary Time Frames

### 5.3 Activity Resource Levels

An activity typically requires different resources in various quantities for its performance. Varying the type of resource or the number of resources assigned to an activity may result in the activity taking more or less time to complete. For example, a pipelaying activity can be performed with a crane and pipes, a pipelayer and pipes, or two pipelayers operating simultaneously and pipes. Resource levels are used in MoveSchedule to model alternative durations for each activity. A **resource level** is comprised of a combination of resources of different types or quantities thus describing different methods or crew-sizes for performing the activity.

A minimum of one and maximum of three (minimum, normal, and maximum) resource levels can be user-defined for each activity to describe up to three alternative durations (maximum ( $d_{max}$ ), normal ( $d_{norm}$ ), and minimum ( $d_{min}$ ); note inverse order; where  $d_{max} > d_{norm} > d_{min}$ ) and space requirements for its performance. Each resource level consists of a combination of Profile-A, -B, or -C resources. For each level, the user inputs the required resources and the corresponding activity duration. Depending on the profile type of the resources, the input for each level is as summarized in Table 5.2.

Resource Level	Duration	Profile-A	Profile-B	Profile-C
Minimum	$d_{max}$	TA, L/W ratio	$A_{min}$	L/W ratio L, W
Normal	$d_{norm}$		$A_{norm}$	
Maximum	$d_{min}$		$A_{max}$	

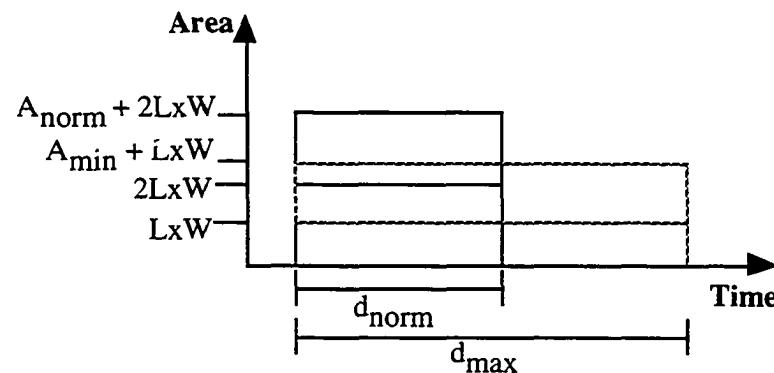
Table 5.2  
Input Required in Activity Resource Levels

If the same Profile-A resource is used in several resource levels, then the total area (TA) required as input for it will not change across these levels. If, however, the same Profile-B resource is used in more than one level, then the user should provide the areas ( $A_i$ ) corresponding to the activity durations ( $d_i$ ) in accordance with the assumptions described in section 5.2.1.2. If the same Profile-C resource is used in several resource levels, then its dimensions (L and W) must be input and these do not vary with the activity duration.

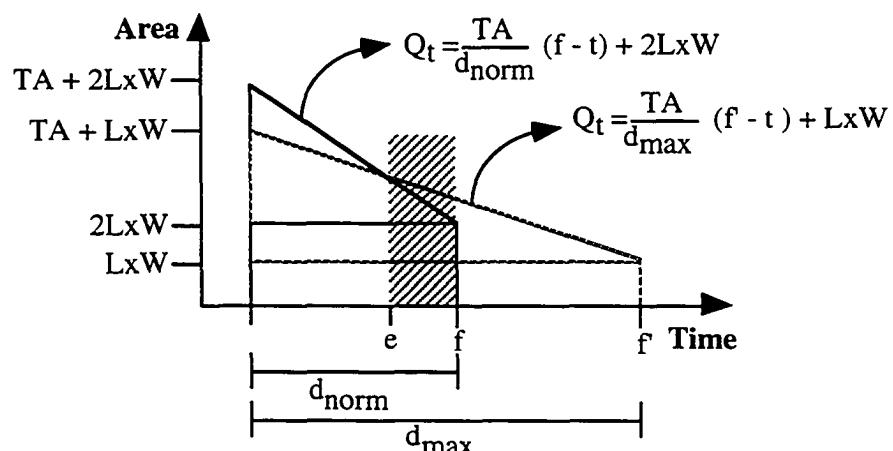
Varying the resource level results in varying the activity space requirement over time depending on the chosen resource area profiles. For example, lowering the resource level of the above pipelaying activity from two pipelayers and pipes to a single pipelayer and pipes affects the area profiles of the two types of resources involved in different ways. If the pipes were modeled with Profile-B, lowering the resource level implies a reduction in the area requirement of pipes over the duration of the activity (by definition of Profile-B the area required by the resource is smaller when the activity is performed at a slower rate:  $A_{min} < A_{norm} < A_{max}$ ). If modeled with Profile-A, the effect will be an increase in area requirement over the activity duration. A single pipelayer modeled as Profile-C would not be affected by the change in activity duration as it always occupies the same amount of

space on site. However, the total area required for the activity changes since one pipelayer instead of two is now needed.

The total area required for the activity is the sum of the individual area requirements of the resources at the assigned resource level. Lowering the resource level of an activity can decrease or increase its total area requirement as this depends on the type of resource profiles used (see Fig. 5.5).



(a) Total Area for Profile-B and -C Resource Level



(b) Total Area for Profile-A and -C Resource Level

Figure 5.5  
Total Area Requirement of an Activity for Different Resource Profiles

Fig. 5.5 illustrates the variation in total area requirement of the above pipelaying activity when its resource level is changed from normal to minimum, i.e., the activity duration changes from normal ( $d_{norm}$ ) to maximum ( $d_{max}$ ). Assume that the maximum level consists of pipes and a crane. The normal level consists of two pipelayers requiring an area  $2LxW$  (where  $L$  and  $W$  are the dimensions of a single pipelayer) and pipes

requiring an area  $TA$  or  $A_{norm}$  depending on the profile used to model pipes. The minimum level consists of a single pipelayer requiring an area  $L \times W$  and pipes requiring an area  $TA$  or  $A_{min}$  again depending on the profile used. The total area requirement of the activity at any time  $t$  is denoted by  $Qt$  and is computed differently for each level as shown in Fig. 5.5. It is expected that the total area requirement of the activity is lower when the activity is performed at a lower resource level. However, this is not always the case. Fig. 5.5 (a) shows that, when the two resource levels consist of Profile-B and Profile-C, the total area requirement of the activity is lower at the lower level. Fig. 5.5 (b), in contrast, shows that, when the two resource levels consist of Profile-A and Profile-C, the total area requirement of the activity is lower at the minimum level for only a portion of the activity duration (see shaded time interval) but then is higher for the remaining duration. That is, for any PTF that starts at or after time  $e$ , the total area requirement of the activity at the minimum level is higher than at the normal level.

When one or several Profile-A resources are used, the relationship between the activity duration ( $d_{min}$ ,  $d_{norm}$ , and  $d_{max}$ ) and the activity total area requirement at any time is not a monotonic function as the activity's total area requirement does not necessarily decrease when performing the activity with a lower resource level. Furthermore, as the resource profiles and levels reflect, there is not a single area requirement per activity. Hence, MoveSchedule needs to compute the total area requirement of each activity on a timely basis, as needed in the course of space scheduling.

## 5.4 Spatial Conflict Resolver (SCR)

SCR is one module in MoveSchedule. This module resolves spatial conflicts that arise when the layout of a PTF (e.g., PTF-e-f) cannot meet all hard constraints or accommodate stationary resources at their positions in previously constructed layouts. Its objective is to reduce the total area requirement in the *problematic time frame* PTF-e-f at a minimum increase in project duration. Reducing the total area requirement in the problematic time frame will increase the amount of available site space and this may help solve the spatial conflict in this time frame.

SCR assumes the following:

- unlimited resources but for space
- a resource cannot have different area profiles in different PTFs
- activities cannot be interrupted

- no changes can be made to the start date of an activity or to its resource level once the activity has started
- activities can be delayed, i.e., there are no hard constraints that force activities to start and finish at some mandatory dates.

The limitations of these assumptions and suggestions to overcome them are presented in section 5.8.

### 5.4.1 Time-Space Tradeoff Strategies

SCR selects a strategy from the following set of time-space tradeoff strategies (see Fig. 5.6) to reduce the total area requirement in the problematic time frame at minimum increase in project duration. These strategies do not include one for crashing the duration of earlier activities but as it will be noted in section 5.6 and 5.8, this strategy is not needed with the approach used in MoveSchedule for solving the space scheduling problem.

#### 5.4.1.1 Strategy A: Delay Activity

Strategy A delays an activity that starts at the start of PTF-e-f (i.e., time e), to start at the end of PTF-e-f (i.e., time f) in order to delay its space requirements in time. From those that start at time e, it selects the activity which, if delayed, will cause the least increase in project duration. If two or more activities tie then it selects the one that has the largest area requirement in PTF-e-f because delaying it will maximize the amount of freed space in this time frame. Let,

- $candidates := \{ \text{activities that start at time } e \}$
- $delay := f - e$ .
- $\text{remaining-total-float}_i := \text{total float of activity } i - delay$
- $\text{current-area-requirement}_i := \text{sum of area requirements of the resources in the current resource level of activity } i \text{ at time } e$ .
- $\text{decrease-in-area-requirement}_i := \text{current-area-requirement}_i - \text{total area requirement of all Profile-C resources of } i \text{ that were brought to site before } e \text{ (i.e., Profile-C resources associated with activities that started before time } e)$ .

This strategy selects from  $candidates$  the one

- 1) that has the maximum value of  $\text{remaining-total-float}$ . In case two or more activities are tied, then from those tied select the one

- 2) that has the largest value of *decrease-in-area-requirement*. If activities are still tied, then from those tied select
- 3) one at random.

Fig. 5.6 shows the ramification of this strategy as implemented in MoveSchedule. It shows how this strategy may or may not increase the project duration depending on the values of the total float and *remaining-total-float* of a candidate activity. For example, if the activity is not critical (branch A.1) and its *remaining-total-float* is positive (branch A.1.1), then delaying this activity will not increase the project duration. In contrast, if the activity is not critical but its *remaining-total-float* is negative (branch A.1.2) or the activity is critical (A.2), then delaying that activity will definitely increase the project duration. Note that the *remaining-total-float* of any critical activity is equal to the duration of PTF-e-f. Hence, strategy A skips step 1 and selects from candidates the one that has the largest value of *decrease-in-area-requirement*.

#### **5.4.1.2 Strategy B: Lower Resource Level of Activity**

Strategy B lowers the resource level of an activity that starts at the start of PTF-e-f and therefore lengthens its duration. From those activities that qualify, it selects the one which, if performed at a lower level, causes the least increase in project duration. If two or more activities tie then it selects the one with the largest difference in total area requirement between its current resource level and the lower one. Let,

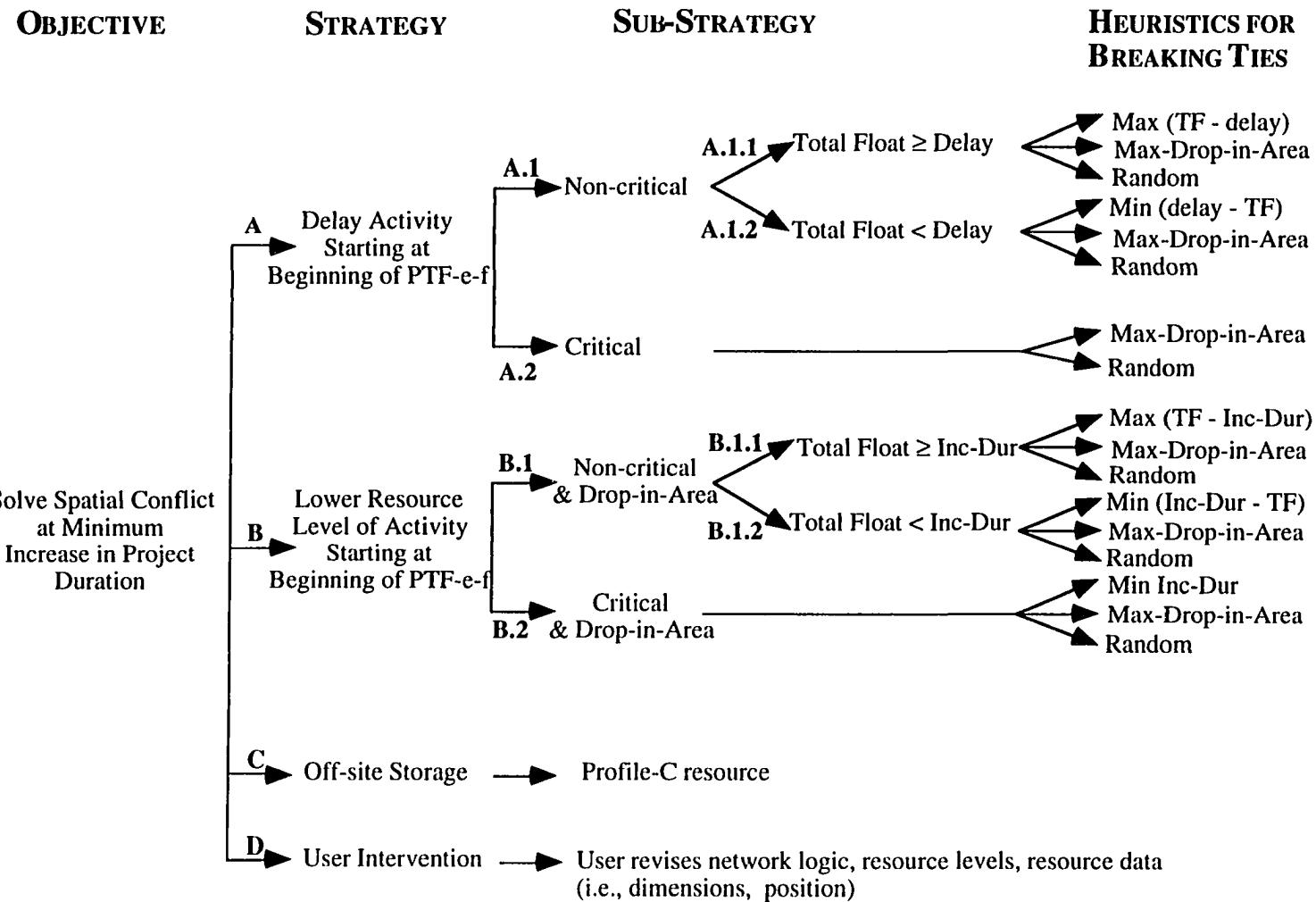
- $candidates := \{activities \text{ that start at time } e \text{ and whose total area requirement in PTF-e-f is lower if performed at a lower resource level}\}$ .
- $increase-in-activity-duration_i := \text{duration of activity } i \text{ at the current resource level} - \text{duration of activity } i \text{ at the lower resource level}$ .
- $remaining-total-float_i := \text{total float of activity } i - increase-in-activity-duration_i$ .
- $current-area-requirement_i := \text{sum of area requirements of the resources in the current resource level of activity } i \text{ at time } e$ .
- $decrease-in-area-requirement_i := current-area-requirement_i - \text{total area requirement of activity } i \text{ if performed at a lower resource level}$ .

This strategy selects from *candidates* the one

- 1) that has the maximum value of *remaining-total-float*. In case two or more activities are tied, then from those tied select the one

Figure 5.6  
Time-Space Tradeoff Strategies for Solving Spatial Conflict

Page 125



- 2) that has the largest value of *decrease-in-area-requirement*. If activities are still tied, then from those tied select
- 3) one at random.

Fig. 5.6 shows the ramification of this strategy as implemented in MoveSchedule. As in strategy A, it shows how strategy B may or may not increase the project duration depending on the values of the total float and *remaining-total-float* of a candidate activity. For example, if the activity is not critical (branch B.1) and its *remaining-total-float* is positive (branch B.1.1), then delaying this activity will not increase the project duration. In contrast, if the activity is not critical but its *remaining-total-float* is negative (branch B.1.2) or the activity is critical (B.2), then delaying that activity will definitely increase the project duration.

#### **5.4.1.3 Strategy C: Off-Site Storage**

Strategy C removes a Profile-C resource that requires space on site if it is idle during the time period e to f. It assumes that, for example, storage off-site is available. Resources modeled with any other profile exist on site for the duration of their associated activity or for a user-defined time period and, hence, are considered in use in PTF-e-f. Before firing this strategy, SCR asks the user to confirm that removal of the resource from site while it is idle is permissible.

#### **5.4.1.4 Strategy D: User Intervention**

Strategy D allows for user intervention in the course of problem solving. User intervention is desired to allow for additions in the activities' resources levels, or changes in the logic of the activity network and the arrival and departure times of independent resources.

### **5.4.2 Strategy Selection**

SCR can be operated in one of two modes: a fully-automated mode and a user-driven mode. In each mode, SCR prioritizes the strategies differently to select the one that reduces the total area requirement in the problematic time frame at a minimum increase in project duration.

#### **5.4.2.1 Mode 1: SCR Fully Automated**

In mode 1, SCR is fully automated and expects only the problematic time frame PTF-e-f as input. It calls strategies A and B. Each strategy may return an activity with a value for the *remaining-total-float* and a value for the expected *decrease-in-area-requirement* in PTF-e-f. SCR selects the strategy-activity combination with the largest value of *remaining-total-float*. In case of a tie, SCR selects the one with largest value of *decrease-in-area-requirement*. If further tied, SCR selects one at random. The expected increase in project duration is equal to

- 0 ; if *remaining-total-float* of selected strategy  $\geq 0$
- 0 – *remaining-total-float* ; if *remaining-total-float* of selected strategy  $< 0$ .

SCR calls strategy D before applying any strategy that results in increasing the project duration to give the user a chance for stopping the space scheduling process when increasing the project duration is unacceptable. SCR in mode 1 can be modified so that strategy D is called before applying any strategy (whether it increases the project duration or not) to give the user the flexibility of changing resource levels or other project data at any stage of problem solving. If the user wants to change the project data then the space scheduling process including dynamic layout construction is restarted using the new data provided by the user.

If operated under mode 1, SCR will always find at least one strategy to apply and thus will always return a modified schedule. Whether this schedule enables the spatial conflict to be solved in PTF-e-f is to be determined by running PTFLCA on PTF-e-f again. Several iterations between PTFLCA and SCR may be needed before the layout of PTF-e-f is successfully constructed.

When no activity starts at PTF-e-f, SCR fails to suggest a strategy. In this case however, a new demand for space can only stem from the arrival of an independent resource whose timing is not affected by a change in the schedule. Note that the area profiles of all activities in progress show either a constant or a declining space need over time. Hence, when an activity's space need could be met in a preceding PTF, then its space needs should be met in the current PTF as well provided that no hard constraints that cannot be satisfied were introduced in the current PTF. This is assumed to be the case here.

#### **5.4.2.2 Mode 2: SCR User Driven**

In mode 2, the user labels a resource R in the layout of PTF-e-f to be removed from the layout. Depending on R's type and profile, SCR has to select a strategy that removes it from the layout at a minimum increase in project duration. Fig. 5.7 shows the decision tree that CSR goes through in selecting a strategy.

As shown in Fig. 5.7, mode 2 includes two cases:

- If R is an independent resource, then changing the schedule has no effect on the timing of R's presence on site and SCR calls for user intervention (i.e., strategy D).
- If R is dependent and
  - if R is modeled with Profile-A or -B, then it is associated with only one activity. Strategy B is discarded because it may not always result in removing R from PTF-e-f as the lower level may include the same Profile-A or -B resource but with smaller area requirements. Strategy A is the only other strategy applicable provided that the activity starts in PTF-e-f. Otherwise, SCR calls strategy D.
  - if R is modeled with Profile-C and
    - if R is idle, then SCR fires strategy C with the user's consent, otherwise SCR calls strategy D.
    - if R is active, then SCR checks if strategy A applies.
      - If it applies, SCR checks if R is associated with activities that started earlier than PTF-e-f.
        - If so, delaying the activities associated with R in PTF-e-f will not result in removing R from this PTF since R, by assumption, will remain on site in-between uses. Hence, SCR calls strategy D after failing to suggest a solution.
        - If R is not associated with activities that occurred earlier than PTF-e-f, then SCR fires strategy A which will delay all activities associated with R in PTF-e-f.
      - Otherwise SCR calls strategy D.

## Spatial Conflict Resolver MODE 2

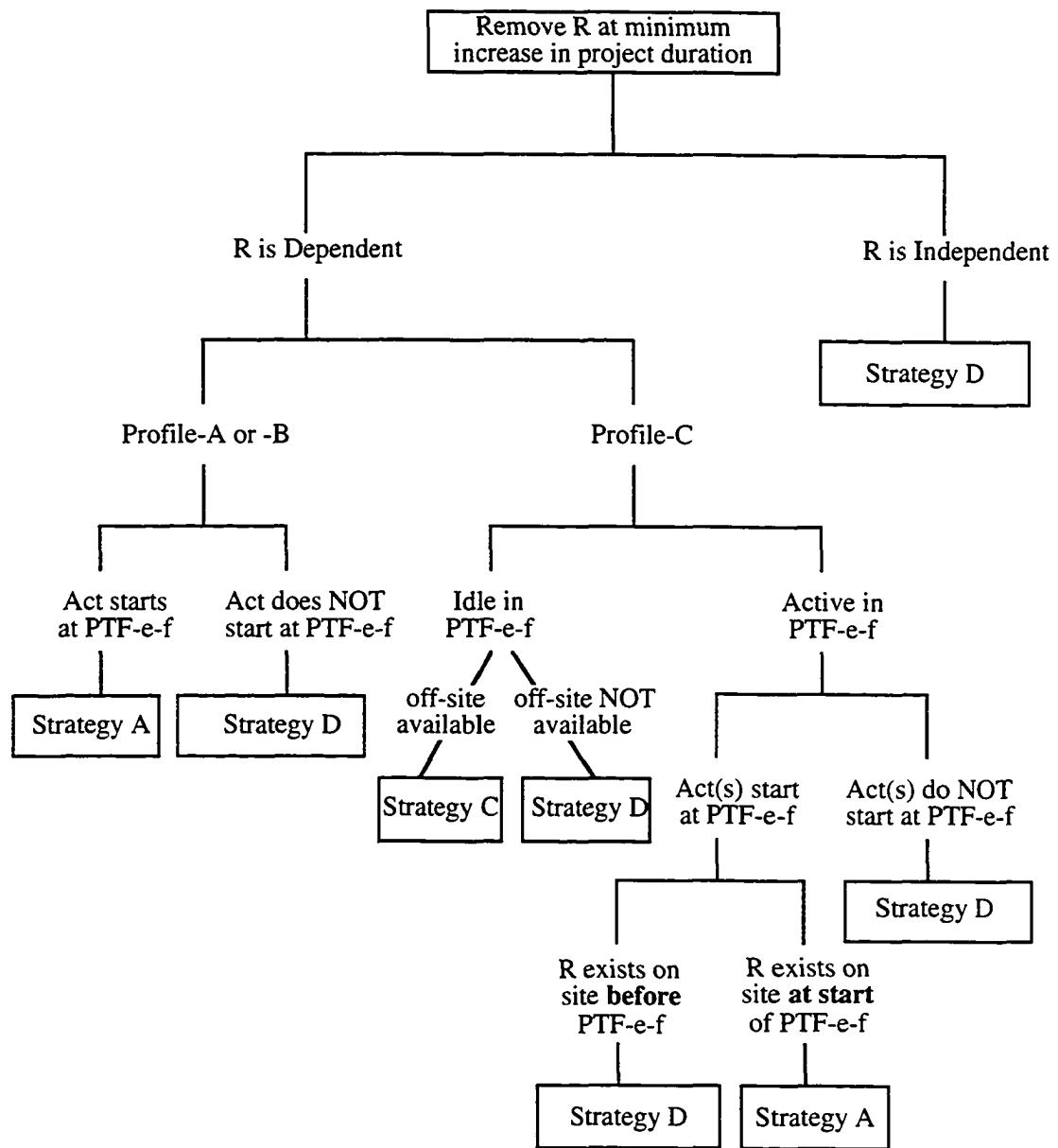


Figure 5.7  
Decision Tree of SCR in Mode 2

## 5.5 Scheduler

The scheduler carries critical path method (CPM) computations by making forward and backward pass calculations to generate early and late start and finish times of activities in the network. While the methods presented in this chapter are applicable to any type of precedence relationships between activities, only finish-to-start type relationships were implemented in MoveSchedule.

The scheduler computes the initial early-start schedule, referred to as the *unconstrained schedule*, and identifies the PTFs based on the schedule and the independent resources data. The scheduler is also responsible for executing the strategies selected by SCR. Depending on the selected strategy, the scheduler:

- Strategy A:** resets the start date of the selected activity to be no earlier than the date suggested by SCR and re-computes the CPM schedule and all PTFs including and following the current one.
- Strategy B:** resets the duration of the selected activity to that of the lower resource level, updates the resources assigned to it, and re-computes the CPM schedule and all PTFs including and following the current one.
- Strategy C:** removes the resource from the list of existing resources in PTF-e-f without changing the schedule or any PTF.
- Strategy D:** waits for the new user input and recomputes the unconstrained schedule given that input.

Note that in the first two cases, the start and end time of the PTFs and the resources present in each PTF can change as a result of changing the start and finish times of activities or their resource levels.

## 5.6 Space Scheduling Algorithm

The space scheduling algorithm is the main algorithm underlying MoveSchedule. It repetitively calls PTFLCA, SCR, and the scheduler to adjust an initial schedule and make it comply with site space limitations. Fig. 5.8 shows a flow chart of the algorithm and it depicts how the three modules of MoveSchedule connect.

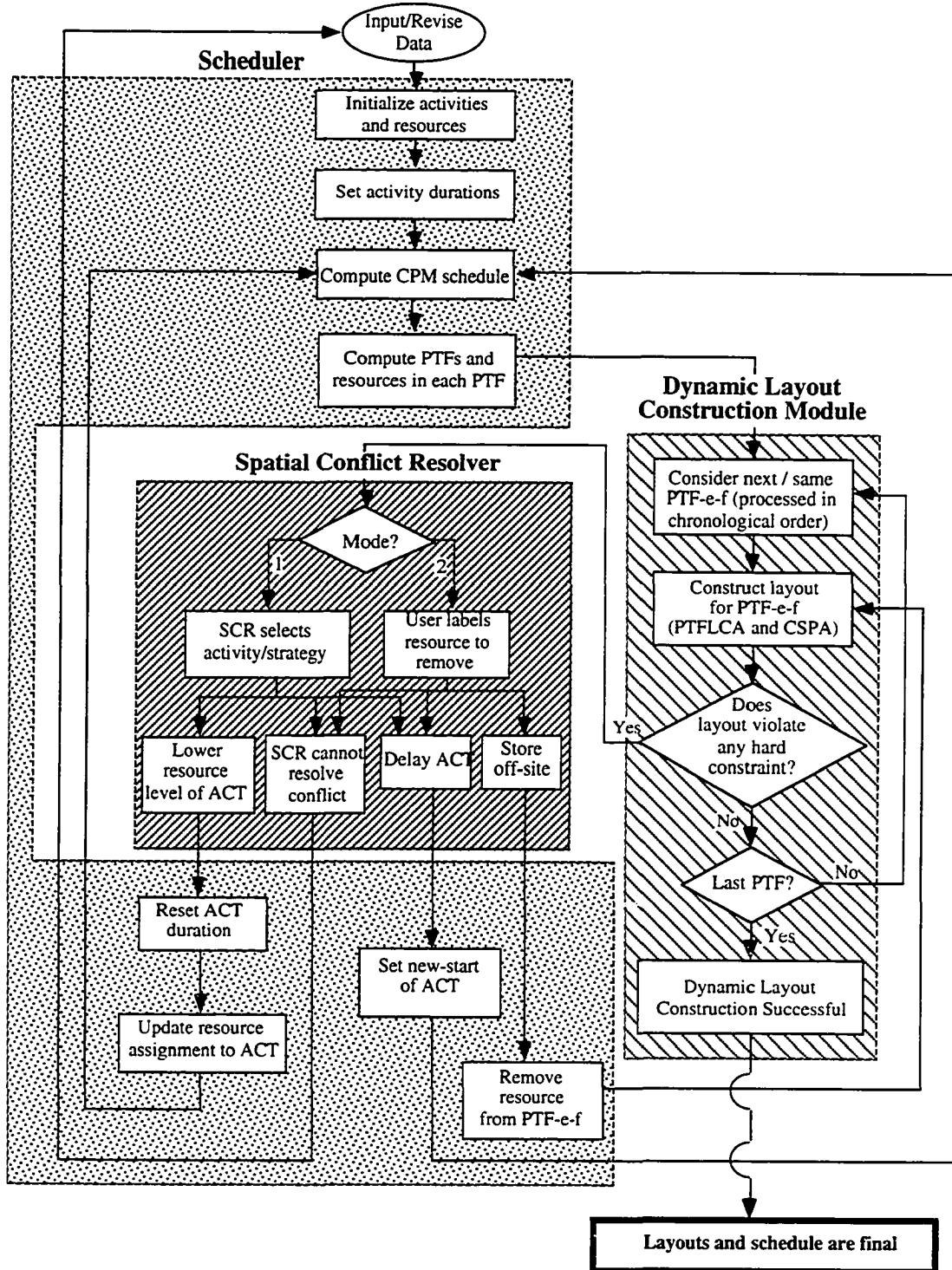


Figure 5.8  
Flow Chart of Space Scheduling Algorithm

Input to the algorithm are the activity network, the arrival and departure times of independent resources, the resource levels for each activity, the relocation weights of resources, and the positions of static resources if applicable.

The algorithm starts from the unconstrained schedule corresponding to the shortest project duration by setting each activity to be performed at its shortest duration (i.e., at the maximum resource level if three resource levels are defined, at the normal level if only two resource levels are defined, otherwise at the minimum level as this will be the only level at which the activity can be performed). It lengthens the project duration as needed during dynamic layout construction to decrease the total space required in problematic time frames. The objective is to construct a feasible sequence of layouts while keeping the project duration minimal. The algorithm goes through the following steps:

1. Call the scheduler to compute the unconstrained schedule corresponding to the shortest project duration and determine the corresponding PTFs.
2. Call the dynamic layout construction module to construct the layout of each PTF, one at a time, in sequential order.
  - 2.a This module applies CSPA and PTFLCA to construct the layout of the next PTF-e-f. Additional input to both algorithms are the hard constraints and proximity weights for PTF-e-f.
  - 2.b If CSPA or PTFLCA could not construct the layout of the current PTF (PTF-e-f), the dynamic layout construction module halts and declares a spatial conflict in that PTF.
3. If a spatial conflict is declared, then ask the user to choose a mode for operating SCR, otherwise go to 8.
4. Call SCR. Depending on the user-selected mode, SCR can suggest one of three strategies to lower the total area requirement or remove the user-labeled resource from PTF-e-f at minimum increase in project duration. SCR calls for user intervention when it cannot suggest a strategy.
5. Call the scheduler to apply the strategy posted by SCR and to update the schedule and corresponding PTFs.
6. If SCR could not suggest a strategy, then go to 1, otherwise go to 7.
7. Call PTFLCA to start anew with the layout construction of PTF-e-f or of some PTF-e-g because changes to the schedule may affect the finish date of PTF-e-f, the resources present in it, and the interactions and hard constraints between them. Go to 2.b.

8. If the layouts for all PTFs are successfully constructed, the current schedule is final and the space scheduling process terminates, otherwise some PTFs remain to be laid out, so go to 2.a.

The motive behind starting with an unconstrained schedule that corresponds to the shortest project duration is that this schedule also corresponds to the highest demand for space and is expected to lead to the largest number of spatial conflicts to be solved.

Note that starting with an early-start and shortest-duration schedule de facto limits the possible changes to either lengthening or delaying activities, thus forcing the changes to be in a forward pass only. Developing a forward pass approach is useful because it can be used to solve spatial conflicts that could not be anticipated at the planning stage (i.e., because of uncertainties associated with the timing of construction operations, deliveries of resources, and estimating space needs of resources) but that can occur when construction is in progress. In that case delaying on-going activities is not feasible and interrupting them may not be practical or useful because their corresponding resources may remain on-site anyway and therefore continue to claim space.

## 5.7 Example Application

### 5.7.1 Input

The following example illustrates the use of the space scheduling algorithm in assessing the feasibility of the schedule for a concrete foundation project and in adjusting this schedule to comply with site space limitations. The activity network shown in Fig. 5.9 describes the construction of the north and east foundation walls of a building. Solid lines between the boxes that represent activities are finish-to-start links. The trenching and excavation work preceding the foundation walls' construction are completed and are not part of the network.

The site dimensions are 22 by 12 units. The north trench is 2 units wide and 20 units long and runs along the northern site limit. The west trench is 2 units wide and 12 units long and runs along the west side. The east side of the site is fenced. The south side is only partially fenced with a reserved area of 4 by 3 located at 12 units from the south-east corner of the site which serves during this phase of construction as site access. The trenches, the fence lines and the site access are as shown in Fig. 5.10.

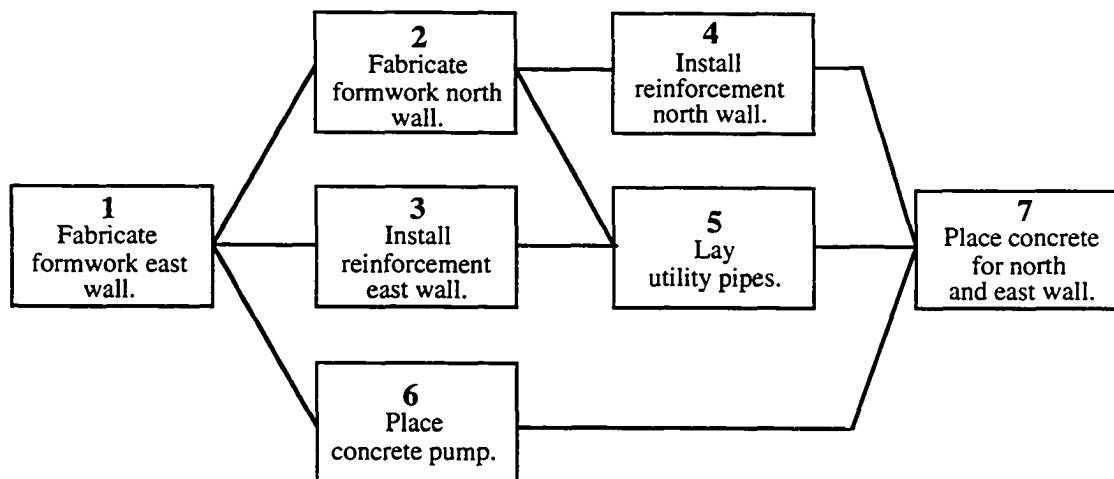


Figure 5.9  
Activity Network for Example Project

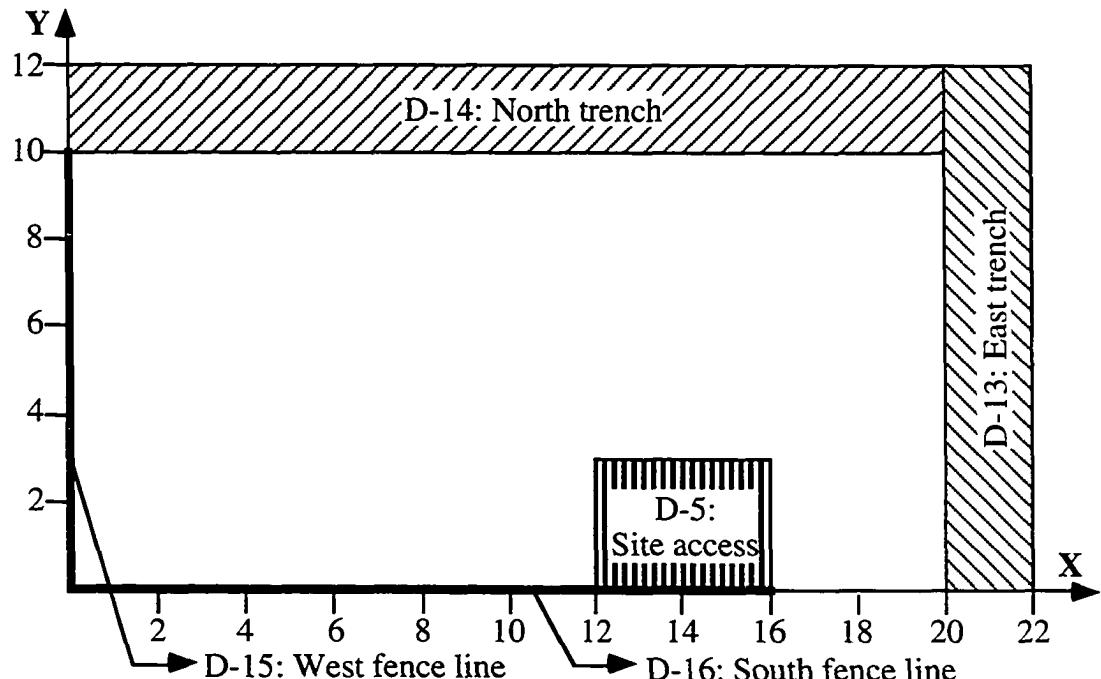


Figure 5.10  
Site Layout with Long-term Static Resources for Example Project

They are modeled as independent resources that exist on site for the duration of this foundation wall construction. Their space requirements are modeled using Profile-D. Furthermore, each of these resources is static and thus has a fixed, user-defined position on site. Table 5.3 summarizes the user input for these resources. In particular, the first column in that table shows the notation used in MoveSchedule for labeling resources: the

letter denotes the area profile of the resource (in this case D), the number is a unique identifier that differentiates between resources of the same profile type.

<b>Profile</b>	<b>Description</b>	<b>L x W</b>	<b>Time Interval</b>	<b>Relocation Weight</b>	<b>Fixed Position [X, Y, Orientation]</b>
D-5	Site access	4 x 3	0 -> 20	Stationary	[14, 1.5, 0]
D-13	East trench	12 x 2	0 -> 20	Stationary	[21, 6, 90]
D-14	North trench	20 x 2	0 -> 20	Stationary	[10, 11, 0]
D-15	West fence line	10 x 0	0 -> 20	Stationary	[0, 5, 90]
D-16	South fence line	16 x 0	0 -> 20	Stationary	[8, 0, 0]

**Table 5.3**  
Independent Resources' Data for Example Project

Note that the duration of this construction phase is not known at the time the independent resources are input to MoveSchedule. In this case, the user can assign an arbitrarily large number to TB to guarantee that the resource will be on site for the duration of the project (20 is chosen for this project). In addition to the above, input to MoveSchedule includes the resource levels for each activity (see Table 5.4), and data defining the resources in each resource level (see Table 5.5). For the purpose of illustration and simplicity, only two resource levels are defined per activity.

Resources with fixed shape and dimensions (e.g., the pipelayer and the concrete pump) and resources used by more than one activity that remain idle on site in-between uses (e.g., the lumber fabrication shop and the welding shop) are modeled using Profile-C. Other dependent resources that get consumed in the construction process (e.g., aggregates, cement, rebar, pipes) are modeled using Profile-A and Profile-B depending on whether the total quantity of the resource is brought to the site in one or multiple deliveries. When the total quantity is brought to the site, then the total area needed by the resource is required as a single input that applies to all resource levels. When multiple deliveries of the resource are made, then the maximum area needed by the resource must be specified for each resource level.

Activity Number	Description	Duration	Resource Level		Area @ Start
			Level	[Type, Area]	
1	Fabricate formwork east wall	2	Normal	[B-2, 8] [C-1], [C-4]	80
		4	Minimum	[B-2, 6] [C-1], [C-4]	78
2	Fabricate formwork north wall	4	Normal	[B-8, 8] [C-1], [C-4]	80
		6	Minimum	[B-8, 6] [C-1], [C-4]	78
3	Install reinforcement east wall	2	Normal	[B-7, 8] [C-3]	15.84
		4	Minimum	[B-7, 4] [C-3]	11.84
4	Install reinforcement north wall	4	Normal	[B-10, 8] [C-3]	15.84
		5	Minimum	[B-10, 6] [C-3]	13.84
5	Lay utility pipes	2	Normal	[A-12, 12] [C-11 C-12]	24
		4	Minimum	[A-12, 12] [C-11]	18
6	Install concrete pump	7	Normal	[C-6]	12
7	Place concrete for north and east wall	2	Normal	[A-1, 6] [B-9, 9] [C-6]	27
		1	Minimum	[A-1, 6] [B-9, 4] [C-6]	22

Table 5.4  
Activities and Resource Levels for Example Project

Formwork lumber normally gets consumed in "fabricate formwork" activities and then builds up in "strip formwork" activities. In this example, however, only the fabrication activities are part of the network and thus only the consumption of lumber needs to be modeled. Profile-A or -B can be used to model the area requirement of lumber. Profile-B is chosen here to reflect the fact that a constant area will be reserved on site to accommodate the lumber for the duration of the corresponding activity. This same profile could be used to model the area needed to store stripped formwork (see section 5.8 for a discussion on how to use the area profiles in MoveSchedule to model the space needs of a resource that builds up on site).

<b>Profile</b>	<b>Description</b>	<b>L/W Ratio</b>	<b>L x W</b>	<b>Relocation Weight</b>	<b>Fixed Position</b>
A-1	Aggregates	1.5		100	—
A-12	Utility pipes	2		100	—
B-2	Lumber east	2		25	—
B-7	Rebar east	2		25	—
B-8	Lumber north	2		25	—
B-9	Cement bags	1		50	—
B-10	Rebar north	2		25	—
C-1	Lumber fabrication shop		8 x 8	Stationary	—
C-3	Welding shop		2.8 x 2.8	0	—
C-4	Tools trailer		4 x 2	100	—
C-6	Concrete pump area		4 x 3	Stationary	—
C-11	Pipelayer 1		3 x 2	0	—
C-12	Pipelayer 2		3 x 2	0	—

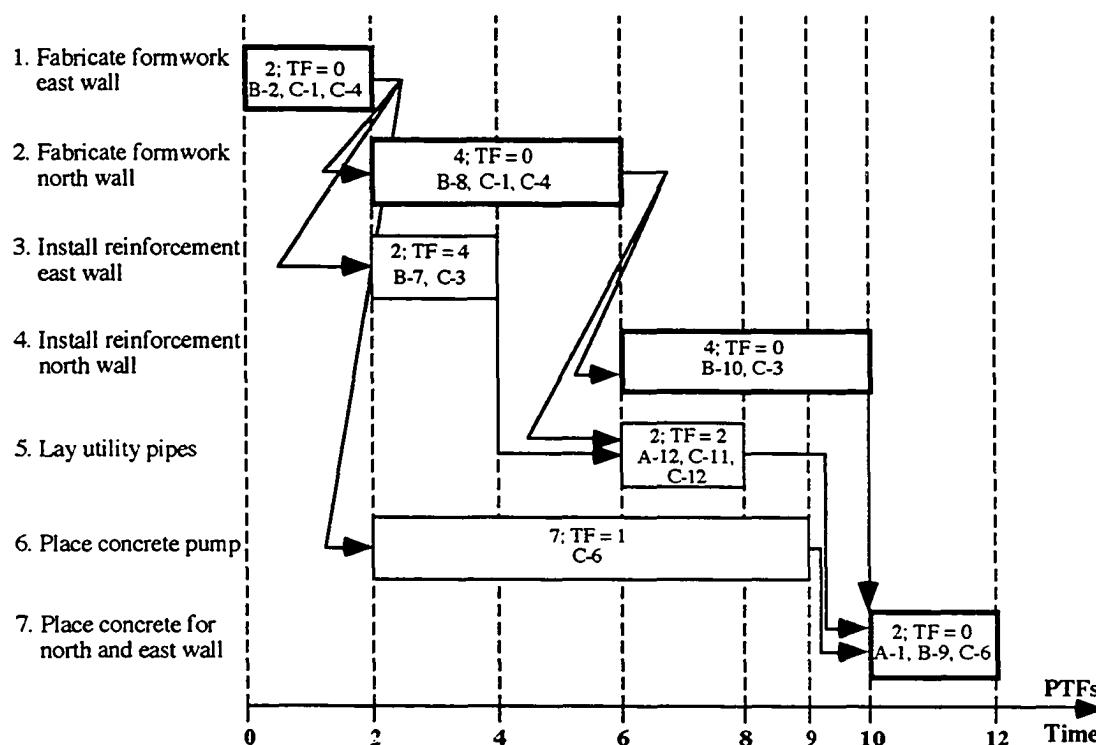
**Table 5.5**  
**Dependent Resource Data for Example Project**

Note that the relocation of the lumber fabrication shop and the concrete pump are restricted in this example. These resources are normally relocatable, but we chose to model them as stationary resources in this example to illustrate the capabilities of PTFLCA in handling stationary resources first and how this constrains the positions of other resources in subsequent layouts.

### 5.7.2 Computations

Initially, all activities are scheduled to be performed at their shortest duration, i.e., at the normal level. The scheduler first computes the corresponding unconstrained schedule and identifies the PTFs (shown in Fig. 5.11).

The feasibility of the unconstrained schedule is assessed by constructing the dynamic layout. For that, PTFLCA constructs the layouts of the individual PTFs in chronological order. Only the layout construction of PTF-0-2 and PTF-2-4 are detailed in this example to illustrate how the space scheduling algorithm works and how SCR solves spatial conflicts in the course of dynamic layout construction.



#### Notation:

Activity description	duration; total float resources
----------------------	---------------------------------

Figure 5.11  
Unconstrained Schedule

The number of trials in PTFLCA is arbitrarily limited to 5 trials per PTF. Ties between candidate resources are broken according to the third tie-breaking rule (see section 4.3.4) which is by highest relocation weight.

### Layout Construction of PTF-0-2

The resources that need positioning in PTF-0-2 are the lumber for the east wall (B-2), the lumber fabrication shop (C-1), and the tools trailer (C-4). Other resources that occupy space on site are the static resources D-5, D-13, D-14, D-15, and D-16. The proximity weights and hard constraints between them are as shown in Table 5.6 and Table 5.7 respectively.

B-2	C-1	C-4	D-5	D-13	D-14	D-15	D-16	
50	—	—	200	—	—	—	—	B-2
	100	—	—	—	—	—	—	C-1
		—	—	—	—	—	—	C-4
			—	—	—	—	—	D-5
				—	—	—	—	D-13
					—	—	—	D-14
						—	—	D-15
							—	D-16

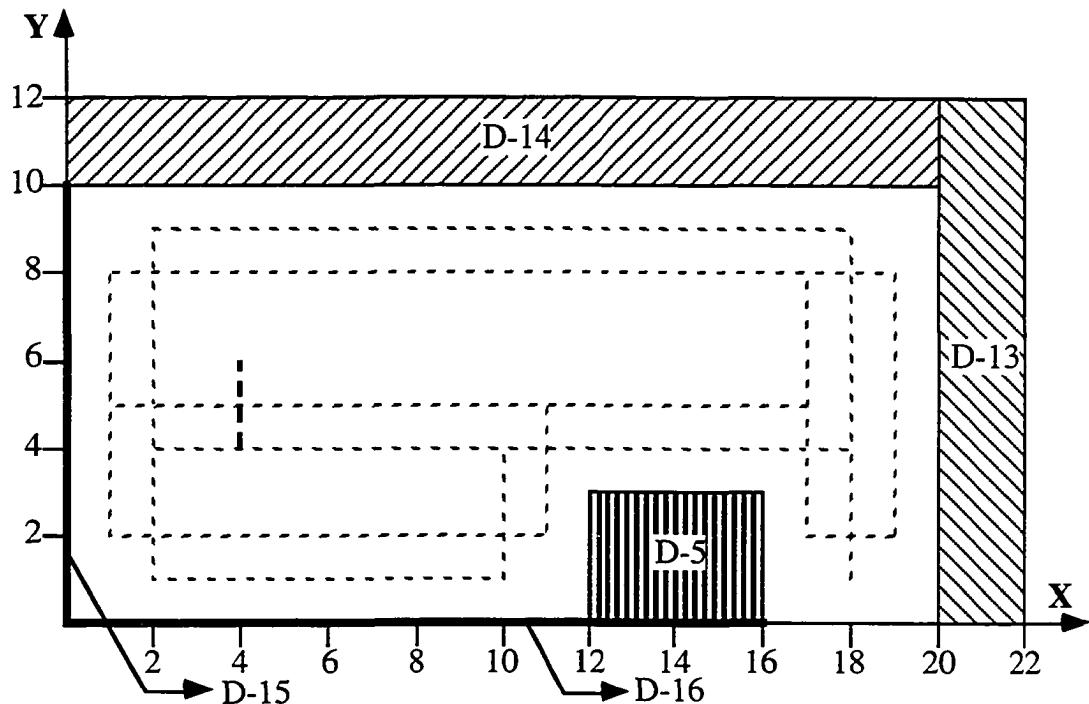
Table 5.6  
Proximity Weights in PTF-0-2

Constraint #	Resource 1	Resource 2	Constraint Type	Value
1	C-1	D-15	max Dx	0

Table 5.7  
Hard Constraints in PTF-0-2

The dimensions of B-2 are computed using its area requirement at time 0 (i.e., 8 from Table 5.4) and its L/W ratio (i.e., 2 from Table 5.5) and are L = 4 and W = 2.

CSPA is run first to determine SPPs of resources in PTF-0-2 that satisfy the hard constraints between them. It initializes  $SPP_{C-1}$ ,  $SPP_{B-2}$ , and  $SPP_{C-4}$  to satisfy the in-zone constraints with the site boundaries and the out-zone constraints with all static resources. Then, it prunes  $SPP_{C-1}$  to satisfy the maximum distance constraint with D-15 (from Table 5.7). The resulting SPPs are shown in Fig. 5.12.



Legend:

----- SPP<sub>C-1</sub> = { {0 {[4 4] [4 6]}} } {90 {[4 4] [4 6]}} }

[-----] SPP<sub>B-2</sub> = SPP<sub>C-4</sub> = { {0 {[2 10] [1 4]} {[2 18] [4 9]} {[18 18] [1 4]}} }  
{90 {[1 11] [2 5]} {[1 17] [5 8]} {[17 19] [2 8]}} }

Figure 5.12  
SPPs of Resources in PTF-0-2 after CSPA

PTFLCA is run next, using the SPPs output by CSPA, to determine single resource positions that minimize proximity and relocation costs. PTFLCA positions all stationary resources first and then all relocatables. C-1 is the only stationary resource in PTF-0-2. Because C-1 has no proximity weights with the static resources, PTFLCA samples at random a single position from SPPC-1 (shown in Fig. 5.12) and positions C-1 at X<sub>1</sub> = 4 and Y<sub>1</sub> = 5.8 @ 0°.

Among the relocatable resources, PTFLCA selects the one that has the highest sum of proximity weights with positioned resources. B-2 which has a total proximity weight of 250 (50 with the lumber fabrication shop C-1 and 200 with the east trench D-13) is selected over C-4, which has a total proximity of only 100 (100 with the lumber fabrication shop C-1). PTFLCA updates SPPB-2 (shown in Fig. 5.12) to reflect the positioning of C-1 using

the default out-zone constraint then further prunes the resulting SPP<sub>B-2</sub> to minimize the increase in  $\Delta$ VFL:

$$\begin{aligned} \min \Delta VFL = 2 \{ & ; \text{duration of PTF-0-2} \\ 50 (|X_2 - 4| + |Y_2 - 5.8|) & ; \text{proximity weight (from Table 5.6) } \times \\ & \quad \text{distance between B-2 and C-1} \\ + 200 (|X_2 - 21| + |Y_2 - 6|) & ; \text{proximity weight (from Table 5.6) } \times \\ & \quad \text{distance between B-2 and D-13} \end{aligned}$$

The values of  $X_2$  and  $Y_2$  that minimize  $\Delta$ VFL, subject to  $\{X_2, Y_2\} \in \text{SPP}_2$ , are  $X_2 = 19$  and  $Y_2 = 6 @ 90^\circ$  with  $\Delta$ VFL = 2,320 and VFL = 2,320.

PTFLCA positions C-4 next. It updates SPP<sub>C-4</sub> (from Fig. 5.12) to satisfy the out-zone constraints with C-1, B-2, and the static resources, then finds a single position for C-4 that minimizes  $\Delta$ VFL:

$$\begin{aligned} \min \Delta VFL = 2 \{ & ; \text{duration of PTF-0-2} \\ 100 (|X_4 - 4| + |Y_4 - 5.8|) & ; \text{proximity weight (from Table 5.6) } \times \\ & \quad \text{distance between C-4 and C-1} \end{aligned}$$

This yields  $X_4 = 9$  and  $Y_4 = 5.8 @ 90^\circ$  with  $\Delta$ VFL = 1,000 and VFL = 2320 + 1000 = 3,320. This step concludes the computations of one trial of PTF-0-2 layout construction.

Table 5.8 summarizes the resources' positions determined by PTFLCA in 5 trials. The order in which resource positions are presented in the third column of Table 5.8 corresponds to the order in which they are positioned in the layout. Note that this order is the same in all 5 trials because the resources did not tie and thus no randomness was needed to break ties. The variations in resource positions and VFL values of the 5 trials are the result of randomly sampling a position for C-1, which is the first resource to enter the layout. C-1's position will, in turn, constrain the positions of all the other resources in PTF-0-2. The layout of the fifth trial has the smallest value of VFL (3,304.6) and is chosen as the layout solution for PTF-0-2 (see Fig. 5.13).

Trial #	Success	Summary of Positions	VFL Value
1	yes	X1 = 4, Y1 = 5.8 @ 0° X2 = 19, Y2 = 6 @ 90° X4 = 9, Y4 = 5.8 @ 90°	3,320
2	yes	X1 = 4, Y1 = 5.6 @ 0° X2 = 19, Y2 = 6 @ 90° X4 = 9, Y4 = 5.6 @ 90°	3,337.4
3	yes	X1 = 4, Y1 = 5.93 @ 0° X2 = 19, Y2 = 6 @ 90° X4 = 9, Y4 = 5.93 @ 90°	3,306.3
4	yes	X1 = 4, Y1 = 5.4 @ 0° X2 = 19, Y2 = 6 @ 90° X4 = 9, Y4 = 5.4 @ 90°	3,360.6
5	yes	X1 = 4, Y1 = 5.95 @ 0° X2 = 19, Y2 = 6 @ 90° X4 = 9, Y4 = 5.95 @ 90°	3,304.6

Table 5.8  
Summary of Layout Solutions for PTF-0-2 in 5 Trials

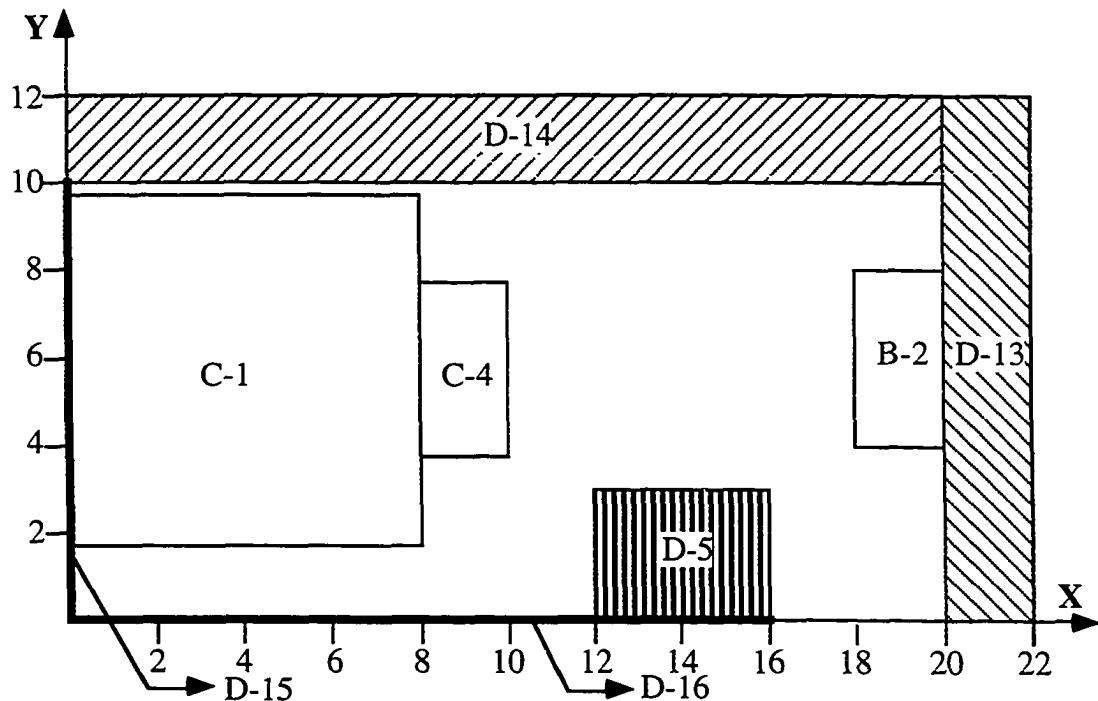


Figure 5.13  
Final Layout of PTF-0-2  
(Cross-hatched Resources are Static and Hollow Resources are Positioned by PTFLCA)

### Layout Construction of PTF-2-4

The resources that need positioning in PTF-2-4 are the relocatable and stationary resources from the layout of PTF-0-2 C-1 and C-4, and the new resources B-7, B-8, C-3, and C-6. The independent and static resources D-5, D-13, D-14, D-15, D-16 remain at their known positions.

