BATCH SCHEDULING OF INCOMPATIBLE JOBS ON A SINGLE REACTOR WITH DYNAMIC ARRIVALS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

GEDİZ KORKMAZ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE

IN

INDUSTRIAL ENGINEERING

JUNE 2004

Approval of the Graduate School of Natural and Applied Sciences

Prof. Dr. Canan Özgen
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science/Arts / Doctor of Philosophy.

Prof. Dr. Çağlar Güven
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Prof.Dr. Ömer Kırca Co-Supervisor	Prof.Dr. Sinan Kayalıgil Supervisor

Examining Committee Members

Prof. Dr. Meral Azizoğlu	METU	
Prof. Dr. Sinan Kayalıgil	METU	
Prof. Dr. Ömer Kırca	METU	
Asst. Prof. Dr. Haldun Süral	METU	
Asst. Prof. Dr. Mehmet R. Taner	Bilkent U.	

ABSTRACT

BATCH SCHEDULING OF INCOMPATIBLE JOBS ON A SINGLE REACTOR WITH DYNAMIC ARRIVALS

Korkmaz, Gediz

M.Sc., Department of Industrial Engineering

Supervisor : Prof. Dr. Sinan Kayalıgil

Co-Supervisor: Prof. Dr. Ömer Kırca

June 2004, 158 pages

In this study, a single machine batch-scheduling problem with incompatible jobs and dynamic arrivals is examined. The objective function is the minimization of the total flow time of the jobs. For solving problems a case specific branch and bound algorithm with a heuristic upper bound scheme and two alternative lower bound procedures is used. An extensive computational experiment is conducted to investigate the effects of certain parameters on the computation time. For the most difficult parameter combination branch and bound algorithm can solve the problems about 25 jobs with 4 different job types in a 10 minutes time on average. For the problem types with higher number of jobs and the most difficult parameter combination proposed upper bound heuristic can be used to obtain near optimal solutions.

Keywords: Batch scheduling, dynamic arrival, branch and bound, incompatible jobs.

ÖZ

DİNAMİK VARIŞLI ORTAMLARDA GEÇİMSİZ İŞLERİN TEK REAKTÖRDE GRUP ÇİZELGELENMESİ

Korkmaz, Gediz
Yüksek lisans, Endüstri Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. Sinan Kayalıgil
Ortak Tez Yöneticisi: Prof. Dr. Ömer Kırca

Haziran 2004, 158 sayfa

Bu çalışmada farklı zamanlarda sipariş verilen (dinamik varışlı) uyumsuz işlerin tek reaktörde parti çizelgelenmesi yöntemiyle işlenmeleri incelenmiştir. Çalışmadaki amaç fonksiyonu işlerin toplam akış sürelerinin enazlanmasıdır. Bu tip problemleri çözebilmek için problem şartları dikkate alınarak hazırlanmış özel bir dal-sınır algoritması kullanılmıştır. Bu algoritma sezgisel bir üst sınır bulma yöntemi ve iki farklı alt sınır bulma yöntemi ile desteklenmiştir. Belirli parametrelerin hesaplama süresi üzerindeki etkilerini görmek için geniş bir sayısal analiz yapılmıştır. En zor parametre kombinasyonu geçerliyken 25 iş ve 4 çeşit işle yapılan denemlerde çözüm zamanının ortlama 10 dakika civarında olduğu görülmüştür. Daha yüksek iş sayısının bulunduğu ve en zor parametre kombinasyonunun geçerli olduğu durumlarda ise üst sınır bulma yöntemi kullanılabilir.

Anahtar Kelimeler: Parti çizelgeleme, dinamik varış,dal-sınır, uyumsuz işler.

ACKNOWLEDGEMENTS

I would like to thank to Prof.Dr Sinan Kayalıgil and Prof.Dr. Ömer Kırca for their guidence and continual support throughout the study.

Special thanks go to Prof. Dr. Meral Azizoğlu for her help to reach an important study made on the subject. Thanks to my friend Melih for his critical helps and tips during the execution of the program.

I thank to my brother Ilgaz and Deniz for their valuable support, help and encouragement. Finally I would like to thank my parents, who always supported me with their endless patience and love.

TABLE OF CONTENTS

ABSTRACT IV

ÖZ V

ACKNOWLEDGEMENTS...VI

TABLE OF CONTENTS VII

LIST OF TABLES IIX

LIST OF FIGURES XVII

1.INTRODUCTION 1

2.AN OVERVIEW...6

2.1 The Literature on Family Scheduling Models.. 6

2.2 The Literature on Batch Scheduling Models.. 8

3.PROBLEM DEFINITION, PROPERTIES OF THE PROBLEM AND SOLUTION APPROACH 15

3.1 Assumptions.. 15

3.2 Properties of the Problem... 19

3.3 Approach for Selecting Available Jobs from Families 20

3.4 Upper Bound Heuristic.. 21

3.5 Lower Bound Procedures 25

3.5.1 Family Independent Lower Bound Procedure (FILB) 26

3.5.2 Family Dependent Lower Bound Procedure (FDLB) 35

3.6 Branch and Bound Algorithm 43

4.RESULTS AND DISCUSSION 45

4.1 Experimental Design 45

4.2 Performance Measures 46

4.3 Results and Discussion 47

4.3.1 Total Number of Nodes and Solution Times of the Problems 47

4.3.2 Terminated problems 57

4.3.3 Performances of Upper and Lower Bound Procedures 58

4.3.4 Performance of Proposed Branch and Bound Algorithm 68

5.CONCLUSION 70

REFERENCES 74

APPENDIX A 77

APPENDIX B 120

APPENDIX C 123

APPENDIX D 126

APPENDIX E 128

APPENDIX F 134

APPENDIX G 142

APPENDIX H 147

LIST OF TABLES

Table 3.1: Data of the sample problem of upper bound heuristic 23
Table 3.2: Data of the sample problem of family independent lower bound procedure 30
Table 3.3: Rearranged pattern of the consolidated jobs through interval [32,42) for LB[1,1] 31
Table 3.4: Rearranged pattern of the consolidated jobs through interval [42,52) for LB[1,1] 31
Table 3.5: Rearranged pattern of the consolidated jobs through interval [6,16) for LB[1,2] 33
Table 3.6: Rearranged pattern of the consolidated jobs through interval [16,26) for LB[1,2] 33
Table 3.7: Rearranged pattern of the consolidated jobs through interval [26,36) for LB[1,2] 34
Table 3.8: Rearranged pattern of the consolidated jobs through interval [36,46) for LB[1,2] 34
Table 3.9: Data of the sample problem of family dependent lower bound procedure 39
Table 3.10: Rearranged pattern of the jobs through interval [12,22) for LB[1,1] 39
Table 3.11: Rearranged pattern of the jobs through interval [22,32) for LB[1,1] 40
Table 3.12: Rearranged pattern of the jobs through interval [32,42) for LB[1,1] 40
Table 3.13: Rearranged pattern of the jobs through interval [42,52) for LB[1,1] 41
Table 3.14: Rearranged pattern of the jobs through interval [6,16) for LB[1,3]. 42
Table 4.1: Average completion time of a problem with 50 jobs. 48
Table 4.2: Average of total number of nodes created for a problem with 50 jobs 49
Table 4.3: Performance measures of problem set J50/F2/S/A 50
Table 4.4: Performance measures of problem set J50/F2/B/L 50
Table 4.5: Average completion time of a problem with 33 jobs. 51
Table 4.6: Average of total number of nodes created for a problem with 33 jobs 52
Table 4.7: Average of completion time of a problem with 25 jobs.................... 53
Table 4.8: Average of total number of nodes created for a problem with 25 jobs 54

Table 4.9: Average of completion time of a problems with 20 jobs 54
Table 4.10: Average of total number of nodes created for a problem with 20 jobs 55
Table 4.11: Average of completion time of a problems with 16 jobs 55
Table 4.12: Average of total number of nodes created for a problem with 16 jobs 56
Table 4.13: Allocation of the problems, whose optimal solutions and upper bound values are equal, according to the volumes and interarrival times of jobs. 60
Table 4.14:Average percent deviation of upper bound from the optimal solution for a 50 arrivals problem 60
Table 4.15:Average percent deviation of upper bound from the optimal solution for a 33 arrivals problem 61
Table 4.16:Average percent deviation of upper bound from the optimal solution for a 16 arrivals problem 62
Table 4.17:Average percent deviation of the minimum lower bound at the root node from the optimal solution for a 50 arrivals problem.......................... 63
Table 4.18: Average percent deviation of the minimum lower bound at the root node from the optimal solution for a 33 arrivals problem.......................... 64
Table 4.19: Average percent deviation of the minimum lower bound at the root node from the optimal solution for a 25 arrivals problem.......................... 64
Table 4.20: Average percent deviation of the minimum lower bound at the root node from the optimal solution for a 20 arrivals problem.......................... 65
Table 4.21: Average percent deviation of the minimum lower bound at the root node from the optimal solution for a 16 arrivals problem.......................... 65
Table A.1: Detailed performance measures of branch and bound algorithm for problem set J50/F2/S/F... 78
Table A.2: Detailed performance measures of branch and bound algorithm for problem set J50/F1/S/F... 78
Table A.3: Detailed performance measures of branch and bound algorithm for problem set J50/F2/S/A... 78
Table A.4: Detailed performance measures of branch and bound algorithm for problem set J50/F1/S/A... 79
Table A.5: Detailed performance measures of branch and bound algorithm for problem set J50/F2/S/L 79
Table A.6: Detailed performance measures of branch and bound algorithm for problem set J50/F1/S/L 79
Table A.7: Detailed performance measures of branch and bound algorithm for problem set J50/F2/B/F 80
Table A.8: Detailed performance measures of branch and bound algorithm for problem set J50/F1/B/F 80
Table A.9: Detailed performance measures of branch and bound algorithm for problem set J50/F2/B/A 80
Table A.10: Detailed performance measures of branch and bound algorithm for problem set J50/F1/B/A 81

Table A.11: Detailed performance measures of branch and bound algorithm for problem set J50/F2/B/L 81
Table A.12: Detailed performance measures of branch and bound algorithm for problem set J50/F1/B/L 81
Table A.13: Detailed performance measures of branch and bound algorithm for problem set J50/F2/M/F 82
Table A.14: Detailed performance measures of branch and bound algorithm for problem set J50/F1/M/F 82
Table A.15: Detailed performance measures of branch and bound algorithm for problem set J50/F2/M/A 82
Table A.16: Detailed performance measures of branch and bound algorithm for problem set J50/F1/M/A 83
Table A.17: Detailed performance measures of branch and bound algorithm for problem set J50/F2/M/L 83
Table A.18: Detailed performance measures of branch and bound algorithm for problem set J50/F1/M/L 83
Table A.19: Detailed performance measures of branch and bound algorithm for problem set J33/F1/S/F 84
Table A.20: Detailed performance measures of branch and bound algorithm for problem set J33/F2/S/F 84
Table A.21: Detailed performance measures of branch and bound algorithm for problem set J33/F3/S/F 84
Table A.22: Detailed performance measures of branch and bound algorithm for problem set J33/F1/S/A 85
Table A.23: Detailed performance measures of branch and bound algorithm for problem set J33/F2/S/A 85
Table A.24: Detailed performance measures of branch and bound algorithm for problem set J33/F3/S/A 85
Table A.25: Detailed performance measures of branch and bound algorithm for problem set J33/F1/S/L 86
Table A.26: Detailed performance measures of branch and bound algorithm for problem set J33/F2/S/L 86
Table A.27: Detailed performance measures of branch and bound algorithm for problem set J33/F3/S/L 86
Table A.28: Detailed performance measures of branch and bound algorithm for problem set J33/F1/B/F 87
Table A.29: Detailed performance measures of branch and bound algorithm for problem set J33/F2/B/F 87
Table A.30: Detailed performance measures of branch and bound algorithm for problem set J33/F3/B/F 87
Table A.31: Detailed performance measures of branch and bound algorithm for problem set J33/F1/B/A 88
Table A.32: Detailed performance measures of branch and bound algorithm for problem set J33/F2/B/A 88
Table A.33: Detailed performance measures of branch and bound algorithm for problem set J33/F3/B/A 88