The user is asked to input proximity weights (shown in Table 5.9) and hard constraints (shown in Table 5.10) between pairs of resources in PTF-2-4. The hard constraints concern the location of

- the welding shop (C-3) which should be at least 8 units of distance away from any combustible material storage (here, the lumber fabrication shop (C-1)), and 7 units of distance away from the concrete pump area (C-6) for safety reasons.
- the concrete pump (C-6) which should be along the southern fence line (D-16) for easy access.

B-7	B-8	C-1	C-3	C-4	C-6	D-5	D-13	D-14	D-15	D-16	
-	-	100	-	-	-	200	-	-	-	-	B-7
	50	-	-	-	-	-	200	-	-	-	B-8
		-	100	-	-	-	-	-	-	-	C-1
			-	-	-	-	-	-	-	-	C-3
				-	-	-	-	-	-	-	C-4
					-	-	-	-	-	-	C-6
						-	-	-	-	-	D-5
						-	-	-	-	-	D-13
						-	-	-	-	-	D-14
						-	-	-	-	-	D-15
						-	-	-	-	-	D-16

Table 5.9  
Proximity Weights in PTF-2-4

First, the dimensions and area requirements of B-7 and B-8 in PTF-2-4 are determined using their area requirement at time 2 and their respective L/W ratio. The area required by B-7 is 8 (Table 5.4), its L/W is 2 (Table 5.5), and its dimensions are therefore L = 4 and W = 2. Similarly, the dimensions of B-8 are found to be L= 4 and W= 2. The dimensions of all Profile-C and Profile-D resources are as input.

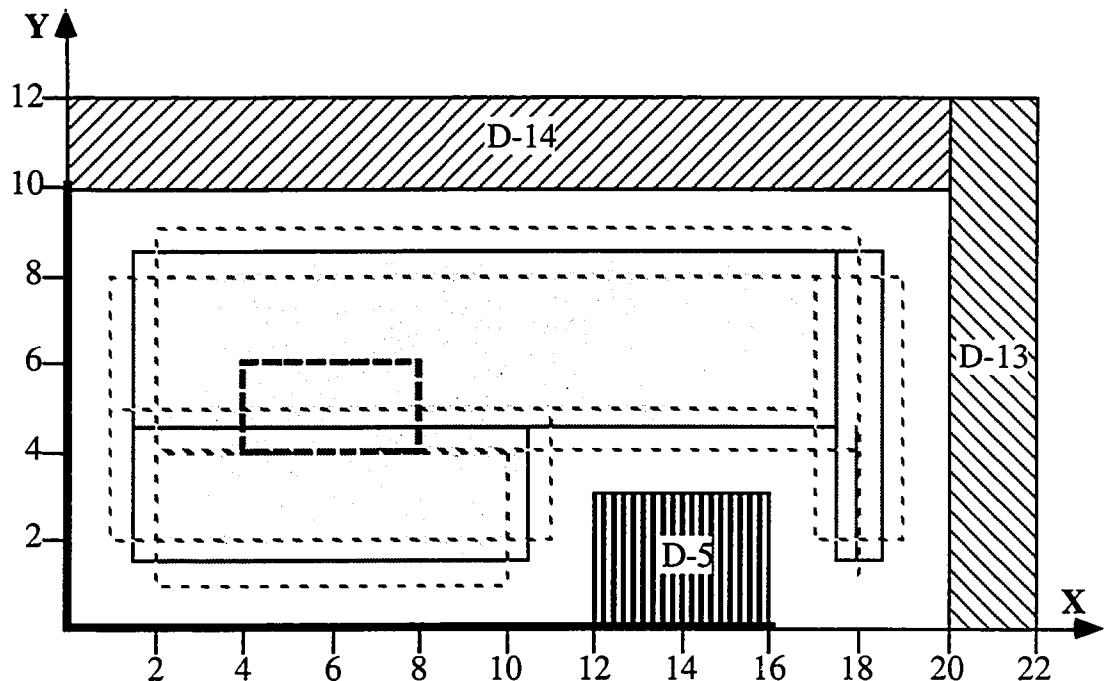
Constraint #	Resource 1	Resource 2	Constraint Type	Value
1	C-3	C-1	min Dx	8
2	C-3	C-6	min Dx	7
3	C-6	D-16	max Dy	0

Table 5.10  
Hard Constraints in PTF-2-4

CSPA initializes the SPP of B-7, B-8, C-1, C-3, C-4, and C-6 to satisfy the default in-zone constraints with the site boundaries and the default out-zone constraints with the static resources (shown in Fig. 5.14). Then, CSPA further prunes SPPC-1, SPPC-3, SPPC-4, SPPC-6, SPPB-7, and SPPB-8 to satisfy the hard constraints of Table 5.10. The resulting SPPs are as shown in Fig. 5.15.

Next, PTFLCA checks if the previously-positioned stationary resources can occupy the same positions in this layout. SPPC-1 (from Fig. 5.14) indeed includes C-1's position in the layout of PTF-0-2 so PTFLCA keeps it at that position.

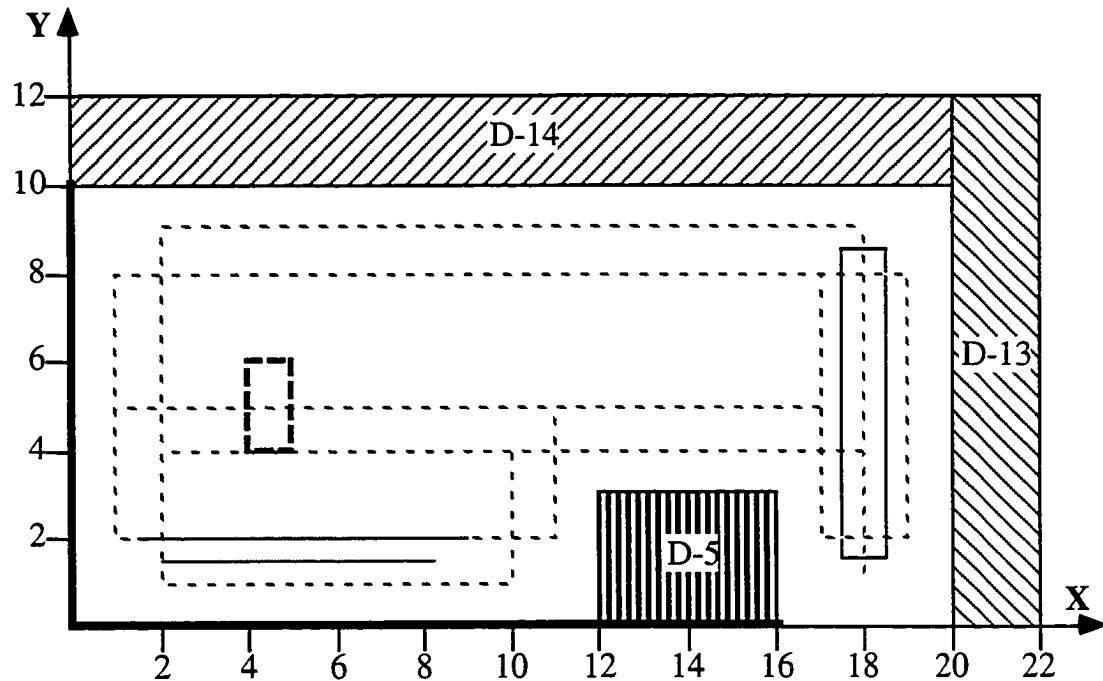
PTFLCA then positions all stationary resources that have not been positioned yet in any layout. C-6 is the only candidate. PTFLCA updates SPP6 (from Fig. 5.15) to satisfy the default out-zone constraint with C-1 at its position  $X_1 = 4$  and  $Y_1 = 5.95$  (fifth trial in Table 5.8). This results in a null set of feasible positions for C-6, which means that C-6 cannot be positioned without overlapping with C-1. Hence, PTF-2-4 layout construction halts and the spatial conflict is sent to SCR.



**Legend:**

- [---]  $SPP_{B-7} = \{ \{0 \{([2 18] [4 9]) ([2 10] [1 4]) ([18 18] [1 4])\} \}$   
 $\{90 \{([1 11] [2 5]) ([1 17] [5 8]) ([17 19] [2 8])\} \} \}$
- [---]  $SPP_{C-1} = \{ \{0 \{([4 8] [4 6])\} \{90 \{([4 8] [4 6])\} \} \}$
- [---]  $SPP_{C-3} = \{ \{0 \{([1.4 10.6] [1.4 4.6]) ([1.4 17.4] [4.6 8.6]) ([17.4 18.4] [1.4 8.6])\} \}$   
 $\{90 \{([1.4 10.6] [1.4 4.6]) ([1.4 17.4] [4.6 8.6]) ([17.4 18.4] [1.4 8.6])\} \} \}$
- [---]  $SPP_{C-6} = \{ \{0 \{([2 10] [1.5 4.5]) ([2 18] [4.5 8.5]) ([18 18] [1.5 4.5])\} \}$   
 $\{90 \{([1.5 10.5] [2 5]) ([1.5 17.5] [5 8]) ([17.5 18.5] [2 8])\} \} \}$

Figure 5.14  
 SPPs of Resources in PTF-2-4 before Satisfying Hard Constraints



**Legend:**

- [---]  $SPP_{B-7} = SPP_{B-8} = SPP_{C-4} = \{ \{0 \{([2 18] [4 9]) ([2 10] [1 4]) ([18 18] [1 4])\} \} \{90 \{([1 11] [2 5]) ([1 17] [5 8]) ([17 19] [2 8])\} \} \}$
- [---]  $SPP_{C-1} = \{ \{0 \{([4 5.2] [4 6])\} \{90 \{([4 5.2] [4 6])\} \} \}$
- [ ]  $SPP_{C-3} = \{ \{0 \{([17.4 18.6] [1.4 8.6])\} \{90 \{([17.4 18.6] [1.4 8.6])\} \} \}$
- $SPP_{C-6} = \{ \{0 \{([2 8.2] [1.5 1.5])\} \{90 \{([1.5 8.7] [2 2])\} \} \}$

Figure 5.15  
SPPs of Resources in PTF-2-4 after Satisfying Hard Constraints

**Solving Spatial Conflict in PTF-2-4**

MoveSchedule asks the user to select a mode for operating SCR. The user selects mode 1. In this mode, SCR looks for a strategy-activity combination to decrease the total area requirement in PTF-2-4 at minimum increase in project duration.

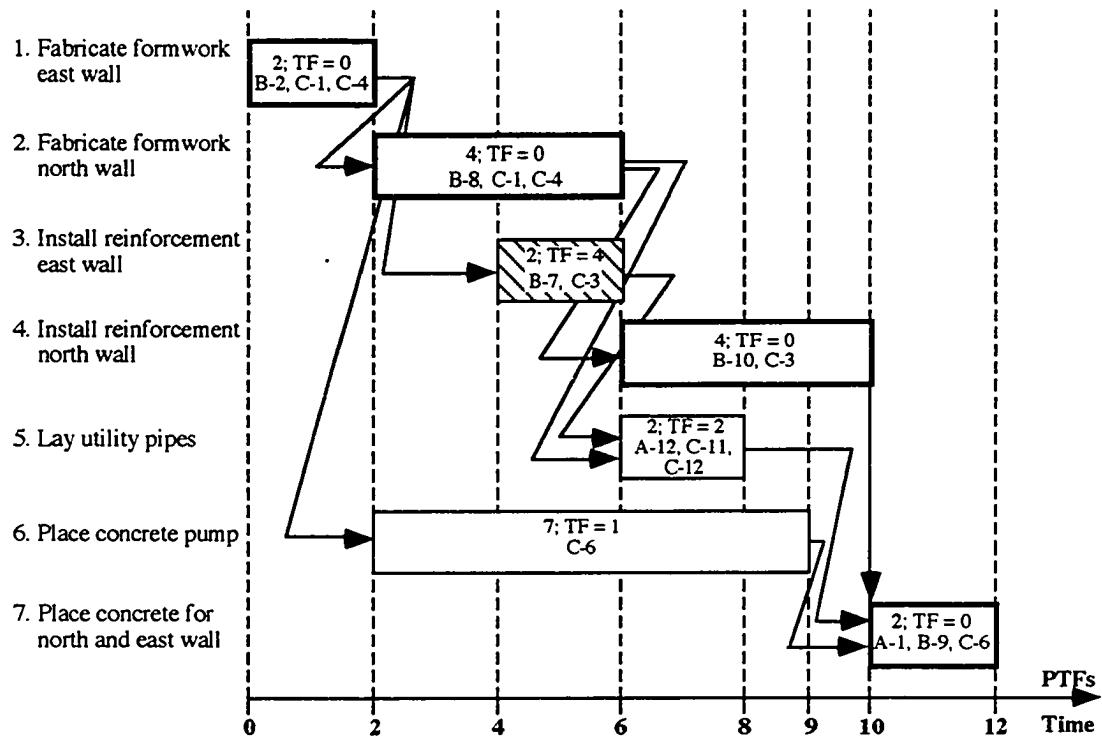
SCR calls strategy A to suggest an activity to be delayed. The expected delay is equal to the duration of PTF-2-4 which is 2. The *candidates* are activities 2, 3, and 6 with total float values of 0, 4, and 1 respectively. If delayed, their *remaining-total-float* values are -2 (i.e., 0 the total float of activity 2 - 2 the delay), 2 (i.e., 4 - 2), and -1 (i.e., 1 - 2) respectively. Strategy A returns activity-3 because it has the largest value of remaining-

total-float (2). If delayed, activity 3 will reduce the total area requirement in PTF-2-4 by 15.84 (Table 5.4).

SCR also calls strategy B to suggest an activity to lower its resource level. The candidate activities are activities 2 and 3 because both start at time 2 and their area requirement is smaller if performed at a lower resource level (Table 5.4). Although activity 6 also starts at time 2, it is not considered because it cannot be performed at a lower resource level (Table 5.4). The decrease in the area requirement of activities 2 and 3 are 2 (i.e., 80 - 78 from Table 5.4) and 4 (i.e., 15.84 - 11.84 from Table 5.4) respectively; and the increase-in-activity-duration are 2 (i.e., 6 - 4 from Table 5.4) and 2 (i.e., 4 - 2 from Table 5.4) respectively. Hence the *remaining-total-float* of activity 2 is 0 (its total float) minus 2 (its *increase-in-activity-duration*) which equals -2. Similarly, the *remaining-total-float* of activity 3 is 4 (its total float) minus 2 (its *increase-in-activity-duration*) which equals 2. Strategy B returns activity-3 because it has the largest value of *remaining-total-float* (2) and lowering its resource level will reduce the total area requirement in PTF-2-4 by 4.

SCR selects the strategy-activity combination with the highest value of remaining-total-float. Both strategies A and B tie with a value of 2. SCR breaks the tie by selecting the one that causes the largest decrease in the total area requirement in PTF-2-4 and thus selects strategy A and activity-3.

The scheduler is then called to delay activity-3 to start at time 4 which is the end of PTF-2-4. The scheduler updates the unconstrained schedule to reflect this change and it recomputes the PTFs (Fig. 5.16).



**Notation:**

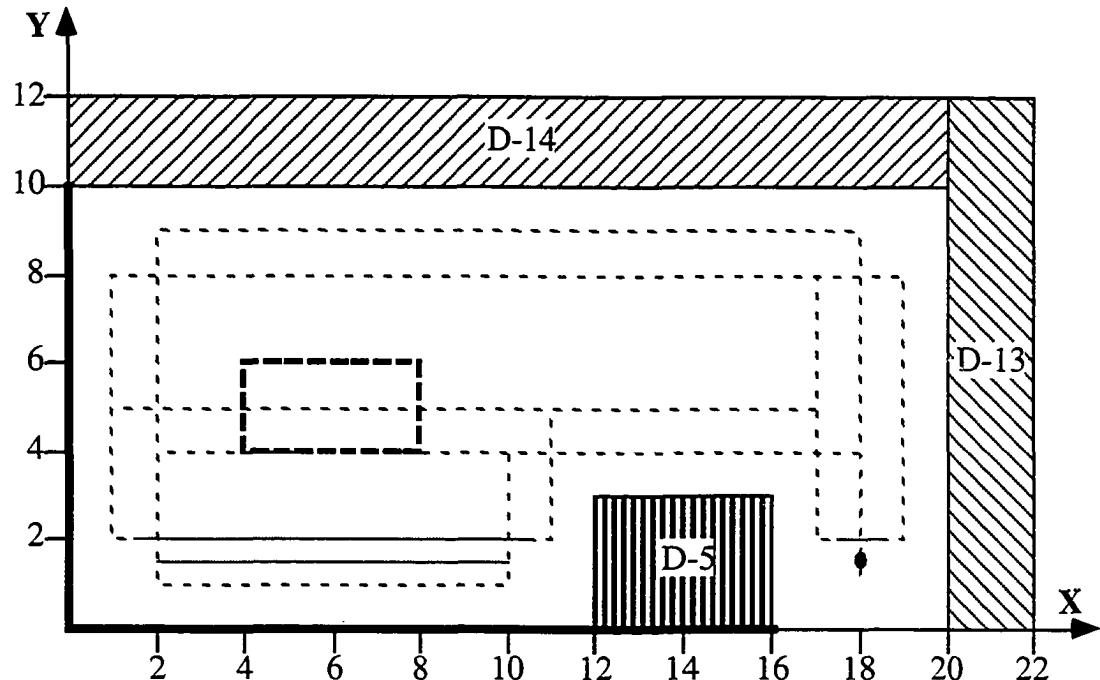
Activity description	duration; total float resources
----------------------	---------------------------------

Figure 5.16  
Updated Schedule and PTFs

**Layout Construction of PTF-2-4 after Schedule Modification**

PTFLCA starts anew with the layout construction of PTF-2-4, now including only B-8, C-1, C-4, and C-6 in addition to the static resources. The user is prompted to make changes to the proximity weights and hard constraints in PTF-2-4 if needed. Assume that the proximity weights remain as shown in Table 5.9, except that those involving C-3 and B-7 will be ignored because these resources no longer exist in this PTF. Similarly, the hard constraints shown in Table 5.10 remain, except for those involving C-3, which leaves only the third constraint.

The area of B-8 is as computed earlier for time 2. CSPA is called again to prune the SPPs of B-8, C-1, C-4, and C-6. The resulting SPPs are as shown in Fig. 5.17.



Legend:

- [ ] SPP<sub>B-8</sub> = SPP<sub>C-4</sub> = { {0 {[2 18] [4 9]} {[2 10] [1 4]} {[18 18] [1 4]} } } {90 {[1 11] [2 5]} {[1 17] [5 8]} {[17 19] [2 8]} } }
- [ ] SPP<sub>C-1</sub> = { {0 {[4 8] [4 6]} } {90 {[4 8] [4 6]} } }
- SPP<sub>C-6</sub> = { {0 {[2 10] [1.5 1.5]} {[18 18] [1.5 1.5]} } } {90 {[1.5 10.5] [2 2]} {[17.5 18.5] [2 2]} } }

Figure 5.17  
SPPs of Resources in PTF-2-4 after Spatial Conflict and after Satisfying Hard Constraints

PTFLCA first positions stationary resources that were positioned in previous layouts, then all other stationaries. It positions C-1 at the same position it occupied in PTF-0-2 after checking if SPP<sub>1</sub> (from CSPA in Fig. 5.17) allows this. Next it positions the stationary C-6. It updates SPP<sub>C-6</sub> (Fig. 5.17) to satisfy the out-zone constraint with C-1. The resulting SPP<sub>C-6</sub> is { {0 {[10 10] [1.5 1.5]} {[18 18] [1.5 1.5]} } } {90 {[9.5 10.5] [2 2]} {[17.5 18.5] [2 2]} } }. C-6 does not have proximity weights with any resource in PTF-2-4, hence, PTFLCA randomly selects a position for it from SPP<sub>C-6</sub>. The remaining resources to be positioned are the relocatables B-8 and C-4. B-8 is selected first because it has the highest sum of proximity weights (250) with the positioned resources. SPP<sub>B-8</sub> (Fig. 5.17) is updated to satisfy the out-zone constraints with C-1 and C-6. The resulting SPP<sub>B-8</sub> is { {0 {[10 18] [4 9]} {[18 18] [1 4]} } } {90 {[9 17] [5 8]} {[17 19] [2 8]} } }.

The position of B-8 that minimizes  $\Delta VFL = 2 \{ 200 (|X_8 - 10| + |Y_8 - 11|) + 50 (|X_8 - 4| + |Y_8 - 5.95|) \}$  is  $X_8 = 10$  and  $Y_8 = 9 @ 0^\circ$ . PTFLCA selects C-4 last. Similarly SPPC-4 is updated to satisfy the out-zone constraints with C-1, C-6, and B-8 then to minimize  $\Delta VFL = 2 \{ 100 (|X_4 - 4| + |Y_4 - 5.95|) \} + 100 (|X_4 - 9| + |Y_4 - 5.95|)$ . A single position for C-4 at  $X_4 = 9$  and  $Y_4 = 5.95 @ 90^\circ$  minimizes  $\Delta VFL$ . This position happens to be C-4's position in PTF-0-2, so this resource need not be relocated. Table 5.11 summarizes the resources' positions found in 5 trials.

Trial #	Success	Summary of Positions	VFL Value
1	yes	$X_1 = 4, Y_1 = 5.95 @ 0^\circ$ $X_6 = 10, Y_6 = 1.5 @ 0^\circ$ $X_8 = 10, Y_8 = 9 @ 0^\circ$ $X_4 = 9, Y_1 = 5.95 @ 90^\circ$	<b>2,704.6</b>
2	yes	$X_1 = 4, Y_1 = 5.95 @ 0^\circ$ $X_6 = 18, Y_6 = 1.5 @ 0^\circ$ $X_8 = 10, Y_8 = 9 @ 0^\circ$ $X_4 = 9, Y_1 = 5.95 @ 90^\circ$	2,704.6
3	yes	$X_1 = 4, Y_1 = 5.95 @ 0^\circ$ $X_6 = 10, Y_6 = 1.5 @ 0^\circ$ $X_8 = 10, Y_8 = 9 @ 0^\circ$ $X_4 = 9, Y_1 = 5.95 @ 90^\circ$	2,704.6
4	yes	$X_1 = 4, Y_1 = 5.95 @ 0^\circ$ $X_6 = 10, Y_6 = 1.5 @ 0^\circ$ $X_8 = 10, Y_8 = 9 @ 0^\circ$ $X_4 = 9, Y_1 = 5.95 @ 90^\circ$	2,704.6
5	yes	$X_1 = 4, Y_1 = 5.95 @ 0^\circ$ $X_6 = 18.5, Y_6 = 2 @ 90^\circ$ $X_8 = 10, Y_8 = 9 @ 0^\circ$ $X_4 = 9, Y_1 = 5.95 @ 90^\circ$	2,704.6

Table 5.11  
Summary of Layout Solutions for PTF-2-4 in 5 Trials

The order in which resource positions are presented in the third column corresponds to the order in which resources are positioned in the layout. Note that this order is the same in all 5 trials because resources did not tie and thus no randomness was needed to break ties. Furthermore, note that the resources' positions are the same in all 5 trials except for C-6's position. C-6 does not have proximity weights with any resource in PTF-2-4 and thus an alternative position was sampled from its SPP in each trial. In contrast, B-8 and C-4 have proximity weights with other resources in PTF-2-4 and

SPPC-4 and SPPB-8 that minimize  $\Delta VFL$  were reduced to single positions for B-8 and C-4 in all 5 trials. Hence, because resources that have proximity weights with other resources in PTF-2-4 are positioned at the same positions in all 5 trials, each trial yields an alternative layout solution with the same VFL value of 2,704.6. The layout of the first trial is selected as the layout solution for PTF-2-4 and is shown in Fig. 5.18.

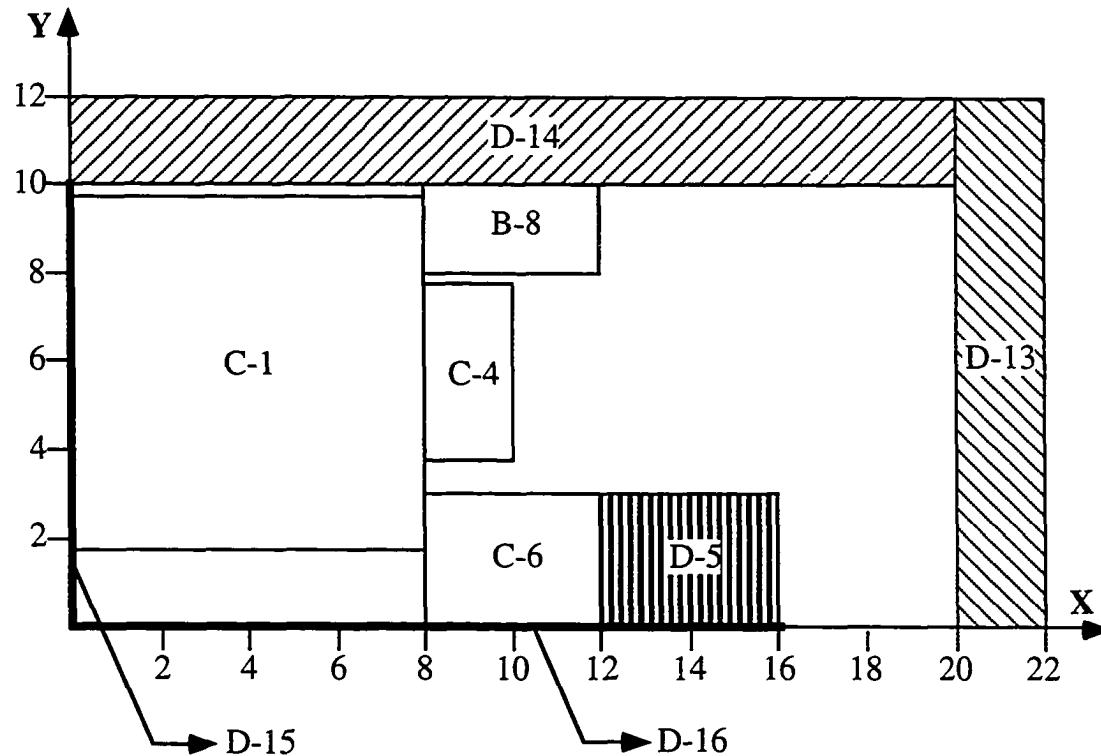


Figure 5.18  
Final Layout of PTF-2-4 from First Trial

### 5.7.3 Output

The output of MoveSchedule thus far are feasible site layouts for the time interval 0 to 2 and 2 to 4 and an adjusted activity schedule for this time period. The layout solution for PTF-0-2 and PTF-2-4 are as shown in Fig. 5.13 and Fig. 5.18 respectively. The adjusted schedule is as shown in Fig. 5.16. For this schedule to be final, feasible layouts for the remaining PTFs should be constructed.

## **5.8 Capabilities, Limitations, and Possible Extensions**

The following section rationalizes about the capabilities and limitations of the model, approach, and schedule-layout evaluation presented in this chapter. It also suggests ways of extending the current system as possible future research directions.

### **5.8.1 Model**

- **Use of Area Profiles as a Conservative Approximation of More Exact Area Profiles**

The resource area profiles modeled in MoveSchedule characterize the space needs of a wide variety of resources whose area requirements decrease or stay constant for the duration of an activity, are constant and independent of the duration of any activity, or are constant and independent of the timing and duration of any construction activity.

These area profiles are by no means exhaustive but can be used as a conservative approximation of more exact area profiles. For example, the area requirement of a resource that builds-up on site as the corresponding activity progresses (e.g., earth during excavation or forms during stripping formwork activities) can be conservatively modeled in MoveSchedule using Profile-B or Profile-C where a constant area is reserved on site to accommodate the resource at some average or at its maximum space requirement.

The space requirement of a resource that is used by more than one activity but does not occupy space on site when idle (because it can be stored off-site or it can be brought to site on an as-needed basis) can also be conservatively modeled in MoveSchedule. The resource can be divided into individual unit resources where each unit is associated with a single activity requiring its use and the space requirement of the unit is modeled using Profile-B or -C. These units are independent and will be on site for the duration of their corresponding activities only. The user, however, should be aware of the fact that, from MoveSchedule's standpoint, these units are different entities and the system will treat them as such. That is, if SCR happens to delay one of the activities using a unit of that resource, forcing it to overlap in time with another activity that uses another unit of the same resource, then MoveSchedule will reserve the space needed by two resources even though it may be acceptable that the two activities share a single resource. Again the model is

conservative as it may overestimate the space needed by the resources of the two activities (see discussion on shared resources in section 5.8.2).

Of course using the area profiles as a conservative approximation to resource space requirements may unduly constrain the layout problem and cause spatial conflicts. Solving these conflicts may result in delaying project completion. Future extensions to MoveSchedule should include an increasing area profile as this will not only accurately model the space needs of a resource that builds-up on site but will complete the basic building blocks required for constructing complex area profiles needed to accurately represent space needs of resources over time.

- **Modeling Mobilization and Demobilization Times**

A common modeling assumption that characterizes the area profiles of a dependent resource in MoveSchedule is that mobilization and demobilization times of the resource are zero. This assumption can be limiting if these time periods are relatively long, as in the case of a tower crane which may take a few days to be erected before it can be used.

To overcome this limitation, mobilization and demobilization times can be modeled as distinct activities that precede and succeed the production activity(ies) requiring the resource. The resource in this case should be modeled using Profile-C because it will be shared by the production, mobilization, and demobilization activities. A disadvantage to modeling mobilization and demobilization times as additional activities is the fact that they can be drawn apart from the main production activity when SCR delays the latter, thus forcing the resource to be on site for a longer time period. Future extensions to MoveSchedule should investigate mechanisms that keep designated activities together.

Alternatively, mobilization and demobilization times can be included in the duration of the activity requiring the resource. The resource in this case should be modeled using Profile-B or -C. It cannot be modeled using Profile-A because this profile suggests that the resource is consumed over the entire duration of the activity which is not true at start where the resource builds-up on site during mobilization time, and then gets consumed in the production activity. By modeling mobilization and demobilization times as an integral part of the activity duration, we ensure that SCR will not separate these from the main activity.

- **Resources Can Occupy Space from Early Start to Late Finish of Corresponding Activities**

The current model assumes that a dependent resource exists on site from the early start to the early finish of the corresponding activity(ies). Because of uncertainties in the timing and duration of activities, however, it may be that resources will actually occupy space on site for a time period longer than planned, for example, from the early start to the late finish of the corresponding activities.

The current model can be easily extended so that dependent resources exist on site for any time period related to the start and finish dates (early, late, or some conventional time) of corresponding activities. The only adjustment needed is in the way PTFs are computed. For example, instead of computing PTFs based on early start and finish dates of activities in the schedule, PTFs can be computed based on early start and late finish of activities. Information regarding both early and late times of activities is currently provided by the scheduler and can be readily used to determine PTFs according to some new definition. All other methods presented in this chapter and the previous one for adjusting the schedule as needed during dynamic layout construction are applicable and do not need to be adjusted.

- **Resource Levels**

The resource levels in MoveSchedule describe alternative construction methods or crew sizes for performing an activity with different durations. A minimum of one and up to three resource levels, where three is arbitrary, are defined for each activity in the network. SCR and the space scheduling algorithm can handle any number of resource levels provided that they are ordered in increasing order of activity duration and that the activity is initially scheduled to be performed at its shortest duration.

- **Simplified Strategy Selection**

By limiting the resource levels to comprise only multiples of the same resource, the following simplifications in the model are observed:

- Strategy B would not be applicable if all three resource levels comprise only a Profile-A resource because lengthening the activity duration does not affect the total area requirement of a Profile-A resource at the start of the activity.
- Strategy A would not be applicable if the current resource level of the activity comprises only a Profile-C resource that is also assigned to some earlier

activity(ies). This is so, because a Profile-C resource, by definition, exists on site from the start of the first activity to the finish of the last activity requiring its use. Hence, delaying the current activity would not result in removing the resource from site nor would it change the total area required on site.

### **5.8.2 Approach**

- **Modeling Shared Resources**

MoveSchedule allows the user to define resources that are shared by more than one activity which may or may not take place concurrently. Because it is the user that defines the resources in the different resource levels of an activity, it remains the user's responsibility to check that a resource is not initially over-assigned to activities. MoveSchedule can assist the user in this task by returning all the activities using a given resource in a given time frame and asking the user to confirm this assignment or refute it.

Throughout problem solving, MoveSchedule (SCR) does not check for resource availability and percentage use of resources when delaying activities as in strategy A or when changing the resource assignment to activities as in strategy B. If a resource cannot be shared by concurrent activities then additional precedence constraints should be defined between the activities to which the resource is assigned to prevent SCR from scheduling these activities simultaneously.

- **Compensate for the Drawbacks of a Myopic SCR**

SCR in the fully automated mode (mode 1) suggests one strategy at a time to solve a spatial conflict in a given problematic time frame. More than one strategy might be needed to solve that conflict, in which case SCR is repeatedly called in the process of the PTF layout construction until the spatial conflict(s) is solved. SCR does not undo the actions of previously applied strategies if a spatial conflict persists in the problematic time frame because it cannot judge whether the applied strategy was ineffective (i.e., it did not contribute to solving the conflict in the problematic time frame), or that more than one strategy is needed to solve the conflict. Hence, SCR can most benefit from mechanisms that allow it to reason about the cause of a conflict so that it can intelligently select strategies to undo.

Mode 2 in SCR is intended to address this weakness by allowing the user to pinpoint the cause of conflict by labeling resources for removal. In mode 2, SCR can

either delay the activities requiring the labeled resource or store the resource off-site with the user's approval.

In some cases however, mode 2 may result in delaying the conflict in time without ever solving it. For example, if the resource is too big to fit within the site boundaries, then delaying the activities using this resource would never solve the problem. This problem can be remedied by augmenting Mode 2 with strategies that change the construction method of an activity so that the new method does not require the labeled resource. This strategy is essentially similar to strategy B except that SCR would need to scan all resource levels of the activity and not just the next level down to select the level that corresponds to a suitable construction method.

Mode 2 can also be extended to accept more user-input, such as which strategies to select or which activities or resources to avoid selecting.

- **Reassess SCR's Modeling Assumptions**

Some assumptions underlying SCR necessarily limit the quality of its solutions. For example, spatial conflicts de facto slow down or delay activities; activities can be delayed at will, thus forcing due dates to be shifted which is unrealistic. The following discussion reassesses their limitations and the implications of relaxing them on the general applicability of the presented methods. It also suggests ways for extending the capabilities of the current system as future research directions.

#### ◊ **Interrupt On-going Activities**

SCR assumes that activities cannot be interrupted. This assumption is commonly made in most algorithmic approaches that solve a limited resource scheduling problem. Relaxing this assumption is relatively easy as it requires making changes to SCR and the scheduler only; no changes in the space scheduling algorithm or the dynamic layout construction module are needed. To this end, SCR should be extended to include a strategy that interrupts an on-going activity for the duration of the problematic time frame and schedules it to re-start at the beginning of the next time frame. Consequently, the heuristics that prioritize strategies should be revised. The scheduler should also be extended so that it can process preempted activities and compute intermediate start and finish dates for these activities.

#### **◊ Change Resource Area Profiles over the Activity Duration**

SCR assumes that the methods and resources applied for performing an activity are not changed in the course of performing it. Again, this assumption is commonly made in most algorithmic approaches that solve a limited resource scheduling problem. This assumption was taken a step further in SCR by assuming that the area profiles of the resources used by an activity do not change in the course of performing it. For example, the space occupied by a resource cannot decrease over part of the activity duration and remain constant for the remaining duration. Relaxing this assumption involves developing mechanisms that can lump one or more of the four area profiles in MoveSchedule together; these mechanisms can also be used for building more complex area profiles. It would also involve developing additional time-space tradeoff strategies that justify the change in area profile and investigating the implication of this change on the activity duration.

#### **◊ Make Changes to On-going Activities**

The time-space tradeoff strategies in SCR select activities that have started at the problematic time frame. This assumption is not unrealistic for the reasons discussed in section 5.6. Relaxing this assumption—by allowing on-going activities that had started in earlier PTFs to be delayed or lengthened—may change the total area requirement and the resources present in preceding PTFs and the start and end times of preceding PTFs. This in turn would require recomputing the PTFs from the start date of the selected activity onward, determining which resources exist in these new PTFs, and constructing their layouts to assess their feasibility. Hence, PTFLCA and the space scheduling algorithm should be augmented with backtracking capabilities before this assumption can be relaxed.

#### **◊ Constrain Start or Finish Dates of Activities**

SCR assumes that activities can be delayed at will; it does not limit activities to start or finish before some specified due date. In practice, there may be seasonal deadlines or fixed milestones (e.g., a subcontractor that can perform the work only in a given time window because of other commitments, etc.) that would prohibit delaying certain activities; or there may be a high cost associated with delaying them which should be taken into consideration before any decision can be made about delaying them.

The methods in MoveSchedule cannot be easily extended to accommodate constraints on start and finish dates of activities. At best, with the current methods,

activities can be tagged so that they do not get selected to be delayed, and so that if the delay or the lengthening of other activities in the network would cause these activities to have a negative float then such actions would not be pursued and SCR would have to seek other actions to resolve the spatial conflict at hand.

- **Additional Time-Space Tradeoff Strategies**

A limited set of time-space tradeoff strategies are implemented in MoveSchedule. These are by no means exhaustive, and future extensions of MoveSchedule should investigate other strategies. The previous discussion identified a few. The following suggests others which are in line with the prevailing modeling assumptions in MoveSchedule.

One important strategy is to shorten the duration of an on-going activity (i.e., increase its resource level) so that it finishes before the problematic time frame and therefore frees up the space it occupies on site during that time frame. Alternatively, this strategy may shorten the duration of an activity that finished prior to the start of the problematic time frame as this may expedite the completion of subsequent activities and possibly free up space in the problematic time frame.

This strategy was not implemented in the current system because the space scheduling algorithm starts from the minimum project duration schedule where all activities are initially scheduled to be performed at their minimum duration. Hence, shortening the duration of an activity that starts at or before the current problematic time frame is not possible. Had the space scheduling algorithm started from some conventional-duration schedule, this strategy would be needed because it does not increase the project duration.

Another strategy with a similar objective, i.e., to avoid increasing the project duration by shortening an activity duration, is to allow the user to add or redefine resource levels of an activity that occurred prior to the start of the problematic time frame so that it becomes possible to shorten its duration and possibly avoid the increase in project duration. Strategy D in MoveSchedule can be used to this end as it allows for user intervention in the course of problem solving. SCR can be easily modified so that strategy D is called prior to selecting any other strategy. This would allow the user to make changes in activities' resource levels at any stage of problem solving.

The shortcoming of doing so in the current system is that the space scheduling algorithm starts problem solving anew using the new data. Shortening the duration of an activity , or changing its resource levels can change the total area requirement and the

resources present in preceding PTFs. Implementing these strategies, thus requires that PTFLCA be augmented with backtracking capabilities so that it can revisit the layouts of selective, preceding PTFs and adjust them as needed.

Another strategy is to vary the area requirement or shape ratio of a selected resource to make it fit in the available space. For example, loose-shaped resources like sand, or modular resources like pallets of bricks, can be arranged to fit in spaces of different shape ratios and alternative L/W ratios can be used to describe their space requirements. Likewise, some bulk resources can fit in more than one area and their area requirements can be expressed as a range of acceptable values. This strategy requires extending PTFLCA so that it can reason about rectangles with different shape ratios and area requirements to modify a layout or resolve a conflict (see discussion on fixed L/W ratios in section 7.5.1).

Yet another strategy is to save alternative layouts to the next cycle and use them to solve spatial conflicts by rearranging resources in the layout. MoveSchedule generates alternative layouts for each PTF by running PTFLCA a given number of times and chooses the layout with the smallest value of VFL. These remaining layouts are discarded afterwards. This strategy suggests saving these layouts so that MoveSchedule can use them as alternatives to branch from when a spatial conflict is encountered in subsequent PTFs. This type of backtracking does not cost any additional computation time and can be easily implemented in MoveSchedule.

- **Alternate Approaches for Space Scheduling**

The space scheduling algorithm in MoveSchedule starts from the early start schedule corresponding to the shortest project duration and calls SCR to solve spatial conflicts as encountered during dynamic layout construction. SCR applies strategies that delay or lengthen activities that lower the total area requirement in the problematic time frame at a minimum increase in project duration.

Starting with an early-start schedule and shortest-duration schedule as the initial unconstrained schedule may not always reflect reality. The following suggests some alternative initial schedules that MoveSchedule can start from and assesses the adjustments or extensions needed to make the presented methods applicable in each case.

#### **◊ Start with a Late Start Schedule**

One possible initial schedule is a late start schedule corresponding to the shortest project duration and maximum space needs. The easiest way to extend the current methods in MoveSchedule to accommodate this change in initial schedule would be to reverse the current methods as follows. The scheduler will compute PTFs based on the late start and finish dates of activities (as opposed to early start and finish dates) and accordingly identify the resources needed in each PTF. The dynamic layout strategy will construct the layouts of the PTFs in sequential order starting from the last PTF (as opposed to the first) in the sequence. When a spatial conflict is detected, SCR can shift the start date of an activity that finishes at the problematic time frame forward in time (as opposed to delaying an activity that starts at the problematic time frame) so that it finishes (starts) at the start (finish) of the problematic time frame. Alternatively, SCR can lengthen the duration of an activity that finishes (as opposed to starts) at the problematic time frame. These reverse strategies will target activities that have a positive total back float first (as opposed to total float) and will consume this float before considering activities that will increase the project duration.

#### **◊ Start at Some Conventional Project Duration and Space Needs**

Another initial schedule is to start with an early start schedule that corresponds to some conventional project duration and space needs, i.e., activities are initially scheduled to be performed at some arbitrary level (e.g., normal level). By starting with a resource level different from the maximum resource level, strategies like shortening an activity (see additional time-space tradeoff strategies earlier in this section) are possible, or even needed, to maintain a short project duration. Because this approach would involve such strategies, PTFLCA needs to be augmented with backtracking capabilities so that the feasibility of previous layouts can be re-assessed.

#### **◊ Consider Alternate Dynamic Layout Construction Strategies**

Alternative dynamic layout construction strategies for constructing hierarchical and not necessarily sequential layouts were suggested in section 4.6.1 of Chapter 4. It was demonstrated then that the dynamic layout construction module in MoveSchedule can be very easily extended to accommodate these strategies. Assessing their compatibility with the space scheduling algorithm and SCR was deferred for after the presentation of the overall system.

By virtue of sequencing time frames in any order, these dynamic layout strategies can be computationally taxing if used with the current SCR, because they may result in a repeated layout construction of time frames before a feasible schedule and dynamic layout can be obtained. This point is best demonstrated with an example. Consider the scenario where the layouts of PTF-t2-t3, PTF-t3-t4, and PTF-t1-t2 are to be constructed in this order according to some dynamic layout strategy, where  $t_1 < t_2 < t_3 < t_4$ . Consider that the layout of PTF-t2-t3 was successfully constructed with no spatial conflict detected. The layout construction of PTF-t3-t4 was not successful, however, and SCR solved the spatial conflict in PTF-t3-t4. Solving the spatial conflict in PTF-t3-t4 did not perturb the layout of PTF-t2-t3 because SCR adjusts the schedule in a forward pass only. Now assume that the layout construction of PTF-t1-t2 results also in a spatial conflict. Solving this conflict can change the total area required or resources needed in PTF-t2-t3 and PTF-t3-t4. The begin and end times of these time frames may also change. This in turn would require reconstructing the layout of these time frames (or some other sequence of time frames resulting from the adjusted schedule) to assess their feasibility.

The space scheduling algorithm and SCR are closely tailored to suit the dynamic layout construction strategy implemented in MoveSchedule. This strategy constructs the layouts of PTFs in a chronological order and therefore avoids inefficient iterations.

### 5.8.3 Schedule-Layout Evaluation

- Schedule-Layout Costs

Two objectives drive MoveSchedule. The first objective is to minimize overall transportation and relocation costs of resources as characterized by VFL. The second objective is to minimize the increase in project duration while making adjustments to the activity schedule. VFL and the duration of the final schedule can therefore be used to evaluate the dynamic layout-schedule combination.

Recall that it is not within the scope of this research to find feasible solutions subject to budget constraints, rather the focus of this research was on investigating strategies for time-space tradeoffs subject to site space availability constraints. MoveSchedule thus neglects project and schedule-related costs such as the cost (or savings in cost) associated with performing an activity at a different resource level. These costs, however, should be considered when evaluating alternative strategies for time-space tradeoffs. By making the cost of performing an activity at a given resource level explicit in

the system, it is possible to extend the current time-space tradeoffs strategies to pursue time-space-cost tradeoffs (see section 8.3 of Chapter 8 for more details).

Other costs that should be considered when evaluating the dynamic layout-schedule combination are mobilization and demobilization costs. From the dynamic layout construction module's perspective, these costs are constant for a given input schedule and therefore are not modeled in VFL since they would not affect the results obtained by minimizing it. These costs, however, may vary in the overall system when SCR changes an activity's resource level. For example, when an activity is performed at a lower level (i.e., it takes longer to finish), smaller quantities of its Profile-B resources will be delivered to site over a longer time period, therefore possibly increasing total mobilization and delivery costs.

Another cost is the cost of keeping a shared resource idle for a longer time period when delaying its corresponding activities. Such a cost can be attributed to an increase in the rental cost or a loss in profit if the resource is a piece of equipment.

Other costs are also associated with delaying activities, even if the project duration is maintained. These costs are in terms of activity floats since all the strategies in MoveSchedule consume the available total float of activities.

Hence, future improvements of MoveSchedule must include the aforementioned costs into a multicriteria evaluation function of the dynamic layout-schedule combination.

- **Zero Relocation Time**

MoveSchedule does not model the time it takes to relocate a resource and does not consider the impact that this time may have on the duration of the corresponding activity and the overall schedule. It is not unrealistic, though, to assume that such relocation time is zero, as field practitioners may try to schedule the relocation of equipment or other resources that take a long time to relocate during off-hours or second shifts (Parsons and Pachuta 80).

In order to account for relocation times of resources in MoveSchedule, resources can be annotated with a value for the expected delay/unit distance traveled ( $T$ ) so that when the resource is relocated a distance  $D$ , the duration of its corresponding activity can be increased by an amount  $D \times T$ . Future extensions should consider the tradeoffs between the expected savings in layout costs ( $\Delta VFL$ ) and the expected increase in project duration to determine whether or not to relocate a resource.

## **5.9 Conclusions**

The time-space tradeoff model presented here uses three resource levels to describe alternative methods for performing each activity so that the corresponding activity duration can vary. Each resource level may impose a changing demand for space over the activity duration. A heuristic algorithm is developed to make changes to the activity schedule when insufficient space is available. The algorithm changes activities' resource levels or delays the start date of activities to vary the space requirements of activities over problematic time frames.

# **Chapter 6**

## **EXAMPLE APPLICATIONS OF MOVECHEDULE**

### **6.1 Introduction**

This chapter presents two problems as example applications of MoveSchedule. The first example applies PTFLCA, presented in Chapter 4, to optimally solve a MINISUM location problem with 2-dimensional constraints on acceptable locations of the new facility. The second example uses the resource levels and VFL to choose between alternative material delivery schedules for a pipelaying activity.

### **6.2 Example Use of MoveSchedule for Solving Constrained Location Problem**

#### **6.2.1 Problem Statement**

Location problems—in contrast to layout problems—are formulated to ignore the dimensions of the positioned facilities (resources) and abstract them to points. These

dimensions are considered irrelevant to the problem because they are negligible compared to the distances at which the facilities are located and relative to the open space.

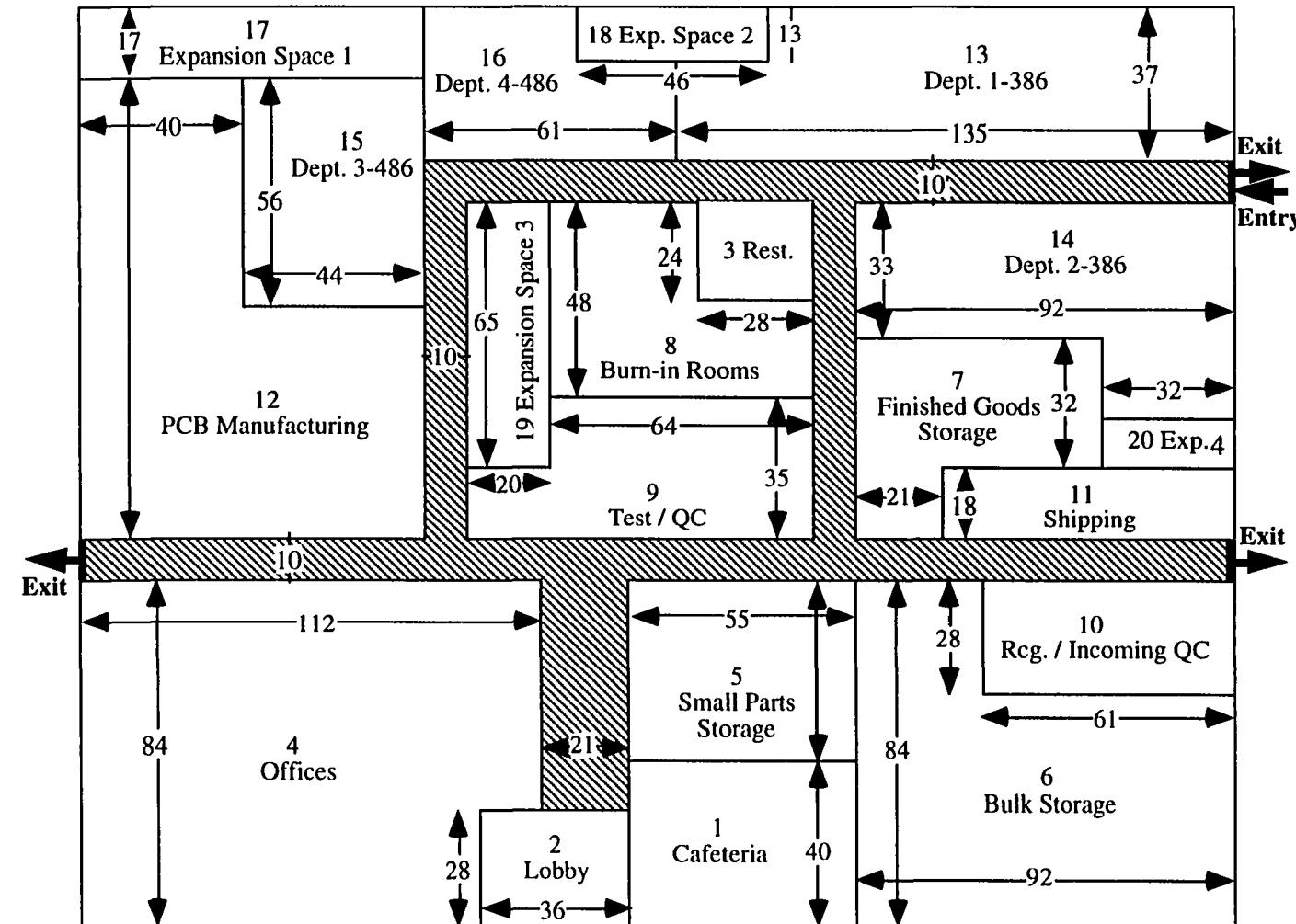
The problem considered here is a MINISUM location problem. It involves determining the location of a single new facility relative to existing ones so that the travel cost between the new facility and the existing ones is minimized. It is assumed that this travel cost is proportional to the rectilinear distance between the points representing the facilities. Hence, the problem is to determine the coordinates of the new facility which minimize the weighted sum of distances traveled. The MINISUM location problem becomes more complex with the addition of constraints on: 1) acceptable locations of the new facility, 2) dimensions of facilities, 3) available space. The latter problem is referred to here as the *constrained MINISUM location problem*.

The following example tests the applicability of PTFLCA for optimally solving a constrained MINISUM location problem. Since no time is being modeled, there is no need to invoice SCR. This example is chosen mainly because of data availability (Bozer 90) and because it is a single-facility location problem with non-overlap constraints and flow between the new facility and existing facilities.

The example is to locate a drinking fountain on the floor of a medium-sized industrial facility where desk-top personal computers are manufactured. The floor layout is as shown in Fig. 6.1. The planning departments and the data associated with each are shown in Table 6.1.

The optimal location of the fountain is to be determined. The fountain should not be located within any of the planning departments; it can only be located in a main aisle along the boundaries of departments. The location of the new fountain should minimize overall employee travel time associated with travel distances between their workspace and the fountain. Distances are measured as rectilinear distances. The number of employees in each department are listed in Table 6.1. It is assumed that all employees carry the same weight and are to be treated equally, and that the employees in a department are uniformly distributed over the area of that department.

Figure 6.1  
Floor Layout of a Manufacturing Facility



Department Number	Description	Number of Employees
1	Cafeteria	0
2	Lobby	0
3	Restrooms	0
4	Offices	90
5	Small parts storage	12
6	Bulk storage	0
7	Finished good storage	0
8	Burn-in rooms	0
9	Test / QC	0
10	Receiving / Incoming QC	0
11	Shipping	0
12	PCB manufacturing	34
13	Dept. 1-386	44
14	Dept. 2-386	60
15	Dept. 3-486	25
16	Dept. 4-486	30
17	Expansion space 1	0
18	Expansion space 2	0
19	Expansion space 3	0
20	Expansion space 4	0

Table 6.1  
Departments and Related Data

### 6.2.2 MoveSchedule's Solution

The floor layout shown in Fig. 6.1 is entered in MoveSchedule as follows. All departments are fixed in location and dimensions; they are therefore modeled using Profile-D. Because all resources in MoveSchedule are modeled as rectangles, L-shaped departments must be split into two sub-departments, each represented by a rectangle. The position of each (sub-)department is entered as the coordinate of the centroid of the rectangle representing it. Table 6.2 lists the dimensions and positions as entered in

MoveSchedule. Split departments are marked with their number followed by the letter a or b. They are shown separated by a dashed line in Fig. 6.2.

The fountain can be modeled with realistic dimensions (e.g., 1 ft x 1 ft) in MoveSchedule. In this problem, however, the intent is to use PTFLCA to solve a location problem, in which case, the fountain would be modeled with zero dimensions. If it is modeled with zero dimensions, MoveSchedule would find feasible positions for it along the shared boundaries of adjacent departments which is not acceptable. Therefore, the fountain had to be modeled with very small dimensions in MoveSchedule (see Table 6.2).

Because the optimal location for the fountain should minimize the total distance traveled by employees between their workplace and the fountain, the number of employees in each department is modeled as a proximity weight between the fountain and that department. As it is always the case in MoveSchedule, proximity weights are associated with the centroids of departments. The weights associated with L-shaped departments can be modeled using several methods, including:

1. distributing the number of employees over sub-departments' areas
2. defining a fictitious department to represent the number of employees at the true geometric centroid of each L-shaped department.