Table A.34: Detailed performance measures of branch and bound algorithm for problem set J33/F1/B/L 89
Table A.35: Detailed performance measures of branch and bound algorithm for problem set J33/F2/B/L 89
Table A.36: Detailed performance measures of branch and bound algorithm for problem set J33/F3/B/L 89
Table A.37: Detailed performance measures of branch and bound algorithm for problem set J33/F1/M/F 90
Table A.38: Detailed performance measures of branch and bound algorithm for problem set J33/F2/M/F 90
Table A.39: Detailed performance measures of branch and bound algorithm for problem set J33/F3/M/F 90
Table A.40: Detailed performance measures of branch and bound algorithm for problem set J33/F1/M/A 91
Table A.41: Detailed performance measures of branch and bound algorithm for problem set J33/F2/M/A 91
Table A.42: Detailed performance measures of branch and bound algorithm for problem set J33/F3/M/A 91
Table A.43: Detailed performance measures of branch and bound algorithm for problem set J33/F1/M/L 92
Table A.44: Detailed performance measures of branch and bound algorithm for problem set J33/F2/M/L 92
Table A.45: Detailed performance measures of branch and bound algorithm for problem set J33/F3/M/L 92
Table A.46: Detailed performance measures of branch and bound algorithm for problem set J25/F2/S/F 93
Table A.47: Detailed performance measures of branch and bound algorithm for problem set J25/F3/S/F 93
Table A.48: Detailed performance measures of branch and bound algorithm for problem set J25/F4/S/F 93
Table A.49: Detailed performance measures of branch and bound algorithm for problem set J25/F2/S/A 94
Table A.50: Detailed performance measures of branch and bound algorithm for problem set J25/F3/S/A 94
Table A.51: Detailed performance measures of branch and bound algorithm for problem set J25/F4/S/A 94
Table A.52: Detailed performance measures of branch and bound algorithm for problem set J25/F2/S/L 95
Table A.53: Detailed performance measures of branch and bound algorithm for problem set J25/F3/S/L 95
Table A.54: Detailed performance measures of branch and bound algorithm for problem set J25/F4/S/L 95
Table A.55: Detailed performance measures of branch and bound algorithm for problem set J25/F2/B/F 96
Table A.56: Detailed performance measures of branch and bound algorithm for problem set J25/F3/B/F 96

Table A.57: Detailed performance measures of branch and bound algorithm for problem set J25/F4/B/F ... 96
Table A.58: Detailed performance measures of branch and bound algorithm for problem set J25/F2/B/A ... 97
Table A.59: Detailed performance measures of branch and bound algorithm for problem set J25/F3/B/A ... 97
Table A.60: Detailed performance measures of branch and bound algorithm for problem set J25/F4/B/A ... 97
Table A.61: Detailed performance measures of branch and bound algorithm for problem set J25/F2/B/L ... 98
Table A.62: Detailed performance measures of branch and bound algorithm for problem set J25/F3/B/L ... 98
Table A.63: Detailed performance measures of branch and bound algorithm for problem set J25/F4/B/L ... 98
Table A.64: Detailed performance measures of branch and bound algorithm for problem set J25/F2/M/F ... 99
Table A.65: Detailed performance measures of branch and bound algorithm for problem set J25/F3/M/F ... 99
Table A.66: Detailed performance measures of branch and bound algorithm for problem set J25/F4/M/F ... 99
Table A.67: Detailed performance measures of branch and bound algorithm for problem set J25/F2/M/A ... 100
Table A.68: Detailed performance measures of branch and bound algorithm for problem set J25/F3/M/A ... 100
Table A.69: Detailed performance measures of branch and bound algorithm for problem set J25/F4/M/A ... 100
Table A.70: Detailed performance measures of branch and bound algorithm for problem set J25/F2/M/L ... 101
Table A.71: Detailed performance measures of branch and bound algorithm for problem set J25/F3/M/L ... 101
Table A.72: Detailed performance measures of branch and bound algorithm for problem set J25/F4/M/L ... 101
Table A.73: Detailed performance measures of branch and bound algorithm for problem set J20/F3/S/F ... 102
Table A.74: Detailed performance measures of branch and bound algorithm for problem set J20/F4/S/F ... 102
Table A.75: Detailed performance measures of branch and bound algorithm for problem set J20/F5/S/F ... 102
Table A.76: Detailed performance measures of branch and bound algorithm for problem set J20/F3/S/A ... 103
Table A.77: Detailed performance measures of branch and bound algorithm for problem set J20/F4/S/A ... 103
Table A.78: Detailed performance measures of branch and bound algorithm for problem set J20/F5/S/A ... 103
Table A.79: Detailed performance measures of branch and bound algorithm for problem set J20/F3/S/L ... 104

Table A.80: Detailed performance measures of branch and bound algorithm for problem set J20/F4/S/L 104
Table A.81: Detailed performance measures of branch and bound algorithm for problem set J20/F5/S/L 104
Table A.82: Detailed performance measures of branch and bound algorithm for problem set J20/F3/B/F 105
Table A.83: Detailed performance measures of branch and bound algorithm for problem set J20/F4/B/F 105
Table A.84: Detailed performance measures of branch and bound algorithm for problem set J20/F5/B/F 105
Table A.85: Detailed performance measures of branch and bound algorithm for problem set J20/F3/B/A 106
Table A.86: Detailed performance measures of branch and bound algorithm for problem set J20/F4/B/A 106
Table A.87: Detailed performance measures of branch and bound algorithm for problem set J20/F5/B/A 106
Table A.88: Detailed performance measures of branch and bound algorithm for problem set J20/F3/B/L 107
Table A.89: Detailed performance measures of branch and bound algorithm for problem set J20/F4/B/L 107
Table A.90: Detailed performance measures of branch and bound algorithm for problem set J20/F5/B/L 107
Table A.91: Detailed performance measures of branch and bound algorithm for problem set J20/F3/M/F 108
Table A.92: Detailed performance measures of branch and bound algorithm for problem set J20/F4/M/F 108
Table A.93: Detailed performance measures of branch and bound algorithm for problem set J20/F5/M/F 108
Table A.94: Detailed performance measures of branch and bound algorithm for problem set J20/F3/M/A 109
Table A.95: Detailed performance measures of branch and bound algorithm for problem set J20/F4/M/A 109
Table A.96: Detailed performance measures of branch and bound algorithm for problem set J20/F5/M/A 109
Table A.97: Detailed performance measures of branch and bound algorithm for problem set J20/F3/M/L 110
Table A.98: Detailed performance measures of branch and bound algorithm for problem set J20/F4/M/L 110
Table A.99: Detailed performance measures of branch and bound algorithm for problem set J20/F5/M/L 110
Table A.100: Detailed performance measures of branch and bound algorithm for problem set J16/F6/S/F 111
Table A.101: Detailed performance measures of branch and bound algorithm for problem set J16/F8/S/F 111
Table A.102: Detailed performance measures of branch and bound algorithm for problem set J16/F10/S/F 111

Table A.103: Detailed performance measures of branch and bound algorithm for problem set J16/F6/S/A ... 112
Table A.104: Detailed performance measures of branch and bound algorithm for problem set J16/F8/S/A ... 112
Table A.105: Detailed performance measures of branch and bound algorithm for problem set J16/F10/S/A ... 112
Table A.106: Detailed performance measures of branch and bound algorithm for problem set J16/F6/S/L ... 113
Table A.107: Detailed performance measures of branch and bound algorithm for problem set J16/F8/S/L ... 113
Table A.108: Detailed performance measures of branch and bound algorithm for problem set J16/F10/S/L ... 113
Table A.109: Detailed performance measures of branch and bound algorithm for problem set J16/F6/B/F ... 114
Table A.110: Detailed performance measures of branch and bound algorithm for problem set J16/F8/B/F ... 114
Table A.111: Detailed performance measures of branch and bound algorithm for problem set J16/F10/B/F ... 114
Table A.112: Detailed performance measures of branch and bound algorithm for problem set J16/F6/B/A ... 115
Table A.113: Detailed performance measures of branch and bound algorithm for problem set J16/F8/B/A ... 115
Table A.114: Detailed performance measures of branch and bound algorithm for problem set J16/F10/B/A ... 115
Table A.115: Detailed performance measures of branch and bound algorithm for problem set J16/F6/B/L ... 116
Table A.116: Detailed performance measures of branch and bound algorithm for problem set J16/F8/B/L ... 116
Table A.117: Detailed performance measures of branch and bound algorithm for problem set J16/F10/B/L ... 116
Table A.118: Detailed performance measures of branch and bound algorithm for problem set J16/F6/M/F ... 117
Table A.119: Detailed performance measures of branch and bound algorithm for problem set J16/F8/M/F ... 117
Table A.120: Detailed performance measures of branch and bound algorithm for problem set J16/F10/M/F ... 117
Table A.121: Detailed performance measures of branch and bound algorithm for problem set J16/F6/M/A ... 118
Table A.122: Detailed performance measures of branch and bound algorithm for problem set J16/F8/M/A ... 118
Table A.123: Detailed performance measures of branch and bound algorithm for problem set J16/F10/M/A ... 118
Table A.124: Detailed performance measures of branch and bound algorithm for problem set J16/F6/M/L ... 119
Table A.125: Detailed performance measures of branch and bound algorithm for problem set J16/F8/M/L ... 119

Table A.126: Detailed performance measures of branch and bound algorithm for problem set J16/F10/M/L 119
Table B.1: Number of problems exceed node limit in problem sets 121
Table C.1: Number of problems, at which the optimal solution is equal to the upper bound value, in a set. 124

LIST OF FIGURES

Figure 3.1: Schedule obtained for the sample problem using upper bound heuristic 25
Figure 3.2: Optimal schedule obtained for the sample problem using proposed B&B algorithm. 25
Figure 3.3: Schedule obtained for LB[1,1] of the sample problem using FILB procedure. 32
Figure 3.4: Schedule obtained for LB[1,3] of the sample problem using FILB procedure. 35
Figure 3.5: Schedule obtained for LB[1,1] of the sample problem using FDLB procedure. 41
Figure 3.6: Schedule obtained for LB[1,3] of the sample problem using FDLB procedure. 42
Figure 4.1: Schedule obtained for the sample problem using LINDO optimization package. 69

CHAPTER 1

INTRODUCTION

Owing to the increase in demand to the low volume specialized products, scheduling of the existing processors (machines, reactors, tanks etc.) became much more important in the manufacturing environment. Parallel to this demand in manufacturing environments a growing area of research has appeared on literature; job grouping.

The motivation of the job grouping may be different for several industries. It is sometimes related to the existence of setup times or changeover times on the machines. Suppose we have many jobs belonging to different job families. The jobs in a family show great similarities with the other jobs in the family thus we can produce them consecutively with a single setup procedure. On the other hand, we need a setup while changing the job family from one to another job family. This model of job grouping is called as family scheduling model. (Webster&Baker, 1995) In family scheduling model, a machine is assumed capable of processing single job at a time.

There are two variants of family scheduling model: the first is item availability model (or job availability model). In item availability model the job can pass to the next processing unit just after its completion. In the second variant, if the job is included in the batch it can pass to the next processing unit only after the completion of all jobs in the batch. This is called as batch availability. (Potts&Kovalyov, 2000)

Another motivation of the job grouping may be the capability of the processing unit (machine) to process several jobs at once. The processing unit can have several jobs in it up to the capacity and process them simultaneously. The nature of the process is batch availability model as mentioned before, whereas the distinguishing feature is not the availability of jobs but the jobs included in the batch. This class of job grouping is called batch scheduling model. For some processes, the jobs in a batch should be identical in terms of their families whereas in some processes this is not an obligation and the jobs from different families may coexist in the same batch.

Before going further we should also give the explanation of the term “family” for this study. A family is a group of jobs, whose technical and processing requirements are the same. The coexistence of two jobs in a batch does not effect processing of each other, if these two jobs belong to the same family. On the other hand, two jobs from different families cannot coexist in a batch. This situation is called as “incompatible jobs” or “incompatible families” in the literature.

Batch scheduling models are applicable in many industries. Especially the integrated circuit manufacturing is a good application area, because we can observe both cases, batches with compatible jobs and batches with incompatible jobs, together in integrated circuit manufacturing. For example, burn-in operation in integrated circuit manufacturing is a good example to the multi family batch processing (compatible families). The final stage in production of the integrated circuits is the burn-in operation, in which chips are loaded onto boards and exposed to high temperature. Each chip has a pre-specified minimum burn-in time, which depends on type or the customer requirements. The burn-in oven has limited capacity thus the boards holding the chips must be divided into batches. Since chips may stay in the oven for a period longer than their minimum required burn-in time, it is possible to place different products in the oven simultaneously. The processing time of each batch will therefore be equal to the longest minimum exposure time among all the products in the batch. (Uzsoy 1995). On the other hand, the operation of diffusion carried out in wafer fabrication, is a batch processing example with single family batches (incompatible families). In this operation a number of wafers are placed in a cylindrical reactor, which is then sealed, heated and filled with a carrier gas to allow dopant atoms present in the gas to diffuse into the exposed layer of the wafers, altering their electrical and chemical characteristics. Due to the chemical nature of the process it is impossible to process jobs with different recipes together in the same batch. (Uzsoy 1995).

In this work our aim is to explore a single machine batch scheduling problem with incompatible families and dynamic job arrivals. The objective function is minimizing the total volume weighted flow time of the jobs. This problem is critical since the batching practice is important for many cases. Moreover the dynamic nature of the arrivals requires a "make to order" approach in the scheduling of the production period. In that respect for minimizing the inventories total flow time of the jobs play an important role.

To be able to solve this problem we should make a combined decision at each decision point:

1. start processing a new batch or wait for the next arrival of a job?
2. if the decision is "starting a new batch" then which one of the existing families should be selected?
3. finally the jobs from the same family should be selected for being processed in the batch.

The main difficulty of the problem caused by its dynamic nature. Since the job arrivals are arbitrary, at any decision point the available jobs and non-released jobs create a non-repeating arrival pattern. This pattern constitutes an enumerative problem, which is difficult to solve. This problem can be compared with static arrival weighted flow time problem with incompatible jobs, where all jobs are simultaneously available at time zero. It is known that this problem is NP-Hard. Considering that in an interval the jobs with different volume weights also constitute a bin-packing problem we can conclude that this problem is also NP-Hard. Moreover it is known that single machine weighted flowtime problem with different release times is NP-Hard even the weights are all unity [3]. Thus this is another comparison that allows us to conclude our problem is also NP-Hard. To solve this problem we used branch and bound method with "depth first" branching strategy. It is coded using Borland Turbo Pascal 7.0 and run on Intel PIV-1.6 Ghz processor under Windows NT 4.0 operating system.

We have developed an upper bound procedure and two alternative lower bound procedures. These are based on problem specific properties.

To explore the problem we observed four parameters; first one is the job volume. We created the input sets in three different job volumes: "small" (the job volumes are determined using uniform distribution in the interval between 0 and 50% of the reactor capacity), "large" (the job volumes are determined using uniform distribution in the interval between 50% of the reactor capacity and 100% of the reactor capacity) and "mixed" (the job volumes are determined using uniform distribution in the interval between 0 and 100% of the reactor capacity).

The second one and the third one of the parameters are the job number and family number, respectively. In this study we determined the job numbers and family numbers simultaneously. The sets are between 50 jobs, 2 families and 16 jobs, 10 families.

The last parameter is the intensity of the arrivals. In this study we used three different frequencies: “frequent” or “tight” (the job arrivals occur in a time interval between 0 and 1/3*(processing time * total job number) with a uniform distribution), “moderate” (the job arrivals occur in a time interval between 0 and 2/3*(processing time * total job number) with a uniform distribution) and “loose” (the job arrivals occur in a time interval between 0 and (processing time * total job number) with a uniform distribution).

CHAPTER 2

AN OVERVIEW

The grouping of jobs is a rather new topic in manufacturing. Therefore the term "batching" can be used for several types of problems in the literature. In this chapter our aim is to address some of the studies made on the scheduling problems with batching of jobs.

2.1 The Literature on Family Scheduling Models

As it mentioned before, we are only interested in single machine batch scheduling problems with incompatible families. But for the sake of the completeness of the work, we also give a short overview for some of the family scheduling model literature.

One of the first and widely known works in this area is the paper of Santos&Magazine (1985). They concentrated on single machine static arrival of n different jobs. The objectives used in the work were minimization of total flow time for item and batch availability models and minimization of total tight flow time for item and batch availability models. They proposed optimal dispatching algorithms for minimization of total flow time problem with item availability and batch availability models. Moreover for total tight flow time minimization problem they proposed that the problem could be reduced to the knapsack problem and if the set-up times are considered equal, a greedy single pass algorithm gives the optimal solution.

In 1987 Dobson et al. studied also a static single machine environment to minimize total flow time of the produced parts. In this study they represented general integer programs for item-flow problem and batch-flow problem. Then for item-flow problems they proposed a dispatching rule solves the problem optimally. In batch-flow problem they separated the problem into two parts. First one is single product batch flow problem. They showed that the general problem can be reduced since there exist single part type and can be solved after this reduction. The second part of the batch-flow problem is represented as multiple product batch flow problem. For this part Dobson et al. (1987) proposed two solution heuristics and two improvement heuristics for the second solution heuristic. To be able to compare the effectiveness of the heuristics, they also proposed a lower bound.

Cheng et al. (1994) proposed a dynamic programming approach to single machine, simultaneously available multiple type items. The jobs have batch availability property and the objective of the study is to minimize the total item flow time. Their DP algorithm runs in time $O(n^{T+1}/T^{T-1})$, where n represents the number of items and T represents number of types. The proposed algorithm becomes polynomial-bounded when T is fixed. In case of arbitrary number of types, T, the problem becomes open to search whether the problem is solvable or NP-hard.

Sung&Joo (1997) published a paper investigating the same type of batch flow problem as that of Dobson et al. (1987) and Cheng et al. (1994) and additionally, considered two types of batch setup times (minor and major setup times) with capacity restricted integral batch volumes. In this paper they propose a DP solution and a DP based heuristic method to solve the minimization of weighted mean flow time problem. But as in the paper of Cheng et al. (1994), Sung&Joo

(1997) states that the problem is time bounded in terms of job numbers when the number of job types is fixed whereas it is open to search when the job type is not fixed. Then they proposed a heuristic method to solve the problem. This heuristic method has a time complexity O(n^2 d log n).

And recently in 2001 Cheng&Kovalyov published a paper having static arrival, batch availability of the jobs with constant set-up times. But their objective function is not unique but studied all these objectives:

- minimizing maximum lateness
- number of late jobs
- total tardiness
- total weighted completion time
- total tardiness

According to the batch capacity they classified the problems into two groups, unbounded model, the batch capacity is equal to the total job number, and bounded model, the batch capacity is less than the total job number. For both bounded and unbounded cases they proposed DP algorithms when all due dates are equal. They stated that the problems can be solved in polynomial time in case of existence of fixed number of distinct due dates or processing times. Moreover, Cheng&Kovalyov (2001) proposed efficient algorithms for some special cases, in which all the processing times and/or due dates are equal, and showed that some of the bounded models are NP-hard.

2.2 The Literature on Batch Scheduling Models

In batch scheduling problem there is a main distinction point in terms of the job specifications. Most of the literature on batch scheduling deals with multifamily batches as in burn-in operations, the final testing stage in semiconductor manufacturing. The jobs in burn-in operations are in solid state and the existence of a job from another family do not affect the jobs in the batch except their completion time.

On the other hand in many production environments, where the operation has a chemical nature and in liquid or gaseous state, the batches should be uniform in terms of families of jobs. Because the jobs are incompatible and the coexistence of these jobs may cause contamination, loss of material or side reactions.

There are limited works on batch scheduling of incompatible jobs in literature. Most of the authors agree that one of the earliest works on batch scheduling appears to be that of Ikura and Gimple (1986). They studied the problem of scheduling a single batching machine in the presence of release times, r_j , and due date, d_j . All jobs assumed to have identical processing times and there exist single job family. Under the assumption of agreeable release times and due dates they provided an algorithm to the problem to determine whether there is a schedule, where all jobs are completed by their due dates.

In 1991 Glassey and Weng presented a paper on “Dynamic batching heuristic for simultaneous processing”. In this paper they examined the problem of scheduling a single batch processing machine with single family in case of dynamic job arrival. Their objective function was minimizing the flow time of a job. They assumed that they have only the information of next L periods and in the scope of this information, they try to decide when to start a batch. They tried the effectiveness of the heuristic using a simulation model.

In 1992 Ahmadi et al. presented a study on scheduling of a system composed of a discrete processor and batch processor in series. They proposed six different solution approaches to minimization of maximum completion time problems and minimization of total completion time problems with different machine configurations. They proposed that five of these problems can be solved in polynomial time and one of them is NP-complete.

Another important work is that of Chandru, Lee and Uzsoy (1993). Briefly this work studied the problem of minimizing total completion time on single and parallel batch processing machines. This problem is motivated by burn-in operations. In this study they assumed that jobs have different processing times and their capacity requirements are equal, all jobs are available at time zero. Chandru et al. proposed an exact solution procedure for the single machine problem and heuristic algorithms for both single and parallel machine problems.

Hochbaum and Landy (1997); they were inspired by the work of Chandru et al. As it mentioned in previous part of this work Chandru et al. proposed a branch and bound algorithm for relatively small number of jobs (35 distinct jobs) and two heuristic methods for larger methods. In addition Chandru et al. considered a restricted version of the problem on a single machine in another paper (the paper is not discussed in this work), in which there is a fixed number of job types and jobs of the same type have the same processing time. This problem is called m-type burn-in problem by the authors. Chandru et al. provided a DP algorithm with running time $O(m^3B^{m+1})$ to the m-type burn-in problem. Hochbaum and Landy state that the DP solution provided by Chandru et al. depends heavily on B, oven capacity, thus in practice the proposed DP can be useful only for small values of m as B get larger. Moreover the B&B proposed for the general problem is again effective for small number of jobs.