The results obtained by running PTFLCA using each method is presented next.

### **Method 1**

The number of employees of an L-shaped department is distributed over the two sub-departments, that represent the original L-shape, in proportion to their area. For example, sub-departments D-4 and D-5 represent department 4 (offices) which has a total area of  $112 \times 56 + 97 \times 28 = 8,988$  and a total number of employees equal to 90 (Tables 6.1 and 6.2). D-4 has an area equal to  $112 \times 56 = 6,272$  (Table 6.2), hence the number of employees proportional to its area is  $90 \times 6,272 / 8,988 = 63$ . The number of employees in D-5 is similarly computed and is equal to  $90 \times 97 \times 28 / 8,988 = 27$ . Table 6.3 shows the number of employees associated with each sub-department.

<b>Profile-D</b>	<b>Department Number</b>	<b>Description</b>	<b>L x W (ft x ft)</b>	<b>Position [X, Y, Orientation]</b>
D-1	1	Cafeteria	55 x 40	[160.5, 20, 0]
D-2	2	Lobby	36 x 28	[115, 14, 0]
D-3	3	Restrooms	28 x 24	[164, 165, 0]
D-4	4 a	Offices	112 x 56	[56, 56, 0]
D-5	4 b	Offices	97 x 28	[48.5, 14, 0]
D-6	5	Small parts storage	55 x 44	[160.5, 62, 0]
D-7	6 a	Bulk storage	92 x 56	[234, 28, 0]
D-8	6 b	Bulk storage	31 x 28	[203.5, 70, 0]
D-9	7 a	Finished good storage	21 x 18	[198.5, 103, 0]
D-10	7 b	Finished good storage	60 x 32	[218, 128, 0]
D-11	8 a	Burn-in rooms	64 x 24	[146, 141, 0]
D-12	8 b	Burn-in rooms	36 x 24	[132, 165, 0]
D-13	9 a	Test / QC	84 x 18	[136, 103, 0]
D-14	9 b	Test / QC	64 x 17	[146, 120.5, 0]
D-15	10	Rcg. / Incoming QC	61 x 28	[249.5, 70, 0]
D-16	11	Shipping	71 x 18	[244.5, 103, 0]
D-17	12 a	PCB manufacturing	113 x 40	[20 x 150.5, 90]
D-18	12 b	PCB manufacturing	57 x 44	[62, 122.5, 90]
D-19	13 a	Dept. 1-386	112 x 37	[224, 205.5, 0]
D-20	13 b	Dept. 1-386	24 x 23	[156.5, 199, 90]
D-21	14 a	Dept. 2-386	92 x 33	[234, 160.5, 0]
D-22	14 b	Dept. 2-386	32 x 20	[264, 134, 0]
D-23	15	Dept. 3-386	56 x 44	[62, 179, 90]
D-24	16 a	Dept. 4-386	61 x 24	[114.5, 199, 0]
D-25	16 b	Dept. 4-386	38 x 13	[103, 217.5, 0]
D-26	17	Expansion space 1	84 x 17	[42, 215.5, 0]
D-27	18	Expansion space 2	46 x 13	[145, 217.5, 0]
D-28	19	Expansion space 3	65 x 20	[104, 144.5, 90]
D-29	20	Expansion space 4	32 x 12	[264, 118, 0]
D-30	New	FOUNTAIN	0.1 x 0.1	?

**Table 6.2**  
**Departments and Related Data as Defined in MoveSchedule**

PTFLCA is used to solve for the location that minimizes travel distances and satisfies non-overlap constraints between the fountain and existing departments. PTFLCA first initializes and prunes the SPP of the fountain to satisfy the non-overlap constraints with all existing static departments. The resulting SPP<sub>30</sub> is equal to:

$$\begin{aligned} \text{SPP}_{30} = & \{ \{0 \{ [(178.05 187.95) (93.95 177.05)] [(112.05 132.95) (28.05 84.05)] \\ & [(84.05 279.95) (177.05 186.95)] [(84.05 93.95) (93.95 177.05)] \\ & [(0.05 279.95) (84.05 93.95)] \} \} \\ & [90 \{ [(178.05 187.95) (93.95 177.05)] [(112.05 132.95) (28.05 84.05)] \\ & [(84.05 279.95) (177.05 186.95)] [(84.05 93.95) (93.95 177.05)] \\ & [(0.05 279.95) (84.05 93.95)] \} \} \}. \end{aligned}$$

This set of rectangles in SPP<sub>30</sub> describes the coordinates of the corner points of the manufacturing floor's aisle structure (see Fig. 6.1) minus a 0.05 clearance which is half of the assumed 0.1 length and width of the fountain.

D-4	D-5	D-6	D-17	D-18	D-19	D-20	D-21	D-22	D-23	D-24	D-25	
63	27	12	21	13	39	5	45	15	25	22	8	D-30

Table 6.3  
Method 1: Proximity Weights with the Drinking Fountain

PTFLCA then solves for the optimal location of the fountain that minimizes

$$\begin{aligned} \text{VFL} = & 8 (|103 - X_{30}| + |217.5 - Y_{30}|) + 22 (|114.5 - X_{30}| + |199 - Y_{30}|) + \\ & 25 (|62 - X_{30}| + |179 - Y_{30}|) + 15 (|264 - X_{30}| + |134 - Y_{30}|) + \\ & 45 (|234 - X_{30}| + |160.5 - Y_{30}|) + 5 (|156.5 - X_{30}| + |199 - Y_{30}|) + \\ & 39 (|224 - X_{30}| + |205.5 - Y_{30}|) + 13 (|62 - X_{30}| + |122.5 - Y_{30}|) + \\ & 21 (|20 - X_{30}| + |150.5 - Y_{30}|) + 12 (|160.5 - X_{30}| + |62 - Y_{30}|) + \\ & 27 (|48.5 - X_{30}| + |14 - Y_{30}|) + 63 (|56 - X_{30}| + |56 - Y_{30}|) \quad (\text{eq. 6.1}) \end{aligned}$$

Subject to  $(X_{30}, Y_{30}) \in \text{SPP}_{30}$ .

Note that the R component of VFL is zero because the problem is static, and the duration of the layout ( $\Delta t$ ) is a constant which has been ignored in eq. 6.1 as it will not affect the minimization. PTFLCA transforms the above program into a linear program and solves it according to the method presented in section 4.3.3. It finds the unconstrained minimum at

$$X_{30}^* = 62 \text{ and } Y_{30}^* = 150.5 \text{ with } \text{VFL} = 38,315.5$$

and the optimal location that is part of SPP<sub>30</sub> at

$$X_{30} = 84.05 \text{ and } Y_{30} = 150.5 \text{ with } \text{VFL} = 38,381.65.$$

## Method 2

Fictitious departments are defined to represent the number of employees (and thus the proximity weight) at the true centroids of the L-shaped departments. They can have any dimension as long as they stay confined within the boundaries of their corresponding departments. This is required so that the added departments would not change the fountain's set of possible positions which should be outside the boundaries of the original departments. No weight will then be associated with the rectangular sub-departments that cover the area of the original L shape. Table 6.4 lists the additional fictitious departments and their related data.

Profile-D	Carrying Weight of Department		L x W	Position [X, Y, Orientation]	Number of Employees
	Dept. #	Description			
D-31	4 f	Offices	1 x 1	[49.54, 43.68, 0]	90
D-32	12 f	PCB manufacturing	1 x 1	[34.96, 138.98, 0]	34
D-33	13 f	Dept. 1-386	1 x 1	[215.9, 204.72, 0]	44
D-34	14 f	Dept. 2-386	1 x 1	[239.4, 155.73, 0]	60
D-35	16 f	Dept. 4-386	1 x 1	[111.54, 203.68, 0]	30

Table 6.4  
Additional Fictitious Departments and Related Data

Proximity weights can then be defined between the new departments and the fountain as shown in Table 6.5.

D-5	D-15	D-31	D-32	D-33	D-34	D-35	
12	25	90	34	44	60	30	D-30

Table 6.5  
Method 2: Proximity Weights with the Drinking Fountain

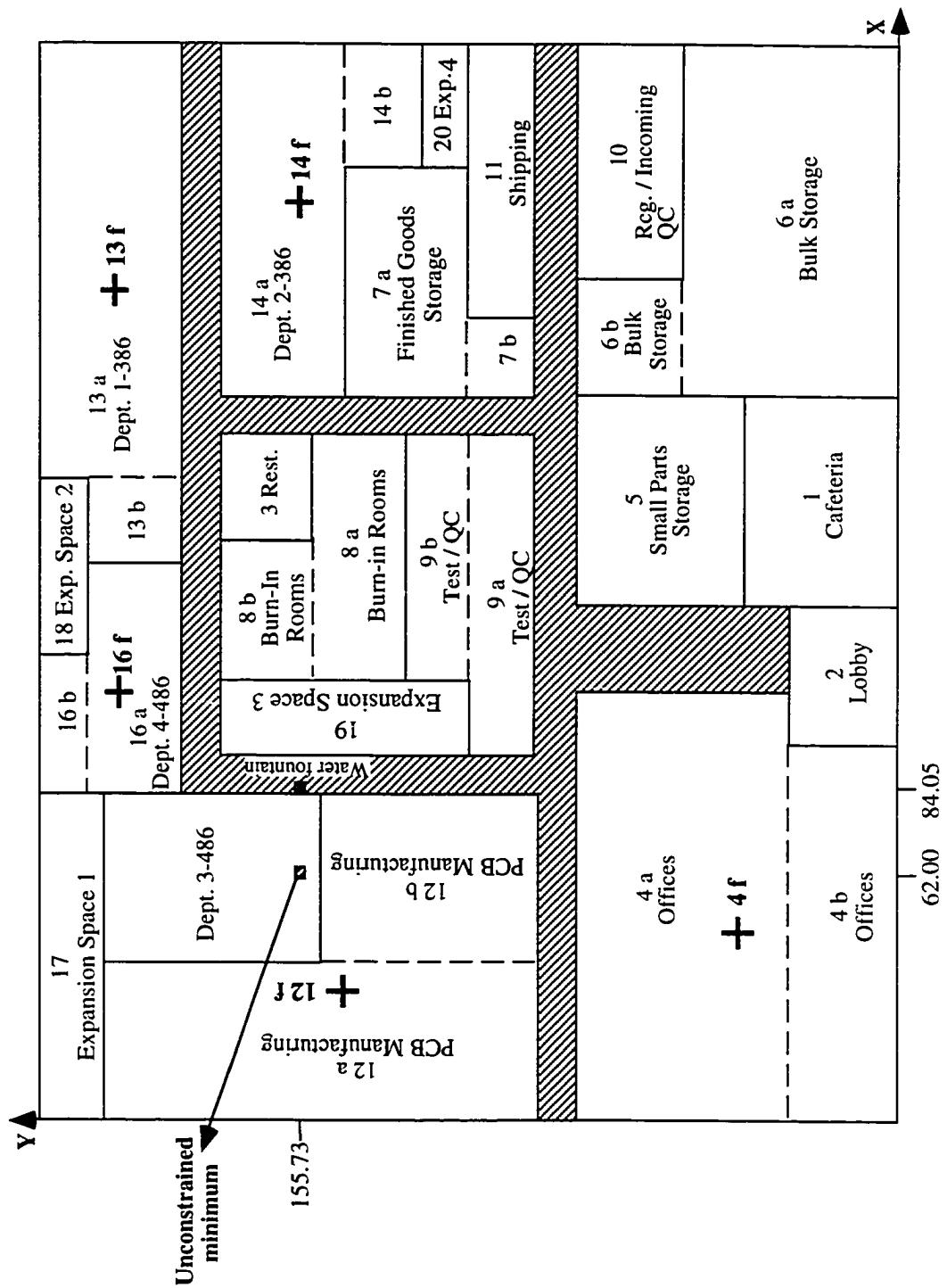


Figure 6.2  
MoveSchedule's Solution to the Drinking Fountain Problem using Fictitious Departments

PTFLCA finds SPP<sub>30</sub> to be identical to the one found when no fictitious departments were used. PTFLCA then solves for the optimal location of the fountain that minimizes

$$\begin{aligned} VFL = & 90(|49.54 - X_{30}| + |43.68 - Y_{30}|) + 12(|160.5 - X_{30}| + |62 - Y_{30}|) + \\ & 60(|239.4 - X_{30}| + |155.73 - Y_{30}|) + 25(|62 - X_{30}| + |179 - Y_{30}|) + \\ & 34(|34.96 - X_{30}| + |138.98 - Y_{30}|) + 44(|215.9 - X_{30}| + |204.72 - Y_{30}|) + \\ & 30(|111.54 - X_{30}| + |203.68 - Y_{30}|) \end{aligned} \quad (\text{eq. 6.2})$$

Subject to  $(X_{30}, Y_{30}) \in \text{SPP}_{30}$ .

PTFLCA finds the unconstrained minimum at

$$X^{*30} = 62 \text{ and } Y^{*30} = 155.73 \text{ with } VFL = 38,079.13$$

and the optimal location that is part of SPP<sub>30</sub> at

$$X_{30} = 84.05 \text{ and } Y_{30} = 155.73 \text{ with } VFL = 38,145.28.$$

Fig. 6.2 shows both points.

Because of the difference in modeling the weights associated with L-shaped departments, PTFLCA found different unconstrained minima and optimal locations for the fountain for each method. This is expected as it has been shown that the solution for the unconstrained minimum will be at one of the points  $h_k$  and  $h'_k$  (see in Chapter 4, section 4.3.3) and these are different in eq. 6.1 and eq. 6.2. Hence, the optima of the two equations are different. Method 2 more realistically represents the input of this problem because it does not redistribute employees over sub-departments and therefore leads to more accurate results.

PTFLCA's solution is compared next against that obtained by solving the problem using traditional methods.

### 6.2.3 Traditional Solution

The fountain location problem can be solved using a traditional two-step approach (Tompkins and White 84). It involves computing the optimum X- and Y-coordinate of the fountain, then using contour lines to find the location that is closest to optimum (i.e., in terms of minimizing travel distances) and that satisfies the constraints on its acceptable positions. Here too, the number of employees are modeled as weights associated with departments' centroids.

In the first step, the algorithm described in Tompkins and White (84) is used to determine the location that minimizes the sum of the weighted distances between the fountain and the departments carrying weights. The weights are carried at the true geometric centroids of departments, hence the objective function to be minimized is the same as when using PTFLCA with fictitious departments (eq. 6.2).

$$\begin{aligned} Z = & 90(|49.54 - X_{30}| + |43.68 - Y_{30}|) + 12(|160.5 - X_{30}| + |62 - Y_{30}|) + \\ & 60(|239.4 - X_{30}| + |155.73 - Y_{30}|) + 25(|62 - X_{30}| + |179 - Y_{30}|) + \\ & 34(|34.96 - X_{30}| + |138.98 - Y_{30}|) + 44(|215.9 - X_{30}| + |204.72 - Y_{30}|) + \\ & 30(|111.54 - X_{30}| + |203.68 - Y_{30}|) \end{aligned} \quad (\text{eq. 6.3})$$

The location that minimizes  $Z$  is found at

$X^*_{30} = 62$  and  $Y^*_{30} = 155.73$  with a value of  $Z = 38,079.13$ , which also was PTFLCA's unconstrained minimum.

This location lies within department 15, not in the aisle, and it is therefore unacceptable. To find a location along the aisles that minimizes  $Z$ , contour lines are used to guide the search for such location. Fig. 6.3 illustrates the method for drawing the contour lines and the final location for the fountain.

The method for drawing contour lines is as follows:

1. Draw vertical and horizontal lines passing through the centroids of all departments with weights (i.e., departments 4, 5, 12, 13, 14, 15, and 16). These lines partition the floor layout into sub-regions as shown by the dashed lines in Fig. 6.3, where the weights are shown in parentheses next to each department number and near its centroid.
2. Add the weights along each line. Let  $C_i$  be the value of the sum along the  $i^{\text{th}}$  dashed vertical line (counting from left to right) and shown at the bottom of it. Let  $D_j$  be the value of the sum along the  $j^{\text{th}}$  dashed horizontal line (counting from bottom to top) and shown to the left of it.
3. Let  $p$  be the total number of vertical lines and  $q$  be the total number of horizontal lines. Compute the following moments:

$$\begin{aligned} M_0 &= -\sum_{i=1}^p C_i & N_0 &= -\sum_{j=1}^q D_j \\ M_1 &= M_0 + 2C_1 & N_1 &= N_0 + 2D_1 \\ &\vdots & &\vdots \\ M_p &= M_{p-1} + 2C_p & N_q &= N_{q-1} + 2D_q \end{aligned}$$

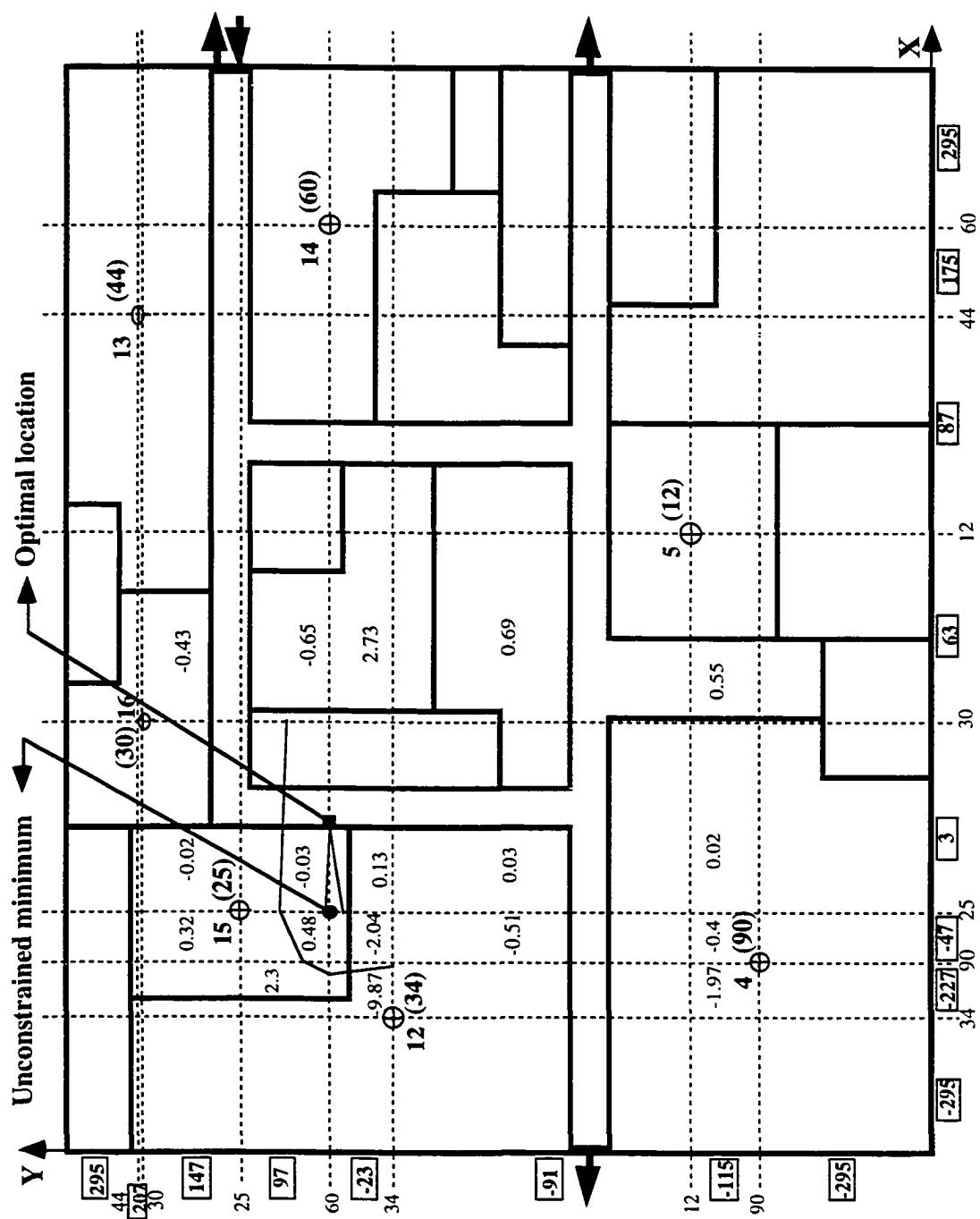


Figure 6.3  
Location of Drinking Fountain using Contour Lines

and place these moments as shown inside the boxes along the X and Y axes in Fig. 6.3, i.e., from left to right for the  $M_j$ s and from bottom to top for the  $N_j$ s. For example,

$$M_0 = -(34 + 90 + 25 + 30 + 12 + 44 + 60) = -295$$

$$M_1 = -295 + 2 \times 34 = -227.$$

4. Let the region  $[i, j]$  be the region delimited by the vertical lines  $i$  and  $i+1$  and the horizontal lines  $j$  and  $j+1$ . The slope of any contour line passing through the region  $[i, j]$  is equal to

$$S_{ij} = -\frac{M_i}{N_j}$$

Slopes in the regions around the optimum,  $X^*_{30} = 62$  and  $Y^*_{30} = 155.73$ , are as shown in Fig. 6.3.

5. Choose a starting point and use the slope in each region to draw the contour lines. It is a trial and error procedure to find the contour that is closest to the optimum and intersects the feasible region, in this example, the aisle-boundaries of departments 12 and 15. By drawing a few contour lines (see Fig. 6.3), one can see that the point  $X_f = 84$ ,  $Y_f = 155.73$  is closest to the optimum without being inside any department.

Table 6.6 shows the value of  $Z$  for selected points: one is the unconstrained minimum, the two following points are acceptable (but not optimal) locations of the fountain along the closest aisle, the fourth point is the solution with the smallest value of  $Z = 38,145.13$ .

Point	X-coordinate	Y-coordinate	Z value
1	62	155.73	38,079.13
2	84	140	38,506.92
3	84	150.5	38,265.42
4	84	155.73	<b>38,145.13</b>

Table 6.6  
Values of  $Z$

Due to the difference in the assumed dimensions of the fountain, the solution found by PTFLCA ( $X_{30} = 84.05$ ,  $Y_{30} = 155.73$ ) is slightly different from the one found using a

traditional method ( $X_{30} = 84$ ,  $Y_{30} = 155.73$ ). Note that the traditional method assumed the fountain to be dimensionless which means that a fountain with large dimensions might not fit around its solution point.

#### **6.2.4 Summary**

The above example showed that PTFLCA can be used to solve MINISUM location problems with 2-dimensional constraints on the acceptable locations of the new facility. PTFLCA's solution was validated by comparing it to that of traditional problem solving methods.

This example illustrates only one class of problems that PTFLCA can solve, where constraints are limited to non-overlap constraints and the dimensions of the new facility are negligible. PTFLCA's main application is layout problems where constraints can be distance, zoning, or orientation constraints, and where the new facility and existing facilities need not be dimensionless.

### **6.3 Evaluating Alternative Material Delivery Schedules**

The following example illustrates the use of MoveSchedule in evaluating alternative material delivery schedules of a pipelaying activity. The alternatives are evaluated based on the value of VFL for each schedule-layout combination.

The example is taken from a case study by Wilkie and Tommelein (92) about the renovation of the University of Michigan Stadium. The project included lowering the football playing field to improve visibility from the first-row spectator seats and replacing the Astroturf artificial carpet by natural grass (Prescription Athletic Turf or PAT). This involved:

- removing the carpet and pad
- removing the asphalt underneath of the carpet and pad
- removing the 2-foot crown of the field to provide a level playing field.
- excavating an additional 5 feet below the level field and providing frost protection
- removing a 36 inch storm sewer that ran across the field

- installing a 24 inch Reinforced Concrete Elliptical Pipe (RCEP) to create a new drainage system
- backfilling the trenches dug for the RCEP and leveling the field
- installing the PAT system.

This example focuses on the installation of the RCEP around the perimeter of the field. The field dimensions are 384 ft long and 208 ft wide (see Fig. 6.4). The only access to the site is through the football player access tunnel located at the center of the north side of the stadium (represented by the thick double arrows in Fig. 6.4). A pipelayer is used for installing the RCEP in the trenches and as a material handling means for moving the pipes from one staging area to another on site. The total number of RCEP segments to be installed by the pipelayer is 144 segments.

The problem to be solved is to find the most cost- and time-effective scenario for delivering the RCEP to site and installing it around the perimeter of the field.

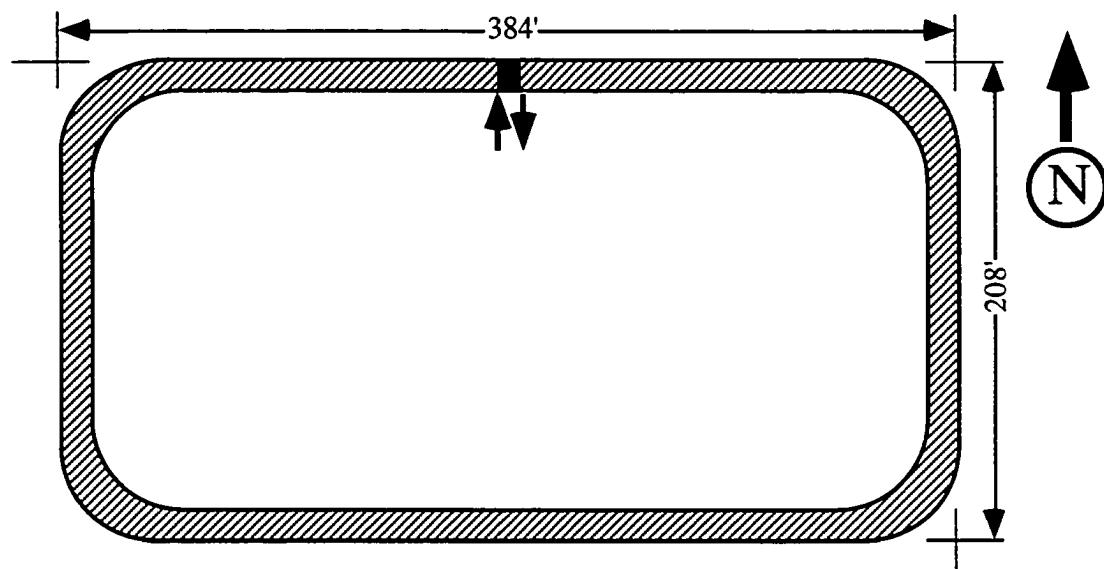


Figure 6.4  
The Excavated Michigan Stadium

### 6.3.1 Modeling Assumptions

The cross-hatched areas in Fig. 6.4 represent the trenches excavated around the field for placement of the RCEP. The trenches are modeled 6 ft wide which is larger than the actual

width of excavation to account for clearance needed to keep equipment farther from the actual limits of the trenches.

The trenches are divided into 6 sections of approximately equal length (shown with different cross-hatching in Fig. 6.5), each requiring 24.8 ft-long pipe segments. (Note that this figure and all following ones showing the site layout for different time frames are screen dumps from MoveSchedule's graphical layout display program. Arrows, labels, and the X- and Y-axis have been added manually.) The trenches are divided into equal-length sections so that equal work is needed to install the pipes in each section. We limited the number of equal-length sections to 6 mainly to limit computations and keep the example simple but to the point. Accordingly, the work is divided into 6 activities, each activity involves laying the pipes of one section. A single pipelayer is used for installing the RCEP, forcing the installation to be sequential. It is assumed that the pipelaying job starts from the access point and proceeds counter-clockwise as shown in Fig. 6.5 and by the activity network in Fig. 6.6. This sequencing ignores the actual slope of the pipes in the trenches, which is acceptable in this example because the solution and materials handling costs are not affected by the order in which the sections are executed.

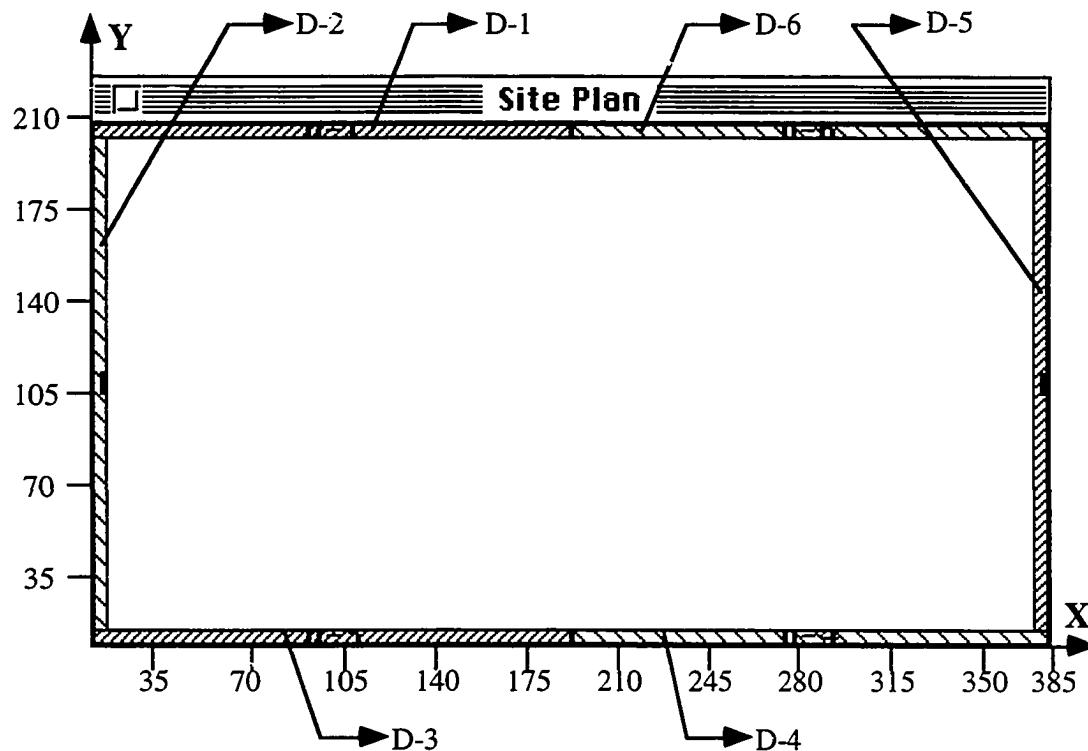


Figure 6.5  
Stadium and Trenches as Modeled in MoveSchedule

The pipelayer is used for installing the RCEP segments in the trenches. A crew of 1 pipelayer, 2 skilled labor, and 2 common labor is chosen to install RCEP segments at the rate of 100 linear ft/day approximately.

The costs modeled in this example are installation and relocation costs of the pipes using the pipelayer. The cost of installing 24 RCEP segments per unit distance per unit time is arbitrarily set to 100. It is assumed that the relocation cost of the 24 RCEP segments is smaller than the installation cost because of additional skilled labor needed for installing the pipes. Accordingly, the cost of relocating 24 RCEP segments is set to 50/unit distance, where 50 is arbitrary. It is important to point out that the findings in this example are not dependent on the actual values of relocation and installation costs, but depends on the relative values of unit installation and relocation costs, i.e., the unit installation cost is higher than the unit relocation cost for this example.

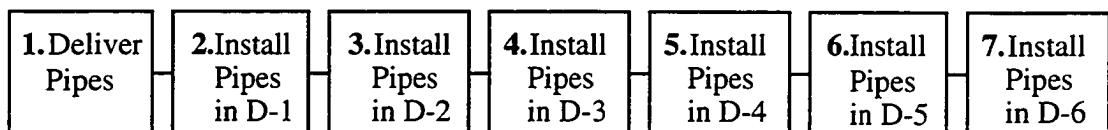


Figure 6.6  
Activity Network for the Pipelaying Job

The 8 ft-long RCEP segments are delivered to site by truck loads of 12 segments. Hence, each trench section requires 2 full truck loads and a total of 12 truck loads are delivered to site. A truck load of pipes can be laid out on site in different possible arrangements, for example, 12 by 16, 24 by 8, or 32 by 6. Because there is ample site space available and limited space is not a concern in this example, we will therefore arbitrarily assume that each truck load of pipes requires a laydown area of 12 by 16. The pipes needed for one section requires twice that area. Here too, the two 12 by 16 areas can be laid out in different arrangements (e.g., 12 by 32 or 24 x 16) and we therefore arbitrarily choose an area of 24 by 16.

Two out of several possible delivery scenarios are considered by the field engineer and describe alternative scenarios for the pipelaying work to be done on site. They are to be evaluated using VFL as a measure of materials handling and installation costs.

**Scenario I:** the 12 truck loads are brought to site and are unloaded at a central location adjacent to one another, in the pattern shown in Fig. 6.7.

**Scenario II:** the 12 truck loads are brought to site and are unloaded at distributed locations around the site as specified by the field engineer and shown in Fig. 6.8.

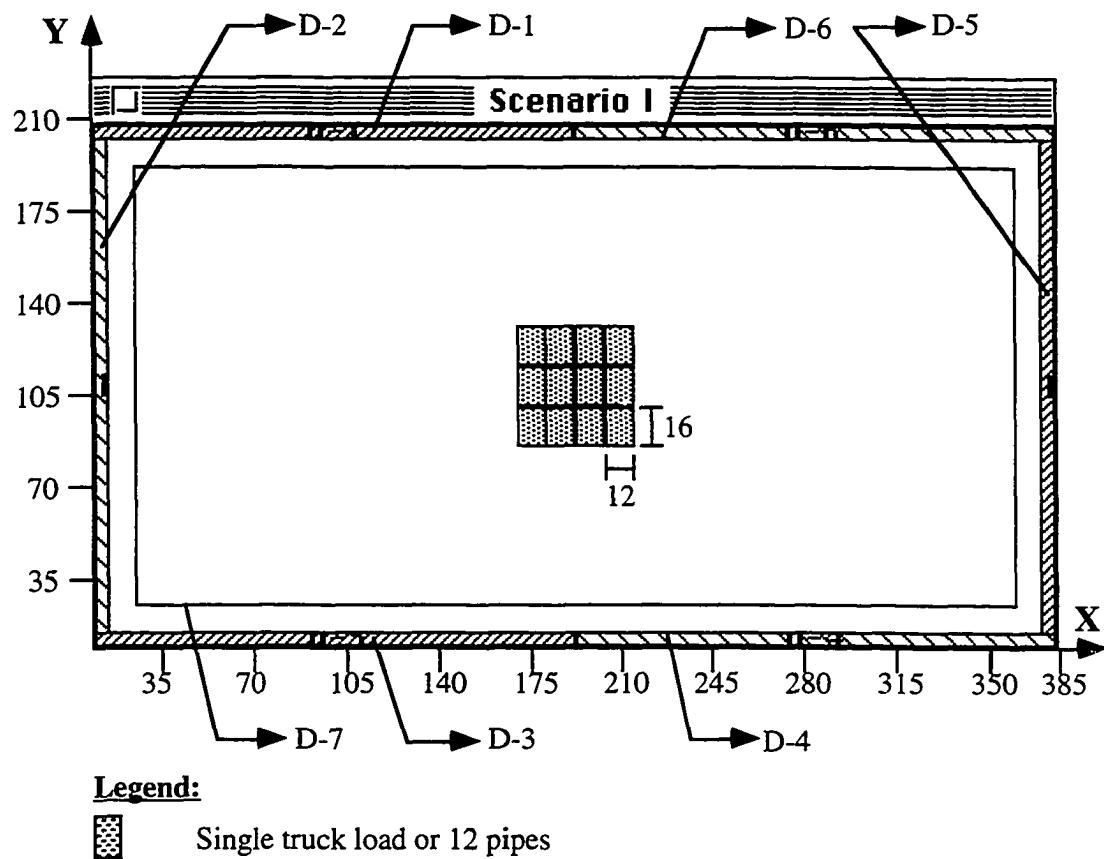


Figure 6.7  
Locations at which Pipes are Unloaded in Scenario I

In addition to installation and materials handling costs which are to be evaluated using VFL, there is the delivery cost of pipes which could be different for the two scenarios as the distance traveled by the trucks to unload the pipes is different. In scenario I, each of the 12 trucks will travel a distance on site of 116 ft approximately, thus a total rectilinear distance of 1,392 ft. In scenario II, the total rectilinear distance traveled is 4 ( $120 + 272 + 280 = 2,688$  ft) ( $> 1,392$ ).

In this example, however, we assume that the trucks will unload at the locations specified for each scenario at no extra cost, i.e., transportation costs associated with distances traveled by the delivery trucks on site are assumed zero.

This assumption is justifiable because the distance traveled by the trucks on site is negligible if compared to the distance traveled from the location of manufacturing to the

stadium site. Furthermore, materials are normally delivered to site by the vendor which means that delivery costs are typically included in the cost of materials.

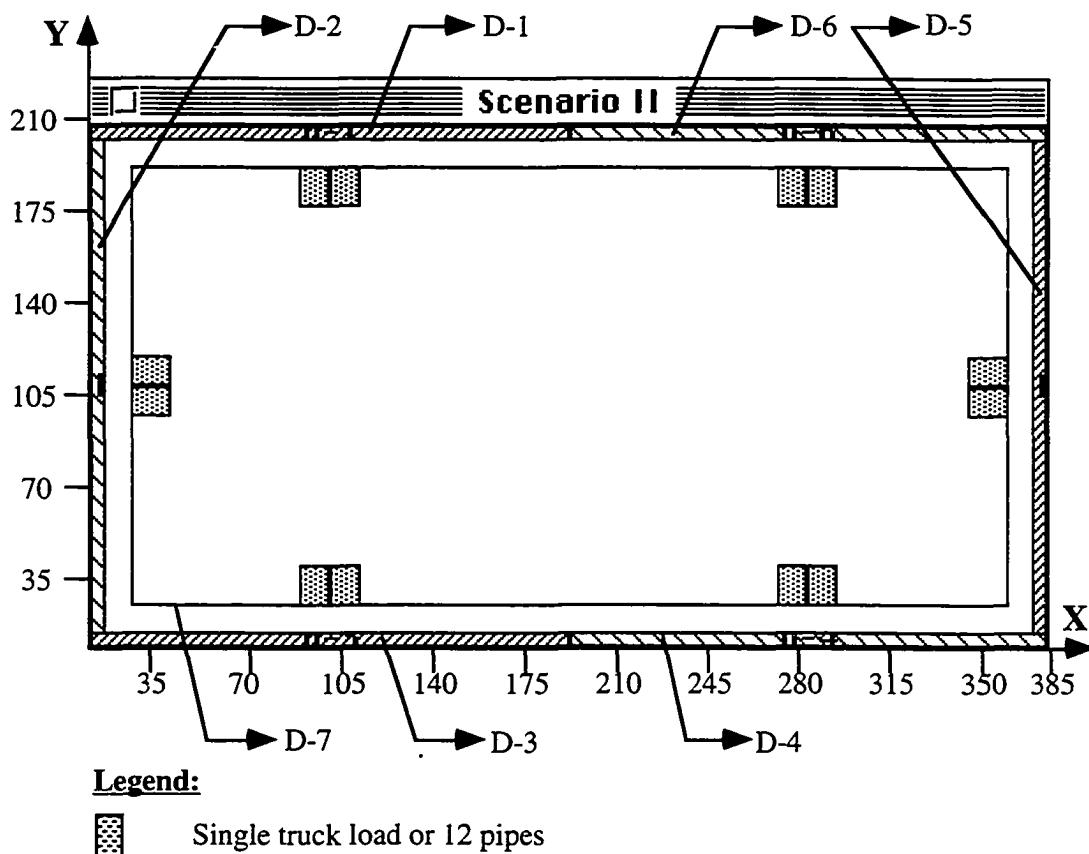


Figure 6.8  
Locations at which Pipes are Unloaded in Scenario II

In some cases, however, a private shipping company could be hired to deliver the materials to site. These companies are in the business of delivering materials to site expediently and fast. If they need to deliver materials at different locations on site, then these locations should be specified before hand and possibly included in the agreement for shipping and handling. In these cases, then, planning laydown areas ahead becomes crucial so that appropriate provisions can be made to ensure and facilitate the delivery to proper locations.

This example is simple and the solution to the problem is expected to be scenario II as the most cost- and time-effective scenario. This example is interesting, though, because it sheds the light on the issues involved in managing materials laydown on construction sites. It also illustrates how the MoveSchedule model accounts for the main variables that

should be considered for economically managing materials on site and which are not considered in other layout or resource scheduling models.

### 6.3.2 Input Data in MoveSchedule

In MoveSchedule, the stadium and the 6 trench sections are modeled as static Profile-D resources. An additional Profile-D resource D-7 is defined to limit the space that can be used as material laydown area to be at least 10 ft away from each trench section. This is needed to ensure that the pipelayer can travel along the trenches unobstructed. Data regarding independent resources as input to MoveSchedule is summarized in Table. 6.7.

<b>Profile-D</b>	<b>Description</b>	<b>L x W</b>	<b>Relocation Weight</b>	<b>Fixed Position [X, Y, Orientation]</b>
D-1	North-west trench	192 x 6	Stationary	[96, 205, 0]
D-2	West trench	208 x 6	Stationary	[3, 104, 90]
D-3	South-west trench	192 x 6	Stationary	[96, 3, 0]
D-4	South-east trench	192 x 6	Stationary	[288, 3, 0]
D-5	East trench	208 x 6	Stationary	[381, 104, 90]
D-6	North-east trench	192 x 6	Stationary	[288, 205, 0]
D-7	Laydown area	352 x 176	Stationary	[192, 104, 0]

Table 6.7  
Input of Independent Resources

The pipes are modeled using Profile-C because they occupy space on site from the time they are unloaded (i.e., during the "deliver pipes" activity) to the time they are installed in the trench (i.e., during the corresponding pipelaying activity). The pipelayer is also modeled using Profile-C because it is shared by all activities.

Each of the two scenarios is represented in a resource level for each activity. The minimum level models the activities' space requirements and duration with scenario I and the normal level that with scenario II. Tables 6.7 and 6.8 summarize MoveSchedule's input data corresponding to the 6 pipelaying activities and dependent resources respectively.

Activity Number	Description	Duration	Resource Level		Area @ Start
			Level	[Type, Area]	
2	Install pipes in D-1	2	Normal	[C-1, 384] [C-20]	465
		4	Minimum		
3	Install pipes in D-2	2	Normal	[C-2, 384] [C-20]	465
		4	Minimum		
4	Install pipes in D-3	2	Normal	[C-3, 384] [C-20]	465
		4	Minimum		
5	Install pipes in D-4	2	Normal	[C-4, 384] [C-20]	465
		4	Minimum		
6	Install pipes in D-5	2	Normal	[C-5, 384] [C-20]	465
		4	Minimum		
7	Install pipes in D-6	2	Normal	[C-6, 384] [C-20]	465
		4	Minimum		

Table 6.8  
Activity Resource Levels

Note that the same resources, namely 24 pipe segments and the pipelayer, are used for performing each pipelaying activity at the minimum level or normal level. The 24 pipe segments require an area equal to  $24 \times 16 = 384$ , the pipelayer requires an area of  $9 \times 9 = 81$ , thus bringing the total area requirement of each pipelaying activity to  $384 + 81 = 465$ .

<b>Profile</b>	<b>Description</b>	<b>L x W</b>	<b>Relocation Weight</b>	<b>Fixed Position</b>
C-1	24 RCEP segments	24 x 16	50	—
C-2	24 RCEP segments	24 x 16	50	—
C-3	24 RCEP segments	24 x 16	50	—
C-4	24 RCEP segments	24 x 16	50	—
C-5	24 RCEP segments	24 x 16	50	—
C-6	24 RCEP segments	24 x 16	50	—
C-20	Pipelayer	9 x 9	0	—

**Table 6.9  
Dependent Resources Data**

The durations of the activities differ for each scenario as they are determined based on the productivity rate of the pipelayer and the length of the trench section to be worked on. At the normal level, the pipes are unloaded next to the trenches where they will be installed. Hence, the duration of a pipelaying activity covering trench D-1, D-3, D-4, or D-6 is equal to 192 ft / 100 linear ft/day which is about 2 days. The duration of an activity covering trench D-2 or D-5 is equal to 208 ft / 100 linear ft/day which is also about 2 days. At the minimum level, the pipes are unloaded at the center of the stadium, and are moved from this central location to some location close to the trenches before installation. It is estimated that the rate at which the pipelayer can install pipes will drop by half because of increased material handling time. Hence, the duration of an activity at the minimum level is longer and is equal to 192 ft / 50 linear ft/day which is about 4 days or 208 ft / 50 linear ft/day which is also about 4 days. Consequently, the overall duration of the pipelaying job with scenario I is expected to be longer than that with scenario II.

Proximity weights between the pipes and the corresponding trenches model the installation costs of pipes. Relocation weights of pipes model their relocation costs. Hard constraints are used to confine the possible positions of the pipes to be within the laydown area D-7, and to limit the positions of the pipelayer to be along and adjacent to the trenches during installation time.

### 6.3.3 Schedule and Layout Costs with Scenario I

Under scenario I, each pipelaying activity takes 4 days (see minimum level in Table 6.8). The schedule and corresponding PTFs are as shown in Fig. 6.9.

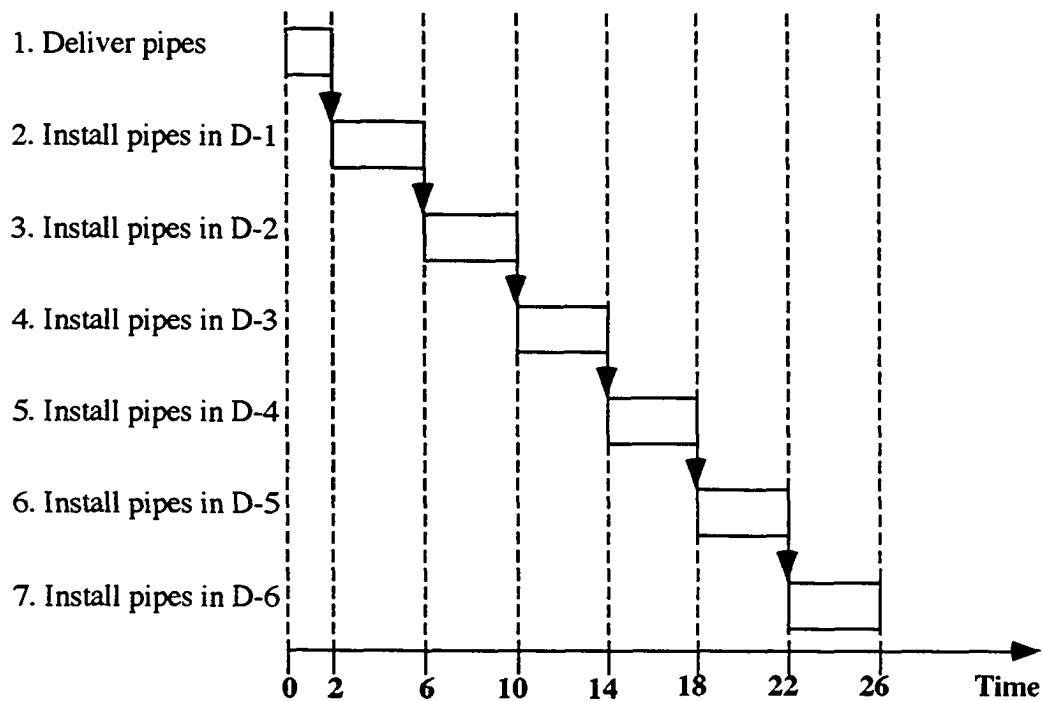


Figure 6.9  
Schedule and PTFs in Scenario I

In PTF-0-2, activity 1 involves delivering the pipes needed by the 6 pipelaying activities to site and unloading them at the positions shown in Fig. 6.10 and summarized in Table 6.10. This layout is normally input to MoveSchedule.

MoveSchedule constructs the layouts of the remaining PTFs to determine the installation and relocation costs of the pipes.

The layout of PTF-2-6 is constructed first. In this time frame, the 24 pipe segments C-1 will be installed in trench D-1. The resources that need space in this PTF are the active resources C-1 and C-20 and the idle resources C-2, C-3, C-4, C-5, and C-6. Note that none of the aforementioned resources is stationary (i.e., PTFLCA can relocate them if it is cost-effective to do so) and PTFLCA will determine their positions in PTF-2-6 so that installation and relocation costs are minimized.

The hard constraints in PTF-2-6 (Table 6.11) describe adjacency requirements between the pipelayer and the trench. The proximity weight in PTF-2-6 (Table 6.11) represent the installation cost of the pipes in the trench using the pipelayer which is equal to 100 (see previous section).

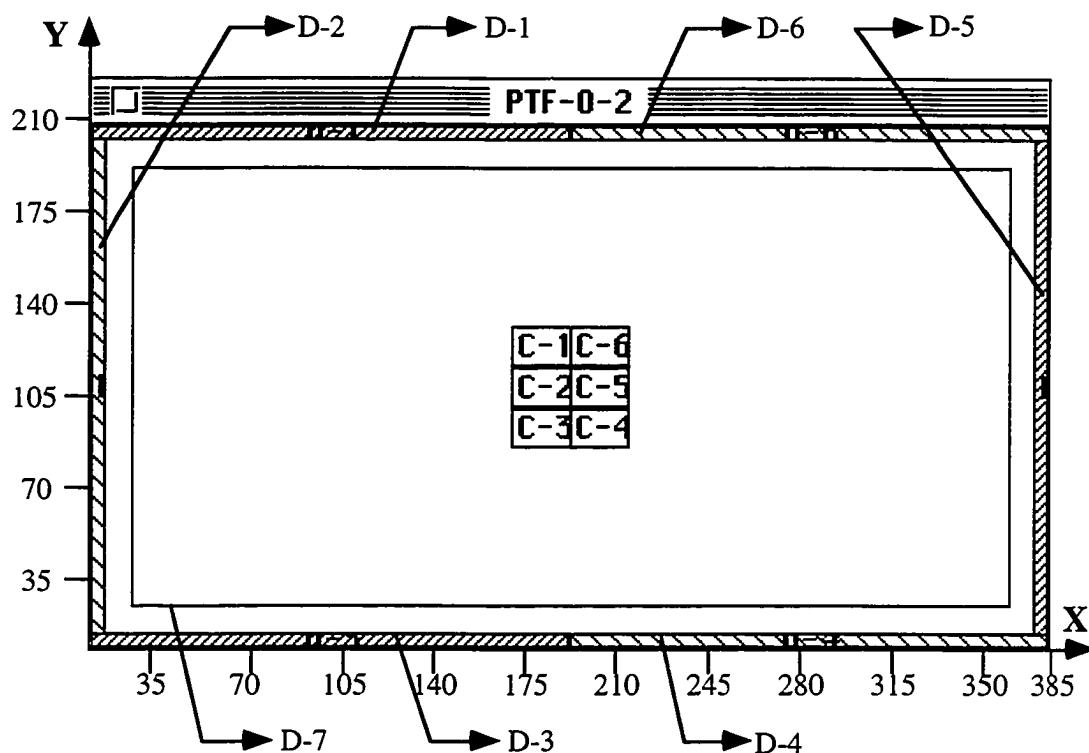


Figure 6.10  
Layout of PTF-0-2 With Scenario I

PTF	Profile-C	Position
PTF-0-2	C-1	[180, 120, 0]
	C-2	[180, 104, 0]
	C-3	[180, 88, 0]
	C-4	[204, 88, 0]
	C-5	[204, 104, 0]
	C-6	[204, 120, 0]

Table 6.10  
Positions of Resources in PTF-0-2 as Input to MoveSchedule for Scenario I

Type	Resource 1	Resource 2	Constraint Type	Value	Weight
Hard	C-20	D-1	max Dx	0	
	C-20	D-1	max Dy	0	
	C-1	D-7	In-zone	—	
Proximity	C-1	D-1			100

Table 6.11  
Proximity Weights and Location Constraints for PTF-2-6

PTFLCA first positions all static resources (D-1 to D-7) at their known positions. Then it positions all previously-positioned stationary resources, but none exist in this PTF.

Next it positions all relocatable resources. Among the relocatables, only C-1 has a proximity weight with another resource D-1 and it is therefore selected first. PTFLCA initializes SPP<sub>1</sub> so that C-1 is positioned within the stadium site and the laydown area D-7. The resulting SPP<sub>1</sub> is shown in Fig. 6.11.

PTFLCA then determines C-1's position that minimizes installation and relocation costs of C-1:

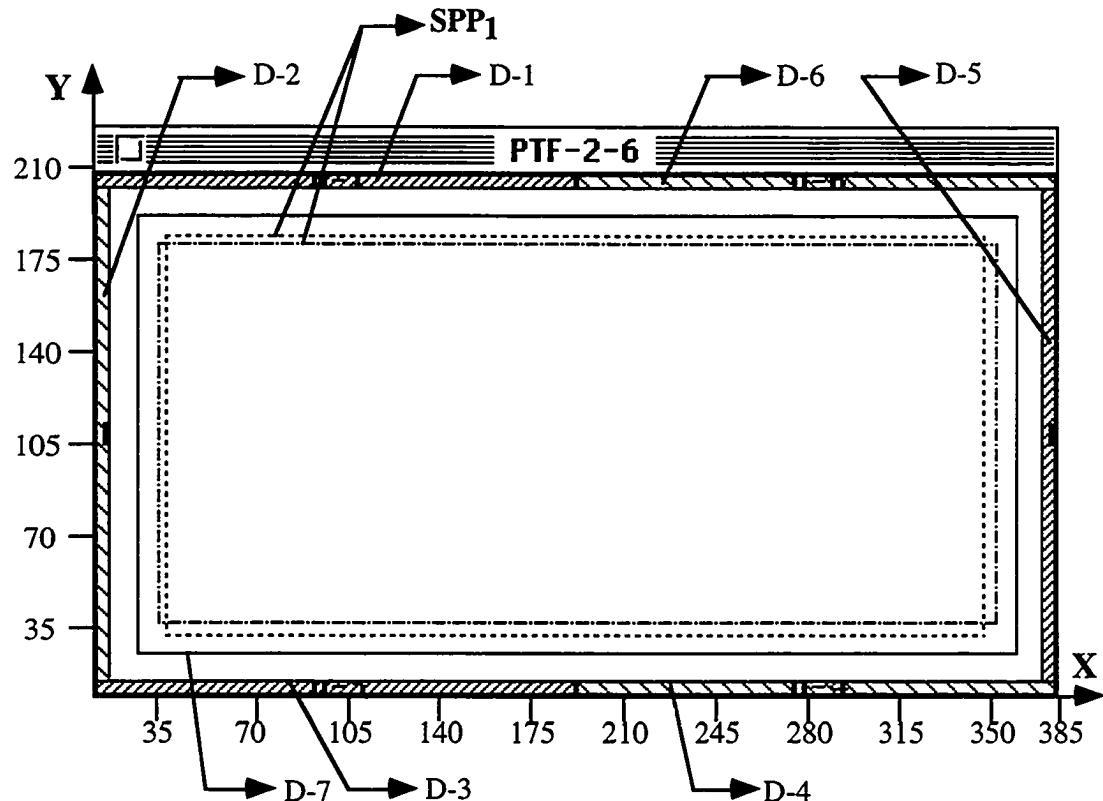
$$\begin{aligned}
 \min \Delta VFL &= 50 &&; \text{relocation weight of C-1 (Table 6.9)} \\
 &\quad (|X_1 - 180| + |Y_1 - 120|) &&; \text{distance from previous position in PTF-0-2} \\
 &\quad + 4 \{ &&; \text{duration of PTF-2-6} \\
 &\quad 100 &&; \text{proximity weight (Table 6.11)} \\
 &\quad (|X_1 - 96| + |Y_1 - 205|) &&; \text{distance between C-1 and D-1}
 \end{aligned}$$

subject to  $\{X_1, Y_1\} \in \text{SPP}_1$ .

It finds  $X_1 = 96$  and  $Y_1 = 184$  @  $0^\circ$  (see Fig 6.11) with  $\Delta VFL = 15,800$ .

None of the resources that remain to be positioned have a proximity weight with those positioned. PTFLCA breaks the tie according to the first tie-breaking rule, i.e., by selecting the first resource in the queue which happens to be C-2. PTFLCA prunes SPP<sub>2</sub> to satisfy the in-zone constraint with D-7, and the out-zone constraints with C-1 and the static resources. The resulting SPP<sub>2</sub> is shown in Fig. 6.12. (Note that the spacing

between the rectangles describing SPP1 in Fig. 6.12 exists because MoveSchedule's layout display program rounds coordinates to the nearest integer.)



Legend:

$$SPP_1 = \{ \{0 \{([28\ 356]\ [24\ 184])\} \} \{90 \{([24\ 360]\ [28\ 180])\} \} \}$$

Figure 6.11  
Partial Layout of PTF-2-6

PTFLCA then determines the position that minimizes relocation costs of C-2:

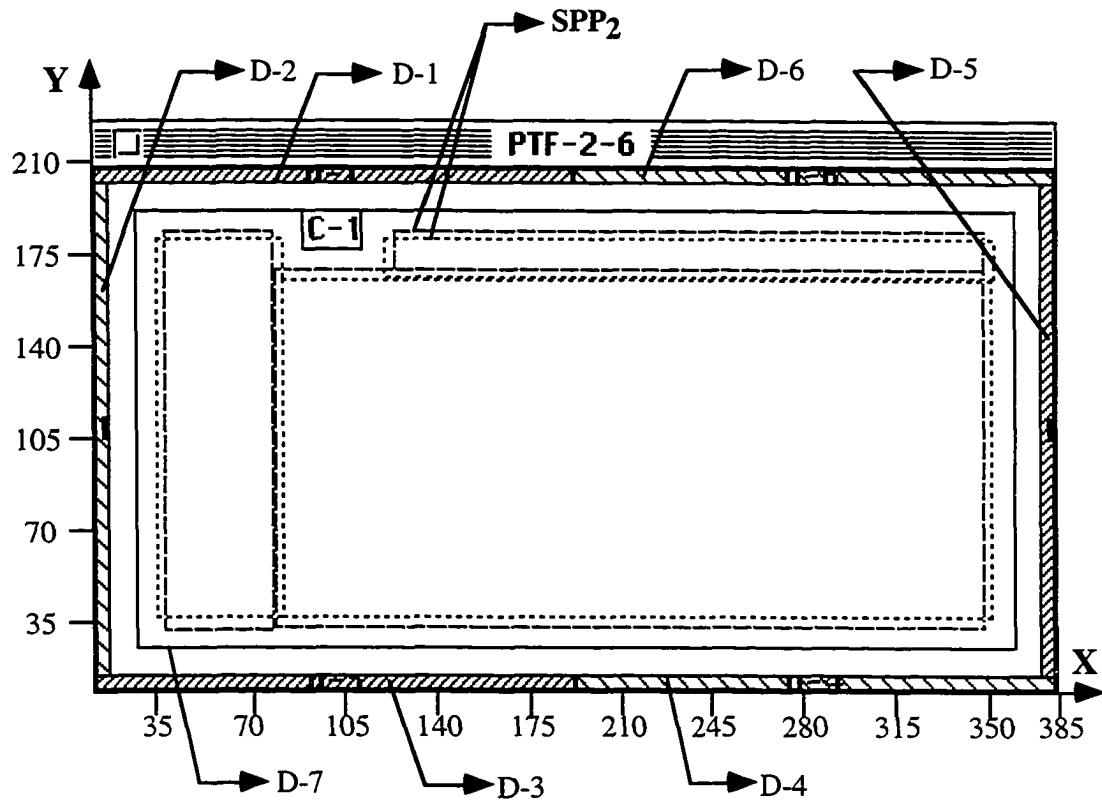
$$\min \Delta VFL = 50 \quad ; \text{relocation weight of C-2 (Table 6.9)}$$

$$(|180 - X_2| + |104 - Y_2|) \quad ; \text{distance from previous position}$$

in PTF-0-2 (Table 6.10)

subject to  $\{X_2, Y_2\} \in SPP_2$ .

PTFLCA keeps C-2 at its previous position  $X_2 = 180$  and  $Y_2 = 104 @ 0^\circ$  (Fig. 6.12) to minimize  $\Delta VFL$ . This is expected because C-2's previous position does not overlap with the new position of C-1. Similarly, PTFLCA keeps C-3, C-4, C-5, and C-6 at their previous positions (see Fig. 6.13). Leaving them at their previous positions brings no increase in the value of  $\Delta VFL$ .



Legend:

$SPP_2 = \{ \{0 \{([28 72] [24 184]) ([72 356] [24 168]) ([120 356] [168 184])\} \}$   
 $\{90 \{([24 76] [28 180]) ([76 360] [28 164]) ([116 360] [164 180])\} \} \}$

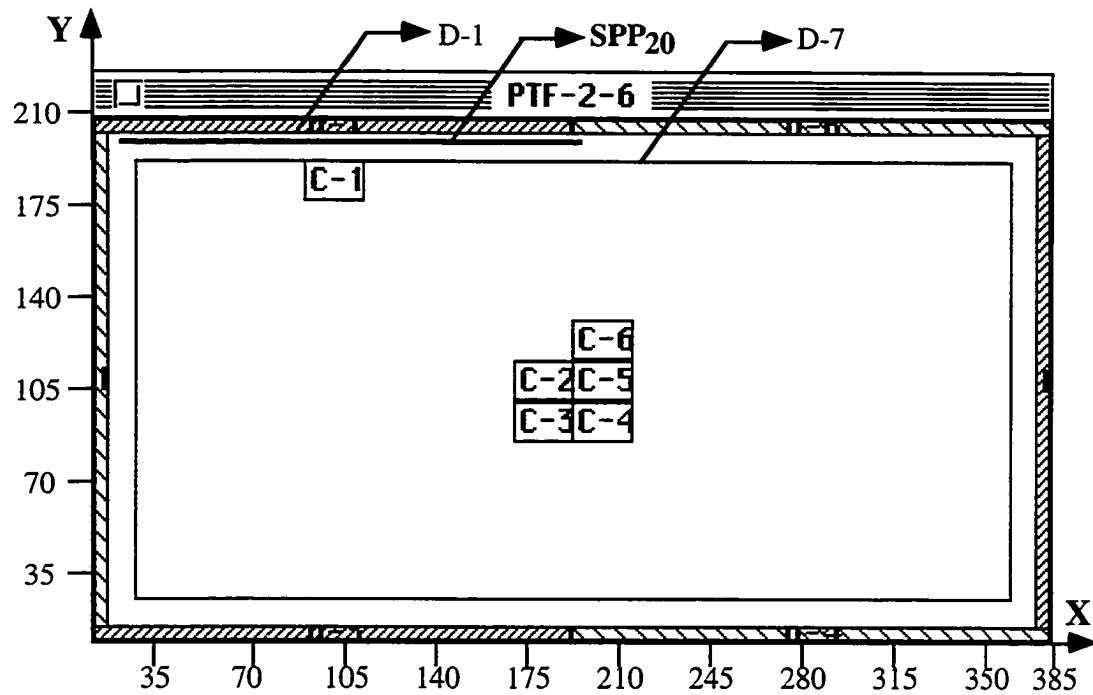
Figure 6.12  
Partial Layout of PTF-2-6 with C-1 Positioned

C-20 is selected last. PTFLCA prunes SPP<sub>20</sub> to satisfy the out-zone constraints with the static (D-1 to D-7) and positioned resources (C-1 to C-6). It then further prunes SPP<sub>20</sub> to satisfy the maximum distance constraints in the X- and Y-direction with D-1. The resulting SPP<sub>20</sub> is shown in Fig. 6.13.

C-20 has no proximity weights with other positioned resources and has a zero relocation weight, hence, PTFLCA randomly samples a single position from SPP<sub>20</sub>. It finds  $X_{20} = 50.25$  and  $Y_{20} = 197.5 @ 0^\circ$ .

The final layout for PTF-2-6 is shown in Fig. 6. 14 with a cumulative  $\Delta VFL$  value of 15,800.

The layout for the remaining PTFs are similarly constructed. Table 6.12 summarizes the hard constraints and proximity weights considered each time.



Legend:

SPP<sub>20</sub> = { {0 {[10.5 196.5] [197.5 197.5]}} } {90 {[10.5 196.5] [197.5 197.5]}} }

Figure 6.13: Partial Layout of PTF-2-6 with C-1 to C-6 Positioned

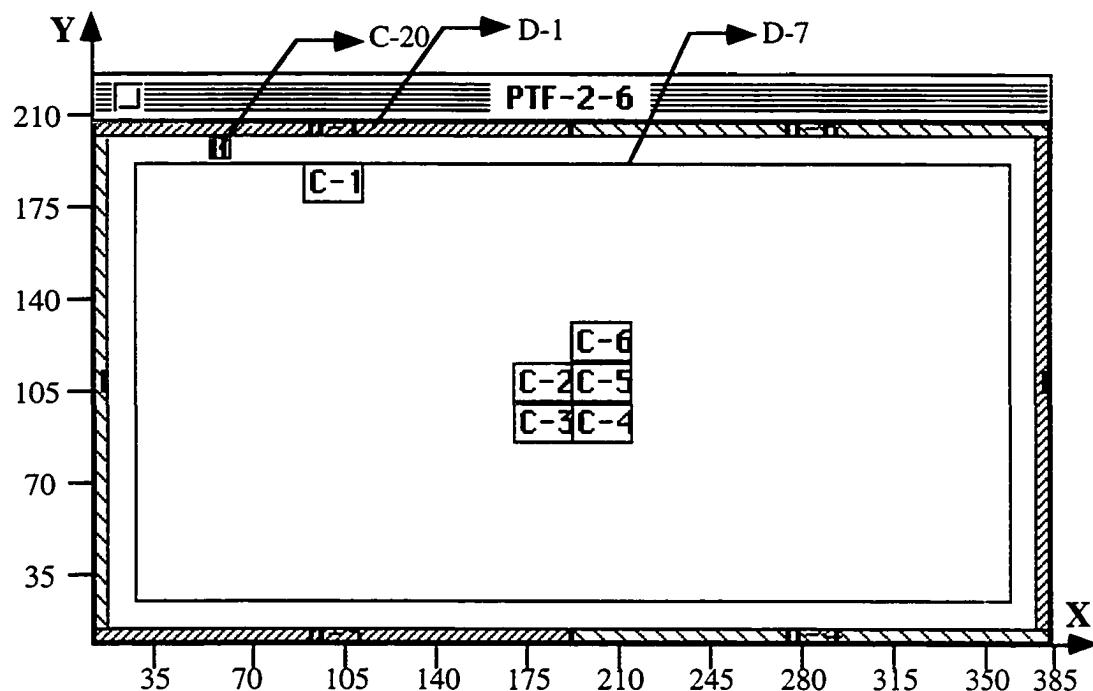


Figure 6.14  
Final Layout of PTF-2-6

<b>PTF</b>	<b>Type</b>	<b>Res-1</b>	<b>Res-2</b>	<b>Constraint Type</b>	<b>Value</b>	<b>Weight</b>
PTF-6-10	Hard	C-20	D-2	max Dx	0	
		C-20	D-2	max Dy	0	
		C-2	D-7	In-zone	—	
	Proximity	C-2	D-2			100
PTF-10-14	Hard	C-20	D-3	max Dx	0	
		C-20	D-3	max Dy	0	
		C-3	D-7	In-zone	—	
	Proximity	C-3	D-3			100
PTF-14-18	Hard	C-20	D-4	max Dx	0	
		C-20	D-4	max Dy	0	
		C-4	D-7	In-zone	—	
	Proximity	C-4	D-4			100
PTF-18-22	Hard	C-20	D-5	max Dx	0	
		C-20	D-5	max Dy	0	
		C-5	D-7	In-zone	—	
	Proximity	C-5	D-5			100
PTF-22-26	Hard	C-20	D-6	max Dx	0	
		C-20	D-6	max Dy	0	
		C-6	D-7	In-zone	—	
	Proximity	C-6	D-6			100

Table 6.12  
Proximity Weights and Location Constraints for all PTFs with Scenario I

Table 6. 12 summarizes the results found by PTFLCA in constructing the layout of the remaining PTFs. It shows, for each PTF (first column) and for each selected resource

(second column), the equation of  $\Delta$ VFL (third column), the single position that minimizes  $\Delta$ VFL (fourth column), and the corresponding  $\Delta$ VFL value (fifth column). When no equation for  $\Delta$ VFL exists, the fourth column shows a position selected at random from the resource's SPP. For each PTF, the resources that remained at their positions relative to the previous layout are not listed. Fig. 6.15 to 6.18 show the final layout of the remaining PTFs for scenario I.

PTF	Res.	$\Delta$ VFL Equation	Position [X, Y, Orientation]	$\Delta$ VFL Value
PTF-0-2			see Table 6.10	0
PTF-2-6	C-1	$50( X_1 - 180  +  Y_1 - 120 ) + 400( X_1 - 96  +  Y_1 - 205 )$	[96, 184, 0]	15,800
	C-20	None	[50.25, 197.5, 0]	0
PTF-6-10	C-2	$50( X_2 - 180  +  Y_2 - 104 ) + 400( X_2 - 3  +  Y_2 - 104 )$	[24, 104, 90]	16,200
	C-20	None	[10.5, 142.5, 90]	0
PTF-10-14	C-3	$50( X_3 - 180  +  Y_3 - 88 ) + 400( X_3 - 96  +  Y_3 - 3 )$	[96, 24, 0]	15,800
	C-20	None	[80.3, 10.5, 90]	0
PTF-14-18	C-4	$50( X_4 - 204  +  Y_4 - 88 ) + 400( X_4 - 288  +  Y_4 - 3 )$	[288, 24, 0]	15,800
	C-20	None	[232.4, 10.5, 90]	0
PTF-18-22	C-5	$50( X_5 - 204  +  Y_5 - 104 ) + 400( X_5 - 381  +  Y_5 - 104 )$	[360, 104, 90]	16,200
	C-20	None	[373.5, 45.4, 0]	0
PTF-22-26	C-6	$50( X_6 - 204  +  Y_6 - 120 ) + 40( X_6 - 288  +  Y_6 - 205 )$	[288, 184, 0]	15,800
	C-20	None	[369.2, 197.5, 0]	0

Table 6.13  
Summary of Resource Positions and VFL Values for PTFs in Scenario I

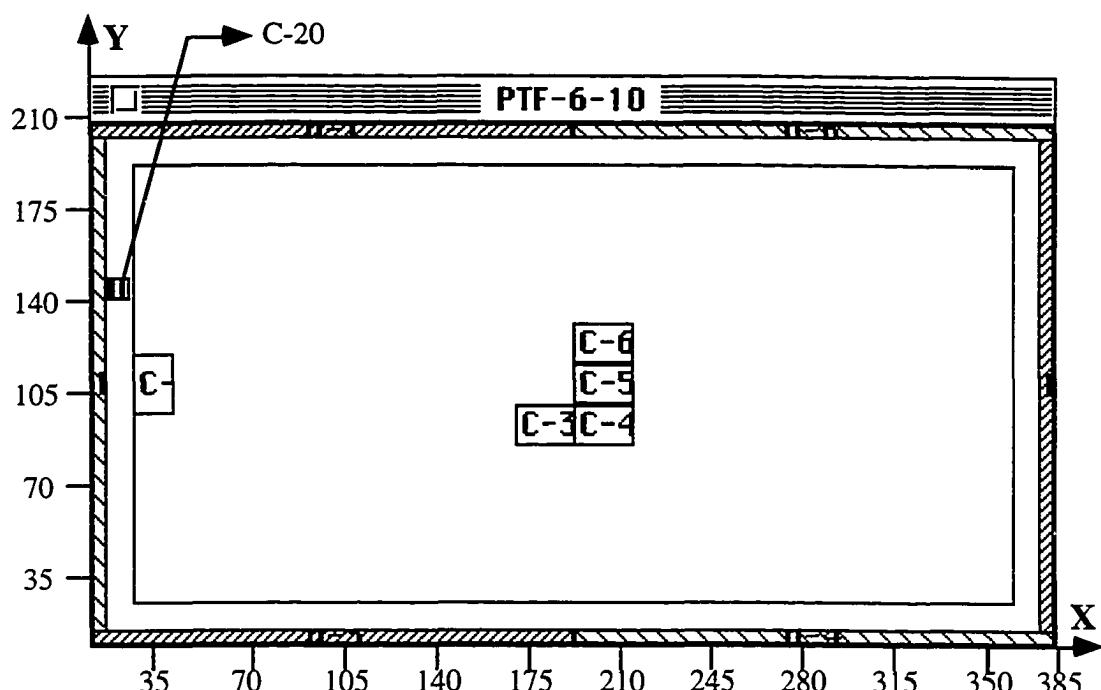


Figure 6.15  
Final Layout of PTF-6-10

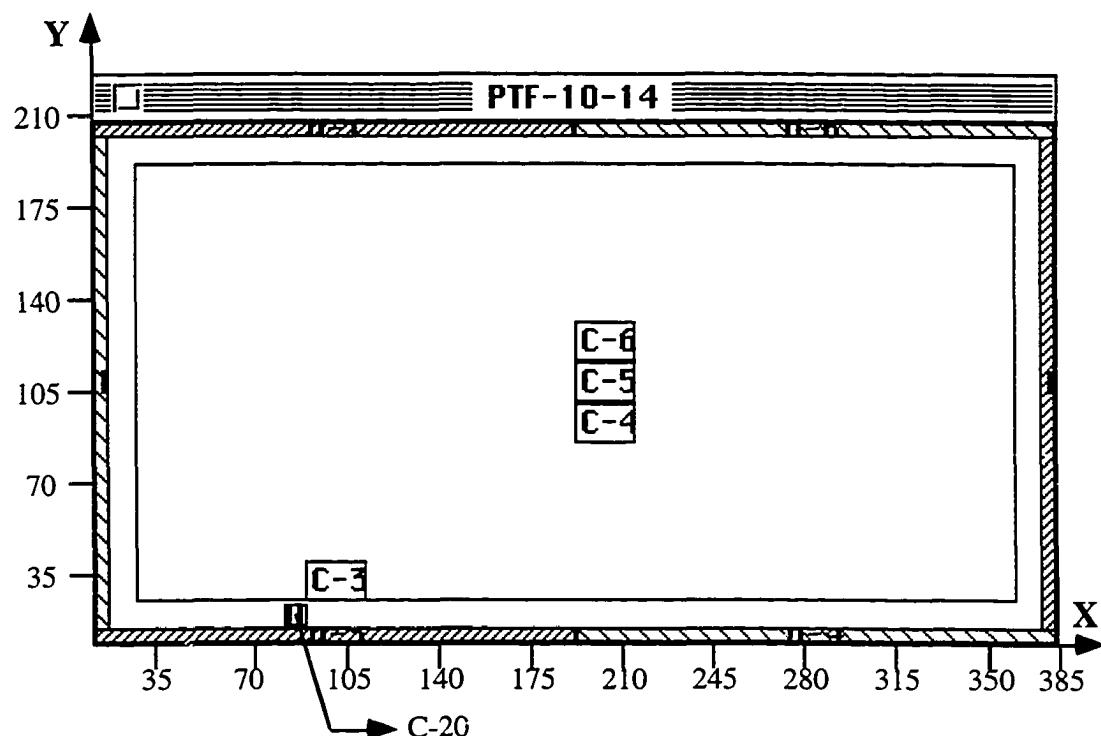


Figure 6.16  
Final Layout of PTF-10-14

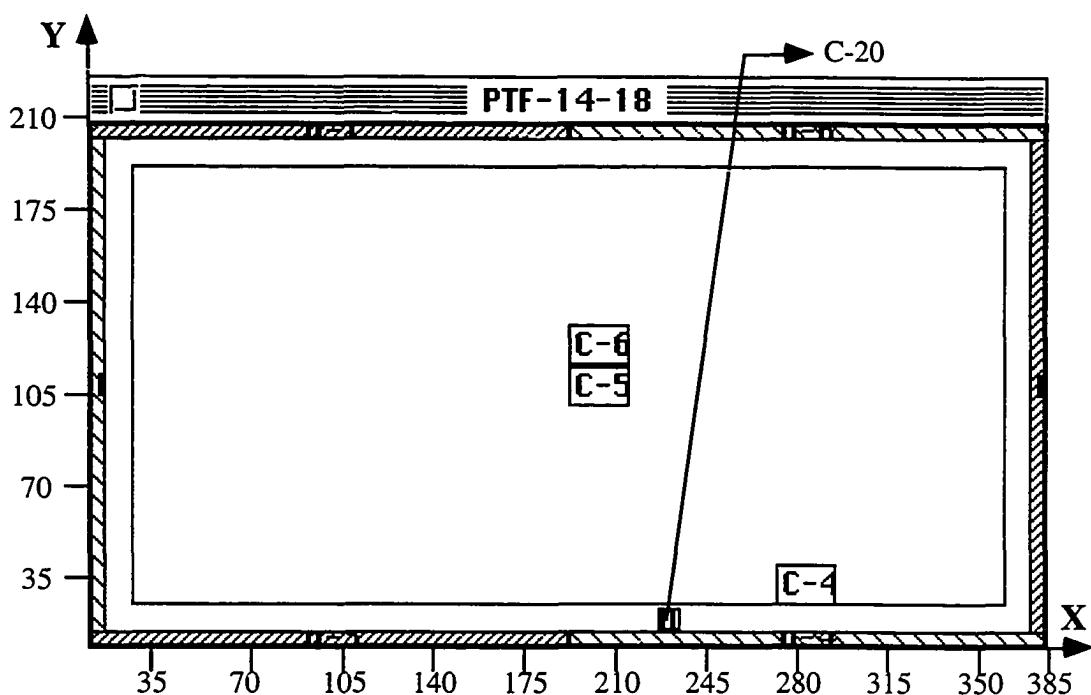


Figure 6.17  
Final Layout of PTF-14-18

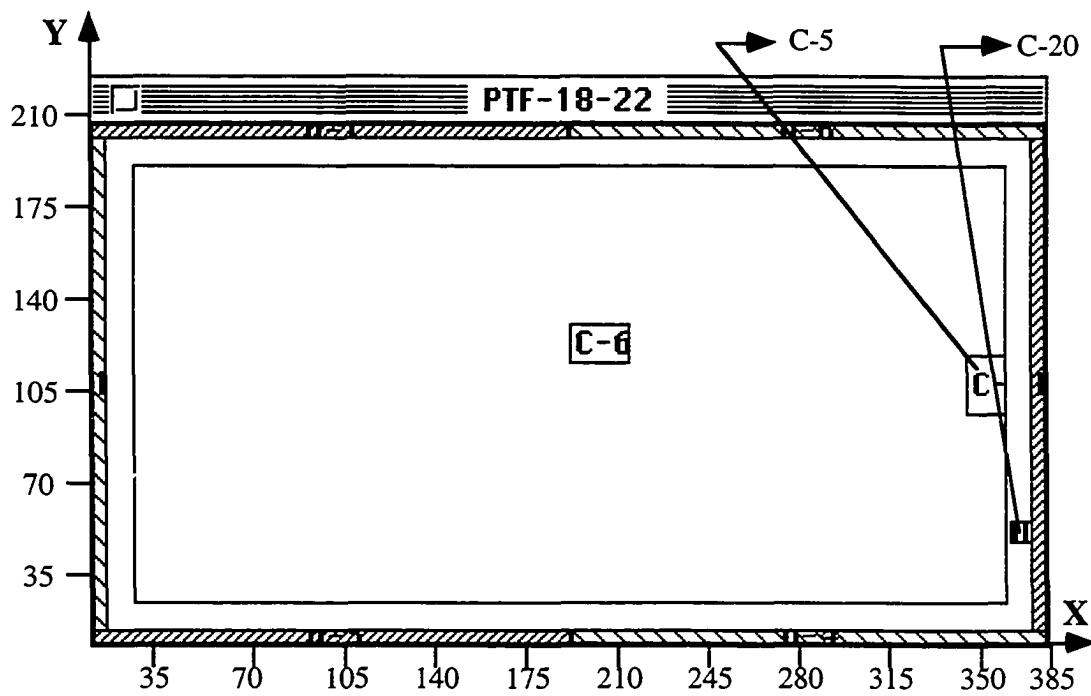


Figure 6.18  
Final Layout of PTF-18-22

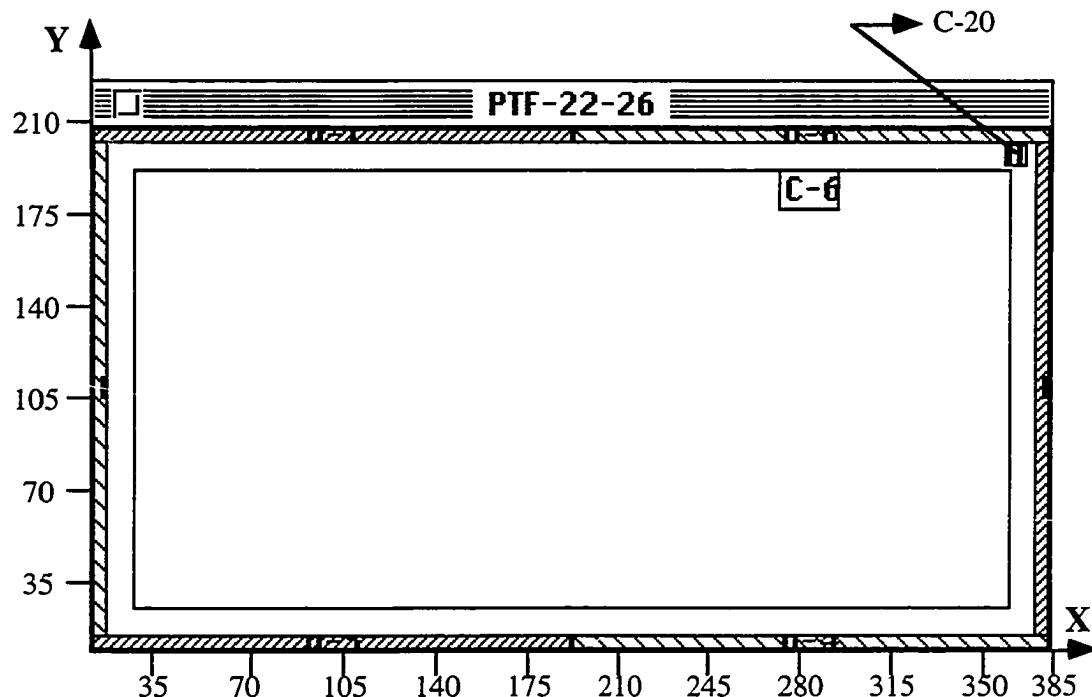


Figure 6.19  
Final Layout of PTF-22-26

The total VFL value is the sum of the  $\Delta$ VFL values in Table 6.13 and is equal to 95,600. It represents the total layout cost, including relocation and installation costs, but excluding delivery costs for scenario I.

#### 6.3.4 Schedule and Layout Costs with Scenario II

Under scenario II, each pipelaying activity takes 2 days (see normal level in Table 6.8). The overall duration of the pipelaying job is 14 days which is shorter than that with scenario I (26 days). The schedule and corresponding PTFs are shown in Fig. 6.20.

In PTF-0-2, activity 1 involves delivering the pipes needed by the 6 pipelaying activities to site and unloading them at the positions shown in Fig. 6.21 and summarized in Table 6.14. This layout is normally input to MoveSchedule.

MoveSchedule then constructs the layouts of the remaining PTFs to determine the installation and relocation costs of the pipes.

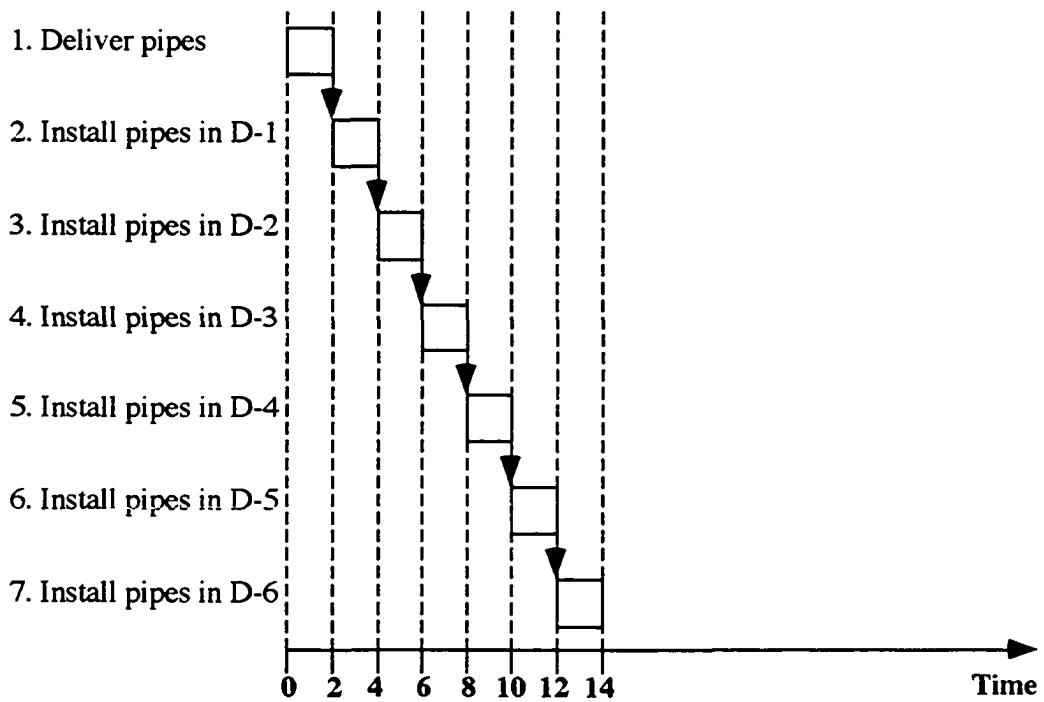


Figure 6.20  
Schedule and PTFs of Scenario II

The layout of PTF-2-4 is constructed first. In this time frame, the 24 pipe segments C-1 will be installed in trench D-1. The resources that need space in this PTF are the active resources C-1 and C-20 and the idle resources C-2, C-3, C-4, C-5, and C-6. Again, none of the aforementioned resources is stationary and PTFLCA may relocate them in PTF-2-4 if it is cost-effective to do so.

The hard constraints and proximity weights in PTF-2-4 are shown in Table 6.15.

PTFLCA first positions all static resources (D-1 to D-7) at their known positions. Then it positions all previously-positioned stationary resources, but none exist in this PTF.

PTFLCA selects C-1 first because it has the highest sum of proximity weights with the static resources. It prunes SPP<sub>1</sub> to satisfy the in-zone constraints with the stadium and the laydown area D-7. The resulting SPP<sub>1</sub> is equal to

$$SPP_1 = \{ \{0 \{([28 356] [24 184])\} \} \{90 \{([24 360] [28 180])\} \} \}.$$

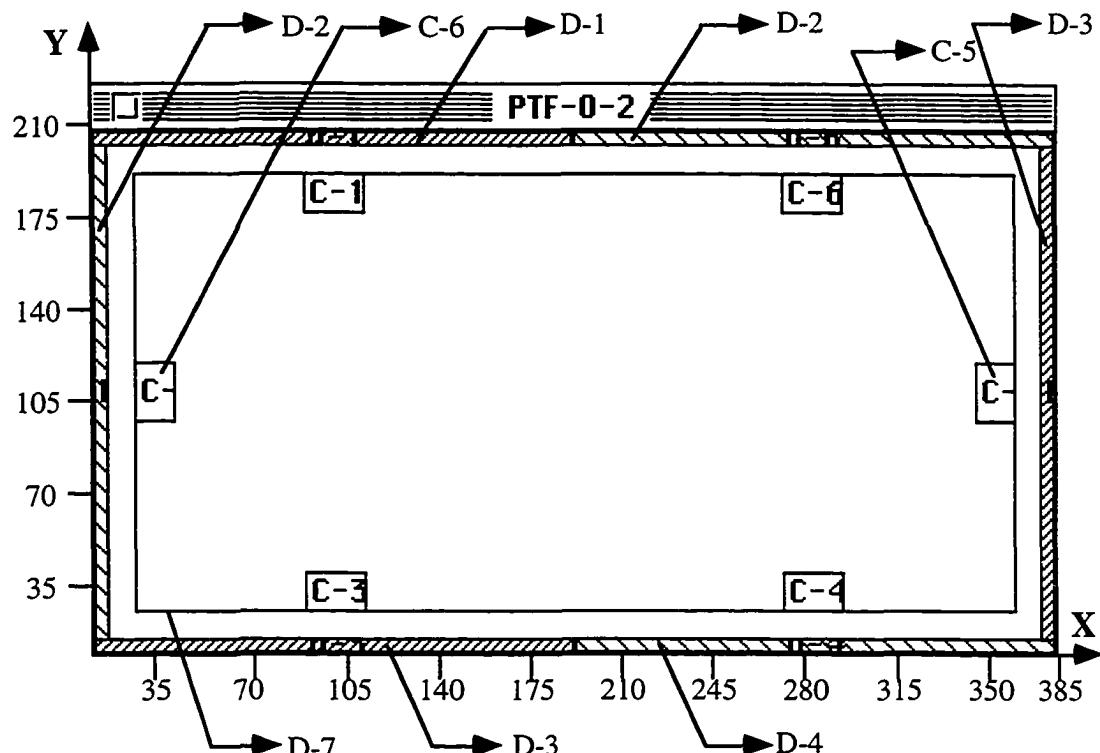


Figure 6.21  
Layout of PTF-0-2 with Scenario II

PTF	Profile-C	Position
PTF-0-2	C-1	[96, 184, 0]
	C-2	[24, 104, 90]
	C-3	[96, 24, 0]
	C-4	[288, 24, 0]
	C-5	[360, 104, 90]
	C-6	[288, 184, 0]

Table 6.14  
Positions of Resources in PTF-0-2 as Input to MoveSchedule for Scenario II

Type	Resource 1	Resource 2	Constraint Type	Value	Weight
Hard	C-20	D-1	max Dx	0	
	C-20	D-1	max Dy	0	
	C-1	D-7	In-zone	—	
Proximity	C-1	D-1			100

Table 6.15  
Proximity Weights and Location Constraints for PTF-2-4

Next, PTFLCA determines C-1's position that

$$\begin{aligned}
 \min \Delta VFL = 50 & ; \text{relocation weight of C-1 (Table 6.9)} \\
 (|X_1 - 96| + |Y_1 - 184|) & ; \text{distance from previous position in PTF-0-2} \\
 & ; (\text{Table 6.14}) \\
 + 2 \{ & ; \text{duration of PTF-2-4} \\
 100 & ; \text{proximity weight (Table 6.15)} \\
 (|X_1 - 96| + |Y_1 - 205|) & ; \text{distance between C-1 and D-1}
 \end{aligned}$$

subject to  $\{X_1, Y_1\} \in SPP_1$ .

It finds  $X_1 = 96$  and  $Y_1 = 184 @ 0^\circ$  with  $\Delta VFL = 4,200$ .

None of the resources that remain to be positioned have a proximity weight with those positioned. PTFLCA breaks the tie by selecting the first resource in the queue which happens to be C-2. PTFLCA prunes SPP<sub>2</sub> to satisfy the in-zone constraint with D-7 and the out-zone constraints with C-1 and the static resources. The resulting SPP<sub>2</sub> is

$$\begin{aligned}
 SPP_2 = \{ & \{0 \{([28 72] [24 184]) ([72 356] [24 168]) ([120 356] [168 184])\}\} \\
 & \{90 \{([24 76] [28 180]) ([76 360] [28 164]) ([116 360] [164 180])\}\}\}.
 \end{aligned}$$

PTFLCA then determines C-2's position that

$$\begin{aligned}
 \min \Delta VFL = 50 & ; \text{relocation weight of C-2} \\
 (|24 - X_2| + |104 - Y_2|) & ; \text{distance from previous position in PTF-0-2} \\
 & ; (\text{Table 6.14})
 \end{aligned}$$

subject to  $\{X_2, Y_2\} \in SPP_2$ .

It keeps C-2 at its previous position in PTF-0-2 to minimize  $\Delta VFL$ , i.e., at  $X_2 = 180$  and  $Y_2 = 104 @ 0^\circ$ . Similarly, it keeps C-3, C-4, C-5, and C-6 at their positions in PTF-0-2. Positioning C-2 to C-6 brings no increase in the value of  $\Delta VFL$  for PTF-2-4.

C-20 is selected last. PTFLCA prunes SPP<sub>20</sub> to satisfy the out-zone constraints with the static (D-1 to D-7) and positioned resources (C-1 to C-6). It then further prunes SPP<sub>20</sub> to satisfy the maximum distance constraints in the X- and Y-direction with D-1. The resulting SPP<sub>20</sub> is

$$\text{SPP}_{20} = \{ \{0 \{([10.5 \ 196.5] [197.5 \ 197.5])\}\} \\ \{90 \{([10.5 \ 196.5] [197.5 \ 197.5])\}\} \}$$

PTFLCA randomly samples a single position from SPP<sub>20</sub> because C-20 has no proximity weight with other positioned resources and has a zero relocation weight. It finds X<sub>20</sub> = 30.3 and Y<sub>20</sub> = 197.5 @ 0°. The final layout for PTF-2-4 is shown in Fig. 6.22 and has a cumulative ΔVFL value of 4,200.

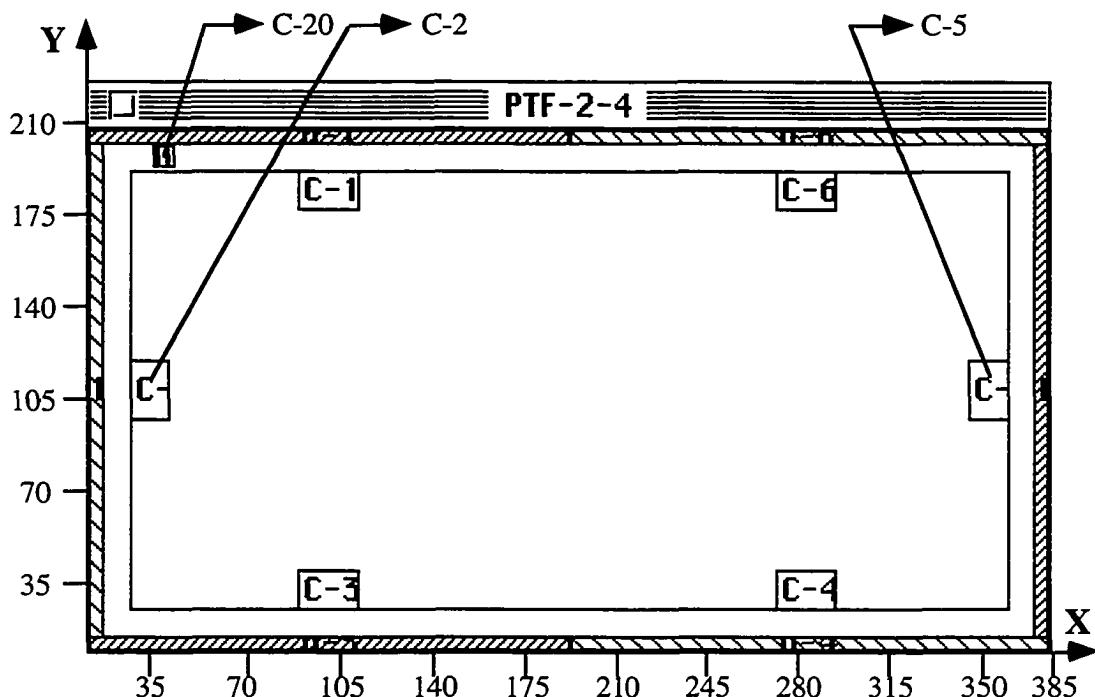


Figure 6.22  
Final Layout of PTF-2-4 With Scenario II

The layouts for the remaining PTFs are similarly constructed. Table 6.16 summarizes the location constraints and proximity weights considered each time.

Table 6.17 summarizes the results found by PTFLCA in constructing the layout of the remaining PTFs. It is similar in structure to Table 6.13. Fig. 6.23 to 6.27 show the final layouts of the remaining PTFs.

<b>PTF</b>	<b>Type</b>	<b>Res-1</b>	<b>Res-2</b>	<b>Constraint Type</b>	<b>Value</b>	<b>Weight</b>
PTF-4-6	Hard	C-20	D-2	max Dx	0	
		C-20	D-2	max Dy	0	
		C-2	D-7	In-zone	—	
	Proximity	C-2	D-2			100
PTF-6-8	Hard	C-20	D-3	max Dx	0	
		C-20	D-3	max Dy	0	
		C-3	D-7	In-zone	—	
	Proximity	C-3	D-3			100
PTF-8-10	Hard	C-20	D-4	max Dx	0	
		C-20	D-4	max Dy	0	
		C-4	D-7	In-zone	—	
	Proximity	C-4	D-4			100
PTF-10-12	Hard	C-20	D-5	max Dx	0	
		C-20	D-5	max Dy	0	
		C-5	D-7	In-zone	—	
	Proximity	C-5	D-5			100
PTF-12-14	Hard	C-20	D-6	max Dx	0	
		C-20	D-6	max Dy	0	
		C-6	D-7	In-zone	—	
	Proximity	C-6	D-6			100

Table 6.16  
Proximity Weights and Location Constraints for all PTFs with Scenario II

<b>PTF</b>	<b>Res.</b>	<b>ΔVFL Equation</b>	<b>Position [X, Y, Orientation]</b>	<b>ΔVFL Value</b>
PTF-0-2			see Table 6.14	0
PTF-2-4	C-1	$50( X_1 - 96  +  Y_1 - 184 ) + 200( X_1 - 96  +  Y_1 - 205 )$	[96, 184, 0]	4,200
	C-20	None	[30.3, 197.5, 0]	0
PTF-4-6	C-2	$50( X_2 - 24  +  Y_2 - 104 ) + 200( X_2 - 3  +  Y_2 - 104 )$	[24, 104, 90]	4,200
	C-20	None	[10.5, 70.6, 90]	0
PTF-6-8	C-3	$50( X_3 - 96  +  Y_3 - 24 ) + 200( X_3 - 96  +  Y_3 - 3 )$	[96, 24, 0]	4,200
	C-20	None	[140.7, 10.5, 90]	0
PTF-8-10	C-4	$50( X_4 - 288  +  Y_4 - 24 ) + 200( X_4 - 288  +  Y_4 - 3 )$	[288, 24, 0]	4,200
	C-20	None	[320.4, 10.5, 90]	0
PTF-10-12	C-5	$50( X_5 - 360  +  Y_5 - 104 ) + 200( X_5 - 381  +  Y_5 - 104 )$	[360, 104, 90]	4,200
	C-20	None	[373.5, 152.5, 0]	0
PTF-12-14	C-6	$50( X_6 - 288  +  Y_6 - 184 ) + 200( X_6 - 288  +  Y_6 - 205 )$	[288, 184, 0]	4,200
	C-20	None	[245.6, 197.5, 90]	0

Table 6.17  
Summary of Resource Positions and VFL Values for PTFs in Scenario II

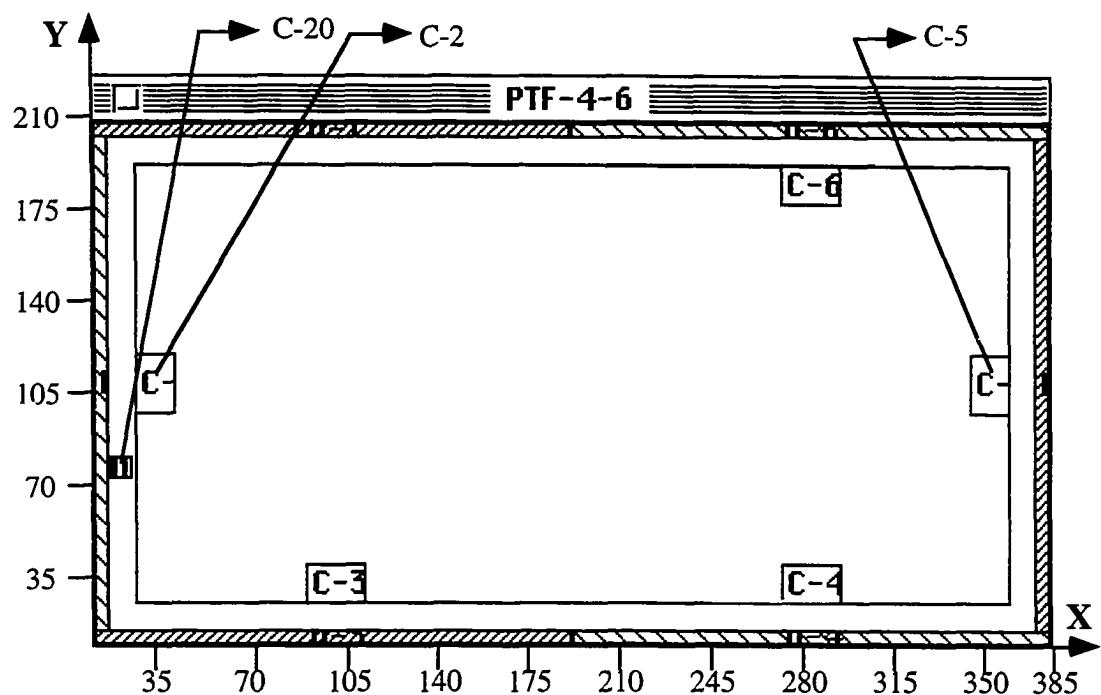


Figure 6.23  
Layout of PTF-4-6 With Scenario II

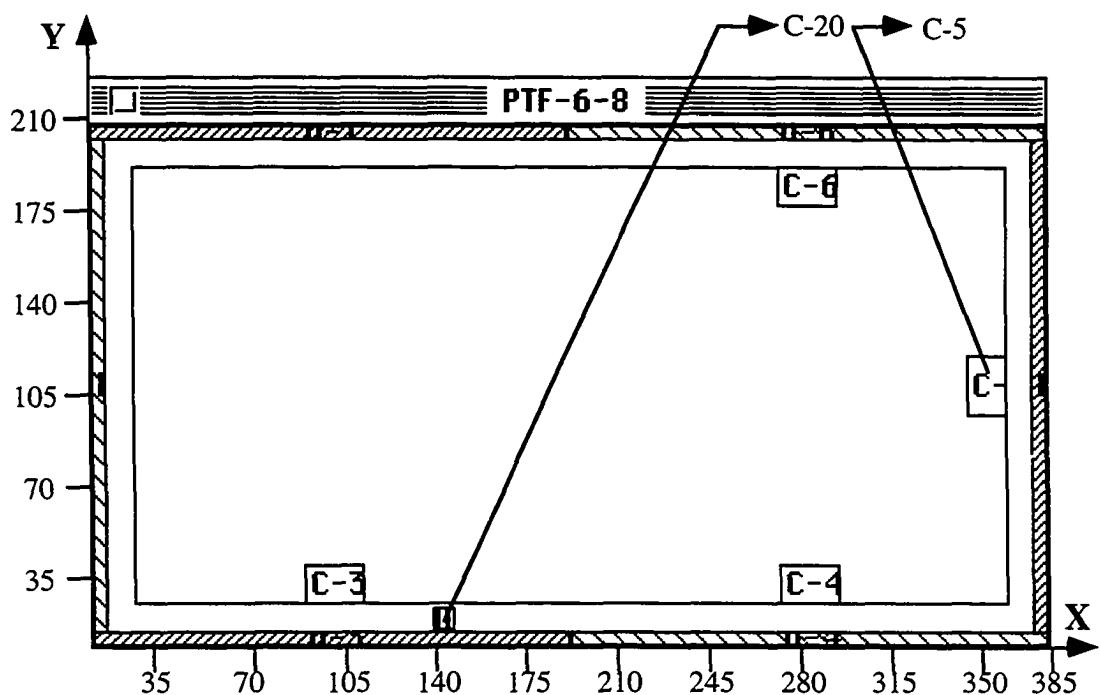


Figure 6.24  
Layout of PTF-6-8 With Scenario II

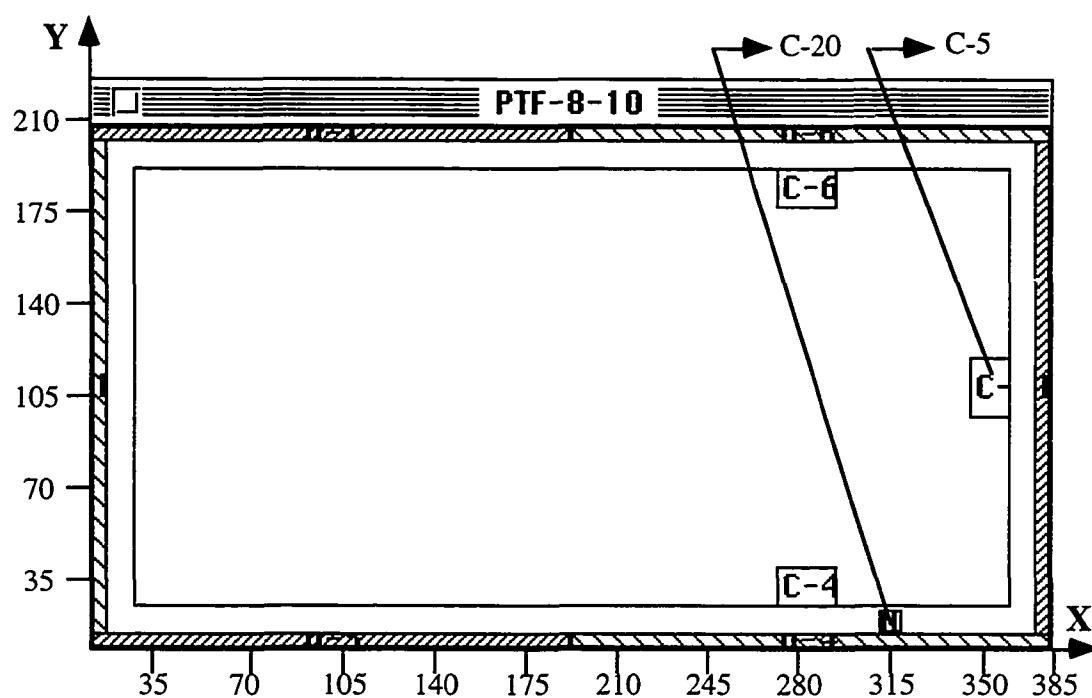


Figure 6.25  
Layout of PTF-8-10 With Scenario II

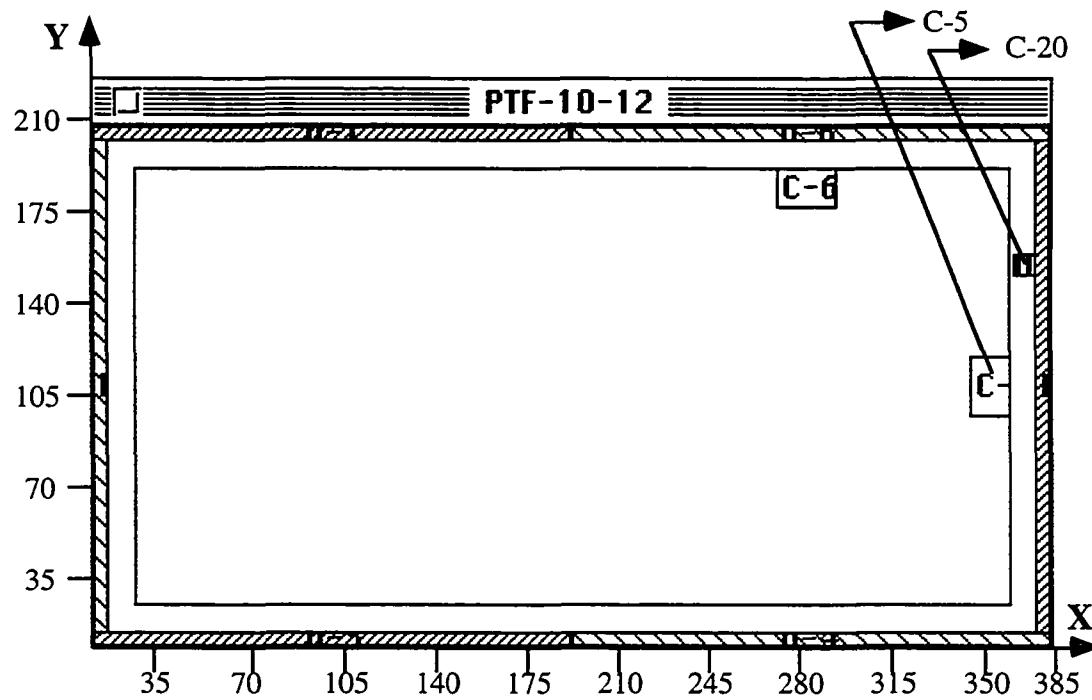


Figure 6.26  
Layout of PTF-10-12 With Scenario II

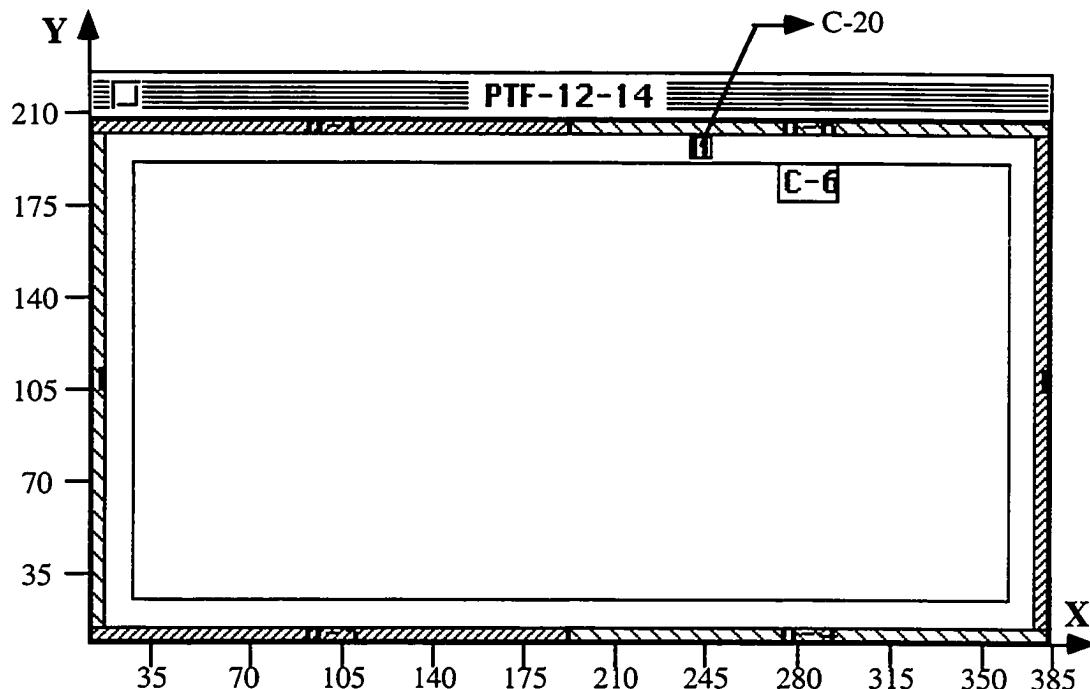


Figure 6.27  
Layout of PTF-12-14 With Scenario II

The total VFL value for this scenario is equal to 25,200. It represents the total layout cost excluding delivery costs for scenario II.

### 6.3.5 Comparison of Scenario I vs. Scenario II and Conclusions

By comparing the project duration and the layout costs of both scenarios, one sees that scenario II (with a project duration of 14 and total layout cost of 25,200) is preferred over scenario I (with a duration of 26 and total layout cost of 95,600). This result is expected and confirms one's intuitions about the preferred alternative. Note that delivery costs are not included in the total VFL value of either scenario. Section 6.3.2 justifies why these costs are equal for both scenarios and therefore will not compensate for the big difference in layout costs:  $95,600 - 25,200 = 70,400$  of the two scenarios.

This example, though simple, illustrates how the MoveSchedule model can be used to

- 1 describe alternative delivery schedules by accounting for the main variables that should be considered when managing materials on site, namely,**
  - installation costs (modeled as proximity weights)
  - relocation costs of materials in between staging areas (modeled as relocation weights)
  - constraints on acceptable laydown positions of material (modeled as hard constraints)
  - location where material is first unloaded (modeled as an initial layout input to MoveSchedule)
  - impact on the progress rate of construction operations (modeled in the resource levels and durations of activities)
- 2 assist the user in evaluating alternatives by**
  - solving for the intermediate locations of material on site before it reaches its final location
  - evaluating intermediate and overall relocation and installation costs of material (represented by VFL).