In the work of Hochbaum and Landy (1997), they proposed an algorithm for the m-type burn-in problem, which has a running time of $O(m^2 3^m)$. The running time is independent of n, number of jobs, and B, the capacity of the oven.
Besides the m-type burn-in problem on single machine Hochbaum and Landy consider also the general burn-in problem. They proposed a DP based heuristic for this general problem, which guarantees a solution that is at most twice the value of optimal solution.

In 1997, DuPont and Ghazvini (1997) presented a B&B algorithm to minimize mean flow time of jobs on a single batch processing machine. As in the previous studies it was focusing on burn-in operations and the job sizes were assumed to be identical and all jobs are present at time zero.

The paper of Brucker et al. (1998) consists of two main parts; the first part is the unbounded model, where the batch capacity is greater or equal to the number of jobs. The second part is the bounded model, where the batch size is less than the number of jobs. In the first part the objective function is not unique, but consider all these objective functions:

- minimizing the number of tardy jobs
- minimizing total weighted completion time
- minimizing the maximum lateness and maximum cost
- minimizing the weighted number of tardy jobs
- minimizing total weighted tardiness

For the unbounded case they presented a generic DP algorithm that solves the minimization problems in pseudo-polynomial time. With the further study and characterization of the minimization problem they proposed more efficient DP algorithms for specific cost functions. They proposed an O(N log N) time algorithm for the minimization of the total weighted completion time.
For the bounded case they proposed an $O(n^{b(b-1)})$ time DP algorithm for minimizing total completion time when b>1; b is the batch capacity. Besides they proposed an $O(b^2m^22^m)$ DP algorithm for

Another paper that discusses the batch scheduling of compatible jobs, is the work of Ghazvini and Dupont (1998). Unlike the others this paper discuss a situation where the jobs are in different volumes and requires the place in burn-in oven relevant to their volumes.

The objective is minimizing the mean flow time. The authors proposed several heuristic algorithms to solve this problem and determined a lower bound to compare the results of the proposed heuristic algorithms. By these comparisons they concluded that DYNA heuristic, an iterative, parametric heuristic, performs best in most of the cases.

And recently in 2002 a study of DuPont and Flipo has appeared in the literature. They proposed an exact algorithm to minimize makespan on a batch machine with non identical job sizes. They proposed three different heuristic methods to solve bin packing problem. By using two of the proposed heuristics they developed a branch and bound algorithm.

We now introduce the papers that address scheduling of incompatible job families. The work of Dobson and Nambimadom (1992) is one of the earliest and widely known researches on the subject. Considering the scarcity of the works on scheduling of incompatible jobs with respect to burn-in problems, the importance of the work increases.

In this working paper the authors presented an integer programming formulation for the problem. Moreover, a lower bound calculation using the partial relaxation of the integer programming is given. Besides these they proposed three heuristics; one is a greedy heuristic, a successive knapsack heuristic and generalized assignment heuristic. The major assumptions made in this work is as follows:
The jobs belonging to different families have to be processed separately. The processing time of the batch depends only on the family but not to the volume of the jobs. The volume of each job can be different. All jobs are available at time zero. The objective of the problem is the minimization of the mean flow time.
Dobson and Nambimadom (1992) concluded that knapsack heuristic is superior to the greedy heuristic; assignment heuristic is better than the knapsack heuristic on average but the performance time of the knapsack heuristic is shorter.

In the same year, 1992, Fowler et al. presented a paper on “Real time control of multiproduct bulk service semiconductor manufacturing processes”. This paper based on the study of Glassey and Weng (1991). But they also studied the multi-product, dynamic arrival batching machine problem with different processing times and incompatible jobs. At the first glance this paper seems very similar to the problem discussed in our study but there is a major distinction in problem definition and two differences in problem conditions. First, the problem conditions: in this paper Fowler et al. assumed the job volumes are fixed and the job processing times are different for the families. Second, the problem definition and nature: In the study of Fowler et al., the future arrival information is not available thus for each family they try to predict the next arrival time and decide to start processing a batch or wait for the next arrival.

Another paper appeared in the literature is the work of Uzsoy (1995). Uzsoy discussed the minimization of makespan (C_{max}), total weighted completion time and minimization of maximum lateness (L_{max}) on single batch reactor for static problems. He offered optimal algorithms to solve these problems. Beside these he also extend the problem to the static, identical parallel machine environment. Finally, he offered an optimal algorithm for the minimization of makespan problem for the dynamic job arrival environment, and proposed some heuristic methods for the minimization of maximum lateness.

In 1998 Mehta and Uzsoy proposed a dynamic programming algorithm to solve the minimzation of total tardiness on a batch processing machine with incompatible job families. Their algorithm can solve the problem in polynomial time when the number of families and machine capacity are fixed. Moreover they proposed two heuristic algorithms to solve the problem in a reasonable computation time.

Finally Azizoğlu and Webster (2001) studied a static batch-processing problem with incompatible job families. In this study the jobs may have different space requirements due to their volumes. The work of Azizoğlu and Webster extends the analysis of the model studied by Dobson and Nambimadom(1992). They developed two properties of an optimal schedule, proposed a lower bound procedure, incorporate these and other known properties and bounds in the design of a branch and bound algorithm.

In literature there exist also some studies on the batch scheduling with dynamic arrivals. The first study that will be introduced is the paper of Sung and Yoon (1997). Unlike the others Sung and Yoon studied a two-batch processing machine flow shop problem with dynamic arrivals. In this study the jobs have different processing times in different machines. There exist no families that can cause an incompatible case or elongation of processing time. They proposed a dynamic programming algorithm with time complexity O(n*c_1), where c_1 is the capacity of the first batching machine.

In 2002, Sung et al. presented another paper with dynamic arrivals. This paper was focusing on minimization of makespan on a single burn-in oven with job families and dynamic arrivals. In this paper authors proposed a dynamic programming algorithm to solve the problem. The time complexity of the proposed algorithm is $O(\prod_{i=1}^{F} n_i c^F)$, where n_i is the number of jobs from family f and F is the number of families.

In this work our aim is to explore the batch-scheduling problem with incompatible job families and dynamic job arrivals for minimizing the total flow time of the jobs. Considering the assumptions made and the solution approach, our work can be regarded as an extension of the work by Azizoğlu and Webster (2001).

CHAPTER 3

PROBLEM DEFINITION, PROPERTIES OF THE PROBLEM AND SOLUTION APPROACH

In this study our motivation was building a parallelism with the liquid phase chemical batch reactions and a general batching machine. To be able to establish the consistency of the problem environment we make the following assumptions.

3.1 Assumptions

1. *Multiple jobs are processed simultaneously in batches; only the jobs from the same family may be processed together (incompatible jobs).*

   This assumption forces us to collect the existing jobs in multiple, unifamilar groups. Thus at any decision point we should evaluate each of these groups and the content of them separately. This property increases the number of decisions. Considering that the processing times of all batches are equal, this assumption is the only assumption leading to the impact of the families.

2. *Preemption is not allowed, the process of the batching machine cannot be interrupted; the jobs cannot be added to the batch or cannot be removed from the batch during the process.*

The main motive of this assumption was the chemical nature of the proposed problem. Unlike the manufacturing industries most of the processes in process industries are irreversible and continuous. This property necessitates the uninterrupted processing of the raw material. Besides this property of chemical processes, the structural and operational conditions of batch reactors are other important factors.

3. *All jobs, regardless of their family, have the same processing time, p.*

   This is a simplifying assumption. This property justifies dividing the time scale into uniform length intervals. By using this grid we can analyze the whole time horizon in distinct short intervals and then we can look for the interactions between these short intervals.

4. *Jobs may have arbitrary volumes between 1 and 100 ($1 \leq$ job volume ≤ 100), where the batch capacity is 100 units.*

   This assumption leads to the indeterminism in batch content. Since the job volumes are arbitrary, theoretically a batch may contain a single job or 100 jobs. Moreover two different batches with n distinct jobs may have different total volumes. Therefore this property brings a batch content decision problem.

5. *Jobs arrive to the processor at arbitrary integer time points and in random order.*

   This assumption is the main characteristic of this study. The arbitrary arrival of jobs creates a non-repeating, non-systematic pattern. The optimal solution of such a pattern can be obtained only by enumerative approaches, which constitutes the main difficulty of this study.

6. *The jobs can be split to different batches, but the completion time of a job is equal to the completion time of its last portion.*

   This property depends heavily on the liquid nature of the reactants and product i.e. in this study we assumed that both raw materials and the product are liquid in nature; therefore they can be divided into all portions. This property brings an extra simplification in using the reactor at its maximum capacity. Besides, with the improved utilization of the reactors the optimal solution will also be improved.

Considering the production environment described above, we can formulate the problem as follows:

Min $\sum_{i=1}^{n} F_i$

Subject to:

$$X_{ik} \leq V_i * J_{ik} \qquad \forall \text{ i,k} \qquad (1)$$

$$\sum_{k=1}^{m} X_{ik} = V_i \qquad \forall \text{ i} \qquad (2)$$

$$\sum_{i=1}^{n} X_{ik} \leq 100 \qquad \forall \text{ i,k} \qquad (3)$$

$$J_{ik} \leq \frac{1}{NF-1} \sum_{f(l) \neq f(i)} \left(\frac{N_{f(l)} - \sum_{l \neq i} J_{lk}}{N_{f(l)}} \right) \qquad \forall \text{ i,k} \qquad (4)$$

$$BS_k \geq BC_{k-1} \qquad \forall \text{ k} \qquad (5)$$

$$BS_k \geq R_i - M(1 - J_{ik}) \qquad \forall \text{ i,k} \qquad (6)$$

$$BC_k = BS_k + p \qquad \forall \text{ k} \qquad (7)$$

$$F_i \geq V_i (BC_k - R_i) - M(1 - J_{ik}) \qquad \forall \text{ i,k} \qquad (8)$$

$BS_k \geq 0$, $BC_k \geq 0$, $X_{ik} \geq 0$, $J_{ik} = \{1,0\}$.

The variables are:

X_{ik} : the volume of job i processed in batch k;

J_{ik} : 1 if any portion of the job i is processed in batch k, 0 otherwise;

BS_k : start time of the processing of batch k

BC_k : completion time of the processing of batch k.

The parameters are:

p : the processing time of a batch

R_i : the release time of job i

V_i : total volume of job i

NF : number of families

$N_{f(i)}$: number of jobs from family f

$f(i)$: index of family of job i

M : very large number.

Constraint 1 and constarint 2 ensures that the fraction of a job is at most identical to the job volume and the sum of the fractions is equal to the job volume.

Constraint 3 ensures that the sum of the volumes of jobs or job fractions is less than the reactor capacity.

Constraint 4 ensures that the jobs from different families are not processed in the same batch.

Constraint 5 ensures that the processing of a batch starts if and only if its predecessor is finished.

Constraint 6 ensures that any batch contain only the jobs which are already released.

Constraint 7 ensures that a batch is completed when the processing of the jobs are completed.

Constarint 8 ensures that volume weighted flow time of a job is calculated when the last portion of the job is completed.

3.2 Properties of the Problem

Using these conditions and specifications stated before for the problem environment, we propose some properties of the optimal solution.

Property 1: Suppose *a batch completion is time t. If we have jobs waiting to be processed, the start of a batch cannot be equal to or later than t+p. (Waiting time of a job can not exceed processing time of a batch when the batch processor is idle.)*

The motivation of this property is that, keeping the machine idle more than a processing time period is not reasonable. The proof of this property can be made easily by contradiction; assuming a batch start at t+p which is optimal and inserting an extra batch to the idle period [t,t+p] in which at least one of the jobs available at time t can be processed.

Property 2: Start time of a new batch can be either a job release time or a batch completion time but no other time.

Proof: Let at time t a batch be completed and we have uncompleted jobs waiting to be processed. Let the next event after this event is at t+n and it is a job arrival. Suppose the optimal solution is obtained in a way that the batch start is not at t or t+n, but at t+m, where n>m. Then we can reduce the total flow time by reducing the flow time of a job, which is available at time t but not started to be processed until t+m. And during this back shift from t+m to t all the other jobs may remain at their optimum positions.
The second property given here can be also found in the paper of Glassey and Weng (1991).

In their paper Glassey and Weng (1991) proposed yet another property: "If the volume of the available jobs exceed (or at least equal to) the capacity of the batch processor and the processor is idle it starts service immediately." This property is very important for the solution of their problem. On the other hand in our problem it cannot be applied. Because the volumes of the jobs in their problem are fixed; and the jobs belong to a single family. These characteristics provide them information about the maximum capacity and the maximum number of jobs in a reactor. In our problem the job volumes are arbitrary this gives rise to another classical combinatorial problem, bin packing problem.

3.3 Approach for Selecting Available Jobs from Families

At a decision point (i.e. at a batch finish time or a job release time) if a family contains more than one job and the sum of the volumes of existing jobs from that family is at least equal to the reactor capacity this case constitutes an instance of bin packing problem. Bin packing problems are hard problems to solve [22]. To overcome this embedded problem we propose a simple rule and get satisfied by an optimal solution among the problems solved by this simple rule. We assume that this restricted optimum solution gives a satisfactory batch schedule. In this study we used the proposed solution approach whenever we met a bin packing problem instance.

Solution Algorithm of Bin Packing Problems

Step1: Determine the available jobs from the same family.

Step 2: For each existing job calculate the ratio $\frac{releasedvolumeofthejob}{existingvolumeofthejob}$.

Step 3: Sort the jobs in decreasing order of this ratio. If there exists a tie, break the tie in favor of the job having higher total volume.

Step 4: Start taking the jobs from the top up to filling the capacity. The last job can be split into two to fill the processor maximally.

After setting the production environment and the bin-packing algorithm, we can propose the heuristic method to solve the single machine batch-scheduling problem. The solution obtained from this heuristic method was used as an upper bound.

Before starting to explain the upper bound procedure it is necessary to introduce some notation about the procedure. These notations are also valid for the lower bound procedures.

V^z_{xy} : Volume of the xth job in interval y belong to family z.
R^z_{xy} : Release time of the xth job in interval y belong to family z.
$m^r(z,y)$: index number of the last job from family z in interval y, whose release time is equal to the interval start time.
$m^*(z,y)$: index number of the last job from family z in interval y
$m(z,y)$: index number of the last job from family z in interval y, which completes the batch capacity to the reactor capacity. If the volume of the last job,say job x, is greater than the remaining capacity the index $m(z,y)$ represents job x-i.
i^*: index of the last interval
$V^z_{xy,opt}$: Volume of the xth job in interval y belong to family z under an optimizing procedure.
Finally we should note that for any family f in any interval i, total volume of the jobs, which are released before the start time of the interval i and not processed yet, are presented by V^f_{0i}.

3.4 Upper Bound Heuristic

This is a rolling horizon type heuristic and its main idea is maximizing the average utilization of the reactor per unit time. It takes a standard batch processing period as the basis and determines for each family a possible batch and batch start time. Then select one of these batches for being processed. This approach provides a feasible start time and content for every batch. This is the brief description of one cycle of the upper bound heuristic. At each cycle of the upper bound procedure it provides a solution for another batch. The description of the heuristic is given below, the pseude-code of the upper bound procedure can be found in Appendix D.

Algorithm of Upper Bound Procedure

Step 1: Set a standart interval with length p.

Step 2: Determine the total volume of available jobs , $\sum_{x=0}^{m^r(f,i)} V^f{}_{xi}$ for each family f;

a. If $\sum_{x=0}^{m^r(f,i)} V^f{}_{xi} \geq$ reactor capacity

   Select the jobs among family f using the bin packing algorithm, establish a batch.

b. If $\sum_{x=0}^{m^r(f,i)} V^f{}_{xi} <$ reactor capacity

   Take all available jobs of family f into the batch, then calculate the capacity utilization of family f $= \dfrac{\sum_{x=0}^{m^r(f,i)} V^f{}_{xi}}{p}$.

   Starting from the job x=m^r(f,i) to m*(f,i) check

   If $V^f{}_{xi} \leq$ remaining batch capacity and

   $\dfrac{V^f{}_{xi} + \sum_{x=0}^{m^r(f,i)} V^f{}_{xi}}{R^f{}_{xi} + p - R^f{}_{0i}} \geq \dfrac{\sum_{x=0}^{m^r(f,i)} V^f{}_{xi}}{p}$ then take the job into the batch; decrease the remaining capacity; check the next job.

Step 3: Select the family f with f = argmax $\{V_f\}$ where V_f is the volume of the batch of family f per unit time.

Step 4: Calculate total weighted flow time of jobs throughout the interval;

If jobs are finished, stop; sum all weighted flow times calculated throughout the intervals, UB is found;

Else Goto Step 1.

Sample of Upper Bound Calculation

Let the processing time of a batch, p, is 10 units and the capacity of the reactor is 100 units. The data of the problem is as given in following table:

Table 3.1: Data of the sample problem of upper bound heuristic

Job #	1	2	3	4	5	6	7	8	9	10	11	12
Rel.time	2	6	8	11	14	20	22	28	33	37	43	46
Family	1	2	1	1	2	2	1	2	2	1	2	1
Volume	50	36	48	21	24	30	44	52	32	33	41	37

The first interval is [2,12), the total volume of available jobs are 50 from family 1 and 0 from family 2.
For family 1: 50 / 10 = 5
(50 + 48) / (8+10-2) = 6.125.
Since volume of the job 4 is greater than the remaining capacity of the batch (i.e.100-98), it is skipped. Since job 4 is the last job of the family 1 in the interval [2,12) we stop at this point and start to evaluate the next family.
For family 2 : 0 / 10 = 0
(0 + 36) / (6+10 - 2) = 2.514.
Job 2 is the last job of the family 2 in the interval.
Max {5 , 6.125, 0, 2.514} = 6.125, then it is decided that
Batch start : 8, Batch finish : 18 Selected family : 1.
Total weighted flow time of the released jobs up to the batch finish time is:
50*16 + 36*12 + 48*10 + 21*7 + 24*4 = 1955.
From this point on the calculations will be given in brief:
Interval is [18,28), volume of available jobs are 21 from family 1 and 60 from family 2.

For family 1: 21 / 10 = 2.1

(21 + 44) / (22 + 10 - 18) = 4.643.

For family 2: 60 / 10 = 6.0

(60 + 30) / (20 + 10 - 18) = 7.50.

Max {2.1, 4.643, 6.0, 7.50} = 7.50, then

Batch start : 20, Batch finish : 30 Selected family : 2.

21*12 + 60*12 + 30*10 + 44*8 + 52*2 = 1728

Interval is [30,40), volume of available jobs are 65 from family 1 and 52 from family 2.

For family 1: 65 / 10 = 6.5

(65 + 33) / (37 + 10 - 30) = 5.76.

For family 2: 52 / 10 = 5.2

(52 + 32) / (33+10 - 30) = 6.46.

Max {6.50, 5.76, 5.20, 6.46} = 6.50, then

Batch start : 30, Batch finish : 40 Selected family : 1.

65*10 + 52*10 + 32*7 + 33*3 = 1493.

Interval is [40,50), volume of available jobs are 33 from family 1 and 84 from family 2.

For family 1: 33 / 10 = 3.3

(33 + 37) / (46 + 10 - 40) = 4.375.

For family 2: 84 / 10 = 8.4

41 > (100-84) therefore skip job 11.

Max {3.30, 4.375, 8.40} = 8.40, then

Batch start : 40, Batch finish : 50 Selected family : 2.

33*10 + 84*10 + 41*7 + 37 * 4 = 1605.

Interval is [50,60), volume of available jobs are 70 from family 1 and 41 from family 2.

For family 1: 70 / 10 = 7.0.

For family 2: 41 / 10 = 4.10.

Max {7.0, 4.10} = 7.0, then

Batch start : 50, Batch finish : 60 Selected family : 1.

70*10 + 41*10 = 1110.

Interval is [60,70), volume of available jobs are 0 from family 1 and 41 from family 2.

For family 1: 0 / 10 = 0.

For family 2: 41 / 10 = 4.10.

Max {0, 4.10} = 4.10, then

Batch start : 60, Batch finish : 70 Selected family : 1.

41*10 =410.

Since batch finish time is greater than the release time of the last job and no available job exist, end the upper bound procedure.

Upper Bound is 8301.

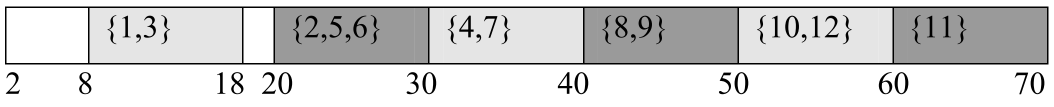

Figure 3.1: Schedule obtained for the sample problem using upper bound heuristic

Note that the optimal solution of this problem is 7942.

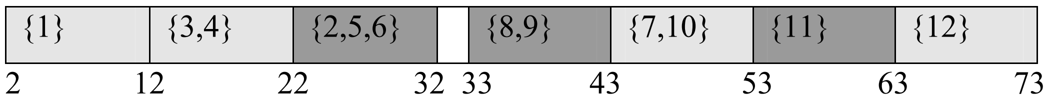

Figure 3.2: Optimal schedule obtained for the sample problem using proposed B&B algorithm.

3.5 Lower Bound Procedures

After setting the upper bound heuristic, we defined two lower bound heuristics to be able to search the problem more efficient.

The flow time of a job can be thought in two parts, the first part is the "operational flow time". Operational flow time is the flow time accumulated during the processing of the job. Therefore it is the minimum possible flow time of the job. The second part is the "waiting flow time". Waiting flow time is the flow time accumulated during the time between release time of the job and start of its last portion's (if any) processing. This part is the variable part and makes the difference between the proposed lower bound procedures. Any procedure that estimates the waiting flow time closer to the optimal result provides tighter lower bound. Since the processing flow time is fixed part of the flow time it can be added later without any difficulty. Throughout this study we took only the waiting flow times in the calculations made for comparison (i.e proofs) due to its convenience.

As it is mentioned before, we proposed two lower bound procedures. The first one is the "Family independent lower bound procedure". The other one is the "Family dependent lower bound procedure".

3.5.1 Family Independent Lower Bound Procedure (FILB)

The main idea of this procedure is operating the reactor at maximum capacity utilization on a continuous basis. To increase the utilization of the reactor following assumptions are relaxed:

- *Multiple jobs are processed simultaneously in batches; only the jobs from the same family can be processed together (incompatible jobs).*

In this procedure we assume that the jobs from different families can be aggregated together in a single batch (i.e. incompatible job assumption is relaxed).