More importantly, this example shows that planning where material should be unloaded and where it gets relocated on site before it ends up at its final location pays off in cost and time savings. Models like MoveSchedule thus are needed because they articulate the effects that materials laydown and delivery schedules have on the project costs and the timeliness of construction operations.

# **Chapter 7**

## **MOVE SCHEDULE VALIDATION**

### **7.1 Introduction**

This chapter concerns the validation of MoveSchedule. The objective is to test the capabilities and limitations of the system in modeling a real project and in adjusting the project's schedule to comply with site space limitations. Recall that it is not within the scope or objectives of this research to rate MoveSchedule and the methods it implements based on the closeness of its solution to optimality, nor is it possible to compare its results to those of existing systems as such systems do not exist. Instead, MoveSchedule is tested with realistic data and its solution is qualitatively assessed by a former project engineer on the project that was selected for this validation. The site location and site space availability of the selected project did constrain the choice of construction methods and the project duration.

The following three sections describe the selected project and detail the steps undertaken by MoveSchedule in constructing the dynamic layout and adjusting the initial unconstrained schedule to comply with site space limitations. The last section of this chapter assesses the capabilities and limitations of MoveSchedule by comparing its solution to that of the project engineer based on three criteria. The first criterion assesses if the model employs an accurate representation of resource space needs and constraints on

resource positions. This criterion is used to validate the model underlying MoveSchedule from the point of view of representation. The second criterion compares the project duration to the actual duration and assesses if changes in resource levels or activity delays are practical or acceptable. This criterion is used to validate SCR and the space scheduling algorithm. The third criterion assesses the quality of the individual layouts. This last criterion is used to validate PTFLCA.

## 7.2 Project Description

The Harvard Square Parking Garage, built in 1986, is located at the heart of an extremely congested urban area in Cambridge, Massachusetts. The construction site is bounded on the south by Elliot Street, on the north by a two-story retail mall, on the east by J.F. Kennedy Blvd., and on the west by the driveway of an old two-story wooden structure. The project floor and elevation plans are documented in Appendix B.

Only the masonry phase of construction is modeled in MoveSchedule because: 1) it is the most affected by the site space conditions and 2) it was on the critical path and there was a great deal of pressure from the owner and Harvard University (the financial guarantor) to have the structure open in time for commencement to accommodate the accompanying flood of automobiles driven by parents of graduates.

The parking structure is essentially a helix of concrete slabs on a steel frame. It is 5 stories high with 108 parking spaces and several small retail shops on the ground level. The south and east elevations have brick facades on cast-in-place concrete. These elevations have many architectural features, including ornamental brickwork, open arches, and precast concrete sills. The south-east corner is a curve connecting the south and east elevations. The south elevation is bounded by a 16 ft-wide sidewalk and Elliot Street: a 4-lane street with no parking allowed. The east elevation is bounded by an 8 ft-wide sidewalk and J.F. Kennedy Blvd.: a 3-lane street with parking meters where space could be rented from the city when needed. The north elevation is a 50 ft-high and 110 ft-long concrete block wall. This wall abuts an existing two-story building and only the top 20 ft are visible from the street and covered with bricks. The west elevation is a plain-brick facade supported by a concrete-block structure. This elevation is 50 ft-high and 95 ft-long, built within 3/8 inch of the property line. For the construction of the brick facade, it was arranged to have access to the neighbor's driveway. A 5 ft-wide strip along the driveway

could be used to erect scaffolding for a limited time period of 12 days for the construction of this wall.

## 7.3 Modeling Project Data in MoveSchedule

The parking structure, the construction site including the sidewalks of the adjacent streets, metered parking spaces along J.F. Kennedy Blvd., and a fraction of the neighbor's driveway are modeled as rectangles in MoveSchedule, as shown in Fig. 7.1.

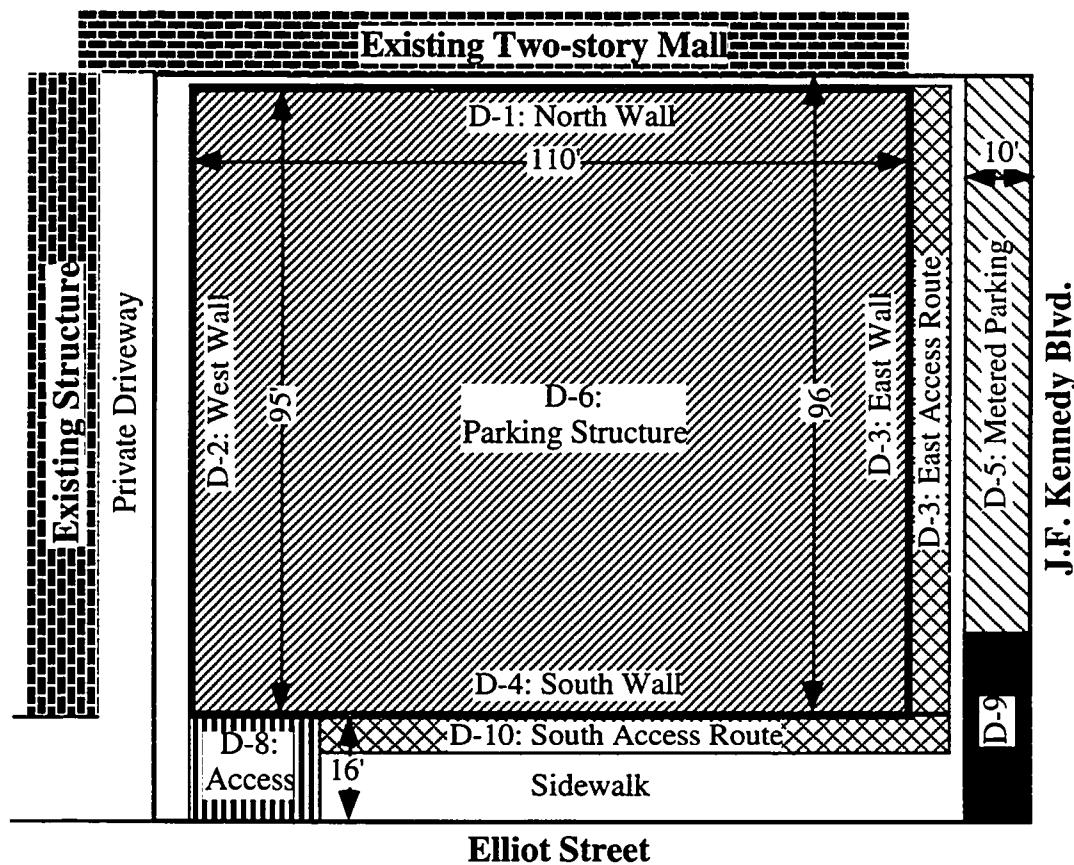


Figure 7.1  
Site Layout with Long-term Static Resources

The patterned rectangles represent reserved areas or laydown areas modeled in MoveSchedule. The vertically striped rectangle at the south-west corner of the site models the access to the parking structure which should be kept open at all times. The two 6 ft-wide double-cross-hatched rectangles along the south and east walls model construction access routes to be used exclusively by personnel or mobile equipment on this project.

These areas are also needed to keep space open along the south and east walls and to provide access from the metered parking area to the parking structure. The cross-hatched rectangle east of the construction site models the metered parking spaces along J.F. Kennedy Blvd. which are rented from the city to house the office trailer. The black rectangle at the south-east corner of the site models the curve.

The parking structure, the reserved areas, and the 4 walls are modeled as static independent resources that are on site for the duration of masonry construction. The office trailer is modeled as an independent but stationary resource whose position is to be determined by MoveSchedule. Data regarding independent resources as input to MoveSchedule is summarized in Table 7.1.

<b>Resource</b>	<b>Description</b>	<b>L x W</b>	<b>Relocation Weight</b>	<b>Fixed Position [X, Y, Orientation]</b>
D-1	North wall	110 x 0	Stationary	[60, 111, 0]
D-2	West wall	95 x 0	Stationary	[5, 63.5, 90]
D-3	East wall	95 x 0	Stationary	[115, 63.5, 90]
D-4	South wall	110 x 0	Stationary	[60, 16, 0]
D-5	Parking meters	80 x 10	Stationary	[128, 72, 90]
D-6	Parking structure	108 x 93	Stationary	[60, 63.5, 0]
D-7	Office trailer	60 x 10	Stationary	—
D-8	Access	20 x 16	Stationary	[15, 8, 0]
D-9	Reserved area	32 x 10	Stationary	[128, 16, 90]
D-10	South access route	96 x 6	Stationary	[68, 13, 0]
D-11	East access route	95 x 6	Stationary	[118, 63.5, 90]

**Table 7.1**  
**Independent Resource Data Input to MoveSchedule**

In addition to the independent resources, input to MoveSchedule includes:

- activities and precedence relationships between them
- up to three resource levels for performing each activity
- space profiles of dependent resources and related data

- positions of static dependent resources (if any)
- relocation weights of dependent resources
- interactions between resources described in the form of proximity weights, and hard constraints on relative positions of resources.

The masonry work for the parking garage consists of 6 main activities:

- Activity-1: the construction of the north concrete block wall
- Activity-2: the construction of the west concrete block wall
- Activity-3: the construction of the west wall's brick facade
- Activity-4: the construction of the north wall's brick facade
- Activity-5: the construction of the east wall's brick facade
- Activity-6: the construction of the south wall's brick facade.

Because of the building location and site space conditions, two alternative construction methods for performing some of the above activities are considered and are modeled by two different resource levels. The normal level includes the area requirements of resources used to perform the activity with conventional methods (i.e., with scaffolding erected on the outside of the parking structure) at a normal duration. The minimum level includes the area requirements of resources used to perform the activity with unconventional methods (i.e., with scaffolding erected from inside the parking structure) at a longer duration. All activities are initially set to be performed with conventional methods.

Data regarding the actual space requirements of resources and activity durations for both methods as they were executed in the field are not available for use by MoveSchedule. Therefore, activity durations are derived based on the quantities of work to be performed and the crews' productivity with conventional and unconventional construction methods. The area requirements of resources are derived from total quantities of material and common practice in procuring and delivering them. For details on these computations the reader is referred to section B.3 in Appendix B. The resulting data, as input to MoveSchedule, are summarized in Table 7.2 and Table 7.3.

Note that the last column of Table 7.2 lists the total area requirement of each activity at the designated resource level when the activity starts. This area requirement is the sum of the area requirements of individual resources at that resource level.

Activity Number	Description	Duration	Resource Level		Area @ Start
			Level	[Type, Area]	
1	North concrete block wall	15	Normal	[A-1 90] [B-1 240], [B-7 16] [C-1], [C-2], [C-3]	732
		25	Minimum	[A-1 90] [B-1 160], [B-7 16] [C-1], [C-2], [C-4]	652
2	West concrete block wall	13	Normal	[A-2 85] [B-2 240], [B-8 16] [C-1, C-2, C-5]	682
		22	Minimum	[A-2 85] [B-2 160], [B-8 16] [C-1], [C-2], [C-6]	593
3	West brick facade	12	Normal	[A-3 155] [B-3 108], [B-9 32] [C-1], [C-2], [C-7]	636
4	North brick facade	11	Normal	[A-4 92] [B-4 162], [B-10 16] [C-1], [C-2], [C-8]	638
		19	Minimum	[A-4 92] [B-4 54], [B-10 16] [C-1], [C-2]	218
5	East brick facade	15	Normal	[A-5 96] [B-5 54], [B-11 16] [C-1], [C-2], [C-9]	507
6	South brick facade	17	Normal	[A-6 102] [B-6 54], [B-12 16] [C-1], [C-2], [C-10], [C-11]	558

Table 7.2  
Activities and Resource Levels Input to MoveSchedule

<b>Profile</b>	<b>Description</b>	<b>L/W</b>	<b>L x W</b>	<b>Relocation Weight</b>	<b>Fixed Position</b>
A-1	Sand	1.7		500	—
A-2	Sand	1.7		500	—
A-3	Sand	1.6		500	—
A-4	Sand	1.7		500	—
A-5	Sand	1.7		500	—
A-6	Sand	1.7		500	—
B-1	Concrete blocks	2.5		1000	—
B-2	Concrete blocks	2.5		1000	—
B-3	Standard brick	3		1000	—
B-4	Standard brick	1.5		1000	—
B-5	Standard brick	1.5		1000	—
B-6	Standard brick	1.5		1000	—
B-7	Masonry cement	1		500	—
B-8	Masonry cement	1		500	—
B-9	Masonry cement	2		500	—
B-10	Masonry cement	1		500	—
B-11	Masonry cement	1		500	—
B-12	Masonry cement	1		500	—
C-1	Prime mover		7 x 4	0	—
C-2	Mortar mixer		7 x 4	750	—
C-3	Scaffolding 50'-H		110 x 3	Stationary	—
C-4	Scaffolding 10'-H		110 x 3	Stationary	—
C-5	Scaffolding 50'-H		95 x 3	Stationary	—
C-6	Scaffolding 10'-H		92 x 3	Stationary	—
C-7	Scaffolding 50'-H		95 x 3	Stationary	—
C-8	Scaffolding 50'-H		110 x 3	Stationary	—
C-9	Scaffolding 50'-H		95 x 3	Stationary	—
C-10	Scaffolding 50'-H		90 x 3	Stationary	—
C-11	Scaffolding 50'-H		20 x 3	Stationary	—

Table 7.3  
Dependent Resource Data Input to MoveSchedule

The scaffolding needed for performing activity-6 at the normal level is modeled using two different Profile-C resources: C-10 and C-11. C-11 is the scaffolding needed to cover the part of the wall that is blocked by D-8 (see Fig. 7.1) and C-10 is the scaffolding needed to cover the remaining part of the wall. Two resources are needed for this in MoveSchedule because different zoning constraints describe acceptable positions of each.

Precedences between activities are as shown in Fig. 7.2. Two milestone activities, M1 and M2, are added to the network of production activities to model the time period over which access to the neighbor's driveway is granted. A dummy static resource, C-12, is associated with both the start activity S and M1 to block the neighbor's driveway for the time period extending from the start of masonry work to the finish of M1. Similarly, another dummy static resource, C-13, is associated with both M2 and the finish activity F to block the neighbor's driveway for the time period extending from the start of M2 to the finish of all masonry work. Data related to the additional milestone activities and dummy resources are listed in Table 7.4 and Table 7.5.

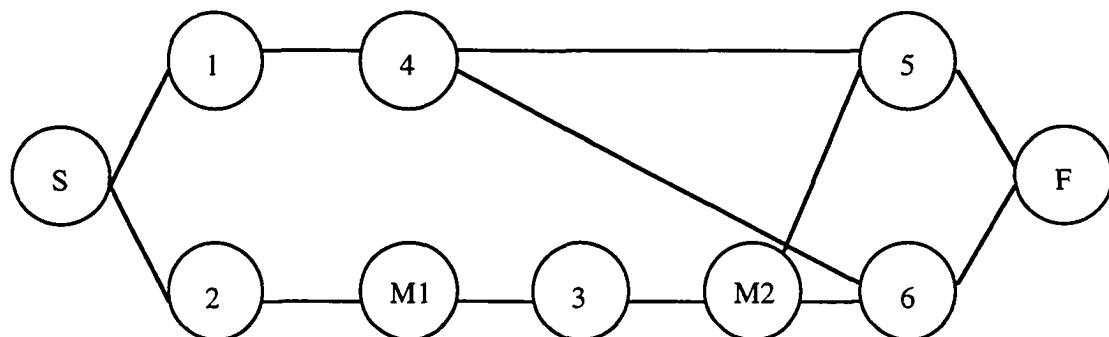


Figure 7.2  
Activity Network for Masonry Work

Activity	Description	Duration	Resource Level		Area @ Start
			Level	[Type, Area]	
S	Start	0	Normal	[C-12]	560
M1	Grant access	0	Normal	[C-12]	560
M2	Terminate access	0	Normal	[C-13]	560
F	Finish	0	Normal	[C-13]	560

Table 7.4  
Additional Milestone Activities

<b>Additional Profile C</b>	<b>Description</b>	<b>L x W</b>	<b>Relocation Weight</b>	<b>Fixed Position [X, Y, Orientation]</b>
C-12	Driveway closed	112 x 5	Stationary	[2.5, 56, 90]
C-13	Driveway closed	112 x 5	Stationary	[2.5, 56, 90]

**Table 7.5**  
**Additional Dummy Resources**

The modeled layout costs are transportation and relocation costs of construction materials. Proximity weights model unit transportation costs of materials and are defined based on the relative number of trips that the prime mover makes in transporting bricks, concrete blocks and mortar from their laydown areas to the work areas next to each wall. Concrete blocks, bricks, and mortar are delivered to the work areas on an as-needed basis in the ratio of 3 to 4 trips of concrete blocks or bricks for one trip of mortar. The cost of a single trip is arbitrarily set to 100. The proximity weight between concrete blocks or bricks and any of the walls is therefore 100; and the proximity weight between the mortar mixer (representing mortar) and any of the walls is  $100/3.5$  which is about 30. Similarly, in mixing mortar, material is delivered in the ratio of 4 loads of sand for every load of cement. Hence, if the proximity weight between sand and the mortar mixer equals 100 then the proximity weight between the cement bags and the mortar mixer equals 25.

Relocation weights model costs associated with moving materials (or other resources) from one staging area to another on site. In this project, relocation weights are defined as a relative measure based on the ease of relocating the resource. Because bricks and blocks are brittle and can be damaged during relocation, they are assigned a higher relocation weight than cement bags or sand which are not. Relocating the mortar mixer is assumed to be easier than relocating blocks and bricks. It is therefore assigned a relocation weight lower than that of bricks and blocks. The relocation weights of these resources are as listed in Table 7.3.

The hard constraints modeled in MoveSchedule concern the location and orientation of the scaffolding with respect to each wall. Hard constraints and the pairs of resources between which proximity weights apply are user-defined for each PTF in the course of dynamic layout construction.

## 7.4 Space Schedule Construction

Initially, all activities are scheduled to be performed at their shortest duration as defined by their normal resource level. The space scheduling algorithm first computes the unconstrained schedule, which corresponds to the shortest project duration, and identifies the corresponding PTFs (Fig. 7.3).

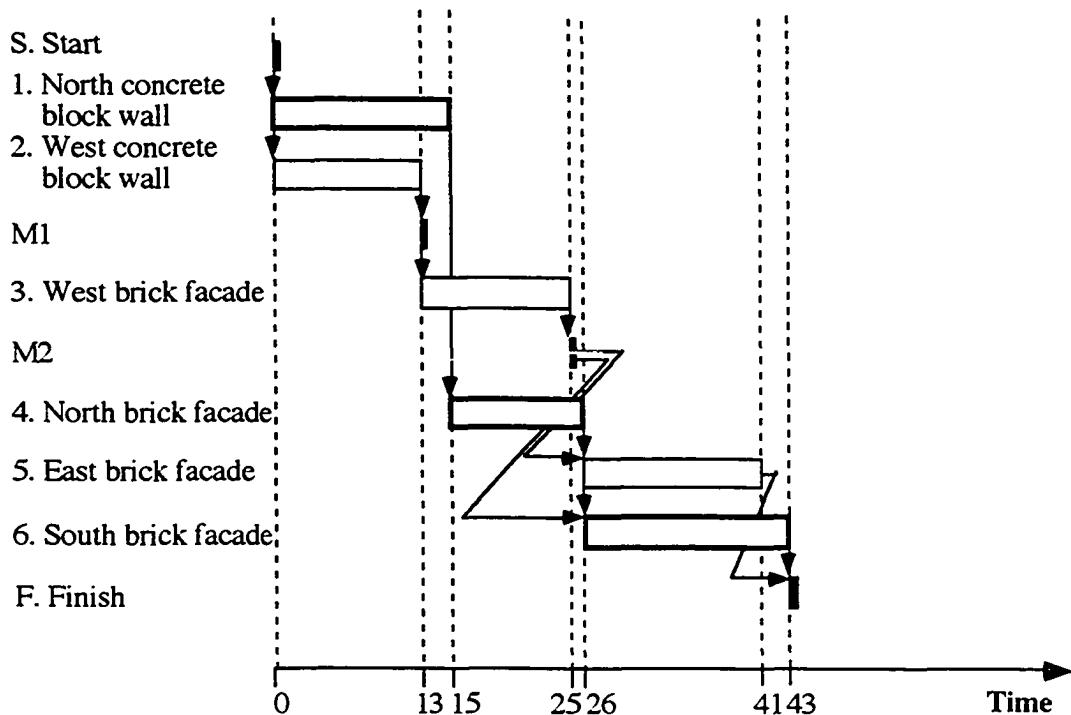


Figure 7.3  
Unconstrained Schedule and Primary Time Frames

The algorithm then assesses the feasibility of this initial schedule by constructing the dynamic site layout. For that, PTFLCA constructs the layout of the individual PTFs sequentially. PTFLCA constructs the layout of a PTF only once (i.e., the number of trials is set to 1 trial per PTF) because repeating a PTF layout construction would increase computations but not necessarily lead to more interesting findings for this project. Ties between candidate resources are broken according to the third and fourth tie-breaking rules (see section 4.3.4), that is, by choosing resources with the highest relocation weight first, and then resources with known positions in previously constructed layouts. If still tied, PTFLCA will choose a resource at random.

PTFLCA will determine the positions of all resources listed in Table 7.3 with the exception of the prime mover C-1. It is assumed that C-1 travels along D-10, D-11, through D-8, and inside the structure D-6. The free movement of C-1 is ensured by keeping these areas open and not using them for laying down construction materials.

#### 7.4.1 Layout Construction of PTF-0-13

In this time frame, the construction of the north (activity-1) and west (activity-2) concrete block walls is scheduled to start with conventional methods. The resources that need positioning are: the office trailer D-7, the scaffolding C-3 and C-5, the mortar mixer C-2, the concrete blocks B-1 and B-2 for the north and west walls D-1 and D-2, sand A-1 and A-2, and masonry cement B-7 and B-8. The proximity weights between them were rationalized in the previous section and are shown in Table 7.6. Note that C-3, C-5, and D-7 have no proximity weights with any other resource and therefore are not shown in Table 7.6.

A-1	A-2	B-1	B-2	B-7	B-8	C-2	D-1	D-2	
	-	-	-	-	-	100	-	-	A-1
	-	-	-	-	100	-	-	-	A-2
	-	-	-	-	-	100	-	-	B-1
	-	-	-	-	-	-	100	-	B-2
	-	-	-	-	25	-	-	-	B-7
	-	-	-	25	-	-	-	-	B-8
	-	-	-	30	30	-	-	-	C-2
	-	-	-	-	-	-	-	-	D-1
	-	-	-	-	-	-	-	-	D-2

Table 7.6  
Proximity Weights in PTF-0-13

Hard constraints (see Table 7.7) are needed to limit acceptable positions of the trailer D-7 to be within the metered parking area D-5, and those of the scaffolding C-3 and C-5 to be parallel and adjacent to D-1 and D-2 respectively.

The layout construction of PTF-0-13 is started by running CSPA to determine the resources' SPPs that satisfy the hard constraints. The algorithm first positions the static resources (i.e., D-1 through D-6, D-9 through D-11, and C-12) at their known positions in the layout and initializes the SPPs of the remaining stationary and relocatable resources in

PTF-0-13 by including them within the site boundaries. It then prunes the SPPs of stationary and relocatable resources to meet the default out-zone and hard constraints with the static resources.

Constraint #	Resource 1	Resource 2	Constraint Type	Value
1	D-7	D-5	In-zone	_
2	D-2	C-5	Parallel	_
3	C-5	D-2	West-of	_
4	C-5	D-2	max Dx	0
5	D-1	C-3	Parallel	_
6	C-3	D-1	North-of	_
7	C-3	D-1	max Dy	0

Table 7.7  
Location Constraints in PTF-0-13

As it turns out, satisfying the out-zone constraint between the stationary C-3 and the static resource C-12 reduces SPPC-3 to the null set. That is, no position can be found for the north scaffolding without overlapping with D-6. This result terminates CSPA and the spatial conflict is sent to SCR.

### Solving First Spatial Conflict

SCR looks for a strategy-activity combination to decrease the total area requirement in PTF-0-13 with a minimum increase in project duration. To this end, SCR first targets non-critical activities that start in PTF-0-13. Activity-2 is the only such activity, which has a total float value of 1. Two strategies apply to activity-2:

1. Delaying activity-2 (strategy A) to start at time 13 will increase the project duration by 12 days (i.e., 13 days, the duration of PTF-0-13, minus 1, activity-2's total float) and will reduce the total area required in PTF-0-13 by 654 (i.e., 682, from Table 7.2, minus 28, the area required by the prime mover C-1 which would still occupy space on site even if activity-2 were delayed).

2. Lowering the resource level of activity-2 (strategy B) to the minimum level will increase its duration by 9 days (i.e., 22 - 13, the difference between activity-2's duration at the minimum and the normal level from Table 7.2) which in turn will increase the project duration by 8 days (i.e., 9 - 1) and reduce the total area required by 89 (i.e., 682 - 593 from Table 7.2).

The second strategy is the best one, because it minimizes the increase in project duration. SCR posts this combination and calls the scheduler to schedule activity-2 at the minimum level and update the current schedule to reflect the change in activity-2's duration. The updated schedule (referred to as *schedule 1*) and the resulting PTFs are shown in Fig. 7.4.

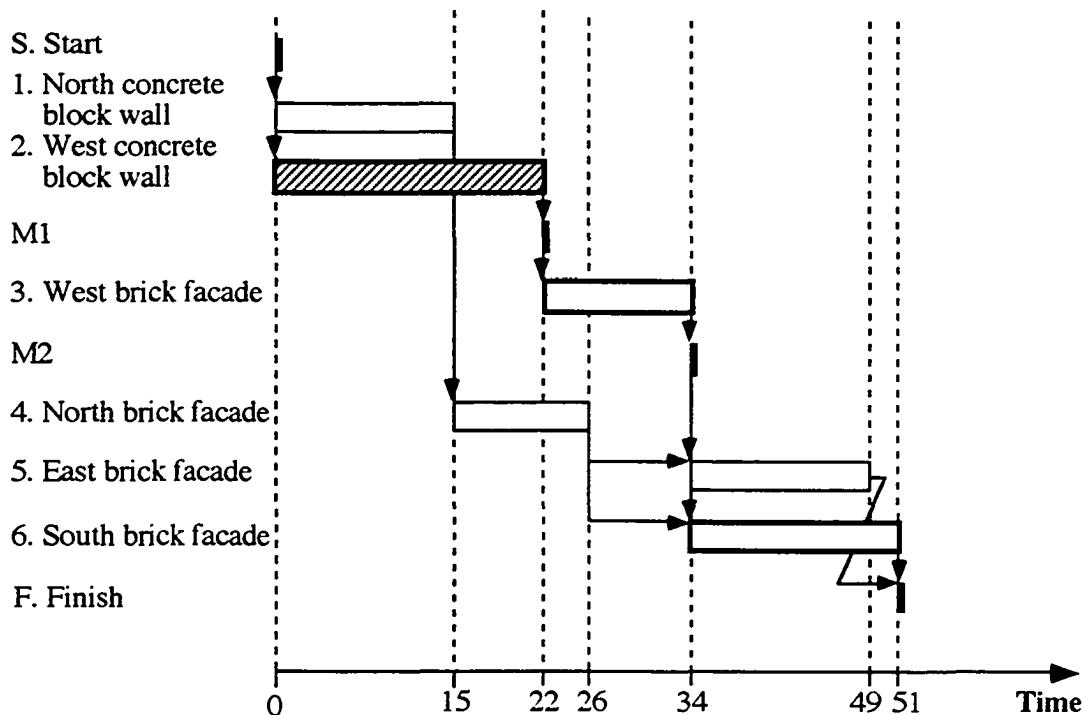


Figure 7.4  
Schedule 1: Adjusted Schedule after First Conflict

Note that the PTFs changed and PTF-0-13 is now PTF-0-15. Dynamic layout construction starts anew for the construction of PTF-0-15.

### 7.4.2 Layout Construction of PTF-0-15

In this time frame, the west concrete wall (activity-2) is constructed from inside the parking structure, as a result of changing the construction method of activity-2, and the north concrete wall is constructed from the outside, as originally planned. The resources present in PTF-0-15 are the same as those in PTF-0-13, with the exception of C-5, which is replaced by C-6 as a result of changing the construction method of activity-2.

The hard constraints are adjusted to reflect the change in zoning and orientation constraints of the west walls' scaffolding (see constraints #2, 3, and 4 in Table. 7.8).

Constraint #	Resource 1	Resource 2	Constraint Type	Value
1	D-7	D-5	In-zone	-
2	C-6	D-6	In-zone	-
3	D-2	C-6	Parallel	-
4	C-6	D-2	max Dx	0
5	C-6	D-4	max Dy	0
6	D-1	C-3	Parallel	-
7	C-3	D-1	North-of	-
8	C-3	D-1	max Dy	0

Table 7.8  
Location Constraints in PTF-0-15

The proximity weights (see Table 7.9) are adjusted to reflect the change in the construction method of the west wall. The concrete blocks B-2, and mortar C-2 needed in the construction of the west wall are delivered to the wall from inside the parking structure through the access D-8: hence, the proximity weights are re-defined to relate B-2 and D-8, and C-2 and D-8. Note that the distance traveled inside the structure can be ignored, because it is constant and will not vary with the position of B-2 on site. This is in contrast to the distance traveled outside the structure, which depends on the relative location of B-2 and C-2 from D-8.

The layout construction of PTF-0-15 starts by calling CSPA to compute the SPPs of resources that satisfy the default out-zone and hard constraints of Table 7.8.

Unfortunately, satisfying the out-zone constraint between C-3 and C-12 reduces SPPC-3 to the null set again. This occurs because lowering the resource level of activity-2 did not eliminate C-3 from the layout, nor did it affect the type of hard constraints it has with the static resources. Hence, it did not solve the spatial conflict in PTF-0-13.

A-1	A-2	B-1	B-2	B-7	B-8	C-2	D-1	D-8	
						100	—	—	A-1
						100	—	—	A-2
						100	—	—	B-1
						—	100	—	B-2
						25	—	—	B-7
						25	—	—	B-8
						—	30	30	C-2
						—	—	—	D-1
						—	—	—	D-8

Table 7.9  
Proximity Weights in PTF-0-15

Fortunately, SCR solved what could have been another conflict in PTF-0-13, that is, finding feasible positions for the scaffolding C-5. Indeed, satisfying any of the orientation constraints or the adjacency constraint (see Table 7.7) between C-5 and the west wall D-2 would reduce SPPC-5 to the null set causing a spatial conflict in PTF-0-13. In this case, though, lowering the resource level of activity-2 directly addresses the cause of this conflict because it replaces C-5 with C-6, which is subject to different zoning and hard constraints.

In short, SCR failed to address the cause of conflict in PTF-0-13, so it is called again to resolve the same spatial conflict in PTF-0-15.

### Solving Second Spatial Conflict

SCR looks at modifying schedule 1 to decrease the total area requirement in PTF-0-15 with minimum increase in project duration. SCR first targets the non-critical activities that start

in PTF-0-15. It finds activity-1, which has a total float value of 8. Two strategies apply to activity-1:

1. Delaying activity-1 (strategy A) to start at time 15 will increase the project duration by 7 days (i.e., 15 days, the duration of PTF-0-15, minus 8, activity-1's total float) and will reduce the total area required in PTF-0-15 by 704 (i.e., 732, from Table 7.2, minus 28, the area required by the prime mover C-1 which would still occupy space on site even if activity-1 were delayed).
2. Lowering the resource level of activity-1 (strategy B) will increase its duration by 10 days (i.e., 25 - 15, the difference between activity-1's duration at the minimum and normal level from Table 7.2), which in turn will increase the project duration by 2 days (i.e., 10 - 8) and reduce the total area required in PTF-0-15 by 80 (i.e., 732 - 652, from Table 7.2).

The second strategy is the best one because it minimizes the increase in project duration. SCR posts this combination and calls the scheduler to schedule activity-1 at the minimum level and update the current schedule to reflect this change. The updated schedule (referred to as *schedule 2*) and the resulting PTFs are shown in Fig. 7.5

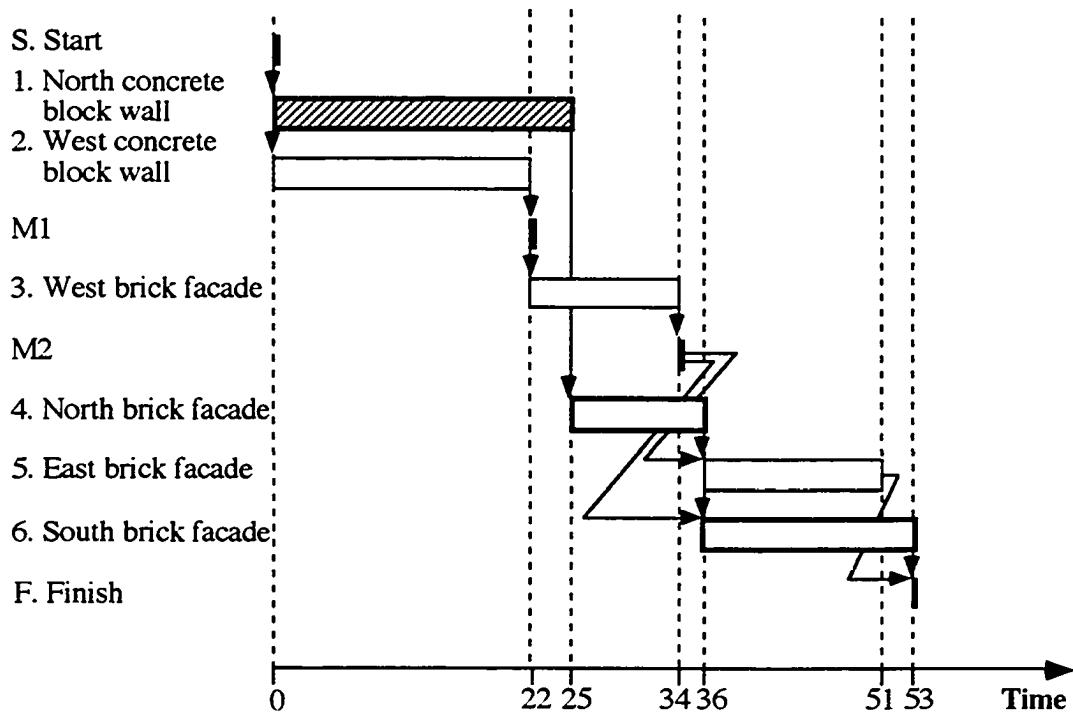


Figure 7.5  
Schedule 2: Adjusted Activity Schedule after Second Conflict

Note that the PTFs have changed again and dynamic layout construction starts anew for the construction of PTF-0-22.

### 7.4.3 Layout Construction of PTF-0-22

At this stage of problem solving, SCR had changed the construction method of activity-2 and activity-1 (see Solving First and Second Spatial Conflicts) so that both the north and west concrete block walls are constructed from inside the parking structure. As a result of this change, new zoning and orientation constraints are added to describe acceptable positions of the scaffolding C-4 (see Table 7.10).

Constraint #	Resource 1	Resource 2	Constraint Type	Value
1	D-7	D-5	In-zone	_
2	C-6	D-6	In-zone	_
3	D-2	C-6	Parallel	_
4	C-6	D-2	max Dx	0
5	C-6	D-4	max Dy	0
6	C-4	D-6	In-zone	_
7	D-1	C-4	Parallel	_
8	C-4	D-1	max Dy	0
9	C-4	D-3	max Dx	0

Table 7.10  
Location Constraints in PTF-0-22

The proximity weights between B-1 and D-1, and C-2 and D-1 are redefined to be between B-1 and D-8, and C-2 and D-8 (see Table 7.11) because deliveries of mortar and blocks for the north wall are done through D-8 now. Note that the weight between C-2 and D-8 has doubled in value because C-2 serves both the north and west walls through D-8.

A-1	A-2	B-1	B-2	B-7	B-8	C-2	D-8	
	—	—	—	—	—	100	—	A-1
		—	—	—	—	100	—	A-2
			—	—	—	—	100	B-1
				—	—	—	100	B-2
					—	25	—	B-7
						25	—	B-8
							60	C-2
								D-8

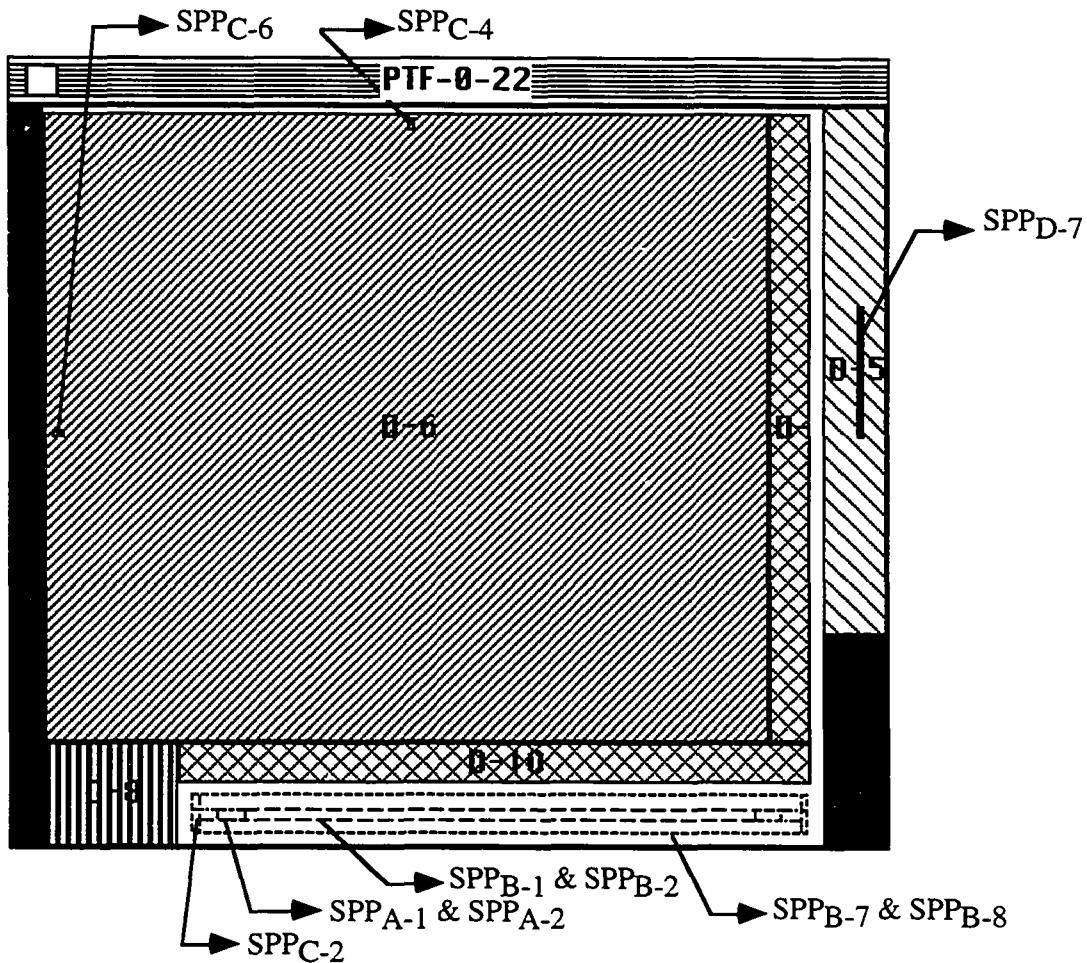
Table 7.11  
Proximity Weights in PTF-0-22

The resources to be positioned in PTF-0-22 are A-1, A-2, B-1, B-2, B-7, B-8, C-2, C-4, C-6, and D-7. CSPA starts anew to determine resources' SPPs that satisfy the default out-zone with static resources and all other hard constraints. This time around, it terminates successfully and the resulting resources' SPPs are as shown in Fig. 7.6. (Note that this figure and all following ones showing the layouts of PTFs are screen dumps from MoveSchedule's graphical layout display program. Arrows and labels have been added manually.)

PTFLCA is run next, for a selected resource, to find its positions in this time frame that minimize its transportation and relocation costs. This is done in two steps:

1. PTFLCA identifies all hard constraints between the selected resource and the positioned resources in the layout and determines the SPP that satisfies these constraints.
2. PTFLCA generates  $\Delta VFL$  (eq. 4) based on the position of the selected resource in the previous layout and proximity preferences with positioned resources in this PTF. It reduces the resource's SPP to minimize  $\Delta VFL$ . If more than one position is found, a single position is selected at random from its SPP.

Table 7.12 summarizes the steps undertaken by PTFLCA in constructing the layout of PTF-0-22. It shows, for each selected resource (first column in the table), its dimensions in that PTF (second column), the equation of  $\Delta VFL$  (third column), the SPP that minimizes  $\Delta VFL$  (fourth column), and its final position in the layout (fifth column). The order in which the resources are listed in Table 7.12 corresponds to the order in which they are positioned in the layout.



Legend:

- A-1 -> {{0 {[31.18 116.82] [3.64 6.36]}} {90 {}}}
- A-2 -> {{0 {[31.01 116.99] [3.54 6.46]}} {90 {}}}
- B-1 -> {{0 {[35.0 113.0] [4.0 6.0]}} {90 {}}}
- B-2 -> {{0 {[35.0 113.0] [4.0 6.0]}} {90 {}}}
- B-7 -> {{0 {[27.0 121.0] [2.0 8.0]}} {90 {[27.0 121.0] [2.0 8.0]}}}}
- B-8 -> {{0 {[27.0 121.0] [2.0 8.0]}} {90 {[27.0 121.0] [2.0 8.0]}}}}
- C-2 -> {{0 {[28.5 119.5] [2.0 8.0]}} {90 {[27.0 121.0] [3.5 6.5]}}}}
- C-4 -> {{0 {[60.0 60.0] [109.5 109.5]}} {90 {}}}
- C-6 -> {{0 {}} {90 {[6.5 6.5] [62.0 62.0]}}}}
- D-7 -> {{0 {}} {90 {[128.0 128.0] [62.0 82.0]}}}}

Figure 7.6  
SPPs of Resources at the End of CSPA for PTF-0-22

#	L x W	$\Delta VFL /100$	SPPs that Minimize $\Delta VFL$	Single Position [X, Y, Orientation]
D-7	60 x 10	—	$\{ \{ 0 \{ \} \} \\ \{ 90 \{ ([128 128] [62 82]) \} \} \}$	[128, 69.5, 90]
C-4	110 x 3	—	$\{ \{ 0 \{ ([60 60] [109.5 109.5]) \} \} \\ \{ 90 \{ \} \} \}$	[60, 109.5, 0]
C-6	92 x 3	—	$\{ \{ 0 \{ \} \} \\ \{ 90 \{ ([6.5 6.5] [62 62]) \} \} \}$	[6.5, 62, 90]
B-2	20 x 8	$22 ( X - 15  +  Y - 8 )$	$\{ \{ 0 \{ ([35 35] [6 6]) \} \} \\ \{ 90 \{ \} \} \}$	[35, 6, 0]
B-1	20 x 8	$22 ( X - 15  +  Y - 8 )$	$\{ \{ 0 \{ ([55 55] [6 6]) \} \} \\ \{ 90 \{ \} \} \}$	[55, 6, 0]
C-2	7 x 4	$13.2 ( X - 15  +  Y - 8 )$	$\{ \{ 0 \{ ([68.5 68.5] [8 8]) \} \} \\ \{ 90 \{ ([67 67] [6.5 6.5]) \} \} \}$	[68.5, 8, 0]
A-2	12 x 7	$22 ( X - 68.5  +  Y - 8 )$	$\{ \{ 0 \{ ([78 78] [6.5 6.5]) \} \} \\ \{ 90 \{ \} \} \}$	[78 6.5 0]
A-1	12.4 x 7.3	$22 ( X - 68.5  +  Y - 8 )$	$\{ \{ 0 \{ ([90.2 90.2] [6.4 6.4]) \} \} \\ \{ 90 \{ \} \} \}$	[90.2, 6.4, 0]
B-8	4 x 4	$5.5 ( X - 68.5  +  Y - 8 )$	$\{ \{ 0 \{ ([68.5 68.5] [4 4]) \} \} \\ \{ 90 \{ ([68.5 68.5] [4 4]) \} \} \}$	[68.5, 4, 90]
B-7	4 x 4	$5.5 ( X - 68.5  +  Y - 8 )$	$\{ \{ 0 \{ ([98.4 98.4] [8 8]) \} \} \\ \{ 90 \{ ([98.4 98.4] [8 8]) \} \} \}$	[98.4, 8, 0]

Table 7.12  
Solution Steps in the Layout Construction of PTF-0-22

PTFLCA positions all stationary resources first, before any relocatable resources. In PTF-0-22, C-4, C-6, and D-7 are the stationary resources. They have no proximity weights with any of the static resources in this PTF. Hence, PTFLCA sequences randomly.

PTFLCA selects D-7 first and prunes its SPP (determined by CSPA and shown in Fig. 7.6) to satisfy all hard constraints with the resources which have been positioned thus

far. Since none has been positioned yet, SPPD-7 is not changed. PTFLCA randomly selects a single position from SPPD-7 and binds D-7 at that position (see Table 7.12).

PTFLCA selects C-4 next. It prunes SPPC-4 (determined by CSPA and shown in Fig. 7.6) to satisfy the default out-zone constraint with D-7. SPPC-4 is not changed because it does not overlap with D-7. SPPC-4 is a singleton, hence, PTFLCA positions C-4 at the position determined by CSPA (see Table 7.12). Similarly, PTFLCA positions C-6 at the position determined by CSPA.

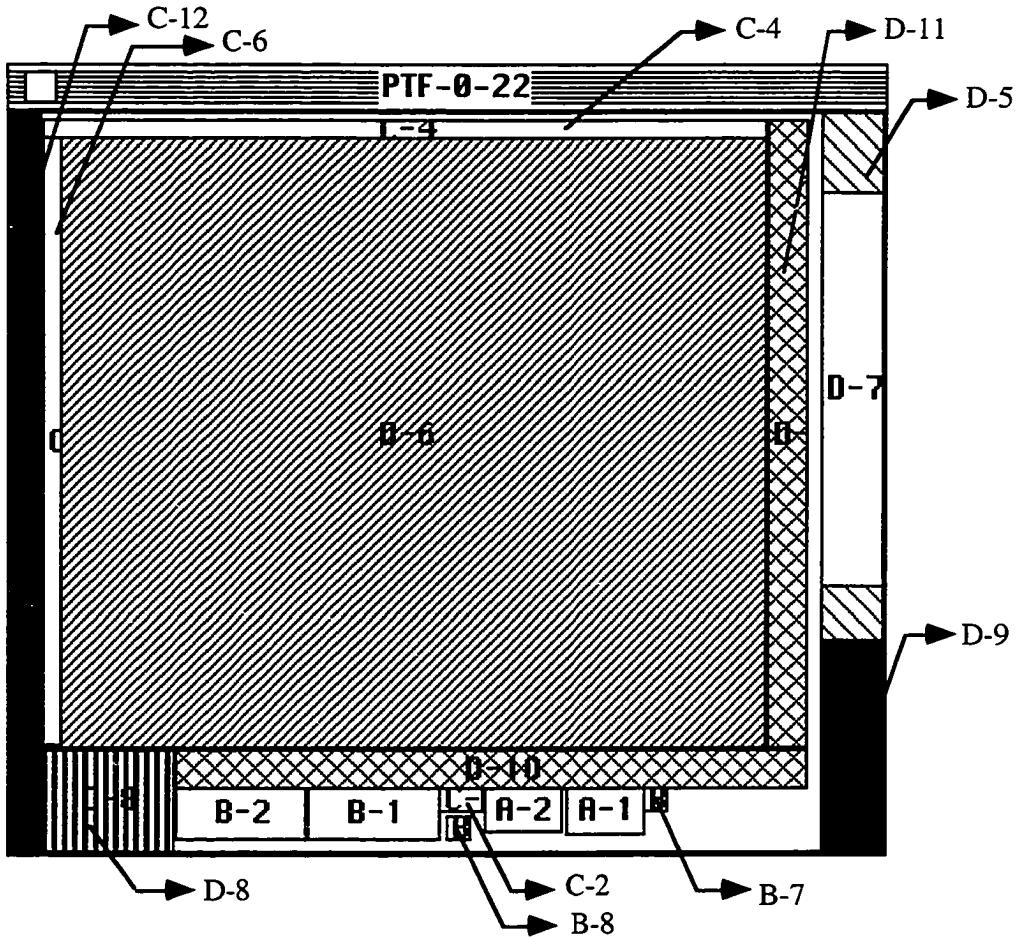
Among the relocatable resources, B-1, B-2, and C-2, PTFLCA selects the one that has the highest sum of proximity weights with positioned resources. B-1 and B-2, each of which has a proximity weight of 100 with D-8, are selected over C-2 though they are in a tie for being selected first. If one had been positioned in a previous layout, it would be given priority over the other, but neither one has a previous position. PTFLCA, therefore randomly picks one of the two, which in this case happens to be B-2.

The dimensions of B-2 are determined using its area requirement at time 0 (i.e., 160, activity-2 at minimum level in Table 7.2) and its L/W ratio (i.e., 2.5 from Table 7.3). SPPB-2 (from Fig. 7.6) is updated to account for the positioning of D-7, C-4, and C-6 by satisfying the default non-overlap constraints. No changes in SPPB-2 are noted. PTFLCA, then, generates  $\Delta VFL$ . In this case, it is the product of the weight (100 from Table 7.11) by the duration of PTF-0-22 (22) by the distance separating the positions of B-1 and D-8 ( $|X-15| + |Y - 8|$ ). PTFLCA minimizes  $\Delta VFL$  which results in a single position for B-2, namely, the position adjacent and closest to the centroid of D-8 (see Table. 7.12 and Fig. 7.7).

In a similar way, PTFLCA finds a position for B-1 that minimizes the distance between B-1 and D-8 (see Table 7.12 and Fig. 7.7).

C-2 is selected after B-1. PTFLCA finds two positions for it that minimize  $\Delta VFL$  (see fourth column in Table 7.12). The two positions have different orientations and both are adjacent to B-1. PTFLCA randomly selects a single position (see fifth column) for C-2; this position will constrain in a different way the positions of the remaining resources in this layout and in subsequent layouts.

The positions of A-2, A-1, B-8, B-7 are determined similarly. Fig. 7.7 shows all resources at their final positions in PTF-0-22. The VFL value for this layout is 305,712.



**Figure 7.7**  
Final Layout of PTF-0-22

(Note that the spacing between A-1 and A-2, and between C-2 and B-8 exists because MoveSchedule's layout display program rounds coordinates to the nearest integer)

#### 7.4.4 Layout Construction of PTF-22-25

In this time frame, the construction of the north wall (activity-1) continues and the construction of the west wall's brick facade (activity-3) is started. Prior to starting activity-3, permission for using the neighbor's driveway was obtained (i.e., the event M1 occurred) which means that C-12 no longer exists in this layout and that the space it occupied can be used. The static and the stationary resources positioned in PTF-0-22 (C-4 and D-7) will remain at their known positions. The resources that need positioning in this time frame are the relocatable resources from the previous layout: A-1, B-1, B-7, and C-2, and the new resources: A-3, B-3, B-9, and C-7.

Additional location constraints and proximity weights are defined for the new resources and are shown in Table 7.13 and Table 7.14.

Constraint #	Resource 1	Resource 2	Constraint Type	Value
1	D-2	C-7	Parallel	-
2	C-7	D-2	max Dx	0
3	C-7	D-4	min Dy	0
4	C-7	D-4	max Dy	0

Table 7.13  
Location Constraints in PTF-22-25

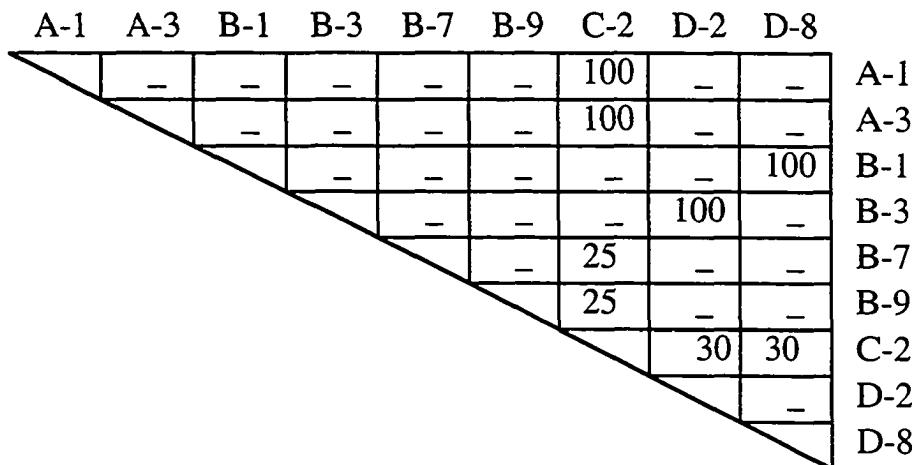


Table 7.14  
Proximity Weights in PTF-22-25

The construction of the west facade is done from outside the structure. Therefore, proximity weights are defined between bricks B-3, mortar C-2 and the west facade D-2. Note that the proximity weight between C-2 and D-8 is kept to model deliveries of mortar inside the structure as needed for the construction of the north concrete block wall. The weight's value, however, is reduced by half because mortar for the west wall is no longer delivered through D-8.

Next, the dimensions of Profile-A and -B resources are computed using their area requirement at time 22 and their respective L/W ratio (shown in Table 7.15). For example, the area required by A-1 at time 22 is  $90 \times (25 - 22)/25 = 10.8$ , its L/W is 1.7 (from Table

7.3) and its dimensions are 4.3 x 2.5. The area required by A-1 in PTF-22-25 is much smaller than that in PTF-0-22 because A-1 was consumed by activity-1 during the latter time frame.

CSPA determines the SPPs of resources in PTF-22-25 that satisfy the hard constraints of Table 7.13. Next, PTFLCA positions the stationary resource C-7 at the position found by CSPA (see Table 7.15).

#	L x W	$\Delta VFL /100$	SPPs that Minimize $\Delta VFL$	Single Position [X, Y, Orientation]
C-7	95 x 3	—	$\{\{0 \{\}\}$ $\{90 \{([3.5 3.5] [63.5 63.5])\}\}$	[3.5, 63.5, 90]
B-1	20 x 8	$10( X - 55  +  Y - 6 ) + 3( X - 15  +  Y - 8 )$	$\{\{0 \{([55 55] [6 6])\}\}$ $\{90 \{\}\}$	[55, 6, 0]
B-3	18 x 6	$3( X - 5  +  Y - 63.5 )$	$\{\{0 \{([34 34] [7 7])\}\}$ $\{90 \{\}\}$	[34, 7, 0]
C-2	7 x 4	$7.5( X - 68.5  +  Y - 8 ) + 0.9( X - 15  +  Y - 8 ) + 0.9( X - 5  +  Y - 63.5 )$	$\{\{0 \{([68.5 68.5] [8 8])\}\}$ $\{90 \{\}\}$	[68.5, 8, 0]
A-1	4.3 x 2.5	$5( X - 90.2  +  Y - 6.4 ) + 3( X - 68.5  +  Y - 8 )$	$\{\{0 \{([90.2 90.2] [6.4 6.4])\}\}$ $\{90 \{\}\}$	[90.2, 6.4, 0]
A-3	15.7 x 9.8	$3( X - 68.5  +  Y - 8 )$	$\{\{0 \{([79.9 79.9] [5 5])\}\}$ $\{90 \{\}\}$	[79.9, 5, 0]
B-7	4 x 4	$5( X - 98.4  +  Y - 8 ) + 7.5( X - 68.5  +  Y - 8 )$	$\{\{0 \{([98.4 98.4] [8 8])\}\}$ $\{90 \{([98.4 98.4] [8 8])\}\}$	[98.4, 8, 0]
B-9	8 x 4	$7.5( X - 68.5  +  Y - 8 )$	$\{\{0 \{\}\}$ $\{90 \{([94.4 94.4] [6 6])\}\}$	[94.4, 6, 90]

Table 7.15  
Solution Steps in the Layout Construction of PTF-22-25

B-1 and B-3 are the next candidate resources for entering the layout. PTFLCA selects B-1 because it has a position in the layout of PTF-0-22 and B-3 does not. This rule

gives higher priority to resources with previous positions to increase the likelihood that they remain at their previous position when relocating them is costly. PTFLCA then generates  $\Delta VFL$  (see Table 7.15) and finds that the position [55, 6, 0], which is B-1's position in PTF-0-22, minimizes  $\Delta VFL$ . Note that the space freed by B-2 was not used by PTFLCA to relocate B-1. This is so because the potential savings of positioning B-1 adjacent to D-8 (i.e., the second term in  $\Delta VFL$ ) does not outweigh the high cost of relocating B-1 (i.e., the first term in  $\Delta VFL$ ). The space occupied by B-2 in PTF-0-22 is reused, however, to accommodate B-3 (see Fig. 7.8).