- *A job can be split to different batches, but the completion time of a job is equal to the completion time of its last portion.*

Another relaxation is made on the completion time of the jobs. We assume that any finished portion of a job does not contribute towards flow time. Only the uncompleted part charges the flow time with respect to its volume.(i.e. let a 80 unit volume job be divided into portions as 20 and 60. The 20 unit portion is already finished and 60 unit portion is waiting to be processed. The flow time of the job is calculated over the 60-unit portion following the completion of 20-unit job)

- *Jobs arrive to the processor at arbitrary integer time points and in random order.*

Finally we relaxed the arbitrary arrival of the jobs. As in the upper bound heuristic, we set intervals to solve the problem and we re-ordered the release times of the jobs according to the interval start time.

Similar to the upper bound heuristic, the family independent lower bound procedure is a rolling horizon type heuristic.

We set interval start time and interval end time for a batch. Interval start time is the first possible time when the machine is idle. As it is in the upper bound procedure, interval length is equal to the batch processing time.

After setting the interval boundaries we determine the available jobs at the interval start for each family and the jobs in the interval.

For the first interval we determine the available jobs according to their families and calculate the flow times as in the optimal solution. But starting from the second interval the volumes of the available jobs are handled as the total volume of all jobs regardless of their families. If the total volume of the available jobs is at least equal to the reactor capacity we start to the process. If the total volume of the available jobs is less than the reactor capacity, we "pull " the jobs from the interval by re-ordering the release times of these jobs, if there exist any release of job throughout the interval. We "pull" the jobs until the volume of available jobs (i.e. jobs in the batch) reaches the reactor capacity or no more jobs remain in the interval. If the volume of the job to be pulled is greater than the remaining capacity, we pull a portion of job, which is equal to the remaining capacity. If there exist no job release in the interval, we start to the processing with available jobs.

Algorithm of Family Independent or Consolidated Family Lower bound Procedure (FILB)

Step 1: Set the standard interval with length p, [t,t+p).

Determine the total volume of available jobs from family f, $\sum_{x=0}^{m^r(f,i)} V^f{}_{xi}$, at time t.

a. If $\sum_{x=0}^{m^r(f,i)} V^f{}_{xi}$ ≥ reactor capacity

Select the jobs among the available jobs using bin packing algorithm provided that the lower bound refer to the restricted optimum as mentioned in section 3.3, establish a batch.

b. If $\sum_{x=0}^{m^r(f,i)} V^f{}_{xi}$ < reactor capacity

Establish a batch with available jobs from family f.

Calculate the total weighted flow time of jobs throughout the interval.

c. If $\sum_{x=0}^{m^r(f,i)} V^f{}_{xi} = 0$

Label LB[j,f] = BIG, f= f+1Goto 1.

Step 2: Set the standard interval with length p.

Step 3: Consolidate all the jobs in single family, f.

Step 4: Determine the total volume of available jobs for the consolidated family, $\sum_{z=1}^{f} \sum_{x=0}^{m^r(z)} V^z{}_{xi}$;

a. If , $\sum_{z=1}^{f} \sum_{x=0}^{m^r(z)} V^z{}_{xi} \geq$ reactor capacity

Select the jobs among available jobs using bin packing algorithm, establish a batch.

b. If , $\sum_{z=1}^{f} \sum_{x=0}^{m^r(z)} V^z{}_{xi} <$ reactor capacity

Pull the jobs that are not released yet, to the interval start until the batch capacity is fully utilized or the last job that will be released before the interval end is pulled to the interval start.

Step 5: Calculate the total weighted flow time of jobs throughout the interval.

Step 6: If jobs are finished then sum all flow times calculated for each interval, LB[j,f] is found.
Else Goto 2.

Step 7: To repeat the same cycle for f+1 Goto 1.

This algorithm has O(n^2 log n) complexity, where n is total number of arrivals. In fact at the worst case the number of comparisons made to obtain a schedule is less than n^2 log n. At the worst case we will have n jobs simultaneously in an interval and sorting of these jobs will bring O(n log n) complexity to the algorithm. Since the number of intervals is at most equal to the number of arrivals we will have at most n intervals in a problem. Therefore the number of comparisons made is always less than n^2 log n. But it does not reduce the order of the complexity.

Proposition 3.1: Under given conditions FILB procedure leads always a lower volume weighted flow time than the restricted optimal schedule.
Proof:
Here we will prove that the family independent lower bound leads always to weighted flow time lower than a restricted optimal schedule by showing that when the job volumes, release times and other operational conditions are identical, job completions in lower bound is necessarily on or before their counterparts in an optimal schedule. For this proof we compared only the cases, where optimal schedule and FILB start at the same time to the processing of jobs. Because the cases in which the optimizing procedure starts at a later time are easier.
Since the size of the proof is rather high we demonstrated it in Appendix G.

Sample of Family Independent Lower Bound Procedure

Let the processing time of a batch, p, be 10 units and the capacity of the reactor be 100 units. The data of the problem is given in the following table.

Table 3.2: Data of the sample problem of family independent lower bound procedure

Job #	1	2	3	4	5	6	7	8	9	10	11	12
Rel.time	2	6	8	11	14	20	22	28	33	37	43	46
Family	1	2	1	1	2	2	1	2	2	1	2	1
Volume	50	36	48	21	24	30	44	52	32	33	41	37

The first interval is [2,12), the available jobs are 50 from family 1 and 0 from family 2. The only family that can be processed is family 1. Therefore label the lower bound of the family 2 as BIG. Start calculating the lower bound of family 1.

Flow time of the jobs throughout the interval is,

50*p + 36*6+48*4+21*1 = 929.

To represent the quantity processed throughout the interval, it is multiplied by p, where p = 10.

Total volume of jobs not processed by t=12 is 36+48+21=105.

The second interval is [12,22), the sum of volumes of available jobs is 105. Flow time of the jobs throughout the interval is,

100*p+5*10+24*8+30*2 = 1302.

Total volume of jobs not processed by t=22 is 5+24+30=59.

The third interval is [22,32), the sum of volumes of available jobs is 103. Flow time of the jobs throughout the interval is,

100*p+3*10+52*4 = 1238.

Total volume of jobs not processed by t=32 is 3+52 =55.

The fourth interval is [32,42), the sum of volumes of available jobs is 55.
Since the total volume of available jobs is less than the reactor capacity we examine the jobs, which will be released throughout the fourth interval. Then pull the jobs to the interval start time (this means taking the job into the batch without waiting its release time) starting from the nearest release of job to the interval start.
Rearrange the jobs throughout the interval as in the table below:

Table 3.3: Rearranged pattern of the consolidated jobs through interval [32,42) for LB[1,1]

Job #										10	11	12
Rel.time									32	37	43	46
Volume									100	20	41	37

Flow time of the jobs throughout the interval is,
100*p+20*5 = 1100
Total volume of jobs not processed by t=42 is 20.

The fifth interval is [42,52), the sum of volumes of available jobs is 20.
Rearrange the jobs throughout the interval as in the table below;

Table 3.4: Rearranged pattern of the consolidated jobs through interval [42,52) for LB[1,1]

Job #											11	12
Rel.time										42	43	46
Volume										98	0	0

Flow time of the jobs throughout the interval is,

98*p = 980.

At the end of the fifth interval (or fifth batch) all jobs are processed. So we can calculate the minimum total weighted flow time by the summation of all weighted flow times calculated for each interval. The result obtained gives the minimum bound of the problem if the job from family 1 is processed immediately.

LB[1,1] = 980+1100+1238+1302+929 = 5549.

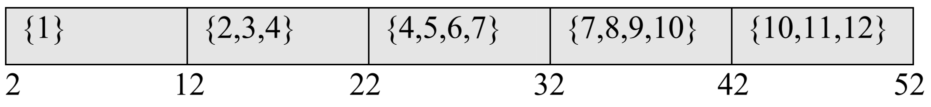

Figure 3.3: Schedule obtained for LB[1,1] of the sample problem using FILB procedure.

Another possibility in calculating the lower bound is accounting for the waiting time of the next job.

The first interval is [2,6) since the next arrival of job is at t=6; there exist 50 unit volume job from family 1.

Flow time of the job throughout the interval is,

50*(6-2) = 200.

Total volume of jobs not processed by t=6 is still 50.

The second interval is [6,16), the sum of volumes of available jobs is 86. Since it is smaller than the reactor capacity rearrange the jobs throughout the interval as given below.

Table 3.5: Rearranged pattern of the consolidated jobs through interval [6,16) for LB[1,2]

Job #		2	3	4	5	6	7	8	9	10	11	12
Rel.time		6	8	11	14	20	22	28	33	37	43	46
Volume		100	34	21	24	30	44	52	32	33	41	37

Flow time of the jobs throughout the interval is,

100*p+34*8+21*5+24*2 = 1425.

Total volume of jobs not processed by t=16 is 34+21+24=79.

The third interval is [16,26), the sum of volumes of available jobs is 79.

Rearrange the jobs throughout the interval as in the table:

Table 3.6: Rearranged pattern of the consolidated jobs through interval [16,26) for LB[1,2]

Job #						6	7	8	9	10	11	12
Rel.time					16	20	22	28	33	37	43	46
Volume					100	9	44	52	32	33	41	37

Flow time of the jobs throughout the interval is,

100*p+9*6+44*4 = 1230.

Total volume of jobs not processed by t=26 is 9+44=53.

The fourth interval is[26,36), the sum of volumes of available jobs is 53.

Rearrange the jobs throughout the interval as:

Table 3.7: Rearranged pattern of the consolidated jobs through interval [26,36) for LB[1,2]

Job #								8	9	10	11	12
Rel.time							26	28	33	37	43	46
Volume							100	5	32	33	41	37

Flow time of the jobs throughout the interval is,

100*p+8*5+32*3 = 1136.

Total volume of jobs not processed by t=36 is 5+32=37.

The fifth interval is [36,46), the sum of volumes of available jobs is 37.

Rearrange the jobs throughout the interval as:

Table 3.8: Rearranged pattern of the consolidated jobs through interval [36,46) for LB[1,2]

Job #										10	11	12
Rel.time									36	36	43	46
Volume									67	33	11	37

Flow time of the jobs throughout the interval is,

100*p+11*3 = 1033.

Total volume of jobs not processed by t=46 is 11+37=48.

The sixth interval is[46,56), the sum of volumes of available jobs is 48.

48*p = 480.

LB[1,3] = 480 + 1033 + 1136 + 1230 + 1425 + 200 = 5504.

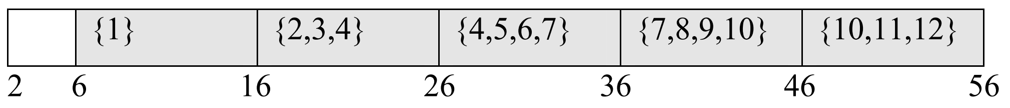

Figure 3.4: Schedule obtained for LB[1,3] of the sample problem using FILB procedure.

At the end of the fifth interval (or fifth batch) all jobs are processed. And lower bound for “waiting the next arrival” (= do nothing) is found less than starting to process 50 unit jobs immediately. In the branch and bound procedure branching the possibility with the minimum lower bound is the most promising choice. Therefore we branch towards the node that has “waiting the next arrival”.

3.5.2 Family Dependent Lower Bound Procedure (FDLB)

This procedure heavily depends on the family independent lower bound procedure. The main idea is the same with the family independent lower bound procedure, operating the reactor with minimum idle time and at maximum capacity. To increase the utilization of the reactor the following assumptions are relaxed:

- *jobs can be split to different batches, but the completion time of a job is equal to the completion time of its last portion.*
- *jobs arrive to the processor at arbitrary integer time points and in random order.*

These are also the two assumptions relaxed for obtaining the family independent lower bound procedure.