After B-3, C-2 is the only resource that has proximity weights with positioned resources D-2 and D-8, so it is selected next. PTFLCA keeps C-2 at its previous position because this position is the closest one to D-2 and D-8.

Both A-1 and A-3 have a weight of 100 with C-2 and, therefore, are both candidates to enter the layout. PTFLCA selects A-1 first because it has a previous position. Here too, PTFLCA does not find it cost-effective to relocate A-1 closer to C-2. The free space left by A-2 and the shrunk A-1 is reused to accommodate A-3.

Similarly, the positions of B-7 and B-9 are determined. PTFLCA positions B-7 at its previous position. It then positions B-9 in between A-1 and B-7 as this position minimizes the distance separating B-9 from C-2. Here too, the space freed by A-1 is reused by B-9. The final layout is as shown in Fig. 7.8 and has a VFL value of 69,393.

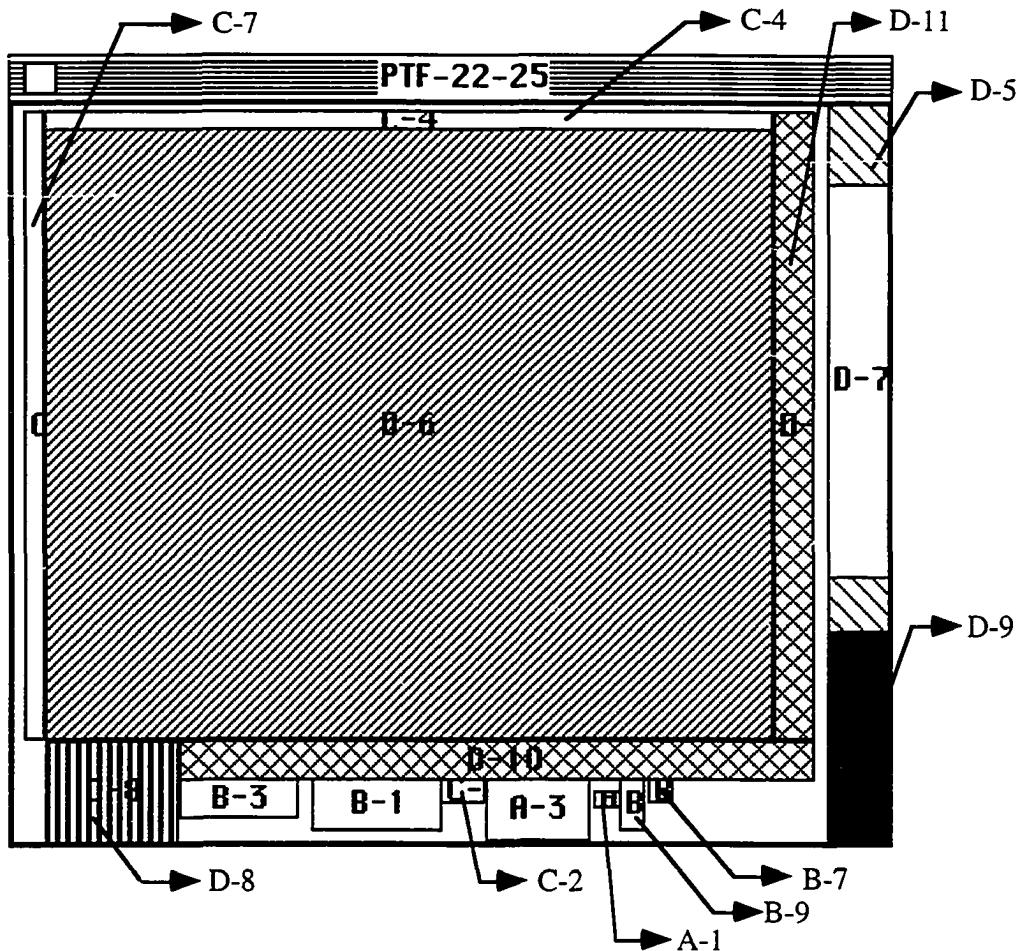


Figure 7.8  
Final Layout of PTF-22-25

#### 7.4.5 Layout Construction of PTF-25-34

In PTF-25-34, the construction of the west facade (activity-3) continues and the construction of the north facade (activity-4) is scheduled to start with conventional methods. The new resources introduced to the layout in this time frame are: A-4, B-4, B-10, and C-8. Additional proximity weights (see Table 7.16) are defined between sand A-4, masonry cement B-9 and the mortar mixer C-2, and between the bricks B-4, C-2 and the north wall D-1.

A-3	A-4	B-4	B-3	B-9	B-10	C-2	D-1	D-2	
	-	-	-	-	-	100	-	-	A-3
		-	-	-	-	100	-	-	A-4
			-	-	-	-	-	100	B-3
				-	-	-	100	-	B-4
					-	25	-	-	B-9
						25	-	-	B-10
							30	30	C-2
								-	D-1
									D-2

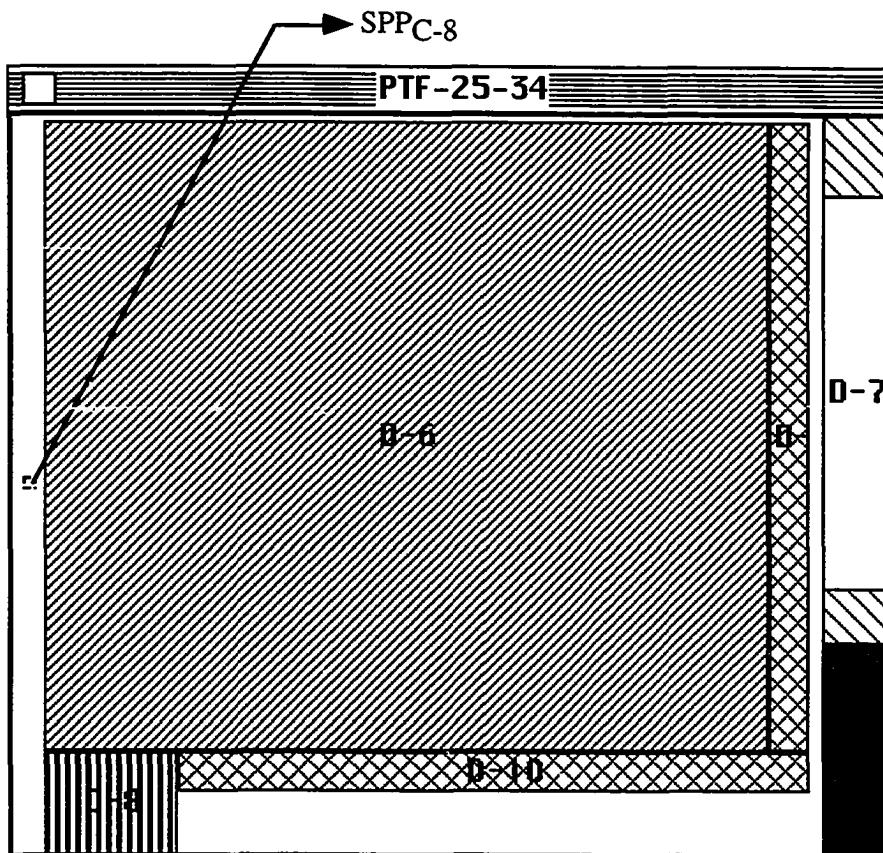
Table 7.16  
Proximity Weights in PTF-25-34

Additional orientation and adjacency constraints (see Table 7.17) are defined to limit the acceptable positions of the scaffolding C-8 needed to construct the north brick facade.

Constraint #	Resource 1	Resource 2	Constraint Type	Value
1	D-1	C-8	Parallel	-
2	C-8	D-1	North-of	-
3	C-8	D-1	max Dy	0

Table 7.17  
Location Constraints in PTF-25-34

The static and the previously-positioned stationary resources remain at their position. CSPA computes SPPs of resources that satisfy the hard constraints of Table 7.17. Fig. 7.9 shows SPPC-8 after applying the default out-zone constraints with the static resources but before applying the out-zone constraint with C-7 and the parallel constraint with D-1 (constraint #1). CSPA then fails to find a position that satisfies these last two constraints. The spatial conflict is sent to SCR.



**Legend:**

C-8 -> {{0 {}} {90 {[1.5 3.5] [55 57]}}}}

Figure 7.9  
SPPC-8 in PTF-25-34 before Applying the Parallel Constraint with D-1  
and the Out-zone Constraint with C-7

**Solving Third Spatial Conflict**

SCR looks at modifying schedule 2 to decrease the total area required in PTF-25-34 with a minimum increase in project duration. To this end, SCR first targets non-critical activities that start in PTF-25-34, but it finds none.

Next, SCR considers critical activities that start at time 25. It finds only activity-4. Two strategies apply to activity-4:

1. Delaying activity-4 (strategy A) to start at time 25 will increase the project duration by 9 days (i.e., the duration of PTF-25-34), and will reduce the total area requirement in this time frame by 610 (i.e., 638, from Table 7.2, minus 28,

the area required by the prime mover which will remain on site even if activity-4 is delayed).

2. Lowering the resource level (strategy B) of activity-4 will increase its duration and, consequently, the project duration by 8 days (i.e., 19 - 11, the difference between activity-4's duration at the minimum and normal level from Table 7.2) and will reduce the total area required in PTF-25-34 by 420 (i.e., 638 - 218, from Table 7.2).

The second strategy is the best one, because it minimizes the increase in project duration. SCR posts this combination and calls the scheduler to schedule activity-4 at the minimum level and update schedule 2 to reflect this change. The updated schedule (referred to as *schedule 3*) and the resulting PTFs are shown in Fig. 7.10

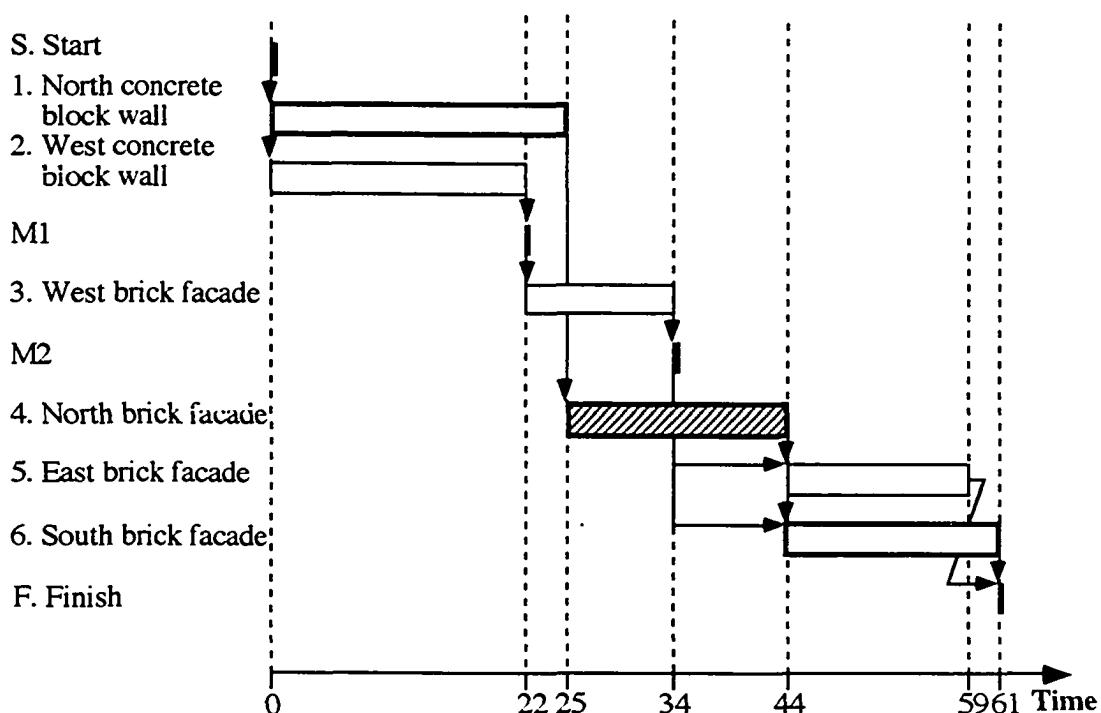


Figure 7.10  
Schedule 3: Adjusted Activity Schedule after Third Conflict

Note that the begin and end dates of PTF-25-34 are unaffected by the change in the schedule and dynamic layout construction starts anew for PTF-25-34.

### Layout Construction of PTF-25-34 After Schedule Modification

Construction of the north brick facade (activity-4) is now performed using a scaffolding system suspended down from the top slab. Proximity weights in PTF-25-34 (see Table 7.18) are, therefore, adjusted to reflect that bricks (B-4) and mortar (C-2) are delivered through D-8 and up to the roof.

A-3	A-4	B-4	B-3	B-9	B-10	C-2	D-2	D-8	
—	—	—	—	—	100	—	—	—	A-3
—	—	—	—	—	100	—	—	—	A-4
—	—	—	—	—	—	100	—	—	B-3
—	—	—	—	—	—	—	100	—	B-4
—	—	—	—	—	25	—	—	—	B-9
—	—	—	—	—	25	—	—	—	B-10
—	—	—	—	—	—	30	30	—	C-2
—	—	—	—	—	—	—	—	—	D-2
—	—	—	—	—	—	—	—	—	D-8

Table 7.18  
Adjusted Proximity Weights in PTF-25-34 after Third Conflict

The hard constraints of Table 7.17 no longer apply because no scaffolding is modeled for activity-4 at the minimum level: the scaffolding cantilevers from the roof down and does not occupy any space on the ground.

PTFLCA checks if the previously-positioned and stationary resource C-7 can occupy the same position it occupied in PTF-22-25. PTFLCA keeps C-7 at its previous position as this position does not conflict with any other positioned resource.

B-3 is selected next, because it has the highest sum of weights and was positioned in an earlier layout. PTFLCA keeps it at its position since there are no other positions that would bring it closer to D-2 (see Table 7.19).

Next, PTFLCA determines optimal positions of B-4 that minimize the distance with D-8. Two alternative positions that are adjacent to B-3 at 0° and 90° orientations are found. PTFLCA randomly selects the one at 0° orientation. Needless to say, this choice will have repercussions on the positions of the remaining resources in all layouts. It is lucky that PTFLCA chose this orientation, because otherwise the space between C-2 and B-3 would be enough to accommodate only B-4 and not both B-4 and A-4 (see Fig. 7.11).

#	L x W	$\Delta VFL /100$	SPPs that Minimize $\Delta VFL$	Single Position [X, Y, Orientation]
B-3	18 x 6	$10( X - 34  +  Y - 7 ) + 9( X - 5  +  Y - 63.5 )$	$\{ \{ 0 \{ ([34 34] [7 7]) \} \} \\ \{ 90 \{ \} \} \}$	[34, 7, 0]
B-4	9 x 6	$10( X - 55  +  Y - 6 ) + 3( X - 15  +  Y - 8 )$	$\{ \{ 0 \{ ([47.5 47.5] [7 7]) \} \} \\ \{ 90 \{ ([46 46] [5.5 5.5]) \} \} \}$	[46, 5.5, 90]
C-2	7 x 4	$7.5( X - 68.5  +  Y - 8 ) + 2.7( X - 15  +  Y - 8 ) + 2.7( X - 5  +  Y - 63.5 )$	$\{ \{ 0 \{ ([68.5 68.5] [8 8]) \} \} \\ \{ 90 \{ \} \} \}$	[68.5, 8, 0]
A-3	13.6 x 8.5	$5( X - 79.9  +  Y - 5 ) + 9( X - 68.5  +  Y - 8 )$	$\{ \{ 0 \{ ([78.8 78.8] [5.7 5.7]) \} \} \\ \{ 90 \{ \} \} \}$	[78.8, 5.7, 0]
A-4	12.5 x 7.4	$9( X - 68.5  +  Y - 8 )$	$\{ \{ 0 \{ ([58.7 58.7] [6.3 6.3]) \} \} \\ \{ 90 \{ \} \} \}$	[58.7, 6.3, 0]
B-9	8 x 4	$5( X - 94.4  +  Y - 6 ) + 2.25( X - 68.5  +  Y - 8 )$	$\{ \{ 0 \{ \} \} \\ \{ 90 \{ ([94.4 94.4] [6 6]) \} \} \}$	[94.4, 6, 90]
B-10	4 x 4	$2.25( X - 68.5  +  Y - 8 )$	$\{ \{ 0 \{ ([68.5 68.5] [4 4]) \} \} \\ \{ 90 \{ ([68.5 68.5] [4 4]) \} \} \}$	[68.5, 4, 0]

Table 7.19  
Solution Steps in the Layout Construction of PTF-25-34

C-2 is selected after B-4 and is kept at its position in the previous layout. Again, the benefit of moving it closer to D-8 (i.e., adjacent to B-4) does not outweigh the high cost of relocating it (see  $\Delta VFL$  of C-2 in Table 7.19).

Note that the space required by A-3 in PTF-25-34 is smaller than that in PTF-22-25. As a result of this, and because of the relatively low cost of relocating A-3 (see  $\Delta VFL$  of A-3 in Table 7.19), PTFLCA relocates it to keep it adjacent to C-2, with which it shares a proximity weight of 100.

Next, A-4 is positioned adjacent to C-2 to minimize the distance separating them. This was possible because B-4 is oriented at 90°, thus providing enough space to squeeze A-4 in between C-2 and B-4 (see Fig. 7.11).

The positions of B-9 and B-10 are determined in similar fashion. The final layout of PTF-25-34 has a VFL value of 183,303 and is shown in Fig. 7.11.

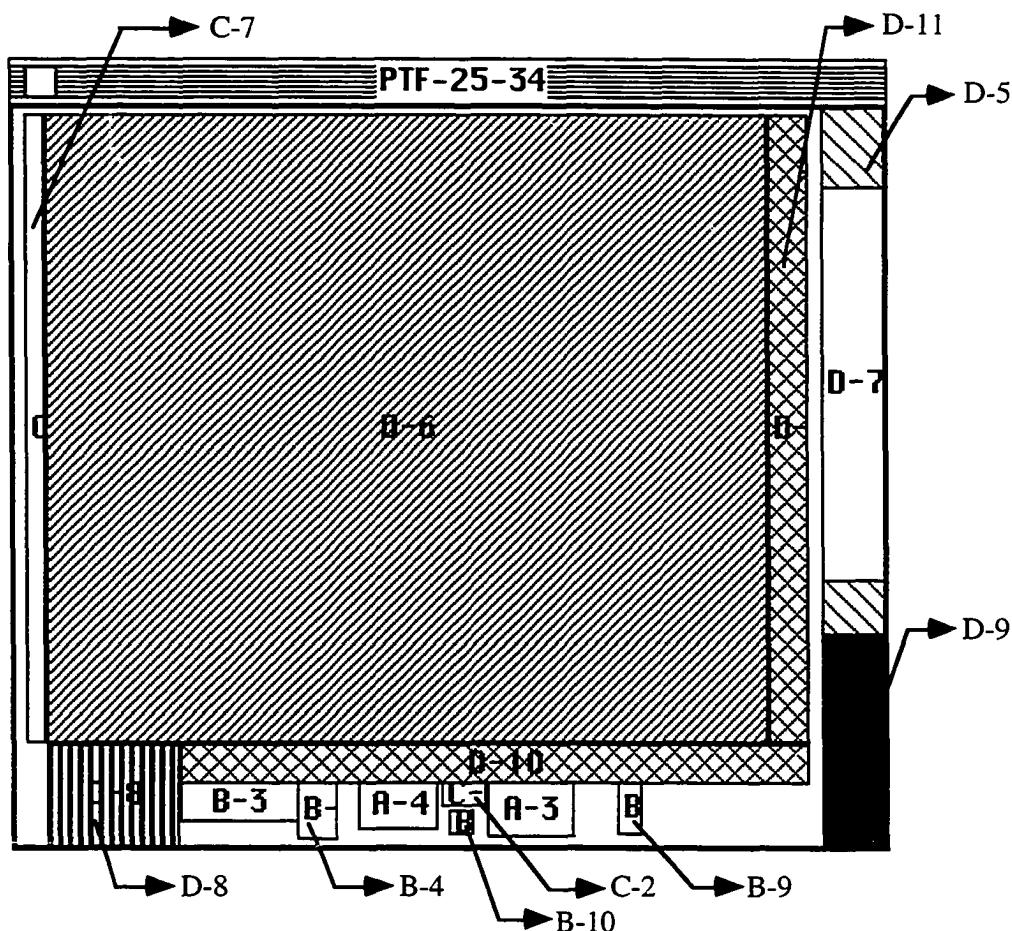


Figure 7.11  
Final Layout of PTF-25-34

#### 7.4.6 Layout Construction of PTF-34-44

In this PTF, activity-3 has finished, permission for using the driveway is withdrawn (i.e., the event M2 has occurred), and activity-4 continues with no other activity scheduled to start. The static resource C-13 is added to the layout to block the neighbor's driveway.

The resources that need positioning in this time frame are the relocatable resources from the previous layout: A-4, B-4, B-10, and C-2.

No additional hard constraints are defined for this PTF. The proximity weights (see Table 7.20) are the same as those defined for PTF-25-34.

	A-4	B-4	B-10	C-2	D-8	
	—	—	100	—	—	A-4
	—	—	—	100	—	B-4
	—	—	25	—	—	B-10
	—	—	—	30	—	C-2
	—	—	—	—	—	D-8

Table 7.20  
Proximity Weights in PTF-34-44

PTFLCA positions B-4 first. It finds alternative optimal positions for B-4 at both orientations that range from its previous position in PTF-24-34 at [46, 5.5, 90] to positions adjacent to D-8 (see SPPB-4 in Table 7.21). PTFLCA is indifferent between these positions since all have the same cost, so it randomly selects a single position [39.8, 5.5, 90] from SPPB-4. Note that a person might have selected the position adjacent to D-8 because it is not evident that these positions are iso-cost positions. If the user can justify the relocation of B-4 then the user can move it to the position closest to D-8.

#	L x W	$\Delta VFL /100$	SPPs that Minimize $\Delta VFL$	Single Position [X, Y, Orientation]
B-4	9 x 6	$10( X - 46  +  Y - 5.5 ) + 10( X - 15  +  Y - 8 )$	{ {0 {[29.5 46] [5.5 7]}}, {90 {[28 46] [5.5 5.5]}} }	[39.8, 5.5, 90]
C-2	7 x 4	$7.5( X - 68.5  +  Y - 8 ) + 3( X - 15  +  Y - 8 )$	{ {0 {[68.5 68.5] [8 8]}}, {90 {}} }	[68.5, 8, 0]
A-4	9 x 5.3	$5( X - 58.7  +  Y - 6.3 ) + 10( X - 68.5  +  Y - 8 )$	{ {0 {[60.5 60.5] [7.3 7.3]}}, {90 {}} }	[60.5, 7.3, 0]
B-10	4 x 4	$5( X - 68  +  Y - 4 ) + 2.5( X - 68.5  +  Y - 8 )$	{ {0 {[68.5 68.5] [4 4]}}, {90 {[68.5 68.5] [4 4]}} }	[68.5, 4, 0]

Table 7.21  
Solution Steps in the Layout Construction of PTF-34-44

C-2 and B-10 remain at their previous positions. A-4 shrinks but stays adjacent to C-2; relocation cost is incurred because A-4's centroid has moved, though the resource stayed within the boundaries of the space it occupied in PTF-25-34. This case is similar to that of A-3 in PTF-25-34. The final layout of PTF-34-44 has a VFL value of 60,620 and is shown in Fig. 7.12.

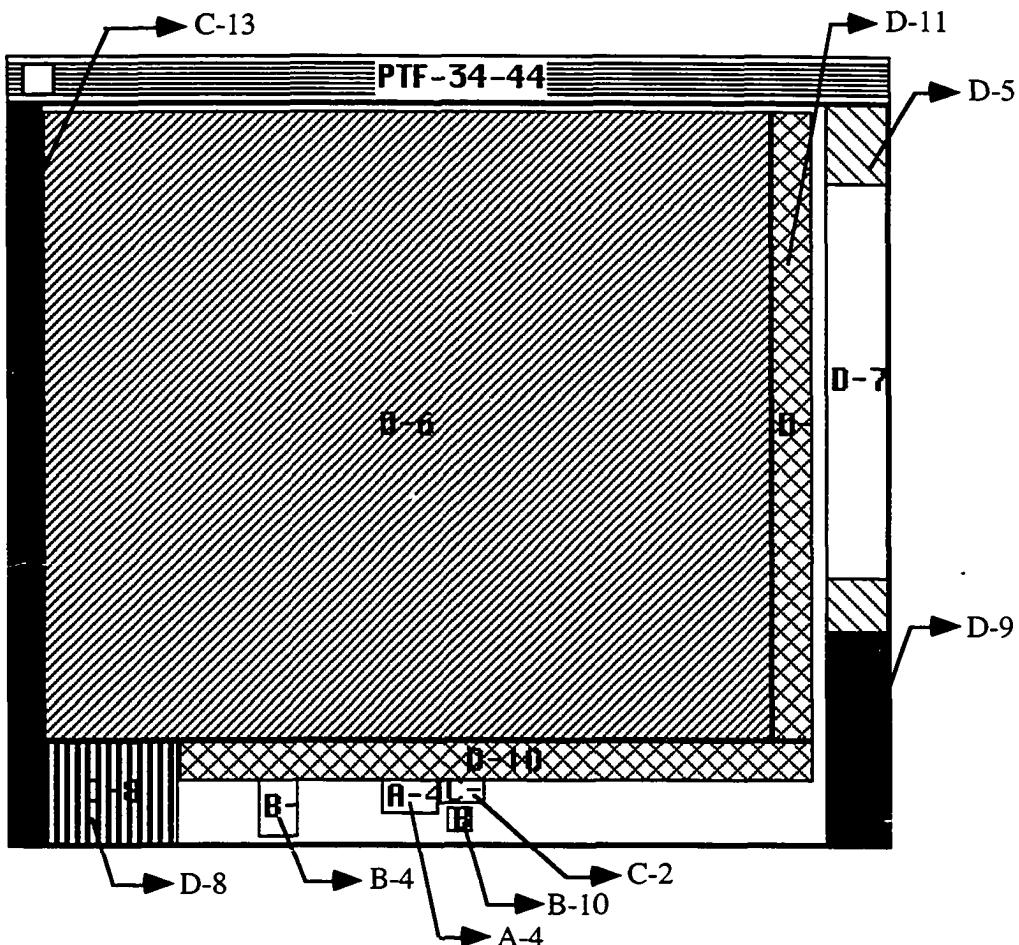


Figure 7.12  
Final Layout of PTF-34-44

#### 7.4.7 Layout Construction of PTF-44-59

In this time frame, the construction of the east and south brick facades (activity-5 and activity-6) are started and all other activities have been completed. Hard constraints (see Table 7.22) are defined to orient the scaffolding C-9 with respect to the east Wall D-3. Similar constraints are defined to orient C-10 and C-11 with respect to the south Wall D-4.

C-10 and C-11 are subject to different zoning constraints as the former overlaps with D-8 and the latter overlaps with D-10.

Constraint #	Resource 1	Resource 2	Constraint Type	Value
1	C-10	D-10	In-zone	-
2	D-4	C-10	Parallel	-
3	C-10	D-4	max Dy	0
4	C-10	C-11	max Dx	0
5	C-11	D-8	In-zone	0
6	C-11	D-4	max Dy	0
7	C-9	D-11	In-zone	-
8	D-3	C-9	Parallel	-
9	C-9	D-3	max Dx	0

Table 7.22  
Location Constraints in PTF-44-59

Proximity weights (see Table 7.23) are defined to model the transportation of bricks and mortar to D-3 and D-4, and the transportation of sand and cement to the mortar mixer.

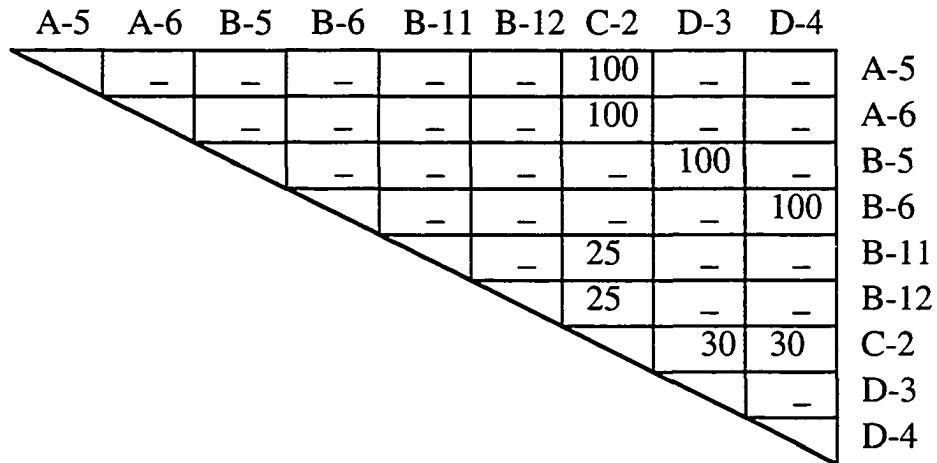


Table 7.23  
Proximity Weights in PTF-44-59

CSPA is run first to determine SPPs of resources that satisfy the default out-zone and hard constraints of Table 7.22. PTFLCA is run next and positions the stationary

resources C-9, C-10, and C-11 at the positions found by CSPA (shown in the fourth column of Table 7.24).

#	L x W	$\Delta VFL /100$	SPPs that Minimize $\Delta VFL$	Single Position [X, Y, Orientation]
C-10	90 x 3	—	$\{ \{ 0 \{ ([70 70] [14.5 14.5]) \} \} \{ 90 \{ \} \} \}$	[70, 14.5, 0]
C-11	20 x 3	—	$\{ \{ 0 \{ ([15 15] [14.5 14.5]) \} \} \{ 90 \{ \} \} \}$	[15, 14.5, 0]
C-9	95 x 3	—	$\{ \{ 0 \{ \} \} \{ 90 \{ ([116.5 116.5] [63.5 63.5]) \} \} \}$	[116.5, 63.5, 90]
B-6	9 x 6	$15( X - 60  +  Y - 16 )$	$\{ \{ 0 \{ ([60 60] [7 7]) \} \} \{ 90 \{ \} \} \}$	[60, 7, 0]
B-5	9 x 6	$15( X - 115  +  Y - 63.5 )$	$\{ \{ 0 \{ ([115 115] [63.5 63.5]) \} \} \{ 90 \{ \} \} \}$	[115, 63.5, 0]
C-2	7 x 4	$7.5( X - 68.5  +  Y - 8 ) + 4.5( X - 60  +  Y - 16 ) + 4.5( X - 115  +  Y - 63.5 )$	$\{ \{ 0 \{ ([68.5 68.5] [8 8]) \} \} \{ 90 \{ \} \} \}$	[68.5, 8, 0]
A-6	13.2 x 7.7	$15( X - 68.5  +  Y - 8 )$	$\{ \{ 0 \{ ([78.6 78.6] [6.1 6.1]) \} \} \{ 90 \{ \} \} \}$	[78.6, 6.1, 0]
A-5	12.8 x 7.5	$15( X - 68.5  +  Y - 8 )$	$\{ \{ 0 \{ ([49.1 49.1] [6.2 6.2]) \} \} \{ 90 \{ \} \} \}$	[49.1, 6.2, 0]
B-12	4 x 4	$3.75( X - 68.5  +  Y - 8 )$	$\{ \{ 0 \{ ([68.5 68.5] [4 4]) \} \} \{ 90 \{ ([68.5 68.5] [4 4]) \} \} \}$	[68.5, 4, 90]
B-11	4 x 4	$3.75( X - 68.5  +  Y - 8 )$	$\{ \{ 0 \{ ([64.5 64.5] [2 2]) \} \} \{ 90 \{ ([64.5 64.5] [2 2]) \} \} \}$	[64.5, 2, 0]

Table 7.24  
Solution Steps in the Layout Construction of PTF-44-59

Next, the relocatable resources are positioned. B-6 is positioned first at a central location adjacent to D-10 to minimize the distance between it and the centroid of D-4. The optimal position of B-5 that minimizes transportation costs with D-3 is the position at the south-east corner of the parking structure (see Fig. 7.13).

C-2 remains at its previous position in PTF-25-34 because relocating it is not cost-effective. A-6, A-5, B-12, and B-11 are selected in this order and positioned as detailed in Table 7.24 and shown in Fig. 7.13. Note the position of A-5 (sand) far from A-6 (sand). This is so because at this position A-5 is closer to C-2, i.e., when measured in centroid-to-centroid distance, than if it were adjacent to A-6.

The final layout of PTF-44-59 has a VFL value of 206,475 and is shown in Fig. 7.13 below.

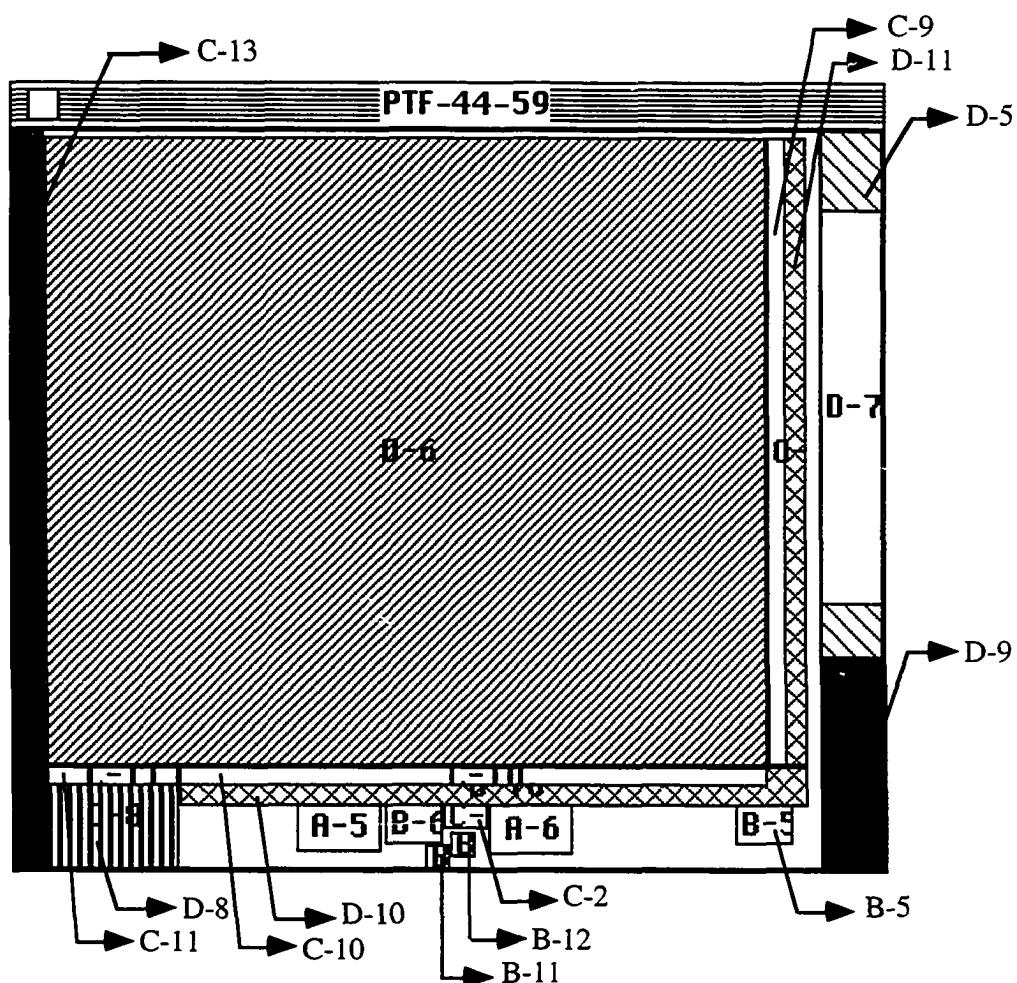


Figure 7.13  
Final Layout of PTF-44-59

#### 7.4.8 Layout Construction of PTF-59-61

In this PTF, activity-6 continues with no other activity scheduled to start. The resources that need positioning in this time frame are the relocatable resources from the previous layout: A-6, B-6, B-12, and C-2.

No additional hard constraints are defined for this PTF. The proximity weights (see Table 7.25) are the same as those defined for PTF-44-59.

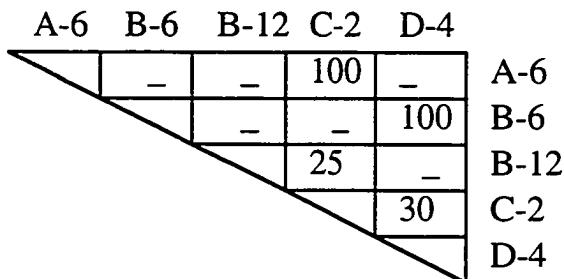


Table 7.25  
Proximity Weights in PTF-59-61

The stationary resources C-10 and C-11 remain at their positions. C-9 is no longer needed and is therefore removed from the layout. B-6 remains at its previous position because this position minimizes the distance with D-4 (see  $\Delta VFL$  in Table 7.26). For the same reason, C-2 also remains at its previous position.

#	L x W	$\Delta VFL /100$	SPPs that Minimize $\Delta VFL$	Single Position [X, Y, Orientation]
B-6	9 x 6	$10( X - 60  +  Y - 7 ) + 2( X - 60  +  Y - 16 )$	{ {0 {[60 60] [7 7]}}, {90 {}} }	[60, 7, 0]
C-2	7 x 4	$7.5( X - 68.5  +  Y - 8 ) + 6( X - 60  +  Y - 16 )$	{ {0 {[68.5 68.5] [8 8]}}, {90 {}} }	[68.5, 8, 0]
A-6	4.5 x 2.6	$5( X - 78.6  +  Y - 6.1 ) + 2( X - 68.5  +  Y - 8 )$	{ {0 {[78.6 78.6] [6.1 6.1]}}, {90 {}} }	[78.6, 6.1, 0]
B-12	4 x 4	$5( X - 68.5  +  Y - 4 ) + 0.5( X - 68.5  +  Y - 8 )$	{ {0 {[68.5 68.5] [4 4]}}, {90 {[68.5 68.5] [4 4]}} } }	[68.5, 4, 90]

Table 7.26  
Solution Steps in the Layout Construction of PTF-59-61

A major part of A-6 was consumed during PTF-44-59, thus reducing its dimensions in PTF-59-61 to 4.5 x 2.6. A-6 shrunk towards its centerpoint as shown in Fig. 7.y. PTFLCA decides to keep A-6 at its previous position because transportation costs are weighed over a short time period (i.e., the duration of PTF-59-61) and, therefore, do not outweigh the higher cost of relocating it (see  $\Delta$ VFL of A-6 in Table 7.26). Last, B-12 is positioned at its previous position.

- The final layout of PTF-59-61 has a VFL value of 5,380 and is shown in Fig. 7.14.

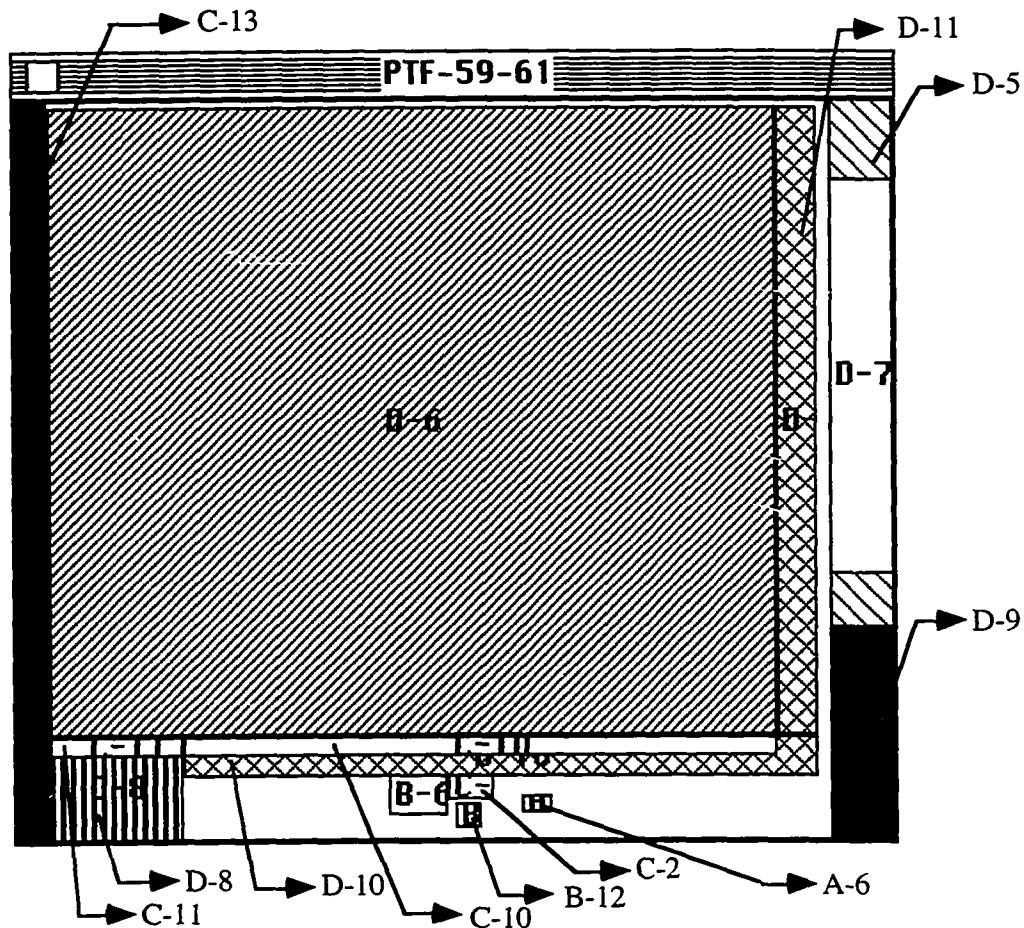


Figure 7.14  
Final Layout of PTF-59-61

PTF-59-61 is the last PTF in the sequence and its layout completes the dynamic layout construction for the masonry phase of this project.

## **7.5 Summary and Validation of Results**

MoveSchedule successfully constructed a feasible sequence of layouts for the masonry phase of this project and changed the user-defined construction method of the north and west concrete block walls and that of the north brick facade so that the overall schedule complies with site space conditions. The sequence of layouts and the final schedule were shown to the former project engineer on this project, Dr. John G. Everett and were evaluated according to: 1) model flexibility, 2) schedule feasibility, and 3) quality of the dynamic layout.

### **7.5.1 Model Flexibility**

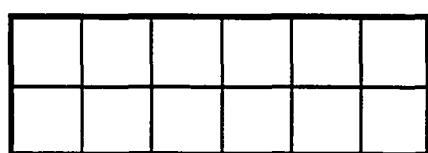
- **Area Profiles Adequately Modeled the Resources' Area Requirements Over Time**

The four area profiles, though conservative, adequately modeled the area required over time by this project's resources. Profile-A modeled changes in the area required over time by consumable and bulk resources, such as sand. MoveSchedule showed piles of sand occupying less space from one PTF to the next, thus releasing space that could be used by other resources. Profile-B modeled the area required by concrete blocks, bricks, and masonry cement over time. This profile is adequate for modeling pallets of material that occupy a constant space, even if the material gets consumed as the activity progresses. Profile-C modeled shared resources, like the mortar mixer, and other dependent resources with fixed and known dimensions, like the scaffolding. Profile-C was also exploited to block out space on site for schedule-dependent phases of construction, like the neighbor's driveway. Profile-D cannot be used for this purpose because it requires that the time period during which the resource exists on site be defined a priori, when defining the resource, instead of at runtime and depending on construction progress. Profile-D modeled the area required by permanent facilities and other restricted areas on site.

- **Model Can Compensate for the Drawback of a Fixed L/W ratio**

Profile-B was not flexible, however, when it came to determining the dimensions of pallets of materials for different area requirements. This is so because the smallest unit of the resource (in this case the pallet) has fixed dimensions and shape, and a single L/W ratio is used to determine the resource dimensions for any area requirement (i.e., any number of

pallets). Fig. 7.15 illustrates this point. Fig. 7.15 (a) and (b) show two alternative arrangements of twelve 4 x 4 pallets for different L/W ratios. The match between the arrangement of individual pallets and the rectangle calculated from the specified area and L/W ratio is perfect. However, this match cannot be perfect for 10 pallets, as is shown in Fig. 7.15 (c) and (d). Ten pallets can only be arranged in a L/W ratio of 10/1 or 5/2 (see Fig. 7.15 (e) and (f)). Hence, when the area required by a Profile-B resource varies with the resource level and a single L/W ratio is used, some areas describe an infeasible arrangement of pallets.



(a)

$$A = 12 \times 16 = 192$$

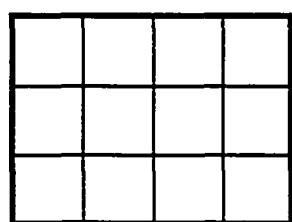
$$L/W = 3$$



(c)

$$A = 10 \times 16 = 160$$

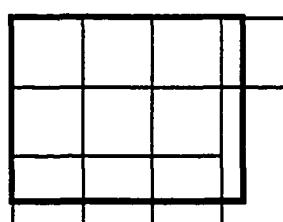
$$L/W = 3$$



(b)

$$A = 12 \times 16 = 192$$

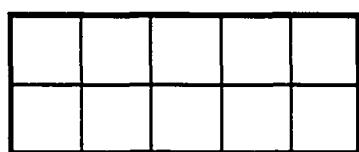
$$L/W = 1.3$$



(d)

$$A = 10 \times 16 = 160$$

$$L/W = 1.3$$



(e)

$$A = 10 \times 16 = 160$$

$$L/W = 2.5$$



(f)

$$A = 10 \times 16 = 160$$

$$L/W = 1$$

Figure 7.15  
Arrangements of Pallets for Different Areas and L/W Ratios

This problem can be avoided using the current model by splitting up a resource into smaller components or modeling it as a Profile-C, or by extending the capabilities of the current model.

#### ◊ **Splitting up Resource into Smaller Components of Fixed Shape**

Modeling each pallet as an individual resource of fixed shape may be advantageous in that it would allow MoveSchedule to come up with L-shaped or other odd-shaped arrangements of pallets, grouped together or spread around the site, as a result of positioning each pallet individually. MoveSchedule will also be able to fit individual pallets in smaller, empty spaces which might not be able to accommodate the entire group of pallets. A disadvantage to breaking down a resource into smaller components is that a larger number of resources will have to be positioned. Another disadvantage is that modeling the decline in area required over the activity duration would not be possible as components have a fixed shape.

#### ◊ **Using Profile-C to Model Resource**

Profile-C can conservatively model the space required by bricks and blocks. This profile requires that the length and width of the resource be explicitly defined and, therefore, solves the problem of defining a single L/W. However, this profile assumes a fixed area requirement throughout the duration of an activity and it does not model the effect that changes in productivity have on the consumption rate of the resource, and, consequently, on its area requirement.

#### ◊ **Extending the Definition of Profile-A and Profile-B**

A more elegant way of solving this problem is to augment the definitions of Profile-A and Profile-B in the current model with information regarding the size of the smallest component or handling unit so that resource dimensions could be computed as matching multiples of the handling unit's size and not as a function of a fixed L/W. Also Profile-A and -B could be modified to accept more than one L/W per resource. For example, the L/W of a Profile-B resource could be defined along with the area requirement of the resource at each resource level. For a Profile-A resource, pairs of compatible L/W ratios and areas could be defined as additional information to characterize the resource.

Clearly, this is a research direction worth pursuing because it can also benefit PTFLCA. Indeed, by extending Profile-A and -B to use ranges or sets of acceptable L/W ratios, PTFLCA can be improved to reason about how to vary L/W ratios in order to solve spatial conflicts or improve the layout solution.

- **Resource Levels Adequately Modeled Alternative Construction Methods**

The resource levels provided the flexibility needed for modeling alternative construction methods and space requirements of activities. Two resource levels described alternative construction methods for the north and west walls to illustrate this capability of MoveSchedule. It would have been possible to define more than two levels per activity, but doing so would have only increased computations and the number of changes made to the schedule and not necessarily have led to different results. For example, assume that activity-1 had an additional maximum resource level that describes the resources needed to perform it with conventional construction methods and with double the crew size. The scaffolding C-3 would still be needed at this level along with the concrete blocks, sand, cement, and the mortar mixer. The activity would be initially scheduled to be performed at its shortest duration, i.e., at this maximum level. Consequently the initial unconstrained project duration would be shorter than 43 days (i.e., the project duration of the unconstrained schedule). The layout construction of the first PTF would have resulted in a spatial conflict as a result of positioning C-3. SCR could solve this conflict by lowering activity-1's resource level to the normal level bringing problem solving to the same point from which we started problem solving in section 7.4.

### **7.5.2 Schedule Feasibility**

- **MoveSchedule Successfully Changed the Schedule to Meet Site Space Constraints**

MoveSchedule started with an unconstrained schedule and a project duration of 43 days. It adequately changed the construction methods for the north and west walls to solve the spatial conflicts encountered in the course of dynamic layout construction. The project duration of the final schedule generated by MoveSchedule is 61 days which corresponds, closely, to the length of time it took the masonry subcontractor to finish masonry construction (approximately 12 weeks or 60 days). The masonry subcontractor had

estimated that the site constraints added approximately 3 weeks (15 days) to what normally would have been a 9-week job (45 days).

- **SCR Fortuitously Solved Spatial Conflicts**

When a spatial conflict was detected, SCR selected the strategy-activity combination that lowers the total area requirement over the given problematic time frame at a minimum increase in project duration. While SCR adequately solved all spatial conflicts encountered in this project, it was fortuitous that it was able to solve a potential conflict in PTF-0-13. The selected strategy-activity did not solve the cause of the conflict (finding a feasible SPP for the scaffolding C-3) but did, however, solve another potential conflict (finding a feasible position for the scaffolding C-5). As a result, SCR was called twice to solve the same conflict in PTF-0-13 and PTF-0-15.

- ◊ **User Can Label Cause of Conflict**

Mode 2 of SCR enables the user to feed the cause of the spatial conflict to SCR by explicitly labeling a resource for removal. SCR can then use this information to select a strategy-activity that removes the labeled resource from the problematic time frame. For example, in the case at hand, the user can label C-3 (which caused the spatial conflict in PTF-0-13) for removal which will then cause SCR to select a strategy-activity combination that will remove C-3 from PTF-0-13 at a minimum increase in project duration.

### **7.5.3 Quality of the Dynamic Layout**

It was not possible to compare the dynamic layout generated by MoveSchedule to the planned or actual site layout of this project because there are no records of the actual site layout and how it evolved over time. Hence, MoveSchedule's solution has been assessed only qualitatively by the project engineer.

Overall, PTFLCA succeeded in constructing a feasible and functional sequence of layouts for the masonry phase of construction.

- **Static Resources Model Access and Pathways of Other Mobile Resources**

D-9 and D-10 block a 6 ft-wide area that can be used only by personnel and the prime mover to access the parking structure and the work areas adjacent to the south and east

walls. In this case, D-9 and D-10 also happen to provide access to the materials laydown areas and the mortar mixing area which are distributed along D-10. Normally, planning such access depends on the materials handling means used, which can be the prime mover, laborers or other hand-operated lifting tools, and involves planning trajectories along which these means travel.

- **Size and Hard Constraints Restrict Possible Positions of Stationary Resources**

Hard constraints restricted the possible positions of stationary resources, namely the office trailer and scaffolding. PTFLCA positioned the office trailer along Elliot street within the metered parking area where it was actually located for the duration of this project. PTFLCA positioned all scaffolding at the right position with respect to each wall and kept them at these positions during the construction of their corresponding walls.

- **Positions Minimize Transportation Costs of Materials**

The positions of all other relocatable resources, namely construction materials, were determined by PTFLCA for each PTF so as to minimize transportation costs.

PTFLCA positioned concrete blocks and bricks as close as possible to the access D-8 when serving the north and south walls and as close as possible to the south and east walls when serving those. In this process, PTFLCA efficiently reused space and avoided the relocation of resources by positioning first those that had been positioned in previous PTFs.

PTFLCA positioned the mortar mixer to minimize transportation cost of mortar to the different work areas, then positioned sand and cement bags around it to minimize the cost of transporting them to the mortar mixer. The mortar mixer is positioned towards the center of the south wall and remains at this position for the duration of masonry construction. In reality, the mortar mixing area, which grouped the mortar mixer, sand, and cement bags, was located at the south-east corner of the site for the duration of masonry construction. The project engineer explained that this location was chosen because the south and east corners of the structure curve-connect thus making the available site space at that location wider than any other place on site. This width is lost in MoveSchedule because all objects are modeled as rectangles. The position determined by PTFLCA for the mortar mixer is closer to the access of the structure, the west wall, and the

south wall, where 80 % of the mortar goes, than the south-east corner, which makes PTFLCA's solution better if transportation costs should be minimized.

- **Hard Constraints Compensate for the Bias for Central Locations**

PTFLCA minimizes the distance between the centroids of any two resources to minimize transportation costs between the two which results in a bias for central locations. For example in PTF-44-59, PTFLCA positioned the bricks B-5 halfway along the south wall D-4 to minimize the cost of transporting bricks to this wall (see Fig. 7.13). While this position is optimal because it minimizes the distance between the centroids of the two resources, it is not the only position that minimizes the actual distance traveled along the wall in delivering bricks. Indeed, because of symmetry and because D-4 is a "long" resource and bricks are delivered to all sections of this wall, any position along the wall is equally good and would result in the same overall distance traveled. In this particular situation it is possible to use adjacency constraints instead of proximity weights to model interactions between the bricks and the wall and, thus, avoid the bias for central locations.

- **Hard Constraints Restrict Positions While Proximity Weights Express Preferences**

Hard constraints, however, may not always solve the problem of bias for central locations. For example, in PTF-44-59, the optimal position of the bricks B-5 that minimizes transportation costs with the east wall D-3 is the position at the south-east corner of the parking structure (see Fig. 7.13). Here again, there would be a bias for a central location, except that another resource D-11 occupies the position where B-5 would be closest to D-3. PTFLCA therefore found the next best position. Had adjacency constraints been used instead of proximity weights to model the interaction between B-5 and D-3, PTFLCA would have found no feasible position for B-5 and consequently no solution for this PTF. This illustrates the fact that using hard constraints may unduly constrain the problem while using proximity weights only expresses preferences.

- **Layouts by MoveSchedule Can be Customized Using an Interactive Graphical Layout Interface**

In most PTFs, PTFLCA reused the site space around the mortar mixer to position sand and masonry cement as close as possible to the mixer (see Fig. 7.7, 7.8, and 7.11). In PTF-44-59 (see Fig. 7.13), however, sand A-5 is positioned on the other side of bricks B-6 and

away from the mortar mixer. This position apparently minimizes the distance between the sand and the mortar mixer.

For practical reasons, however, a human planner might have positioned A-5 next to the other sand A-6 to logically group them together. It would be very hard to encode such knowledge in a system like MoveSchedule. This need is better addressed by an interactive graphical interface (such as MovePlan) that allows a human planner to custom-tailor the layout solutions generated by MoveSchedule.

To this end, the layouts generated by MoveSchedule can be saved as an ASCII file that contains information regarding PTFs, resources, and their positions as determined by PTFLCA. MovePlan, for instance, can read this ASCII file and display the layouts of individual PTFs where all stationary and static resources are displayed as fixed resources, and all others as relocatable whose positions can be edited by the user.

- **Fallacy in Determining Relocation Cost of a Profile-A Resource**

PTFLCA relocated resources only when it was cost-effective to do so. In PTF-25-34 and PTF-34-44, however, PTFLCA may have overestimated the relocation cost of sand (i.e., resources A-3 and A-4 respectively). This is so because PTFLCA measures relocation costs based on the location of the centroid of the resource without considering the overall space that the resource occupies in consecutive time frames.

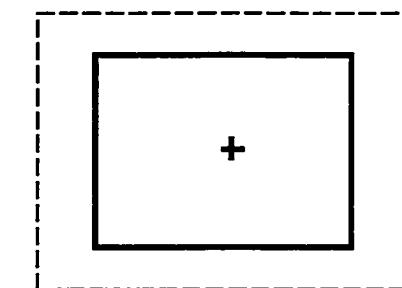
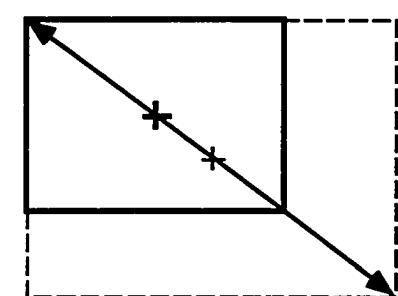
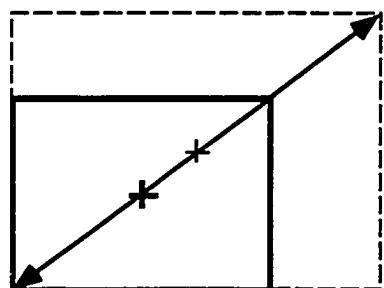
For example, PTFLCA added the cost of relocating A-3 to the layout costs because A-3's centroid has moved in the layouts of PTF-22-25 and PTF-25-34. The centroid of A-3 has moved because its dimensions have changed as a result of it being consumed in the pattern shown in Fig. 7.16 (a). However, A-3 in PTF-25-34 remains fully contained within the boundaries of the space it occupied in the layout of PTF-22-25 and thus has not been physically relocated.

PTFLCA can be improved so that no relocation costs are added when the resource stays within the area it occupied in previous time frames.

- **Complexities in Estimating Actual Relocation Costs**

In many instances PTFLCA could not justify the relocation of resources because their unit relocation costs were relatively high. Unit relocation costs can be estimated based on the relative ease of relocating the resource. Determining exact unit relocation costs, however,

is difficult as it does not only depend on the nature of the resource, but also on the means used for relocation. The choice of an appropriate handling mean depends on the quantity and the distance over which the resource is to be relocated. For example, sand can be relocated using a wheel barrow and a laborer if the quantity to be moved is small and the distance is short; otherwise, a front end loader is more appropriate and can move the sand much faster. In each case, the unit relocation cost is different and depends on the rental cost of the handling mean.



**(a) Relocation cost added**

**(b) No relocation cost added**

**Legend:**



Space that resource occupied in earlier layout



Space that resource requires in current layout



Centroid of the rectangle representing the resource.

**Figure 7.16**  
Effect of Profile-A Consumption on Relocation Costs

# Chapter 8

## CONCLUSIONS

### 8.1 Contributions to Knowledge

This research has successfully developed a heuristic approach for constructing efficient dynamic site layouts and adjusting the schedule of construction activities to comply with site space limitations. By meeting the objectives presented in section 1.3, this research contributes to knowledge in the following ways:

#### **1 Built a model with space and time as explicit variables**

This research characterized the space needs, as they vary over time, of individual resources on a construction site. It identified five basic groups where the space need of a resource is either dependent on construction progress, in which case it is decreasing, increasing, seemingly constant, or constant over time, or independent of construction progress.

Accordingly, this research developed a model that uses four area profiles that conservatively approximate the space needs of construction resources in those five basic groups. That is, the group of resources whose space needs increase with construction progress is not directly modeled but can be approximated by two of the proposed area

profiles as discussed in section 5.8.1 of Chapter 5. The model allows layout objects to be of fixed or variable shape. The shape of a layout object is defined by its length (L) and width (W) when its area requirement does not vary over time. It is computed using a fixed L/W ratio when its area requirement varies over time.

The model characterizes activities in a schedule by different resource levels. Each level describes a combination of resources involved in performing the activity with a specific construction method, crew-size, or delivery schedule (see second example in Chapter 6). By explicitly articulating alternative resource levels to include area profiles of the resources involved in them, a time-space relationship for each activity in the schedule was established. This made it possible to trade off activity durations against space needs but not costs.

The presented model differs from existing space scheduling models in two ways. First, the space requirements of activities are determined from those of individual resources at a given time and for a given activity duration. Information at this level of detail is desired to construct the layout of individual resources on site and determine the layout's feasibility. Second, the model does not require sophisticated input derived from CAD systems or the use of dedicated graphics workstations. It relies on the engineer's or layout designer's experience and knowledge of construction resources to estimate parameters such as the L/W ratio and maximum or average area requirement of resources.

## **2 Developed interactive computational methods for dynamic layout planning and space scheduling**

As part of this research, two independent and interactive computer programs, MovePlan and MovePlan-ConRes, were developed. These programs helped elicit the variables and decisions involved in developing dynamic layouts and space-feasible schedules.

The MovePlan program supports users in the construction of dynamic layouts by providing—among other graphical displays—a display of the site space and the resources needed on site over user-selected time frames. The user would then position or relocate these templates on the site space until a satisfactory layout is reached. MovePlan's contributions are tying layout data to an activity schedule and providing mechanisms to help a user construct consistent dynamic layouts.

The MovePlan-ConRes program allows the MovePlan user to connect to a reactive scheduler that solves spatial conflicts as he/she identifies them during dynamic layout construction. ConRes applies strategies such as delaying activities or decreasing the number of resources to vary the total space need in a specific time frame.

The model underlying MovePlan and ConRes is simplified and uses a constant area profile and fixed dimensions for representing the space needs of resources over time. This representation is extended in MoveSchedule to include four different resource area profiles and three resource levels per activity. Adjustments in the methods and strategies used to resolve spatial conflicts were needed as a result of this extension. The approach in MovePlan and ConRes is interactive and relies on the user for positioning resources and assessing the quality of the layouts. A fully automated algorithmic approach is used in MoveSchedule.

### **3 Developed a heuristic dynamic layout construction algorithm**

The algorithm (PTFLCA) seeks heuristically suboptimal solutions to a constrained dynamic layout problem. PTFLCA uses both 2-dimensional spatial geometric constraints on relative positions of resources, and proximity and relocation weights to model interactions between resources. This approach is the most comprehensive approach for modeling the construction site layout problem considered to date as existing models typically include only one of these two types of interactions.

The objective function (VFL) assesses the quality of an individual layout and of the sequence layouts based on transportation costs of resources between staging and fabrication or installation areas and relocation costs of resources in between staging areas on site.

PTFLCA considers changing sizes of resources over time and relocation of resources. It integrates two widely-used problem solving methods, Constraint Satisfaction and Linear Programming, into a single algorithm for constructing the dynamic layout. Constraint Satisfaction is commonly used in AI to solve large constrained problems. It is used here to satisfy 2-dimensional geometric constraints between resources so that infeasibilities are detected before the positioning of resources is started. The Constraint Satisfaction approach was possible thanks to the use of the bounded interval representation, which keeps track of all possible positions of a resource by storing these positions as a set of convex sub-regions. In turn, the convex sub-regions were used to determine optimal

positions that minimize transportation and relocation costs of a resource using a Linear Programming formulation.

Although PTFLCA performs all computations using a numerical representation, a graphical output was needed to illustrate the final solution and communicate it to system users and industry practitioners. To this end, a stand-alone graphical program was developed to enable the user to visually check SPPs and single positions of resources.

#### **4 Developed a heuristic space scheduling algorithm**

This research developed a space scheduling algorithm that improves an initial schedule to comply with spatial constraints on acceptable positions of resources. The algorithm starts with the shortest project duration and applies strategies that modify start dates or durations of activities to lower the total area requirement over problematic time frames. Strategies are applied one at a time until all spatial conflicts are solved in a given problematic time frame. At each time, the strategy that causes the smallest increase in project duration is applied. Different heuristics are used to break ties among strategies. As a result of applying a strategy, the layout for the given time frame is reconstructed to determine its feasibility and, consequently, to determine the feasibility of the schedule in this time frame.

Three main strategies are used to change the total demand for space over a specific time period. These strategies are to delay or slow down an activity, or to store a selected idle resource temporarily off-site if such a storage space is available. The effect of each strategy on the overall schedule depends on the selected activity or resource.

Strategies are prioritized, so that the strategy/activity combination that lowers the total area requirement over a given time frame at the least increase in project duration is selected. This selection criterion ignores the cause of conflict. To allow a user to bring some insight into the process, the strategy selection can be overridden. The user can stop the space scheduling algorithm or decide on which resource to remove from the layout. That resource is then removed from the layout at the least increase in project duration.

#### **5 Tested MoveSchedule with realistic project data to assess its capabilities**

MoveSchedule is the implementation of the model, including the time-space tradeoff strategies, the layout construction algorithm and the space scheduling algorithm.

MoveSchedule was tested with realistic project data to assess the capabilities and limitations of the model and to validate the solutions generated by the space scheduling algorithm.

Chapter 6 reported on two example applications of MoveSchedule. The first example tested the application of PTFLCA for optimally solving MINISUM location problems with constraints on acceptable positions of the new facility. This example was chosen because it is a single facility (resource) location problem with non-overlap constraints and flow between the new facility and existing facilities. PTFLCA elegantly and accurately found the optimal solution for the problem by using proximity weights and geometric constraints that limit the SPP of the new facility.

The second example showed how alternative delivery scenarios can be modeled in MoveSchedule and how VFL can evaluate different laydown-delivery alternatives. There is more work to be done however before the space scheduling algorithm can choose between alternative delivery scenarios ('Time-space-cost tradeoffs' in section 8.3 of this chapter elaborates on the issues to be considered).

Chapter 7 reported on the validation of the space scheduling algorithm. A real project was selected where the site space availability constrained the choice of construction methods and the project duration. MoveSchedule used the unconstrained schedule to sequentially construct the layout of resources on site for each PTF. The schedule was adjusted by applying one space-time tradeoff strategy at a time during layout construction to lower the total area requirements over PTFs for which no layout could be constructed. MoveSchedule was able to construct a feasible dynamic layout. The end result was an adjusted schedule that can be executed within the existing site space restrictions.

MoveSchedule's solution was qualitatively assessed by a former project engineer on this project. The end schedule is very close to how long it actually took the contractor to complete this phase of construction. The layouts of the individual PTFs were satisfactory and in most cases optimal in terms of minimizing relocation and transportation costs of materials. The solution lacked process-level details such as access and intermediate positions of the mobile lifting equipment.

The example applications and the validation problem revealed some limitations of the model and methods which, in turn, suggested a number of possibilities for future research.

## 8.2 Capabilities and Limitations of MoveSchedule

The last sections of Chapters 4, 5, and 7 listed the capabilities and limitations of the methods and modeling assumptions presented in each chapter. This section summarizes them with respect to: 1) the problem solving method, 2) the layout objects representation, and 3) the modeled variables.

### 1 Problem solving method

PTFLCA is a heuristic layout construction algorithm. A heuristic approach was to be expected because existing formulations of unconstrained static layout construction problems with fixed-shape objects are NP complete. PTFLCA uses an early commitment approach for constructing the layout of a PTF while taking into account previously constructed PTFs. The solution it generates is path-dependent (i.e., it depends on the order in which the resources are selected for layout). A random factor is introduced to get out of local optima and avoid dead ends. Even with randomness, PTFLCA may not find a solution if one exists, but its probability of success is higher than without randomness.

PTFLCA does not have a mechanism that allows it to backtrack to partial layouts and pursue a different path. It could most benefit from a backtracking mechanism to undo previous assignments of resources to positions, or from a control mechanism to vary the order according to which resources are selected to enter the layout.

The dynamic layout construction strategy considers the layouts of PTFs sequentially and paralleling construction time. This first-come-first-served strategy may not be the best as layouts of earlier PTFs may unduly constrain those of subsequent PTFs and layouts of earlier PTFs cannot be revised. Augmenting this strategy and PTFLCA with backtracking capabilities is definitely a research direction worth pursuing. Research in this direction involves developing mechanisms to identify sources of conflicts, select which commitments to undo, and determine how to proceed from a partial solution onward.

A limited set of time-space tradeoff strategies were implemented in the system; these are by no means exhaustive and a combination of these strategies as well as others should be investigated. Additional strategies are discussed in section 5.8.2 of Chapter 5.

SCR selects and applies a strategy irrespective of the source of conflict. If the conflict persists, SCR is called again and another strategy/activity is selected and applied.

In this iterative process, SCR does not undo previous actions because it cannot judge whether such actions were effective or not. SCR needs more flexibility to reason about sources of conflicts so that it can intelligently select or undo actions. For example, if SCR can determine that the cause of the first spatial conflict in the Harvard Square Parking Garage is 'not being able to position the scaffolding on the exterior of the wall', then the system can select strategies that will eliminate the exterior-type scaffolding from the problematic time frame and choose to perform the corresponding activity with some other type of scaffolding.

## 2 Layout object representation

In MoveSchedule, the area of a resource is fixed at any given time and its shape ratio is always constant. Some resources, like bulk and stackable materials, can be arranged to fit in more than one area or shape ratio depending on space availability. Section 7.5.1 in Chapter 7 discusses the possibility of modeling the space requirements of these resources using ranges of acceptable areas and shape ratios. The problem solving method in this case would need to be adjusted so that it can reason about ranges of shape ratios and area requirements. For example, PTFLCA might determine first the optimal location of a dimensionless resource by solving the linear program, and then try to fit the resource at that location by varying its shape ratio and area within an acceptable range. The inverse approach is implemented now.

Furthermore, limiting the shape of resources to rectangles and their orientations to  $0^\circ$  and  $90^\circ$  limits the solution space to a subset which may exclude the optimal solution. Since, in this research, feasible—and not necessarily optimal—solutions were considered satisfactory, this issue was not a concern. It would have been a concern had the bounded interval representation reduced the solution space to the null set, that excluded all feasible solutions.

The area profiles modeled in MoveSchedule do not include a profile where the area requirement of a resource increases over time. In section 5.8.1 of Chapter 5, it is suggested that Profile-C or Profile-B, which are constant area profiles, be used to model the space needs of such a resource. Of course, this is only a conservative approximation which assumes that the resource will need an equal amount of space for the length of time it is on site. This approximation was made because the space scheduling algorithm computes the space need of a resource in a layout-e-f based on the area profile of the resource at time e. Including an increasing area profile would accurately represent its space needs. Another

advantage to including such a profile is to complete the basic building blocks required for constructing complex area profiles, for example, a profile that shows a build-up of the resource first during mobilization time, and then a consumption of the resource as the production activity progresses. Building complex area profiles can be done by aggregating or lumping several activities, each using a basic area profile. Research in this direction involves developing mechanisms that keep activities together.

### **3 Modeled variables**

The model includes variables to represent transportation and relocation costs of resources. It ignores, however, many process-level variables, such as the time it takes to rotate or relocate a resource. MoveSchedule may rotate a resource at runtime to make it fit in the available space, therefore no activity can be defined *a priori* to reflect this time. When this time is large and known *a priori*, additional activities could be added to the schedule to reflect the corresponding durations, which in turn may impact the project duration. The cost of relocating a resource could be refined to include the cost of rotating the resource. Currently, the problem solving method penalizes the layout solution only for the relocation of resources, but does not adjust activity durations or start dates to reflect the time it takes to relocate the resource.

## **8.3 Directions for Future Research**

Future research will comprise: 1) working on overcoming the limitations of MoveSchedule, 2) revisiting some fundamental modeling assumptions, and 3) implementing the following extensions to MoveSchedule.

### **1 Reassessing some modeling assumptions**

MoveSchedule makes assumptions regarding resource limits (all resources, except space are available in unlimited quantities), the sharing of resources (resources can be shared by concurrent activities), hard constraints on start and finish dates of activities (there are no constraints on acceptable start and finish dates of activities), and resource use following an early start schedule (dependent resources exist from early start to early finish of corresponding activities). These assumptions necessarily limit the quality of MoveSchedule's solutions. Section 5.8.2 in Chapter 5 discusses the possibility of reassessing these assumptions as future research directions.

## **2 Knowledge bases for resource management**

MoveSchedule does not contain knowledge to differentiate between activities. The system, as it stands now, differentiates between resources based on their area profiles only. Knowledge to classify activities or resources could be used to determine sources of spatial conflicts and intelligently select a strategy, activity, or resource to resolve spatial conflicts. For example, if the system could distinguish procurement activities from production activities, then it could shorten mobilization or demobilization activities as these activities are deemed contributory but not productive. The system would also be able to reason about the dependencies between production and procurement activities so that delaying production activities can delay procurement activities too. The system could also be augmented with knowledge about procurement policies, and not just individual procurement activities to reason about policy changes and their impact on the layout and schedule. For example, it might change the policy from a single delivery of the total quantity of the resource prior to the activity start to regular deliveries of smaller quantities during the performance of the activity to reduce the resource space needs on site.

Knowledge about resources can include attributes such as bulkiness and weight (e.g., pieces of machinery tend to be bulky resources), fragility (e.g., windows or finishing hardware are fragile resources) and the resource type (e.g., materials, equipment, hoisting means, transportation means, etc.). This knowledge can be used to automatically infer some interactions between resources, without requiring the explicit input of interactions and geometric constraints on resource positions as is currently the case. Some of this input can be articulated as generic or domain-specific (e.g., by type of project) knowledge and integrated in the system as inference rules. For example, laydown areas should be located adjacent to access roads. Some existing knowledge-based systems for site layout include such knowledge; they could extend MoveSchedule's capabilities. In addition, this knowledge can be used to plan resource hoisting and transportation means.

## **3 Time-space-cost tradeoffs**

While some layout costs (i.e., resource transportation and relocation costs) are modeled in MoveSchedule, many project and schedule-related costs (like mobilization, demobilization, delivery, and costs of alternate resource levels or methods) are not. Because the main focus of this dissertation was on modeling schedule-layout interactions, the strategies implemented focused on time-space tradeoffs only. By making project and

schedule-related costs explicit in the model, it is possible to extend the existing strategies to pursue time-space-cost tradeoffs.

For example, by changing the resource level of an activity, its direct costs will change as a result of adding, removing, or changing the type of resources used for its performance. The project indirect costs will increase if the project duration is lengthened as a result of this change. When slowing down an activity involves changing procurement policy, procurement costs may vary too. Temporarily storing a resource off-site may not affect the project duration but may add mobilization and demobilization costs (i.e., costs for moving the resource to and from its temporary storage) in addition to the cost of procuring the off-site storage space to the total project cost.

In addition, by explicitly modeling delivery costs, it is possible to extend the current strategies to include a strategy that varies the delivery schedule of selected resources to trade off space and cost. This addition may require extending the current representation of resources in MoveSchedule to include alternative initial laydown positions (i.e., positions where resources are first unloaded on site). It would also require modeling the time it takes to relocate resources between staging areas so that tradeoffs between time and cost are also possible (at present, this time is assumed to be zero and only the relocation distance is taken into account in VFL).

#### **4 Alternate approaches for dynamic layout construction and space scheduling**

MoveSchedule uses a first-come-first-served strategy for constructing the layouts of PTFs. Future research should investigate alternate dynamic layout construction strategies. In a first attempt, section 4.6.1 in Chapter 4 presented a few strategies that rate resources or PTFs according to some criterion (e.g., by criticality or total space needs) and sequence them in that order to construct a dynamic layout. The result could be a sequence or a hierarchy of layouts. The expected benefits and shortcomings of these strategies and how the current methods in MoveSchedule can be extended to accommodate them were also discussed in that section.

MoveSchedule adjusts an initial unconstrained schedule to comply with site space constraints in a forward pass approach. In a similar fashion, section 5.8.2 in Chapter 5 discussed the limitations of this approach and assessed the complexities involved in extending the current system to accommodate different approaches, namely, starting with

different initial schedules (e.g., a late schedule with maximum space needs, a conventional schedule with conventional space needs).

MoveSchedule uses a construction method that creates a dynamic layout and an improvement method that adjusts the schedule accordingly. Another research direction is to augment MoveSchedule with a layout improvement method to improve the quality of the constructed layouts.

## **5 Interactive interface**

Interactive graphics are desired to allow the user to override the system's decisions or recommendations on resource positioning or conflict resolution. The user may also want to conduct what-if analyses at intermediate stages of problem solving.

Because it is very hard to articulate all variables, decisions, and constraints needed to describe the space scheduling problem and to reach quality solutions, MoveSchedule can benefit from a user-friendly interactive interface. With this added feature, MoveSchedule can be improved by learning from user interaction, can be used for data or knowledge acquisition which can be later encoded into the system, and will definitely be of greater use to practitioners in the field.

# **Appendices**

# Appendix A

## LIST OF ACRONYMS AND DEFINITIONS

**CSPA:** Constraint Satisfaction and Propagation Algorithm.

**Mobile resource:** A resource that is continuously moving around the site.

**PTF-i-j:** A primary time frame that starts at time i and finishes at time j. This time frame is the smallest time interval delimited by the arrival or departure of a resource or by the start and finish of an activity.

**PTFLCA:** Primary Time Frame Layout Construction Algorithm.

**Relocatable resource:** A resource that can be relocated on site provided the benefit from its relocation outweighs the relocation cost.

**Resource:** Any physical (e.g., materials, equipment, trailer, tree) or abstract (e.g., laydown area, excavated trenches) entity that requires space on site.

**SCR:** The Spatial Conflict Resolver is one of three modules in MoveSchedule. It is the implementation of the time-space tradeoffs strategies presented in section 5.4 and of the heuristics used for applying these strategies in both the user-interactive and the automated mode.

**SPP<sub>i</sub>:** Set of possible positions of resource i. This set is represented by a disjunction of rectangles at both 0° and 90° orientations of i.

**Static resource:** A resource that has a user-defined fixed position on site.

**Stationary resource:** A resource that should not be relocated once positioned on site.

**VFL:** Value function of the layout. It is equal to the sum of proximity and relocation costs of the sequence of layouts. Equation of VFL is given in section 4.2.4

# **Appendix B**

## **PROJECT DATA FOR HARVARD SQUARE PARKING GARAGE**

The following contract documents of the Harvard Square Parking Garage are courtesy of C.E. Maguire of Providence, RI, which was the architect for this project with Harold Morsilli project manager. The owner was Trinity Trust Co. of Cambridge, MA. The construction manager was Gilbane Building Company of Providence, RI. Robert Zoglio was Project Manager/Superintendent, James Lanza was Project Engineer, and John G. Everett was Office Engineer; all three worked for Gilbane Building Company.

Section B.1 includes architectural renderings and the elevation plans of the garage. Section B.2 includes the floor plans for the first and second level. Section B.3 includes calculations of activity durations and space requirements of resources needed as input to MoveSchedule.

## B.1 Elevation Plans

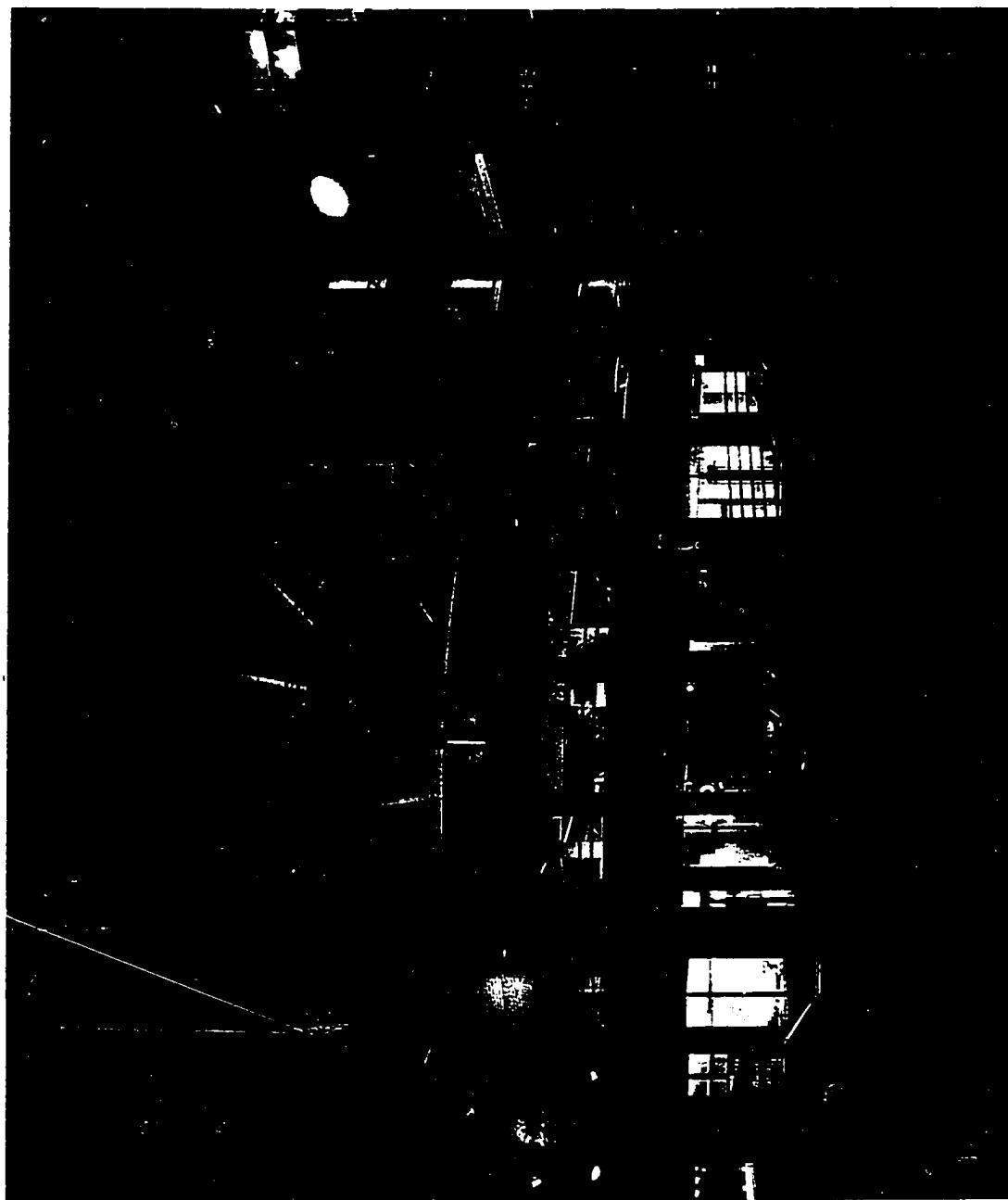


Figure B.1  
East Facade Viewed from J.F. Kennedy Blvd.

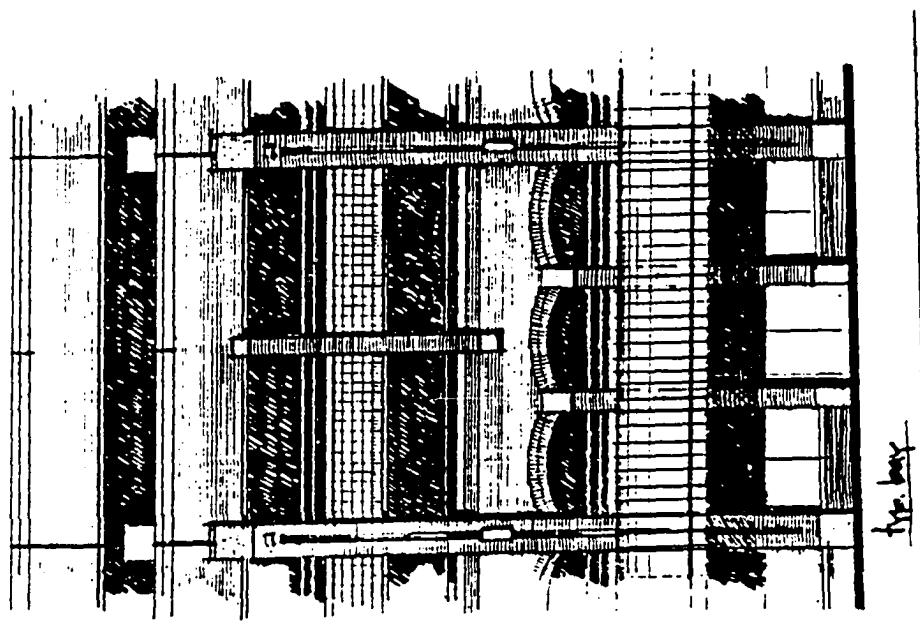
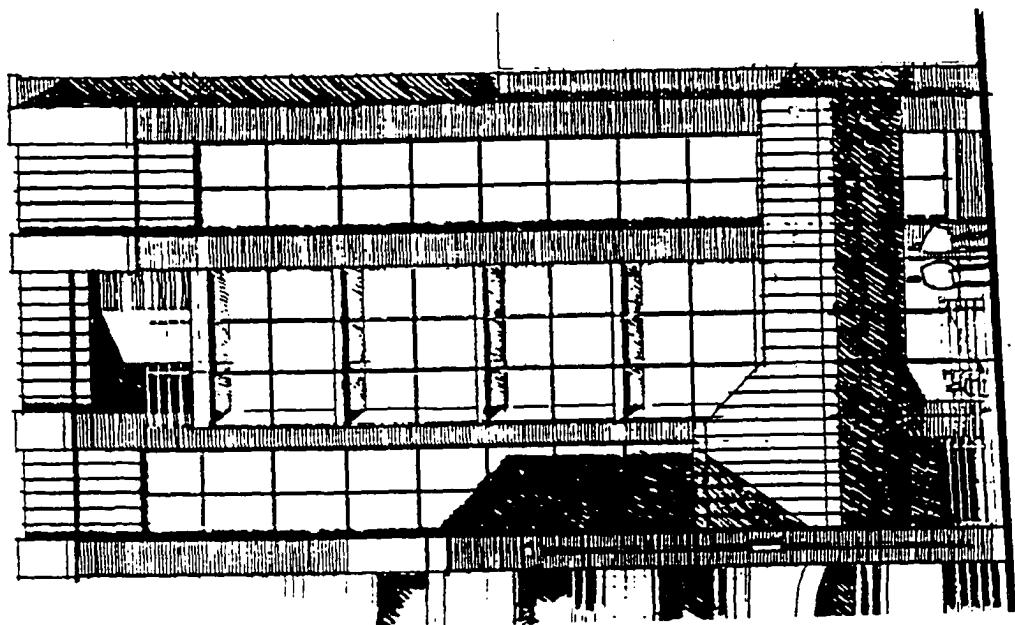


Figure B.2  
Typical South and East Facade

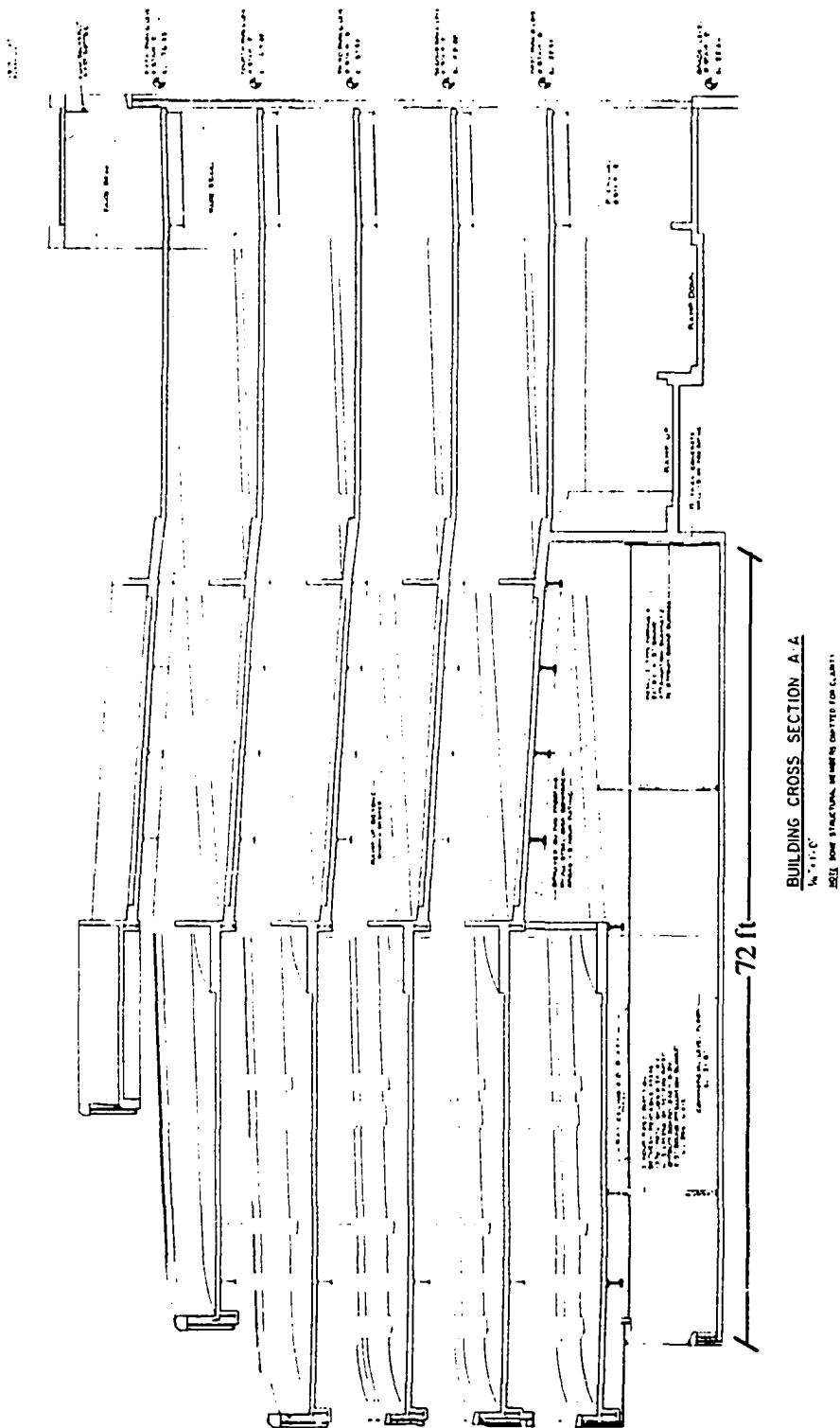


Figure B.3  
Building Cross Section AA  
(Note some structural members are omitted for clarity.)

## B.2 Floor Plans

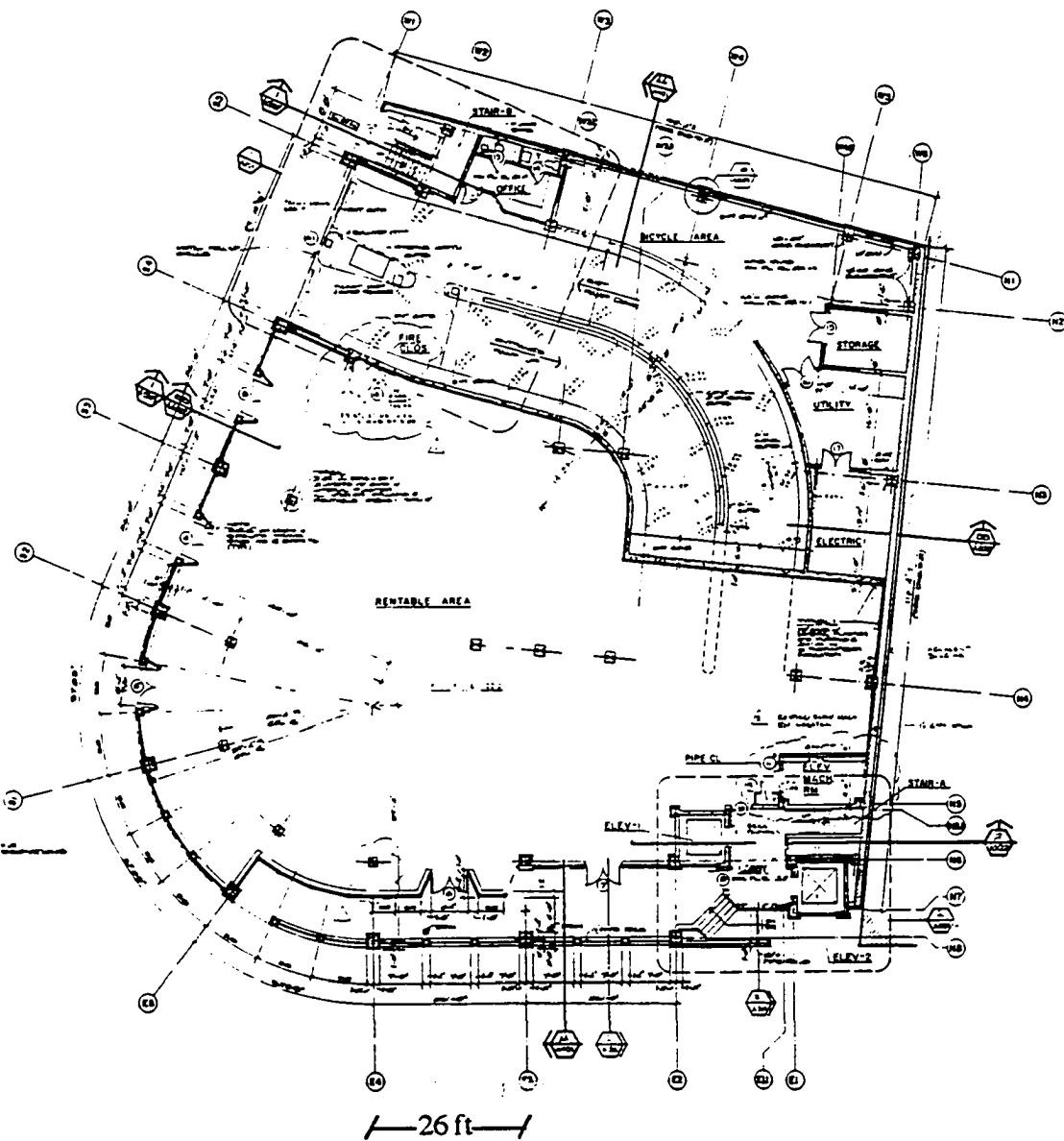


Figure B.4  
Commercial Level Plan

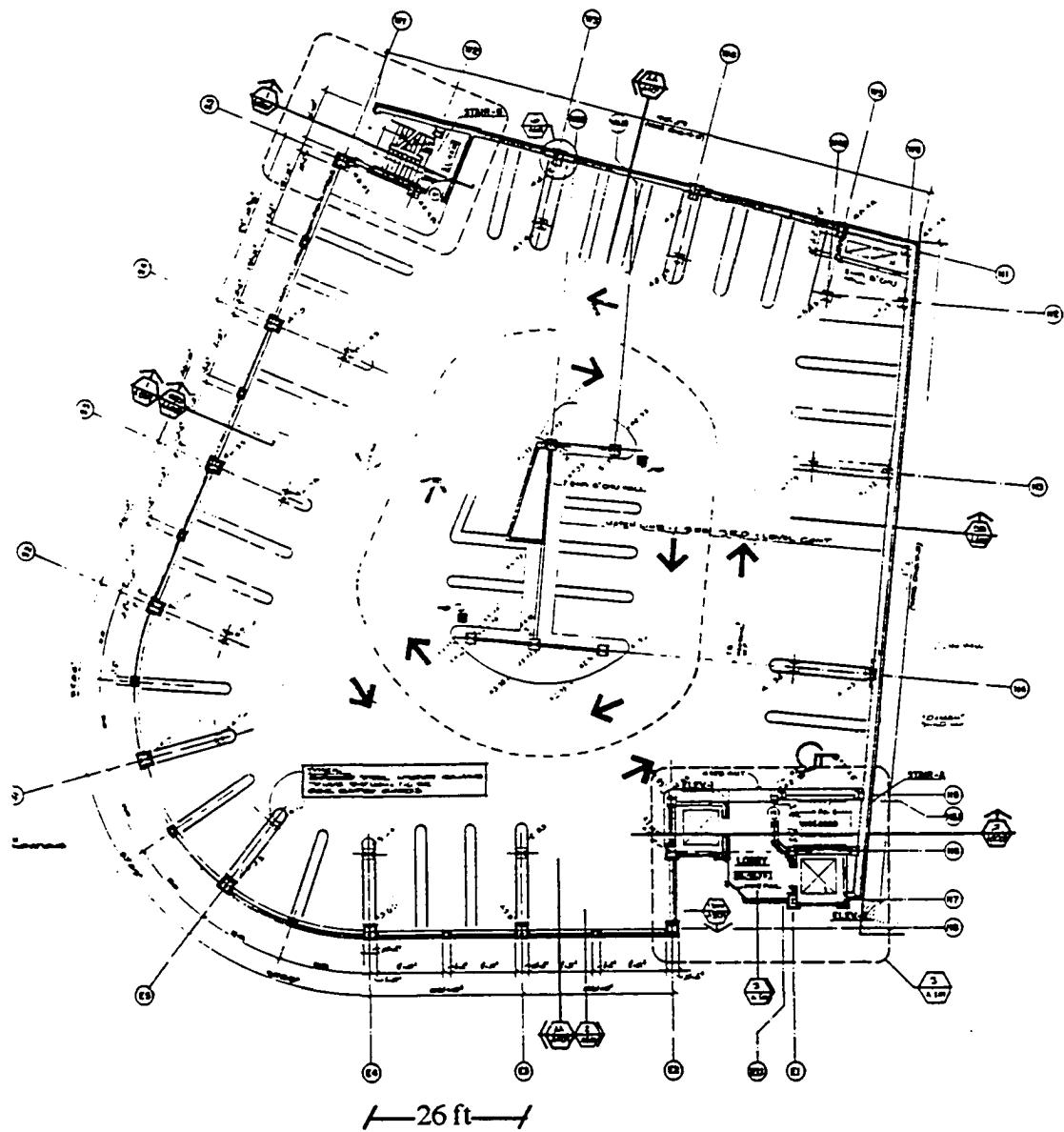


Figure B.5  
Second Parking Level Plan

## B.3 Calculations

Profile-C models the area requirements of the resources commonly used by all masonry activities of this project as it is the only profile in MoveSchedule that can be used to model resources used by more than one activity. These resources are:

- a prime mover (small mobile forklift) used as handling equipment to provide a continuous supply of bricks, blocks, and mortar to the work areas (denoted by C-1). Its dimensions are defined by its footprint, which is 7 ft long and 4 ft wide.
- a mortar mixer (denoted by C-2). Its estimated dimensions are 7 ft long and 4 ft wide including additional work space needed by the crew for maneuvering.

Mortar is another resource used by all masonry activities. Its area requirement is equal to the area needed by the mortar mixer, sand, and masonry cement bags. It could be modeled as a single Profile-C resource that is shared by all masonry activities. Alternatively, the mortar mixer, sand, and masonry cement could be modeled as individual resources for each activity and their space requirements can be computed independently for each. The second alternative is modeled here.

Sand is delivered to site in bulk whereas concrete blocks, bricks, and masonry cement are delivered to site on pallets. Sand is delivered in 12 yd<sup>3</sup> capacity dumptrucks. The area A occupied by a volume V of sand is approximated using the following equations for a given L/W [equations are from Nunnally (77) for a triangular cross-section and an angle of repose equal to 30°]:

- $V = 1/4 \tan 30^\circ \times L^2 \times W$  (1)
- $A = W \times L$

The space required by concrete blocks, bricks, and cement bags is estimated based on the space occupied by their pallets. A truck load of concrete blocks is typically 10 pallets. A pallet holds approximately 90 blocks and occupies a space 4 ft-long x 4 ft-wide x 3 ft-high. A truck load of bricks is typically 12 pallets. A pallet holds approximately 500 bricks and occupies a space 3 ft-long x 3 ft-wide x 3 ft-high. A truck load of masonry cement bags typically holds 10 pallets. The handling unit is a pallet of 40 bags and occupies an area of 4 ft-long x 4 ft-wide x 3 ft-high. Pallets are stacked 2 pallets high on site.

All scaffolding is estimated to be 3 ft-wide which is typical for buildings that are less than 5 stories high [Nolan 90, Div. 015-256 Means 88].

Sand is modeled using Profile-A to show a decrease in its area requirement as the corresponding activity progresses.

Masonry cement, concrete blocks, and bricks are modeled using Profile-B because pallets of blocks will occupy a constant space that does not shrink when they get consumed as the activity progresses.

A crew of 3 bricklayers and 2 bricklayer helpers (D-8 from Means 88) is selected for laying 8 in reinforced concrete blocks. The same crew type is selected for laying 8 in-long x 4 in-wide x 2-2/3 in-high standard brick.

Two construction methods are modeled as the normal and minimum resource levels of an activity. The number of resource levels was limited to a maximum of 2 levels in order to reduce computations and the number of iterations that MoveSchedule will undergo to adjust the schedule.

As a general rule, it is assumed that 3 or 4 deliveries of concrete blocks (bricks) are made when an activity is performed at the normal or minimum level respectively. Exceptions to this rule were made when the total number of truck loads needed by the activity or the activity duration are relatively small to justify so many deliveries as in activity-4.

The following is the derivation of activity durations and area requirements of resources needed by each activity. Field observations, current practice, and estimating handbooks [Means (88), Nolan (90), and Nunnally (77)] are used as references in determining the spatial requirements of resources.

### **B.3.1 Activity-1: Construction of the North Concrete Block Wall**

The area of the north concrete block wall is  $50 \text{ ft} \times 110 \text{ ft} = 5,500 \text{ ft}^2$ . A single D-8 crew is selected for performing this activity.

In addition to the mortar mixer and the prime mover, the resources used by this activity and requiring space on site are:

- 8 in-thick concrete blocks (denoted by B-1)
- sand used in mortar (denoted by B-7)

- masonry cement used in mortar (denoted by A-1)
- scaffolding (denoted by C-3 or C-4)

### Normal Resource Level Computations

The normal resource level models the resources used in constructing the north wall with conventional methods, i.e., with scaffolding erected at the outside of the building:

- crew productivity:  $365 \text{ ft}^2/\text{crew day}$  [042300-310.0200 Means 88]
- activity duration:  $d_{\text{norm}} = 5,500 \text{ ft}^2 / 365 \text{ ft}^2/\text{crew day} = 15 \text{ crew days}$  (2)
- number of blocks consumed per day =  $365 \text{ ft}^2/\text{crew day}$  ( $113 \text{ blocks} / 100 \text{ ft}^2$ )
 
$$= 413 \text{ blocks/day} [59 - \text{Means 88}] \quad (3)$$
- number of pallets consumed per day including a 5% waste factor
 
$$= (1.05 \times 413 \text{ blocks/day}) / 90 \text{ blocks/pallet}$$

$$= 4.8 \text{ pallets/day} = 0.5 \text{ truck load/day} \quad (4)$$

$$(\text{i.e., } 4.8 \text{ pallets/day} / 10 \text{ pallets/truck load})$$
- total number of truck loads =  $0.5 \text{ truck load/day} \times 15 \text{ days} = 7.5 \text{ truck loads.}$

Three deliveries of concrete blocks are made when the activity is performed at the normal level. The equivalent of  $7.5 \text{ truck loads} / 3 = 2.5 \text{ truck loads}$  is therefore needed. Most likely fully loaded trucks will arrive to site, thus the:

- number of pallets that occupy space on site =  $(3 \text{ truck loads} \times 10 \text{ pallets/truck load}) / 2 \text{ pallets/stack} = 15 \text{ pallets}$
- area required by B-1:  $A_{\text{norm}} = 15 \text{ pallets} \times 4 \text{ ft} \times 4 \text{ ft} = 240 \text{ ft}^2$
- assume that L/W of B-1 = 2.5.

The quantities of mortar, sand, and masonry cement bags per block are:

- volume of mortar per block =  $0.024 \text{ ft}^3/\text{block}$  [Nolan 90]
- number of cement bags per block =  $0.008 \text{ bags/block}$  [Nolan 90] (5)
- volume of sand per block =  $0.023 \text{ ft}^3/\text{block}$  [Nolan 90]. (6)

Using (3), (5), and (6), the consumption rates of cement bags and sand are:

- number of cement bags consumed per day =  $413 \text{ blocks/day} \times 0.008 \text{ bags/block}$ 

$$= 3.3 \text{ bags/day} \quad (7)$$
- volume of sand per day =  $413 \text{ blocks/day} \times 0.023 \text{ ft}^3/\text{block} = 9.5 \text{ ft}^3/\text{day.}$  (8)

Using (1), (2), (7), and (8), the total quantities (including a 5% waste) and areas of sand and cement bags are:

- volume of sand =  $1.05 \times 9.5 \text{ ft}^3/\text{day} \times 15 \text{ days} = 150 \text{ ft}^3$
- total number of cement bags =  $1.05 \times 3.3 \text{ bags/day} \times 15 \text{ days}$   
= 52 bags  
= 1.3 pallets (i.e., 52 bags / 40 bags/pallet)
- number of pallets of cement bags that occupy space on site  
= 1.3 pallets / 2 pallets/stack = 1 pallet
- assume that L/W of A-1 = 1.7
- area required by A-1:  $T_A = 90 \text{ ft}^2$
- area required by B-7:  $A_{\text{norm}} = 16 \text{ ft}^2$
- L/W of B-7 = 1 because the area of B-7 is that of one pallet and a pallet is square.

The scaffolding (C-3) is 3 ft wide and 110 ft long.

### Minimum Resource Level Computations

The minimum resource level models the resources used in constructing the north wall with scaffolding erected inside the parking structure. Constructing the wall from the inside greatly slows down productivity because the masons have to constantly adjust and move their scaffolding within the garage and work with limited headroom. When the height of the wall reaches the under side of the deck above, the masons have to reach down from the level above to lay the block. Therefore, a normal crew D-8 is assumed to work at an estimated 60 % efficiency with this method of construction.

- crew productivity =  $0.6 \times 365 \text{ ft}^2/\text{crew day} = 219 \text{ ft}^2/\text{crew day}$
- activity duration:  $d_{\min} = 5,500 \text{ ft}^2 / 219 \text{ ft}^2/\text{crew day} = 25 \text{ days.}$  (9)

- number of blocks consumed per day =  $219 \text{ ft}^2/\text{crew day} (113 \text{ blocks} / 100 \text{ ft}^2)$   
= 248 blocks/day (10)

- number of pallets consumed per day (including a 5 % waste)  
=  $1.05 \times 248 \text{ blocks/day} / 90 \text{ blocks/pallet}$   
= 2.9 pallets/day = 0.29 truck load/day (11)

Smaller quantities can be delivered to keep up with the rate at which the blocks are consumed. Accordingly, 4 deliveries are made at the minimum level. Since a total of 7.5 truck loads are needed for this activity, the equivalent of  $7.5 \text{ truck loads} / 4 = 2 \text{ truck loads}$  need space.

- number of pallets that occupy space on site =  $(2 \text{ truck loads} \times 10 \text{ pallets/truck load}) / 2 \text{ pallets/stack} = 10 \text{ pallets}$
- area required by B-1:  $A_{\min} = 10 \text{ pallets} \times 4 \text{ ft} \times 4 \text{ ft} = 160 \text{ ft}^2$
- L/W of B-1 = 2.5.

The number of cement bags consumed per day will also be lower for the minimum resource level. However, since the smallest area that cement bags can occupy is the area of a pallet, their area requirement remains:

- area of B-7:  $A_{\min} = 16 \text{ ft}^2$
- L/W of B-7 = 1.

The area requirement of sand is the same as for the normal level because the total quantity is brought to site at the start of the activity and will not vary with changes in activity duration.

A different resource (denoted C-4) represents the scaffolding used inside the building because different zoning constraints model the possible positions of C-3 and C-4. Like C-3, it is 3 ft wide and 110 ft long.

### B.3.2 Activity-2: Construction of the West Concrete Block Wall

The area of the west concrete block wall is  $50 \text{ ft} \times 95 \text{ ft} = 4,750 \text{ ft}^2$ . A single D-8 crew is selected for performing this activity. The resources used by this activity and requiring space on site are:

- 8 in-thick concrete blocks (denoted by B-2)
- sand used in mortar (denoted by B-8)
- masonry cement used in mortar (denoted by A-2)
- scaffolding (denoted by C-5 or C-6).

#### Normal Resource Level Computations

The normal resource level models the resources used in constructing the west wall with conventional methods.

- crew productivity:  $365 \text{ ft}^2/\text{crew day}$
- activity duration:  $d_{\text{norm}} = 4,750 \text{ ft}^2 / 365 \text{ ft}^2/\text{crew day} = 13 \text{ crew days}$  (12)

The area requirement of B-2 is obtained using data previously derived for activity-1, using (4) and (12):

- total number of truck loads =  $0.5 \text{ truck load/day} \times 13 \text{ days} = 6.5 \text{ truck loads.}$

Three deliveries of concrete blocks are made when the activity is performed at the normal level and thus  $6.5 \text{ truck loads} / 3 = 2.2 \text{ truck loads}$  is needed. Most likely fully loaded trucks will arrive to site, thus the:

- number of pallets that occupy space on site =  $(3 \text{ truck loads} \times 10 \text{ pallets/truck load}) / 2 \text{ pallets/stack} = 15 \text{ pallets}$
- area required by B-2:  $A_{\text{norm}} = 15 \text{ pallets} \times 4 \text{ ft} \times 4 \text{ ft} = 240 \text{ ft}^2$
- assume that L/W of B-2 = 2.5.

The total quantities (including a 5% waste factor) and areas of sand and cement bags are derived using (1), (7), (8), and (11):

- volume of sand =  $1.05 \times 9.5 \text{ ft}^3/\text{day} \times 13 \text{ days} = 130 \text{ ft}^3$
- total number of cement bags =  $1.05 \times 3.3 \text{ bags/day} \times 13 \text{ days} = 45 \text{ bags} = 1.12 \text{ pallets}$
- number of pallets of cement bags that occupy space on site =  $1.12 \text{ pallets} / 2 \text{ pallets/stack} = 1 \text{ pallet}$
- assume L/W of A-2 = 1.7
- area required by A-2:  $T_A = 85 \text{ ft}^2$
- area required by B-8:  $A_{\text{norm}} = 16 \text{ ft}^2$
- L/W of B-8 = 1 because the area of B-8 is that of one pallet and a pallet is square.

C-5 represents the scaffolding used on the outside of the building. It is 3 ft wide and 95 ft long.

### Minimum Resource Level Computations

The minimum resource level models the resources used in constructing the west wall from inside the parking structure. For the reasons described in the previous section, a normal crew D-8 is assumed to work at an estimated 60 % efficiency with this method of construction.

- crew productivity =  $365 \text{ ft}^2/\text{crew day} \times 0.6 = 219 \text{ ft}^2/\text{crew day}$
- activity duration:  $d_{\min} = 4,750 \text{ ft}^2 / 219 \text{ ft}^2/\text{crew day} = 22 \text{ days.}$  (13)

Smaller quantities can be delivered to keep up with the rate at which the blocks are consumed. Accordingly, 4 deliveries are made at the minimum level. Since 6.5 truck loads in total are needed for this activity, the equivalent of  $6.5 \text{ truck loads} / 4 = 2$  truck loads need space.

- number of pallets that occupy space on site =  $(2 \text{ truck loads} \times 10 \text{ pallets/truck load}) / 2 \text{ pallets/stack} = 10 \text{ pallets}$
- area required by B-2:  $A_{\min} = 10 \text{ pallets} \times 4 \text{ ft} \times 4 \text{ ft} = 160 \text{ ft}^2$
- assume that L/W of B-2 = 2.5.

Since the smallest area that cement bags can occupy is the area of a pallet, their area requirement remains:

- area of B-8:  $A_{\min} = 16 \text{ ft}^2$
- L/W of B-8 = 1.

The area requirement of sand is the same as for the normal level because the total quantity is brought to site at the start of the activity and will not vary with changes in activity duration.