We set interval start time and interval end time for a batch. Interval start time is the first possible time when the machine is idle. As in the upper bound and the former lower bound procedures, interval length is equal to the batch processing time.

After setting the interval boundaries we determine the available jobs at the interval start for each family and the job releases in the interval. For the first interval we determine the available jobs according to their families and calculate the weighted flow times as in the optimal solution. Starting from the second interval we start to “pull” the jobs from the future arrivals. If the total volume of the available jobs from family f* is greater than or equal to the reactor capacity we start to process. If the total volume of the available jobs from family f*is less than the reactor capacity, we “pull “ the jobs from the interval by re-ordering the release times of the jobs. We “pull” the jobs until the volume of available jobs (i.e. jobs in the batch) reaches the reactor capacity. If the volume of the job to be pulled is greater than the remaining capacity, we pull a portion of the job, which is equal to the remaining capacity. But unlike the family independent lower bound procedure, we only pull the jobs from the same family; to be able to fulfill the maximum capacity requirement we also pull the jobs from future intervals.

Algorithm of Family Dependent Lower Bound Procedure

Step 1: Set the standard interval with length p, [t,t+p).

Determine the total volume of available jobs from family f, $\sum_{x=0}^{m^r(f,i)} V^f{}_{xi}$, at time t.

a. If $\sum_{x=0}^{m^r(f,i)} V^f{}_{xi} \geq$ reactor capacity

Select the jobs among the available jobs using bin packing algorithm provided that the lower bound refer to the restricted optimum as mentioned in section 3.3, establish a batch.

b. If $\sum_{x=0}^{m^r(f,i)} V^f{}_{xi} <$ reactor capacity

Establish a batch with available jobs from family f.

Calculate the total weighted flow time of jobs throughout the interval.

c. If $\sum_{x=0}^{m^r(f,i)} V^f{}_{xi} = 0$

Label LB[j,f] = BIG, f= f+1Goto 1.

Step 2: Set a standard interval with length p;

Step 3: Determine the total volume of available jobs, and the jobs that will be released throughout the interval for each family, f; $\sum_{x=0}^{m^*(f,i)} V^f{}_{xi}$ determine the release times of the jobs;

a. If $\sum_{x=0}^{m^*(f,i)} V^f{}_{xi} \leq$ reactor capacity

Take all jobs of family f and calculate the total weighted flow times of these jobs throughout the interval;

b. $\sum_{x=0}^{m^*(f,i)} V^f{}_{xi} >$ reactor capacity

Select the earliest released jobs of family f, whose total volume is equal to the batch capacity, $\sum_{x=0}^{m(f,i)} V^f{}_{xi}$;

Calculate the total flow time of the selected jobs throughout the interval;

Step 4: Choose the family f with the maximum flow time content calculated in step 3, label the family as f*;

Step 5: If total volume of available jobs of family f* < reactor capacity

Pull the jobs of family f*, that are not released yet, to the interval start until the batch capacity is fully utilized;

Step 6: Calculate the total flow time of all jobs throughout the interval.

Step 7: If jobs are finished then sum all flow times calculated for each interval, LB[j,f] is found;

Else Goto 2;

Step 8: To repeat the same cycle for f+1 Goto 1.

As in FILB procedure the complexity of this algorithm is O(n^2 log n), where n is the total number of jobs.

Proposition 3.2: Under given conditions FDLB procedure always leads to a lower volume weighted flow time than the restricted optimal schedule.

Proof:

This procedure takes care of the existence of families. Therefore it is more complicated than the family independent procedure. We will first prove that in case $BS^{z}_{i,opt} = BS^{z}_{i,LB}$ for $\forall z$ the lower bound procedure completes more job than the optimizing procedure if both of the procedures decide to process the same family f.

The second part is more complicated than the first part. We will show that in case of $BS^{z}_{i,opt} = BS^{z}_{i,LB}$ for $\forall z$ given that the optimizing procedure starts to process a batch of family f and lower bound procedure starts to process a batch of family g, the total weighted flow time accumulated by the lower bound at the end is less than that of optimal solution. For this comparison we selected a case at which the volume of completed jobs are less for the lower bound procedure. Because, all other cases dominate the optimal solution in terms of the total volume of completed jobs. But for this case the dominance in terms of total volume of completed jobs in the interval is violated by the optimizing procedure. Therefore this is the most advantageous case of optimizing procedure.

Since the size of the proof is rather high we demonstrated it in Appendix G.

Sample of Family Dependent Lower Bound Procedure

Let the processing time of a batch, p, be 10 units and the capacity of the reactor be 100 units. The data of the problem is as follows:

Table 3.9: Data of the sample problem of family dependent lower bound procedure

Job #	1	2	3	4	5	6	7	8	9	10	11	12
Rel.time	2	6	8	11	14	20	22	28	33	37	43	46
Family	1	2	1	1	2	2	1	2	2	1	2	1
Volume	50	36	48	21	24	30	44	52	32	33	41	37

The first interval is [2,12), the available quantities are 50 from family 1 and 0 from family 2. The only family that can be processed is family 1. Therefore label the lower bound of the family 2 as BIG. Start calculating the lower bound of family 1.

Flow time of the jobs in the interval is,

50*p + 36*6+48*4+21*1 = 929.

To represent the quantity processed in the interval, it is multiplied by p.

Over [12,22), the available quantities are 69 from family 1 and 36 from family 2.

The effective flow times of families are:

69 * 10 = 690 for family 1

36 * 10 + 24 * 8 + 30 * 2 = 612 for family 2,

choose family 1 to be processed in interval [12,22) as 690>612.

Re-arrange the arrivals as follows

Table 3.10: Rearranged pattern of the jobs through interval [12,22) for LB[1,1]

Job #					5	6	7	8	9	10	11	12
Rel.time			12	12	14	20	22	28	33	37	43	46
Family			1	2	2	2	1	2	2	1	2	1
Volume			100	36	24	30	13	52	32	33	41	37

Flow time of the jobs in the interval is,

100*p+36*10+24*8+30*2 = 1612.

Over [22,32), the available quantities are 13 from family 1 and 90 from family 2.

The effective flow times of families are:

13 * 10 = 130	for family 1
90 * 10 + 10 * 4 = 940	for family 2,

choose family 2 to be processed in interval [22,32).

Re-arrange the arrivals as follows

Table 3.11: Rearranged pattern of the jobs through interval [22,32) for LB[1,1]

Job #							7	8	9	10	11	12
Rel.time						22	22	28	33	37	43	46
Family						2	1	2	2	1	2	1
Volume						100	13	42	32	33	41	37

Flow time of the jobs in the interval is,

100*p+13*10+42*4= 1298.

Over [32,42), the available quantities are 13 from family 1 and 42 from family 2.

The effective flow times of families are:

13 * 10 + 33 * 5= 295	for family 1
42 * 10 + 32 * 9 = 708	for family 2,

choose family 2 to be processed in interval [32,42).

Re-arrange the arrivals as follows

Table 3.12: Rearranged pattern of the jobs through interval [32,42) for LB[1,1]

Job #										10	11	12
Rel.time								32	32	37	43	46
Family								1	2	1	2	1
Volume								13	100	33	15	37

Flow time of the jobs in the interval is,

100*p+13*10+37*5= 1315.

Over [42,52), the available quantities are 46 from family 1 and 0 from family 2.

The effective flow times of families are:

46 * 10 + 37 * 6= 682	for family 1
15 * 9 = 135	for family 2,

choose family 1 to be processed in interval [42,52).

Re-arrange the arrivals as follows

Table 3.13: Rearranged pattern of the jobs through interval [42,52) for LB[1,1]

Job #											11	
Rel.time									42	42	43	
Family									1	2	2	
Volume									83	0	15	

Flow time of the jobs in the interval is,

83*p+15*9= 965.

Over [52,62), the available quantities are 0 from family 1 and 15 from family 2.

The effective flow times of families are:

0	for family 1
15 * 10 = 150	for family 2,

choose family 2 to be processed in interval [52,62).

15*p = 150.

LB[1,1] = 929 + 1612 + 1298 + 1315 + 965 + 150 =6269.

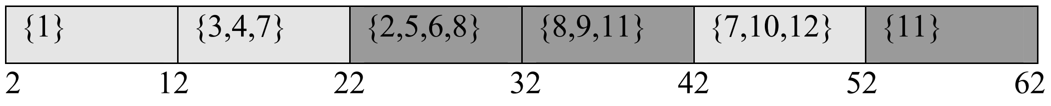

Figure 3.5: Schedule obtained for LB[1,1] of the sample problem using FDLB procedure.

Another possibility is waiting for the next job. Start calculating the Lower bound of “wait” option.

The first interval is [2,6),

Flow time of the jobs in the interval is,

50*4 = 200.

Over [6,16), the available quantities are 50 from family 1 and 36 from family 2.

The effective flow times of families are:

50 * 10 + 48 * 8 + 2 * 5= 894	for family 1
36 * 10 + 24 * 2 = 408	for family 2,

choose family 1 to be processed in interval [6,16).

Re-arrange the arrivals as follows.

Table 3.14: Rearranged pattern of the jobs through interval [6,16) for LB[1,3]

Job #			2	4	5	6	7	8	9	10	11	12
Rel.time		6	6	11	14	20	22	28	33	37	43	46
Family		1	2	1	2	2	1	2	2	1	2	1
Volume		100	36	19	24	30	44	52	32	33	41	37

Flow time of the jobs in the interval is,

100*p+19*5+36*10+24*2= 1503.

For the other intervals the calculating procedure is similar to the procedure given above. Therefore we skip the remaining part of the detailed solution.

At the end of the interval [56,66) we obtain the following lower bound value for "wait" option

LB[1,3] = 200+1503+1366+1432+1045+480+150 = 6176

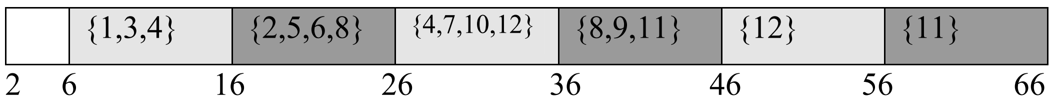

Figure 3.6: Schedule obtained for LB[1,3] of the sample problem using FDLB procedure.

The lower bound found for processing the job from family 1 was 6269 which is greater than 6176. According to the result we conclude that in node 1 “waiting the next arrival” may lead to a smaller total weightad flow time. Thus we select this option first to branch.

3.6 Branch and Bound Algorithm

After describing the upper bound and lower bound schemes we can give the algorithm of the branch and bound algorithm developed to solve this problem.
In this algorithm we have f+1 branches for each node. The first f branches represent the branching possibilities of jobs belong to any family and the last branch (f+1th branch) represents the possibility of waiting the next arrival (i.e., not making any batch). If there exist no available jobs from any family f, we label this branch by giving the lower bound value BIG to this family (described in lower bound procedures) and fathom it. As we said before, the branch and bound algorithm follows depth first strategy. To avoid multiple evaluation of a branch we label again that branch as BIG, when we pass to the next node. For example, let at node j we select family f to branch. When we create node j+1 we label LB[j,f] as BIG. So that we can distinguish the non-evaluated branches at node j when we come back to node j later. We can simply give the Branch and Bound algorithm that is developed to solve the problem as follows:

Step 1: Determine the upper bound value of the problem using Upper Bound Procedure. Use this upper bound value (and schedule) as the initial solution.