A different resource (denoted C-6) represents the scaffolding used inside the building because different zoning constraints model the possible positions of C-5 and C-6. C-6 is 3 ft wide and 92 ft long. The length of C-6 is smaller than that of C-5 because the length of the wall on the inside is smaller.

### B.3.3 Activity-3: Construction of the West Wall Brick Facade

A special permission was obtained for using the neighbor's driveway along the west wall for a limited time period of 12 days which made it possible to construct that brick facade. Therefore, a single normal resource level is defined for this activity and represents the resources needed for its performance at a duration of 12 days. The resources used by this activity are:

- bricks (denoted by B-3)
- sand used in mortar (denoted by B-9)
- masonry cement used in mortar (denoted by A-3)
- scaffolding (denoted by C-7).

### **Normal Resource Level Computations**

The area of the west brick facade is  $50 \text{ ft} \times 95 \text{ ft} = 4,750 \text{ ft}^2$ . Two D-8 type crews are selected for performing this activity.

- crew productivity is 2,800 bricks/crew day [042100-184.0140 Means 88]
- number of bricks per  $\text{ft}^2 = 6.75 \text{ blocks}/\text{ft}^2$  [59 Div.4 Means 88]
- total number of bricks (including a 5 % waste) =  $6.75 \text{ bricks}/\text{ft}^2 \times 4,750 \text{ ft}^2$   
 $\times 1.05 = 33,666 \text{ bricks}$  (14)
- activity duration:  $d_{\text{norm}} = 33,666 \text{ bricks} / 2,800 \text{ bricks/crew day}$   
 $= 12 \text{ days}$  (15)
- total number of pallets =  $33,666 \text{ bricks} / 500 \text{ bricks/pallet} = 68 \text{ pallets}$  (16)
- total number of trucks =  $68 \text{ pallets} / 12 \text{ pallets/truck load} = 6 \text{ truck loads}$  (17)

Three deliveries of bricks are made for this activity, i.e., 2 truck loads need space on site.

- number of pallets that occupy space on site =  $(2 \text{ truck loads} \times 12 \text{ pallets/truck load}) / 2 \text{ pallets/stack} = 12 \text{ pallets}$
- area required by B-3:  $A_{\text{norm}} = 12 \text{ pallets} \times 3 \text{ ft} \times 3 \text{ ft} = 108 \text{ ft}^2$
- assume that L/W of B-3 = 3.

The quantities of mortar, sand, and masonry cement bags per brick are:

- volume of mortar per brick =  $0.0103 \text{ ft}^3/\text{brick}$  [59 Means 88] (18)
- number of cement bags per brick =  $0.8 \text{ bags}/2.4 \text{ ft}^3 \text{ of mortar} \times 0.0103 \text{ ft}^3/\text{brick}$   
 $= 0.0034 \text{ bags/brick}$  [Nolan 90] (19)
- volume of sand per brick =  $2.3 \text{ ft}^3 \text{ of sand} / 2.4 \text{ ft}^3 \text{ of mortar} \times 0.0103 \text{ ft}^3/\text{brick}$   
 $= 0.01 \text{ ft}^3/\text{brick}$  [Nolan 90] (20)

Using (1), (14), (19) and (20), the total quantities (including a 5% waste) and area requirements of sand and cement bags are:

- volume of sand =  $0.01 \text{ ft}^3/\text{brick} \times 33,666 \text{ bricks} = 350 \text{ ft}^3$
- total number of cement bags =  $0.0034 \text{ bags/brick} \times 33,666 \text{ bricks}$   
 $= 120 \text{ bags} = 3 \text{ pallets}$
- number of pallets of cement bags that occupy space on site  
 $= 3 \text{ pallets} / 2 \text{ pallets/stack} = 2 \text{ pallets}$
- assume that L/W of A-3 = 1.6
- total area of sand A-3:  $TA = 155 \text{ ft}^2$

- area of cement bags B-9:  $A_{norm} = 2 \times 4 \text{ ft} \times 4 \text{ ft} = 32 \text{ ft}^2$
- L/W of B-9 = 2 because the area of B-9 is that of 2 pallets which can only be arranged in that L/W ratio.

The scaffolding (C-7) is 3 ft wide and 95 ft long.

### **B.3.4 Activity-4: Construction of the North Wall Brick Facade**

This activity involves laying standard-type bricks on the upper 20 ft of the north facade. The area of the north brick facade is  $20 \text{ ft} \times 110 \text{ ft} = 2,200 \text{ ft}^2$ . A single D-8 crew is selected for this job.

- total number of bricks (including 5% waste)  $= 1.05 \times 6.75 \text{ bricks}/\text{ft}^2 \times 2,200 \text{ ft}^2$   
 $= 15,593 \text{ bricks.}$  (21)

The resources needed by this activity are:

- bricks (denoted by B-4)
- sand (denoted by A-4)
- masonry cement (denoted by B-10)
- scaffolding (denoted by C-8).

#### **Normal Resource Level Computations**

The normal resource level models the resources used in constructing the north facade with scaffolding erected at the outside of the building:

- crew productivity is 1,400 bricks/crew day
- activity duration:  $d_{norm} = 15,593 \text{ bricks} / 1,400 \text{ bricks}/\text{crew day} = 11 \text{ days}$  (22)

Using data previously derived for activity-3, the total number of truck loads of bricks needed for this activity are:

- total number of pallets  $= 15,593 \text{ bricks} / 500 \text{ bricks/pallet} = 32 \text{ pallets}$  (23)
- total number of trucks  $= 32 \text{ pallets} / 12 \text{ pallets/truck load} = 3 \text{ truck loads.}$  (24)

As an exception to the generally adopted rule which assumes that 3 deliveries are made at the normal level and 4 at the minimum level, a single delivery of bricks is made for this activity at the normal level and 3 deliveries are made at the minimum level. This is so

because the activity duration is short and the total number of truck loads is small to justify multiple deliveries.

- number of pallets that occupy space on site =  $(3 \text{ truck loads} \times 12 \text{ pallets/truck load}) / 2 \text{ pallets/stack} = 18 \text{ pallets}$
- area required by bricks B-4:  $A_{\text{norm}} = 18 \text{ pallets} \times 3 \text{ ft} \times 3 \text{ ft} = 162 \text{ ft}^2$
- assume that L/W of B-4 = 1.5.

Using (1), (19), (20), and (21), the total quantities (including a 5% waste) and area requirements of sand and masonry cement bags are:

- volume of sand =  $1.05 \times 15,593 \text{ bricks} \times 0.01 \text{ ft}^3/\text{brick} = 164 \text{ ft}^3$
- total number of cement bags =  $1.05 \times 15,593 \times 0.0034 \text{ bags/brick} = 56 \text{ bags}$   
= 1.4 pallets
- number of pallets of cement bags that occupy space on site  
=  $1.4 \text{ pallets} / 2 \text{ pallets/stack} = 1 \text{ pallet}$
- assume that L/W of A-4 = 1.7
- total area required by sand A-4:  $T_A = 92 \text{ ft}^2$
- area required by cement B-10:  $A_{\text{norm}} = 1 \text{ pallet} \times 4 \text{ ft} \times 4 \text{ ft} = 16 \text{ ft}^2$
- L/W = 1.

The scaffolding (C-8) is 3 ft wide and 110 ft long.

### **Minimum Resource Level Computations**

The minimum level models the resources used in constructing the wall using a cantilevered-type scaffolding suspended from the top slab. A normal crew D-8 is assumed to work at an estimated 60% efficiency with this method of construction.

- Crew productivity =  $1,400 \text{ bricks/crew day} \times 0.6 = 840 \text{ bricks/crew day}$
- activity duration:  $d_{\text{min}} = 15,593 \text{ bricks} / 840 \text{ bricks/crew day} = 19 \text{ days.}$  (25)

To balance the reduction in the crew productivity at the minimum level, 3 deliveries of bricks are made over the duration of the activity at this level. Since a total of 3 truck loads are needed for this activity, the equivalent of 1 truck load needs space.

- total number of pallets that occupy space on site =  $(1 \text{ truck load} \times 12 \text{ pallets/truck load}) / 2 \text{ pallets/stack} = 6 \text{ pallets}$
- Area required by B-4:  $A_{\text{min}} = 6 \text{ pallets} \times 3 \text{ ft} \times 3 \text{ ft} = 54 \text{ ft}^2$
- assume that L/W of B-4 = 1.5.

The area requirements of sand and masonry cement bags are the same as for the normal level because the total quantity is brought to site at the start of the activity and will not vary with changes in activity duration.

The cantilevered scaffolding is not represented by a resource as it does not occupy space on the ground.

### B.3.5 Activity-5: Construction of the East Wall Brick Facade

The east facade faces the street; there are no adjacent buildings or access restrictions to hamper construction. Therefore, a single normal resource level is defined for this activity. The facade has many openings and the effective wall area to be covered with bricks is only half of the total wall area. The effective area of the east facade =  $0.5 \times 50 \text{ ft} \times 95 \text{ ft} = 2,375 \text{ ft}^2$ .

- total number of bricks (including 5 % waste) =  $1.05 \times 6.75 \text{ bricks}/\text{ft}^2 \times 2,375 \text{ ft}^2$   
= 16,833 bricks (26)

A single D-8 crew is selected for this job. The crew is expected to work at 80% efficiency because this wall has many architectural details, including ornamental brickwork and arches.

- crew productivity =  $0.8 \times 1,400 \text{ bricks}/\text{crew day} = 1,120 \text{ bricks}/\text{crew day}$ .

The resources needed by this activity are:

- bricks (denoted by B-5)
- sand (denoted by A-5)
- masonry cement (denoted by B-11)
- scaffolding (denoted by C-9).

#### Normal Resource Level Computations

The normal level models the resources used in constructing the east facade with scaffolding erected at the outside of the building:

- activity duration:  $d_{\text{norm}} = 16,833 \text{ bricks} / 1,120 \text{ bricks}/\text{crew day}$   
= 15 days (27)

- total number of pallets of bricks =  $16,833 \text{ bricks} / 500 \text{ bricks/pallet}$   
= 34 pallets (28)

- total number of truck-loads =  $34 \text{ pallets} / 12 \text{ pallets/truck load}$   
 $= 3 \text{ truck loads.}$  (29)

Three deliveries of bricks are made for this activity, thus the equivalent of 1 truck load or 6 pallets (because pallets are stacked double) needs space on site.

- area required by bricks B-5:  $A_{\text{norm}} = 6 \times 3 \text{ ft} \times 3 \text{ ft} = 54 \text{ ft}^2$
- assume that L/W of B-5 = 1.5.

Using (1), (19), (20), and (26), the total quantities (including 5% waste) and area requirements of sand and masonry cement bags are:

- volume of sand =  $1.05 \times 0.01 \text{ ft}^3/\text{brick} \times 16,833 \text{ bricks} = 177 \text{ ft}^3$
- total number of cement bags =  $1.05 \times 0.0034 \text{ bags/brick} \times 16,833 \text{ bricks}$   
 $= 57 \text{ bags} = 1.4 \text{ pallets}$
- number of pallets that occupy space on site =  $1.4 \text{ pallets} / 2 \text{ pallets/stack} = 1 \text{ pallet}$
- assume that L/W of A-5 = 1.7
- total area required by sand A-5:  $TA = 96 \text{ ft}^2$
- area required by cement B-11:  $A_{\text{norm}} = 1 \times 4 \text{ ft} \times 4 \text{ ft} = 16 \text{ ft}^2$
- L/W of B-11 = 1.

The scaffolding C-9 is 3 ft wide and 95 ft long.

### B.3.6 Activity-6: Construction of the South Wall Brick Facade

This wall is similar in all respects to the east wall, except for its orientation and dimensions. A single D-8 crew is used for its construction.

- effective area of the south facade =  $0.5 \times 110 \text{ ft} \times 50 \text{ ft} = 2,750 \text{ ft}^2$
- total number of bricks (including 5 % waste) =  $1.05 \times 6.75 \text{ bricks}/\text{ft}^2 \times 2,750 \text{ ft}^2$   
 $= 19,490 \text{ bricks}$  (30)
- crew productivity =  $0.8 \times 1,400 \text{ bricks/crew day} = 1,120 \text{ bricks/crew day.}$

The resources needed by this activity are:

- bricks (denoted by B-6)
- sand (denoted by A-6)
- masonry cement (denoted by B-12)
- scaffolding (denoted by C-10 and C-11)

### **Normal Resource Level Computations**

The normal level models the resources used in constructing the south facade with scaffolding erected at the outside of the building:

- activity duration:  $d_{norm} = 19,490 \text{ bricks} / 1,120 \text{ bricks/crew day} = 17 \text{ days}$  (31)
- total number of pallets of bricks =  $19,490 \text{ bricks} / 500 \text{ bricks/pallet}$   
= 39 pallets
- total number of truck loads =  $39 \text{ pallets} / 12 \text{ pallets/truck load}$   
= 4 truck loads. (32)

It is assumed that 4 deliveries of bricks are made for this activity, thus the equivalent of 1 truck load or 6 pallets (because pallets are stacked double) needs space on site.

- area required by bricks B-6:  $A_{norm} = 6 \times 3 \text{ ft} \times 3 \text{ ft} = 54 \text{ ft}^2$
- assume that L/W of B-6 = 1.5.

Using (1), (19), (20), and (30), the total quantities (including 5% waste) and area requirements of sand and masonry cement bags are:

- volume of sand =  $1.05 \times 0.01 \text{ ft}^3/\text{brick} \times 19,490 \text{ bricks} = 195 \text{ ft}^3$
- total number of cement bags =  $1.05 \times 0.0034 \text{ bags/brick} \times 19,490 \text{ bricks}$   
= 66 bags = 1.6 pallets
- number of pallets that occupy space on site =  $1.6 \text{ pallets} / 2 \text{ pallets/stack} = 1 \text{ pallet}$
- assume that L/W of A-6 = 1.7
- total area required by sand A-6:  $TA = 102 \text{ ft}^2$
- area required by cement B-12:  $A_{norm} = 1 \times 4 \text{ ft} \times 4 \text{ ft} = 16 \text{ ft}^2$
- L/W of B-12 = 1

The scaffolding needed is 3 ft wide and 110 ft long. It is modeled with 2 Profile-C resources: C-10 which is 90 ft long and C-11 which is 20 ft long. This is needed, in MoveSchedule, because part of the scaffolding will run on top of the Garage main entrance (D-8) which makes C-10 and C-11 subject to different zoning constraints.

# **Bibliography**

## BIBLIOGRAPHY

- Abdul-Sater, H., Delchambre, A. (1994). "Computer-Aided Planning in Masonry Construction." *Proceedings 11th International Symposium on Automation and Robotics in Construction*, ISARC, Brighton, England, U.K., Elsevier Science Publishers, pp. 141-147.
- Anson, M., Yip, P. (1992). "Graphics for Short Term Site Planning." *3rd International Conference on Modern Techniques in Construction, Engineering, and Project Management*, Singapore, pp. 8-13, March.
- Aoki, T., Kimura, T., Momozaki, K., Suzuki, A. (1993). "Comprehensive Site Monitoring Through Model Based Reasoning." *Proceedings 5th International Conference in Computing in Civil & Building Engineering*, ASCE, Vol. 1, pp. 153-160, New York, NY, June.
- Apple, J.M. (1977). *Plant Layout and Material Handling*. Wiley, 3rd Edition.
- Armour, G.C., Buffa, E.S. (1963). "A Heuristic Algorithm and Simulation Approach to Relative Location of Facilities." *Management Science*, Vol. 9, No. 2, pp. 294-309.
- Ashley, D.B., Uehara, K., Burke, E.R. (1983). "Critical Decision Making During Construction." *Journal of Construction Engineering and Management*, ASCE, Vol. 109, No. 2, pp. 146-162, June.
- Banerjee, P., Montreuil, B., Moodie, C.L., Kashyap, R.L. (1990). "A Qualitative Reasoning-Based Interactive Optimization Methodology For Layout Design." *Working*

*Draft 90-14*, Faculté des Sciences de l'Administration, Université Laval, Québec, Canada.

Bechtel Software, Inc. (1988). "3DM CAD/E for Designers." 4pp., and "Walkthru 3-D Animation and Visualization System." 4 pp., Acton, MA.

Beliveau, Y.J., Dal, T., Dixit, S. (1993). "CAD Applications for Material Handling in the Construction Environment." *Proceedings 5th International Conference in Computing in Civil & Building Engineering*, ASCE, Vol. 1, pp. 125-128, New York, NY.

Bell, L.C., McCullouch, B.G. (1988). "Bar Code Application In Construction." *Journal of Construction Engineering and Management*, ASCE, Vol. 114, No. 2, pp. 263-278, June.

Bell, L.C., Stukhart, G. (1986). "Attributes of Materials Management Systems." *Journal of Construction Engineering and Management*, ASCE, Vol. 112, No. 1, pp. 14-21, March.

Bell, L.C., Stukhart, G. (1987). "Costs and Benefits of Materials Management Systems." *Journal of Construction Engineering and Management*, ASCE, Vol. 113, No. 2, pp. 222-234, June.

Bernold, L.E., Salim, M. (1993). "Placement-Oriented Design and Delivery of Concrete Reinforcement." *Journal of Construction Engineering and Management*, ASCE, Vol. 119, No. 2, pp. 323-335, June.

Biro, M. (1990). "Object-Oriented Interaction in Resource Constrained Scheduling." *Information Processing Letters*, Vol. 36, pp. 65-67, North-Holland, October.

Bjornsson, H.C. (1986). "Management of Construction Material Procurement." *Proceedings CIB-W65*, pp. IV-60 to IV-73, Washington, DC, May.

Bohinsky, J.A., Fails, D.W. (1991). "Computer Aided Rigging Design System." *7th Conference on Computing in Civil Engineering and Symposium and Data Bases*, ASCE, pp. 710-718, May.

Bookbinder, J.H., Gervais, D. (1992). "Material-Handling Equipment Selection Via an Expert System." *Journal of Business Logistics*, Vol. 13, No. 1, pp. 149-171.

Bozer, Y.A. (1990). *Industrial and Organizational Engineering Facility Planning*. Class Project, Industrial and Operations Engineering Department, The University of Michigan, Ann Arbor, MI.

Bozer, Y.A., Meller, R.D., Erlebacher, S.J. (1994). An Improvement-Type Layout Algorithm for Single and Multiple-floor Facilities. *Management Science*, Vol. 40, No. 7, pp. 918-932.

Bridgewater, C., Griffin, M., Retik, A. (1994). "Use of Virtual Reality in Scheduling and Design of Construction Projects." *11th International Symposium on Automation and Robotics in Construction*, ISARC, Brighton, England, U.K., Elsevier Science Publishers, pp. 249-257.

Buffa, E.S., Armour, G.C., Vollman, T.E. (1964). "Allocating Facilities With CRAFT." *Harvard Business Review*, pp. 136-158, March/April.

Chandler, I.E. (1978). "The Planning of Storage and Movement of Materials on Site." *Building Technology and Management*, pp. 14-16, October.

Chandrasekaran, B. (1990). "Design Problem Solving: A Task Analysis." *Artificial Intelligence Magazine*, Vol. 11, No. 4, pp. 59-71, Winter.

Cheng, M.-Y. (1992). *Automated site layout of temporary facilities using geographic information systems (GIS)*. Ph.D. Dissertation, University of Texas at Austin, Austin, TX, December.

Cheng, M.Y., Varghese, K., O'Connor, J.T. (1992). "Management of spatial information for construction planning and design using geographical information systems (GIS)." *Proceedings 9th International Symposium on Automation and Robotics in Construction*, ISARC, JIRA, Vol. 2, pp. 393-402.

Cheng, M.-Y., O'Connor, J.T. (1993). "Site layout of construction temporary facilities using enhanced-geographic information system (GIS)." *Proceedings 10th International Symposium on Automation and Robotics in Construction*, ISARC, Watson, G.H., Tucker, R.L., and Walters, J.K. (eds.), Elsevier Science Publishers, pp. 399-406.

Chhajed, D., Montreuil, B., Lowe, T.J. (1992). "Flow Network Design for Manufacturing Systems Layout." *European Journal of Operational Research*, Vol. 57, No. 2, pp. 145-161.

- Choi, B. (1994). *Development of Interactive Design Systems: from Paradigm to Practice*. MS Thesis, Department of Civil and Environmental Engineering, Carnegie Mellon University, Pittsburgh, PA.
- Confrey, T., Daube, F. (1988). "GS2D: A 2D Geometry System." *Knowledge Systems Laboratory, KSL Report* No. 88-15, Computer Science Department, Stanford University, Stanford, CA.
- Davies, E.M. (1973). "New Criterion for Resource Scheduling." *Transportation Engineering Journal of ASCE*, Proceedings of ASCE, Vol. 99, No. TE4, pp. 741-756, November.
- Dechter, R., Pearl, J. (1988). "Network-Based Heuristics for Constraint-Satisfaction Problems." *Artificial Intelligence*, Vol. 34, pp. 1-38.
- Dechter, R., Pearl, J. (1989). "Research Note. Tree Clustering for Constraint Networks." *Artificial Intelligence*, Vol. 38, pp. 353-366.
- Donaghey, C.E. (1986). "A Department Location System for Micro-Computers." *Proceedings International Industrial Engineering Conference*, Institute of Industrial Engineers, Vol. 22, pp. 113-117.
- Drolet, J., Montreuil, B., Moodie, C. (1990). "Virtual Cellular Manufacturing Layout Planning." *Proceedings International Industrial Engineering Conference*, Institute of Industrial Engineers, pp. 236-341, May.
- Dzeng, R.J., Tommelein, I.D. (1994). "Case Storage of Planning Knowledge for Power Plant Construction." *Proceedings 1st Congress in Computing in Civil Engineering*, ASCE, Vol. 1, pp. 293-300.
- Easa, S.M. (1989). "Resource Leveling in Construction by Optimization." *Journal of Construction Engineering and Management*, ASCE, Vol. 115, No. 2, pp. 302-316, June.
- Eastman, C.M. (1972). "Preliminary Report on a System for General Space Planning." *Communications of the ACM*, Vol. 15, No. 2, pp. 76-87, February.
- Eastman, C.M. (1975) (editor). *Spatial Synthesis in Computer-Aided Building Design*. Halsted Press, New York, NY.

- ENR (1990). "Huge Scrubber 'Rolled In'." *Engineering News Record*, The McGraw-Hill Construction Weekly, 10/4/90, pp. 13.
- European (1992). Special Issue Facility Layout. *European Journal of Operational Research*, Vol. 57, No. 2.
- Flemming, U., Coyne, R.F., Glavin, T., Hsi, H., Rychener, M.D. (1988). "A Generative Expert System for the Design of Building Layouts - Version 2." Gero, J.S. (editor), *Artificial Intelligence in Engineering: Design*, Elsevier, pp. 445-464.
- Foster, D.P., Vohra, R.V. (1989). "Probabilistic Analysis of a Heuristics for the Dual Bin Packing Problem." *Information Processing Letters*, Vol. 31, pp. 287-290, North-Holland, June.
- Francis, R.L., White, J.A. (1974). *Facility Layout & Location - An Analytical Approach*. Prentice-Hall, Inc., Englewood Cliffs, NJ.
- Garey, M.R., Johnson, D.S. (1979). *Computers and Intractability; a Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, New York, NY.
- Gavish, B., Pirkul, H. (1991). "Algorithms for the Multi-Resource Generalized Assignment Problem." *Management Science*, Vol. 37, No. 6, pp. 695-713, June.
- Geis, A.J. (1986). "Printing Plant Layout and Facility Design." *Proceedings International Industrial Engineering Conference*, Institute of Industrial Engineers, pp. 181-190, Fall.
- Gore, M.C. Jr. (1995). DuPont Engineering, Wilmington, DE, Personal Communication.
- Gray, C., Little, J. (1985). "A Systematic Approach to the Selection of an Appropriate Crane for a Construction Site." *Construction Management and Economics*, No. 3, pp. 121-144.
- Haddadi, A. (1988). *Project Planning and Resource Movement*. Abstract, Ph.D. Thesis, University of Nottingham, Nottingham, England, July.
- Halpin, D.P., Riggs, L.S. (1992). *Planning and Analysis of Construction Operations*. John Wiley & Sons, New York, NY.

- Hamiani, A. (1987). *CONSITE: A Knowledge-Based Expert System Framework for Construction Site Layout*. Ph.D. Dissertation, Department of Civil Engineering, University of Texas, Austin, TX.
- Handa, V., Lang, B. (1988). "Construction Site Planning." *Construction Canada*, Vol. 30, No. 3, pp. 43-49, May.
- Handa, V., Lang, B. (1989). "Construction Site Efficiency." *Construction Canada*, Vol. 31, No. 1, pp. 40-48.
- Handa, V.K. (1986). "Construction Production Planning." *Journal of Construction Engineering and Management*, ASCE, Vol. 112, No. 2, pp. 163-177, June.
- Harris, R.B. (1978). *Precedence and Arrow Networking Techniques for Construction*. John Wiley & Sons, Inc., New York, NY.
- Haussman, W.H., Schwarz, L.B., Graves S.C. (1976). "Optimal Storage Assignment in Automatic Warehousing Systems." *Management Science*, Vol. 22, No. 6, February.
- Hayes-Roth, B., Johnson, M.V., Garvey, A., Hewett, M. (1986). "Application of the BB1 Blackboard Control to Arrangement-Assembly Tasks." *Artificial Intelligence Magazine*, Vol. 1, No. 2, pp. 85-94.
- Henderson, E.M. (1976). "The Use of Scale Models in Construction Management." *Technical Report No. 213*, The Construction Institute, Department of Civil Engineering, Stanford University, Stanford, CA.
- Hendler, J., Tate, A., Drummond, M. (1990). "AI Planning Systems and Techniques." *Artificial Intelligence*, Vol. 11, No. 2, pp. 61-77, Summer.
- Hendrickson, C., Zozaya-Gorostiza, C., Rehak, D., Baracco-Miller, E., Lim, P. (1987). "Expert System for Construction Planning." *Journal of Computing in Civil Engineering*, ASCE, Vol. 1, No. 4, pp. 253-269, October.
- Heragu, S.S., Kusiak, A. (1988). "Machine Layout Problem in Flexible Manufacturing Systems." *Operations Research*, Vol. 36, No. 2, pp. 258-268, March-April.
- Heragu, S.S., Kusiak, A. (1991). "Efficient Models for the Facility Layout Problem." *European Journal of Operational Research*, Vol. 53, No. 1, pp. 1-13.

- Hillier, F.S. (1963). "Quantitative Tools for Plant Layout Analysis." *Journal of Industrial Engineering*, Vol. 14, No. 1, pp. 33-40.
- Hirasawa, G., Matsumura, S., Yamazaki, Y., Kataoka, M. (1993). "Knowledge and Information Processing with Drawing Objects in Construction Phase." *Proceedings 5th International Conference in Computing in Civil & Building Engineering*, ASCE, Vol. 1, pp. 131-136, New York, NY, June.
- Hoffman, R.R. (1987). "The Problem of Extracting the Knowledge of Experts from the Perspective of Experimental Psychology." *Artificial Intelligence Magazine*, Vol. 8, No. 2, pp. 53-67, Summer.
- Humphreys, K.K. (1991). *Jelen's Cost and Optimization Engineering*. Chap. 16, pp. 476-481, 3rd Edition, McGraw-Hill, Inc., New York, NY.
- Hyvonen, E. (1992). "Constraint Reasoning Based on Interval Arithmetic: The Tolerance Propagation Approach." *Artificial Intelligence*, Vol. 58, pp. 71-112.
- Immer, J. R. (1953). Warehousing Activities. *Materials Handling*, Chap. 29, pp. 407-421, McGraw-Hill, New York, NY.
- Irtishad, A., Morad, A. (1993). "Computer-Aided Decision Model for Selecting a Contractual System." *Proceedings 5th International Conference in Computing in Civil & Building Engineering*, ASCE, Vol. 1, pp. 255-261, New York, NY.
- Johnson, R.V. (1982a). "SPACECRAFT for Multi-Floor Layout Planning." *Management Science*, Vol. 28, No. 4, pp. 407-417.
- Johnson, R.V. (1982b). "Building Shapes that Minimize Internal Traffic Costs." *Paper submitted to the School of Business Administration*, University of Michigan, Ann Arbor.
- Kameyama, K., Kondo, K., Ohtomi, K. (1990). "An Intelligent Interactive Layout CAD System for Industrial Plants." *2nd International Conference on Design Theory and Methodology*, ASME, Vol. 27, pp. 33-38.
- Kartam, N.A., Levitt, R.E. (1990). "Intelligent Planning of Construction Projects." *Journal of Computing in Civil Engineering*, ASCE, Vol. 4, No. 2, pp. 155-176, April.

- Kasprowicz, T. (1994). "Multi-Objective of Construction Schedules." *Proceedings 1st Congress in Computing in Civil Engineering*, ASCE, Vol. 1, pp. 185-190.
- Kay, E. (1968). *A Mathematical Model For Handling in a Warehouse*. New York: Pergamon Press.
- Ketcham, R.L., Malstrom, E.M. (1984). "A Computer Assisted Facilities Layout Algorithm using Graphics." *Proceedings International Industrial Engineering Conference*, Institute of Industrial Engineers, pp. 88-95.
- King, C.U., Fisher, E.L. (1986). "Object-Oriented Shop-Floor Design-Simulation and Evaluation." *Proceedings International Industrial Engineering Conference*, Institute of Industrial Engineers, pp. 131-137.
- Kouvelis, P., Kiran, A.S. (1991). "Single and Multiple Period Layout Models for Automated Manufacturing Systems." *European Journal of Operational Research*, Vol. 52, pp. 300-314, North-Holland.
- Krause, K., Larmore, L.L., Volper, D.J. (1987). "Packing Items from a Triangular Distribution." *Information Processing Letters*, Vol. 25, pp. 351-361, North-Holland, May.
- Kumar, V. (1992). "Algorithms for Constraint-Satisfaction Problems: A Survey." *Artificial Intelligence Magazine*, pp. 32-44, Spring.
- Kunigahalli, R., Russell, J.S. (1994). "CONCPLANNER: An Automated Process Planning System for Concrete Construction." *Proceedings 11th International Symposium on Automation and Robotics in Construction*, Vol. 1, pp. 117-123.
- Kunigahalli, R., Russell, J.S. (1994). "Visibility Graph Approach to Detailed Path Planning in CNC Concrete Placement." *Proceedings 11th International Symposium on Automation and Robotics in Construction*, ISARC, Brighton, England, U.K., Elsevier Science Publishers, pp. 141-147.
- Kusiak, A. (1987). "Artificial Intelligence and Operations Research in Flexible Manufacturing Systems." *INFOR*, Vol. 25, No. 1, pp. 2-12.
- Kusiak, A., Heragu, S.S. (1987). "The Facility Layout Problem." *European Journal of Operational Research*, Vol. 29, pp. 229-251.

- Kusiak, A., Heragu, S.S. (1988). "Knowledge-Based System Guides Machine Layout in Flexible Manufacturing System." *Industrial Engineering*, Vol. 20, No. 11, pp. 48-53, November.
- Lee, H.L., Rosenblatt, M.J. (1987). "Simultaneous Determination of Production Cycle and Inspection Schedules in a Production System." *Management Science*, Vol. 33, No. 9, pp. 1125-1136, September.
- Lee, R.C., Moore, J.M. (1967). "CORELAP-Computerized Relationship Layout Planning." *Journal of Industrial Engineering*, Vol. 18, No. 3, pp. 195-200.
- Leung, J.Y. (1989). "Bin Packing with Restricted Piece Sizes." *Information Processing Letters*, Vol. 31, pp. 145-149, North-Holland, May.
- Liu, L.Y. (1991). *COOPS - Construction Object-Oriented Process Simulation System*. Ph.D. Dissertation, Department of Civil Engineering, The University of Michigan, Ann Arbor, MI.
- Lundberg, E.J., Beliveau, Y.J. (1989). "Automated Laydown Yard Control System-ALYC." *Journal of Construction Engineering and Management*, ASCE, Vol. 115, No. 4, pp. 535-544, December.
- Maimon, O., Nof, S. (1984). "Analysis and Control of Flexible Multi-Robot Operations." *Proceedings International Industrial Engineering Conference*, Institute of Industrial Engineers, pp. 17-28.
- Malakooti, B. (1987). "Computer-Aided Facility Layout Selection (CAFLAS) with Applications to Multiple Criteria Manufacturing Planning Problems." *Large Scale Systems*, No. 2, pp. 109-123.
- Malmborg, C.J., Simons, G.R., Agee, M.H. (1986). "Knowledge Engineering Approaches to Material Handling Equipment Specification." *Proceedings International Industrial Engineering Conference*, Institute of Industrial Engineers, pp. 148-151.
- Marcus, S., Stout, J., McDermott, J. (1988). "VT: An Expert Elevator Designer That Uses Knowledge-Based Backtracking." *Artificial Intelligence Magazine*, Vol. 9, No. 1, pp. 95-111, Spring.

- Mawdesley, M.J., Cullingford, G., Haddadi, A. (1988). "Site Layout and Resource Scheduling—An Approach to Modelling Movement around Sites." *Proceedings 5th International Symposium on Robotics in Construction*, ISARC, Tokyo, Japan, pp. 391-399.
- McDermott, J. (1991). "Preliminary Steps Toward a Taxonomy of Problem-Solving Methods." *Automating Knowledge Acquisition for Expert Systems*, Kluwer Academic Publishers, Draft.
- Means (1988). *Building Construction Cost Data*. 4th Annual Edition, R.S. Means Company, Inc., Construction Consultants & Publishers, Kingston, MA.
- Meller, R.D. (1992). *Layout Algorithms for Single and Multiple Floor Facilities*. Ph.D. Dissertation, Department of Industrial & Operations Engineering, University of Michigan, Ann Arbor, MI.
- Messner, J.I., Sanvido, V.E., Ikeda, M. (1994). "Selecting Precast Concrete Building Systems." *Proceedings 1st Congress in Computing in Civil Engineering*, ASCE, pp. 369-371.
- Mittal, S., Dym, C.L., Morjaria, M. (1986). "PRIDE: An Expert System for the Design of Paper Handling Systems." *Computer*, IEEE, Vol. 19, No. 7, pp. 102-114, July.
- Montreuil, B. (1987). "Integrated Design of Cell Layout, Input/Output, Station Configuration, and Flow Network of Manufacturing Systems." *Chapter of Intelligent and Integrated Manufacturing Analysis and Synthesis*, ASME, PED-Vol. 25, pp. 315-326, December.
- Montreuil, B. (1990). "Requirements for Representation of Domain Knowledge in Intelligent Environments for Layout Design." *Computer-Aided Design*, Vol. 22, No. 2, pp. 97-108, March.
- Montreuil, B., Banerjee, P. (1988). "Object Knowledge Environment for Manufacturing Systems Layout Design." *International Journal of Intelligent Systems*, Vol. 3, pp. 399-410.
- Montreuil, B., Nof, S.Y. (1988). "Approaches for Logical vs. Physical Design of Intelligent Production Facilities." Chapter of *Recent Developments in Production Research*, Ed. by E. Mittal, Elsevier.

- Montreuil, B., Ratliff, H.D. (1988). "Optimizing the Location of Input/Output Stations Within Facilities Layout." *Engineering Costs and Production Economics*, Elsevier Science Publishers, Vol. 14, pp. 177-187.
- Montreuil, B., Ratliff, H.D. (1989). "Utilizing Cut Trees as Design Skeletons for Facility Layout." *IIE Transactions*, Vol. 21, No. 2, pp. 136-143, June.
- Montreuil, B., Ratliff, H.D., Goetschalckx, M. (1987). "Matching Based Interactive Facility Layout." *IIE Transactions*, Vol. 19, No. 3, pp. 271-279, September.
- Montreuil, B., Ratliff, H.D., Goetschalckx, M. (1990). "A Modeling Framework for Integrating Layout Design and Flow Network Design". *Proceedings of the Materials Handling Research Colloquium*, pp. 43-58, Hebron, KY.
- Montreuil, B., Stump, D.J. (1988). "Knowledge Preanalysis for Manufacturing Systems Layout." *Proceedings International Industrial Engineering Conference*, Institute of Industrial Engineers, pp. 180-188.
- Montreuil, B., Venkatadri, U. (1988). "From Gross To Net Layouts: An Efficient Design Model." *Working Draft 88-56*, Faculté des Sciences de l'Administration, Université Laval, Québec, Canada.
- Montreuil, B., Venkatadri, U. (1991). "Strategic Interpolative Design of Dynamic Manufacturing Systems Layouts." *Management Science*, Vol. 37, No. 6, pp. 682-693.
- Montreuil, B., Venkatadri, U., Ratliff, H.D. (1989). "Generating a Layout From a Design Skeleton." *Working Draft 89-01*, Faculté des Sciences de l'Administration, Université Laval, Québec, Canada.
- Morad, A.A. (1990). *Geometric-Based Reasoning System for Project Planning Utilizing AI and CAD Technologies*. Ph.D. Dissertation, Department of Civil Engineering, Virginia Polytechnic Institute and State University, Blacksburg, Virginia.
- Morad, A.A., Beliveau, Y.J. (1991). "Knowledge-Based Planning System." *Journal of Construction Engineering and Management*, ASCE, Vol. 117, No. 1, pp. 1-12, March.
- Morad, A.A., Beliveau, Y.J., Fransisco, V.D., Dixit, S.S. (1992). "Path-Finder: AI-Based Path Planning System." *Journal of Computing in Civil Engineering*, ASCE, Vol. 6, No. 2, pp. 114-128, April.

- Moselhi, O., El-Rayes, K. (1993). "An OOP Model for Scheduling of Repetitive Projects." *Proceedings 5th International Conference in Computing in Civil & Building Engineering*, ASCE, Vol. 2, pp. 939-946, New York, NY, June.
- Muehlhausen, F.B. (1991). "Construction Site Utilization: Impact of Materials Movement and Storage on Productivity and Cost." *American Association of Cost Engineers Transactions*, pp. L.2.1-L.2.9.
- Muther, R. (1973). *Sytematic Layout Planning*. Second Edition, Cahners Books, Boston, MA.
- Muther, R. (1979-80). *Systematic Planning of Industrial Facilities*. Management and Industrial Research Application, Kansas City, MO.
- Navinchandra, D.S., Logcher, R.D. (1988). "GHOST: Project Network Generator." *Journal of Computing in Civil Engineering*, ASCE, Vol. 2, No. 3, pp. 239-253, July.
- Neil, J.M. (1982). *Steam-Electric Generating Station Construction*. Courtesy of M-K Power Group, pp. 7-11 to 7-29, June.
- Newell, A. (1969). "Heuristic Programming: Ill-structured Problems." *Progress in Operations Research*, New York, John Wiley, pp. 360-414.
- Nicol, L.M., Hollier, R.H. (1983). "Plant Layout In Practice." *Material Flow*, Vol. 1, pp. 177-188.
- Nolan, K.J. (1990). *Masonry & Concrete Construction*. Craftsman Book Company, Carlsbad, CA.
- Nunnally S.W. (1977). *Managing Construction Equipment*. Prentice-Hall, Inc., Englewood Cliffs, NJ.
- Odeh, A.M. (1992). *CIPROS Knowledge-Based Construction Integrated Project and Process Planning Simulation System*. Ph.D. Dissertation, Department of Civil Engineering, The University of Michigan, Ann Arbor, MI, June.
- Padilla, E.M., Carr, R.I. (1991). "Resource Strategies for Dynamic Project Management." *Journal of Construction Engineering and Management*, ASCE, Vol. 117, No. 2, pp. 279-293, June.

- Parsons, R.M., Pachuta, J.D. (1980). "System For Material Movement To Work Areas." *Journal of The Construction Division, ASCE*, Vol. 106, No. CO1, pp. 55-71, March.
- Paulson, B.C. (1973). "Project Planning and Scheduling: Unified Approach." *Journal of the Construction Division, Proceedings of the American Society of Civil Engineers*, Vol. 99, No. CO1, pp. 45-58, July.
- Perera, S. (1983). "Resource Sharing in Linear Construction." *Journal of Construction Engineering and Management, ASCE*, Vol. 109, No. 1, pp. 102-111, March.
- Primavera (1992). *Primavera Project Planner® 5.0 Reference*. Primavera Systems, Inc.. Bala Cynwyd, PA.
- Pulat, B.M. (1984). "Microcomputer Software for Evaluating Risk Due to Manual Materials Handling Tasks." *Proceedings International Industrial Engineering Conference*, Institute of Industrial Engineers, pp. 342-349.
- Rad, P.F. (1982). "A Graphic Approach to Construction Job-Site Planning." *Cost Engineering*, Vol. 24, No. 4, pp. 211-217.
- Ramanan, P. (1989). "Average-Case Analysis of the Smart Next Fit Algorithm." *Information Processing Letters*, Vol. 31, pp. 221-225, North-Holland, June.
- Rasdorf, W.J., Abudayyeh, O.Y. (1991). "Cost- and Schedule-Control Integration Issues and Needs." *Journal of Construction Engineering and Management, ASCE*, Vol. 117, No. 3, pp. 486-502, September.
- Rasdorf, W.J., Herbert, M.J. (1990a). "Bar Coding in Construction Engineering." *Journal of Construction Engineering and Management, ASCE*, Vol. 116, No. 2, pp. 261-280, June.
- Rasdorf, W.J., Herbert, M.J. (1990b). "Automated Identification Systems—Focus on Bar Coding." *Journal of Computing in Civil Engineering, ASCE*, Vol. 4, No. 3, pp. 279-296, July.
- Rich, E., Knight, K. (1991). *Artificial Intelligence*. Second edition, McGraw-Hill, Inc.
- Riley, D. R. (1993). "Construction Material Handling: A Review of Research and Industry Practice." *Unpublished Paper, CIC Laboratory*, Department of Architectural Engineering, Pennsylvania State University, University Park, PA.

- Riley, D.R. (1994). *Modeling the Space Behavior of Construction Activities*. Ph.D. Dissertation, Department of Architectural Engineering, Pennsylvania State University, University Park, PA.
- Rodriguez-Ramos, W.E. (1982). *Quantitative Techniques for Construction Site Layout Planning*. Ph.D. Dissertation, University of Florida, Gainesville, Florida.
- Rodriguez-Ramos, W.E., Francis, R.L. (1983). "Single Crane Location Optimization." *Journal of Construction Engineering and Management*, ASCE, Vol. 109, No. 4, pp. 387-397, December.
- Rosenblatt, M.J. (1979). "The Facilities Layout Problem: A Multi-Goal Approach." *International Journal of Production Research*, Vol. 17, No. 4, pp. 323-332.
- Rosenblatt, M.J. (1986). "The Dynamics of Plant Layout." *Management Science*, Vol. 32, No. 1, pp. 76-86.
- Rosenblatt, M.J., Enyan, A. (1989). "Deriving the Optimal Boundaries for Class-Based Automatic Storage/Retrieval Systems." *Management Science*, Vol. 35, No. 12, pp. 1519-1524, December.
- Rosenblatt, M.J., Rothblum, U.G. (1990). "On the Single Resource Capacity Problem for Multi-Item Inventory Systems." *Operations Research*, Vol. 38, No. 4, pp. 686-693, July-August.
- Royer, K. (1986). "The Federal Government and the Critical Path." *Journal of Construction Engineering and Management*, ASCE, Vol. 112, No. 2, pp. 220-225, June.
- Russell, A.D., McGowan, N. (1993). "Linear Scheduling: A Practical Implementation." *Proceedings 5th International Conference in Computing in Civil & Building Engineering*, ASCE, Vol. 1, pp. 279-286, New York, NY, June.
- Sacerdoti, E.D. (1973). "Planning in a Hierarchy of Abstraction Spaces." *IJCAI*, Stanford, CA, pp. 412-422, August.
- Sacerdoti, E.D. (1975). "The Nonlinear Nature of Plans." *IJCAI*, Tbilisi, GA, pp. 206-214, September.

- Sahni, S., Gonzalez, T. (1976). "P-complete Approximation Problem." *Journal of Association for Computing Machinery*, Vol. 23, No. 3, pp. 555-565.
- Sanvido, V.E., Kuntz, K.A., Lynch, T.D., Messner, J.I., Riley, D.R. (1994). "Information Modeling and Applications." *Proceedings 1st Congress in Computing in Civil Engineering*, ASCE, Vol. 1, pp. 787-789.
- Sanvido, V.E., Paulson, B.C. (1991). "Site Analysis Using Controller-Function Charts." *Journal of Construction Engineering and Management*, ASCE, Vol. 117, No. 2, pp. 226-241, June.
- Sathi, A., Fox, M.S., Greenberg, M. (1985). "Representation of Activity Knowledge for Project Management." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. Pami-7, No. 5, pp. 531-552
- Seehof, J.M., Evans, W.O. (1967). "Automated Layout Design Program." *Journal of Industrial Engineering*, Vol. 18, No. 12, pp. 690-695.
- Seibert, J.E., Evans, G.W. (1991). "Time-Constrained Resource Leveling." *Journal of Construction Engineering and Management*, ASCE, Vol. 117, No. 3, pp. 503-520, September.
- Shah, K.A., Farid, F., Baugh Jr., J.W. (1993). "Optimal Resource Leveling Using Integer-Linear Programming." *Proceedings 5th International Conference in Computing in Civil & Building Engineering*, ASCE, Vol. 1, pp. 501-508, New York, NY, June.
- Sirajuddin, A., Mawdesley, M.J. (1989). "A Knowledge Based System For Recognition of Work Required for Construction." *Conference Civil-Comp*, pp. 83-91, London, U.K.
- Skolnick, J.F. (1993). "A CAD-Based Construction Simulation Tool Kit for Construction Planning." *Proceedings 5th International Conference in Computing in Civil & Building Engineering*, ASCE, Vol. 1, pp. 119-124, New York, NY, June.
- Smith, D.M. (1987). *An Investigation of the Space Constraint Problem in Construction Planning*. MS Thesis, Virginia Polytechnic Institute and State University, Blacksburgh, VA.

- Smithers, T. (1989). "AI-Based Design Versus Geometry-Based Design or Why Design Cannot Be Supported by Geometry Alone." *Computer-Aided Design*, Vol. 21, No. 3, pp. 141-150, April.
- Stallman, R.M., Sussman, G.J. (1977). "Forward Reasoning and Dependency-Directed Backtracking in a System for Computer-Aided Circuit Analysis." *Artificial Intelligence*, Vol. 9, pp. 135-196.
- Steels, L. (1990). "Components of Expertise." *Artificial Intelligence Magazine*, Summer.
- Stefik, M. (1981). "Planning With Constraints (MOLGEN: Part 1)." *Artificial Intelligence*, Vol. 16, pp. 111-140.
- Stukhart, G. (1987). *Project Materials Management Handbook*. The Construction Industry Institute, Materials Management Task Force, The University of Texas at Austin, Austin, Texas.
- Sussman, G.J., Steele, G.L. (1980). "CONSTRAINTS-A Language for Expressing Almost-Hierarchical Descriptions." *Artificial Intelligence Journal*, North Holland, Vol. 14, pp. 1-39.
- Tamimi, S., Diekmann, J. (1988). "Soft Logic in Network Analysis." *Journal of Computing in Civil Engineering*, ASCE, Vol. 2, No. 3, pp. 289-254, July.
- Thabet, W.Y., Beliveau, Y.J. (1993). "A Model to Quantify Work Space Availability for Space Constrained Scheduling within a CAD Environment." *Proceedings 5th International Conference in Computing in Civil & Building Engineering*, ASCE, pp. 110-116, New York, NY.
- Thabet, W.Y., Beliveau, Y.J. (1994a). "Modeling Work Space to Schedule Repetitive Floors in Multi-Story Projects." *Journal of Construction Engineering and Management*, ASCE, Vol. 120, No. 1, pp. 96-116, March.
- Thabet, W.Y., Beliveau, Y.J. (1994b). "Alternative Decisions for Space-Based Scheduling in Multi-Story Projects." *Proceedings 1st Congress in Computing in Civil Engineering*, ASCE, Vol. 2, pp. 1164-1171.

- Thabet, W.Y., Beliveau, Y.J. (1994c). "Representation of Vertical Constraints Knowledge for Scheduling Multi-Story Projects." *Proceedings 1st Congress in Computing in Civil Engineering*, ASCE, Vol. 1, pp. 349-354.
- Thabet, W.Y., Morad, A.A. (1994). "An Integrated Computer Model for Project Schedule Updating." *Proceedings 1st Congress in Computing in Civil Engineering*, ASCE, Vol. 2, pp. 743-750.
- Tommelein, I.D. (1989). *SightPlan: An Expert System that Models and Augments Human Decision-Making for Designing Construction Site Layouts*. Ph.D dissertation, Department of Civil Engineering, Stanford University, Stanford, CA, August.
- Tommelein, I.D. (1991). "Site Layout: Where should it go?" *Proceedings of the Construction Congress*, ASCE, pp. 632-637, Cambridge, MA, April.
- Tommelein, I.D. (1993). "Planning and Controlling On-Site Materials Handling Through Space Scheduling." *Proceedings 3rd International Conference on Fossil Plant Construction*, Electric Power Research Institute, Edison Electric Institute, Palm Beach, FL, October.
- Tommelein, I.D. (1994a). "Materials Handling and Site Layout Control." *Proceedings 11th International Symposium on Automation and Robotics in Construction*, Brighton, England, UK, pp. 297-304.
- Tommelein, I.D. (1994b). "MoveCapPlan: An Integrated System for Planning and Controlling Construction Material Laydown and Handling." *Proceedings 1st Congress in Computing in Civil Engineering*, ASCE, Vol. 2, pp. 1172-1179.
- Tommelein, I.D. (1994c). *MovePlan 1.5 User's Manual*. Technical Report No. 92-26, Department of Civil and Environmental Engineering, The University of Michigan, Ann Arbor, MI.
- Tommelein, I.D., Castillo, J.G., Zouein, P.P. (1992a). "Space-Time Characterization for Resource Management on Construction Sites." *Proceedings 8th Conference of Computing in Civil Engineering*, ASCE, pp. 1042-1049, New York, NY.
- Tommelein, I.D., Dzeng, R.J., Zouein, P.P. (1993). "Exchanging Layout and Schedule Data in a Real-Time Distributed Environment." *Proceedings 5th International*

*Conference in Computing in Civil & Building Engineering*, ASCE, pp. 947-954, New York, NY.

Tommelein, I.D., Hayes-Roth, B., Levitt, R.E. (1992b). "Altering the SightPlan Knowledge-based Systems." *Journal of Artificial Intelligence in Engineering, Design, and Manufacturing*, AI EDAM, Vol.6, No. 1, pp. 19-37.

Tommelein, I.D., Levitt, R.E., Hayes-Roth, B., Confrey T. (1991). "SightPlan Experiments: Alternative Strategies for Site Layout Design." *Journal of Computing in Civil Engineering*, ASCE, Vol. 5, No. 1, pp. 42-63.

Tommelein, I.D., Levitt, R.E., Hayes-Roth, B. (1992c). "SightPlan Model for Site Layout." *Journal of Construction Engineering and Management*, ASCE, Vol. 118, No. 4, pp. 749-766.

Tommelein, I.D., Zouein, P.P. (1992). "Activity-Level Space Scheduling." *Proceedings 9th International Symposium on Automation and Robotics in Construction*, ISARC, sponsored by Japan Industrial Robot Association, JIRA, Tokyo, Japan, pp. 411-420, June.

Tommelein, I.D., Zouein, P.P. (1993a). "Interactive Dynamic Layout Planning." *Journal of Construction Engineering and Management*, ASCE, Vol. 119, No. 2, pp. 266-287, New York, NY.

Tommelein, I.D., Zouein, P.P. (1993b). "Space Scheduling for Construction Progress Planning and Control." *Proceedings 10th International Symposium on Automation and Robotics in Construction*, ISARC, Construction Industry Institute, Austin, TX, Elsevier Science Publishers, pp. 415-422.

Tompkins, J.A., White, J.A. (1984). *Facilities Planning*. John Wiley & Sons, Inc., New York, NY.

Tracey, R.G., Fling, R.S. (1989). "Rehabilitation Strategies." *Concrete International*, pp.41-45, September.

Tsai, R.D., Malmstrom, E.M., Meeks, H.D. (1985). "Robotic Unitization of Warehouse Pallet Loads." *Proceedings International Industrial Engineering Conference*, Institute of Industrial Engineers, pp. 222-227.

- Urban, Y.L. (1992). "Computational Performance and Efficiency of Lower-bound Procedures for the Dynamic Facility Layout Problem." *European Journal of Operational Research*, Vol. 57, No. 2, pp. 271-279.
- Van Camp, D.J., Carter, M.W., Vannelli, A. (1992). "A Nonlinear Optimization Approach For Solving Facility Layout Problems." *European Journal of Operational Research*, Vol. 57, No. 2, pp. 174-189.
- Vollman, T.E., Buffa, E.S. (1966). "The Facilities Layout Problem In Perspective." *Management Science*, Vol. 12, No. 10, B-450 to B-468, June.
- Wakefield, R.R., O'Brien, J.B. (1994). "A 'Virtual Reality' Type Simulation System for Construction Automation System Development." *Proceedings 11th International Symposium on Automation and Robotics in Construction*, ISARC, Brighton, England, U.K., Elsevier Science Publishers, pp. 239-247.
- Warszawski, A. (1973a). "Analysis of Transportation Methods in Construction." *Journal of the Construction Division*, ASCE, Vol. 99, No. CO1, pp. 191-202, July.
- Warszawski, A. (1973b). "Multi-dimensional Location Problems." *Operational Research Quarterly*, Vol. 24, No. 2, pp. 165-179.
- Warszawski, A. (1987). "An Expert System for Crane Selection and Location." *Proceedings 4th International Symposium on Robotics and Artificial Intelligence in Building Construction*, pp. 313-324, Haifa.
- Warszawski, A. (1990). "Expert Systems for Crane Selection." *Construction Management and Economics*, No. 8, pp. 179-190.
- Warszawski, A., Peer, S. (1973). "Optimizing the Location of Facilities on a Building Site." *Operational Research Quarterly*, Vol. 24, No. 1, pp. 35-44.
- Wiest, J.D. (1964). "Some Properties of Schedules for Large Projects With Limited Resources." *Operations Research*, Vol. 12, pp. 395-418, May-June.
- Wiest, J.D. (1967). "A Heuristic Model for Scheduling Large Projects With Limited Resources." *Management Science*, Vol. 13, No. 6, pp. 359-377, February.
- Wiest, J.D., Levy, F.K. (1977). *A Management Guide to Pert/CPM: with Gert/PDM/DCPM and other Networks*. Prentice-Hall, Inc., New Jersey, 2nd Ed.

- Wilkie, D., Tommelein, I.D. (1992). *Michigan Stadium Renovation*. Case Study, Civil and Environmental Engineering Department, The University of Michigan, Ann Arbor, MI.
- Yeh, I.C. (1995). "Construction-Site Layout Using Annealed Neural Network." *Journal of Computing in Civil Engineering*, ASCE, Vol. 9, No. 3, pp. 201-208, July.
- Zabilski, R.J., Hall, H.E. (1989). "Presented in 3-D." *Civil Engineering*, ASCE, pp. 48-50, June.
- Zouein, P.P., Tommelein, I.D. (1992). "MovePlan: Allocating Space During Scheduling." *Proceedings CIB 92 World Building Congress*. National Research Council Ottawa, pp. 608-609, Montreal, Canada, May.
- Zouein, P.P., Tommelein, I.D. (1993). "Space Schedule Construction." *Proceedings 5th International Conference in Computing in Civil & Building Engineering*, ASCE, Vol. 2, pp. 1770-1777, New York, NY, June.
- Zouein, P.P., Tommelein, I.D. (1994a). "Automating Dynamic Layout Construction." *Proceedings 11th International Symposium on Automation and Robotics in Construction*, ISARC, Brighton, England, U.K., Elsevier Science Publishers, pp. 409-416.
- Zouein, P.P., Tommelein, I.D. (1994b). "Time-Space Tradeoff Strategies For Space-Schedule Construction." *Proceedings 15th Congress in Computing in Civil Engineering*, ASCE, Vol. 2, pp. 1180-1187.