Step 2: Start the Forward Move Procedure. Increase the node number (j = j+1). Determine the time of the node (TNODE[j]= TNODE[j-1]+p or TNODE[j]= TNODE[j-1]+w, where w is the waiting time). Label LB[j-1,f] as BIG. At node j, determine the lower bound values of of each family and “wait” option using one of the lower bound schemes.

If, for any family f, the value of calculated lower bound is at least equal to the upper bound value, then fathom that branch (i.e., LB[j,f]= BIG)

Step 3: Choose the minimum value of lower bounds (i.e., minimum of LB[j,f] for all f) to branch. If the minimum lower bound value is less than the upper bound value and there exist unprocessed jobs at node j, then go to step 2.

Step 4: If all families have lower bound values at least equal to the upper bound value, then stop. Fathom all branches of node j (i.e., LB[j,f]= BIG for all f). Start the Backward Move Procedure to go to most recent alternative branching to node j.

If any family has lower bound value less than the upper bound value and all jobs are processed at node j then keep this solution as incumbent solution. Revise the upper bound value, upper is equal to the incumbent solution. Fathom the node j.

Start the Backward Move Procedure to go to most recent alternative branching to node j.

Step 5: If at any node j, any family f has the lower bound value less than the upper bound value then stop Backward Move Procedure. Go to step 2.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Experimental Design

In this chapter the figures for the computational experimentation are presented. We performed the experimentation twice for each problem set, with two different lower bound algorithms (i.e. family dependent and family independent schemes). Each problem set contains 10 different problems craeted randomly.We assumed that the reactor capacity is 100 unit volumes and the processing times of the jobs are 10 unit times.The problems are generated for:

- Three different job volume ranges: small (1-50 units), large (51-100 units) and mixed (1-100 units), where the volumes are selected uniformly.

- Three different arrival patterns, where all release times are selected using uniform distribution: tight or frequent; (0 to 1/3*processing time*job number), moderate (0 to 2/3*processing time*job number), loose (0 to processing time*jobnumber).
  The limits of the uniform distribution are selected such that we have interarrival times less than the processing time of a batch on average.

- 5 different arrival stream sizes ,i.e.number of jobs: 50 arrivals , 33 arrivals, 25 arrivals, 20 arrivals and 16 arrivals.

The figures of the job numbers are determined by considering the previously solved problems. There exist problems solved successfully for 25 jobs with 4 families or 30 jobs with 4 families in the literature. On the other hand these problems were all static arrival problems. Depending on the solved problems in previous studies we have expected that the parameters number of jobs and number of families have close relation with the computation time of the problem. Therefore we set the "jobnumber*family number" multiplication as the limit and we set it to 100 except the 16 jobs case. This was a conservative limitation considering the problems with 30 jobs and 4 families. On the other hand we could not predict the effects of dynamic nature of the problem in terms of solution time. With the increasing practice and results obtained we set the limit to 160 for the 16 jobs problems.

For each problem, the solution size (i.e total number of nodes) is limited with 2 million nodes. The program terminates the solution procedure at the 2 millionth node and assumes the last incumbent solution as the optimal solution.

4.2 Performance Measures

In this study the following data are collected for a problem:

- Run time
- Upper bound value.
- First incumbent solution
- Node of the first incumbent solution
- Optimal solution or the last incumbent solution
- Node number of optimal solution or node number of the last incumbent solution.
- Total nodes created to solve the problem.
- Lower bound at the root node

Depending on the data following performance measures are observed:

- Run Time: Solution time for a problem measured in CPU seconds. (Elap.time)
- Percentage deviation of the upper bound from the optimal solution or the last incumbent solution, whichever applies. (UB-Opt.)
- Percentage deviation of the first solution from the optimal solution or the last incumbent solution, whichever applies. (F.S.Opt.)
- Total nodes created to solve the problem.(TotalNode)
- Percentage deviation of the minimum lower bound at the root node from the optimal solution or the last incumbent solution. (LB-Opt.)
- The number of problems in which the upper bound procedure gives optimal solution.
- The number of problems that cannot be solved within node limit (i.e. 2000000 nodes).

4.3 Results and Discussion

4.3.1 Total Number of Nodes and Solution Times of the Problems

Problems with 50 arrivals

In this study the first 18 problem sets contain the problems with 50 job arrivals in total. First 6 sets are composed of the problems with small sized jobs (i.e. jobs of 1-50 unit volumes, S). First two sets are problems with “frequent” or “tight” arrivals (F); sets 3 and 4 are the problems with “moderate” or “average “arrivals (A) and the final two are the “loose” arrival problems (L). The second six sets are composed of the problems with large sized jobs, jobs of 51-100 unit volumes (B). The last six sets are with mixed sizes, jobs of 1-100 unit volumes (M). The intensity pattern is the same as in the first six sets.

Table 4.1: Average completion time of a problem with 50 jobs.

family	FDLB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
2	00:00:11	00:04:00	00:01:34	00:56:34	01:02:32	00:26:36	00:02:49	00:16:24	00:08:04
1	00:00:02	00:00:11	00:00:01	00:00:01	00:00:12	00:03:08	00:00:00	00:00:36	00:00:11
family	FILB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
2	00:00:18	00:03:24	00:01:29	02:20:44	00:42:48	00:15:30	00:03:08	02:37:06	00:06:27
1	00:00:02	00:00:08	00:00:01	00:00:01	00:00:08	00:01:53	00:00:00	00:00:30	00:00:09

When we observe the computation times on Table 4.1 it is apperant that sets with "large" job sizes constitute the majority of the time consuming problems. On the other hand the sets with "small" job sizes are relatively easy to solve when computation time is concerned.

In Table 4.2 it can be seen that the problem sets with 2 families dominate the problem sets with 1 family in terms of the problem size. Six of the problem sets have average total node number larger than 1 million and these are all the sets with two families. Considering Table 4.1, we can conclude that the sets with high solution times are the sets with large total number of nodes. But we cannot determine a direct proportionality of time requirement with total number of nodes. As in the case of set J50/F2/S/A or J50/F2/M/L (here we introduce a notation to specify the problem sets, J50/F2/S/A represents the problem set with 50 arrivals (J50), two families (F2), small job sizes (S) and average arrival rate(A)), these sets are larger in terms of the total number of nodes but smaller in terms of the computational time required. It can be explained by the amount of alternative solutions and the close differences among the values of alternative solutions. For the problems with large volume jobs, each arrival constitutes an alternative branching in an interval.

For problems with small volume jobs, the jobs can be combined easily in a batch; therefore the number of alternative branches reduces in great extent. As it can be seen the problems with small volumes and frequent arrivals are easier than the problems with small volumes and average arrivals. For the small volume problems, as the arrival rate decreases the possibility of combining different jobs in a batch also decreases. This case increases the length of the branch reaching a solution. But branching is not in increase.

On the other hand, for the problems with many alternatives the solution time increases due to the many forward and backward moves on the branch and bound tree. Beside to this increased branch length also increases solution time.

Table 4.2: Average of total number of nodes created for a problem with 50 jobs

family	FDLB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
2	23726,7	1222970	496208	2000000	1826198	1819328	780122	1456817	1490971
1	4982,3	114349	11620,3	531,8	15262,4	294749	265,5	118163	38345,1
family	FILB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
2	45289,7	1404221	762546	2000000	1654186	1827788	114642	1704969	1434488
1	5117,8	93105,2	10762,8	531,8	11313	182859	265,5	96415,3	34387,1

To be able to analyze the problems we require detailed information of the data sets. On the other hand due to the high variety of the parameters we cannot analyze all data sets in this chapter. Therefore we selected two of the sets and represented them in this chapter. Since the organization of data is tha same for all type of problems it will be sufficient to explore one of these tables.The tables can be found in Appendix 1 in complete. In this chapter we represented the tables of problem sets J50/F2/S/A and J50/F2/L/L as sample problems.

At the top of the table the it is possible to find the operational parameters about the problem such as number of jobs, number of families, limits of arrival time range and limits of volume range.

In these tables, “Node of F.S.” is the node number at which the first incumbent solution has found. If upper bound is the optimal solution or the size of the problem exceeds the node limit without any other solution, it becomes equal to

upper bound "Node of Inc." is the node number at which the incumbent solution has found. "Total Node" is the total number of nodes created to solve the problem. "UB-Opt." represents the deviation of the upper bound value from the optimal solution or the last incumbent solution. "LB-Opt." represents the deviation of the minimum lower bound value at the root node from the optimal solution. "F.S.-Opt." is the deviation of the first incumbent solution from the optimal solution. Finally "Elap.time" is the computation time of the problem.
Note that these figures are the averages of ten different problems. It is also possible to see the maximum and minimum values and standard deviation of the data.

Table 4.3: Performance measures of problem set J50/F2/S/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	44,9	222076,3	1222970,4	10,7	31,0	4,5	00:04:00
Max	49,0	770011,0	2000000,0	16,1	36,1	8,2	00:08:29
Min	43,0	5399,0	74677,0	6,4	28,0	2,5	00:00:19
St.Dev.	2,4	274227,0	823555,7	3,7	2,4	1,8	00:02:51
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	17208,4	588325,1	1404221,2	10,1	30,9	5,8	00:03:24
Max	166185,0	1670818,0	2000000,0	16,1	36,0	13,8	00:06:34
Min	41,0	265,0	120702,0	3,1	28,0	0,0	00:00:19
St.Dev.	52373,5	570159,2	766985,2	4,4	2,3	3,9	00:02:02

Table 4.4: Performance measures of problem set J50/F2/B/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	316988,4	695628,7	1819328,3	7,3	45,7	3,0	00:26:36
Max	1638853,0	1842868,0	2000000,0	23,8	51,7	7,8	01:21:53
Min	0,0	0,0	193283,0	0,0	40,8	0,0	00:06:08
St.Dev.	558284,1	648337,8	571334,1	9,3	3,2	3,2	00:21:04
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	374036,9	707780,1	1827787,9	7,2	37,9	4,4	00:15:30
Max	1972331,0	1973754,0	2000000,0	23,8	45,0	18,9	00:42:36
Min	0,0	0,0	277879,0	0,0	32,7	0,0	00:04:08
St.Dev.	694683,7	697910,5	544582,5	9,4	4,1	6,8	00:10:26

Problems with 33 arrivals

Table 4.5: Average completion time of a problem with 33 jobs.

family	Family Dependent								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
1	00:00:00	00:00:00	00:00:00	00:00:00	00:00:00	00:00:02	00:00:00	00:00:01	00:00:00
2	00:00:00	00:00:02	00:00:02	00:01:19	00:00:43	00:00:46	00:00:02	00:00:34	00:00:11
3	00:00:01	00:00:24	00:00:02	00:29:40	00:18:57	00:07:36	00:04:44	00:00:29	00:01:22
family	Family Independent								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
1	00:00:00	00:00:00	00:00:00	00:00:00	00:00:00	00:00:01	00:00:00	00:00:01	00:00:00
2	00:00:00	00:00:02	00:00:02	00:01:40	00:01:02	00:00:41	00:00:02	00:00:22	00:00:10
3	00:00:02	00:00:30	00:00:03	00:25:58	00:20:53	00:07:47	00:06:26	00:00:55	00:01:06

When we observe Table 4.5 it can be seen that four of the sets are dominating the others in terms of the computation times. These are J33/F3/B/F, J33/F3/B/A, J33/F3/B/L and J33/F3/M/F; all of them are 3 family sets. Moreover most time consuming problems are the problems with 3 families and "large" volumes.
When we compare the sets in three groups according to the job sizes (small, big [or large] and mixed), the easiest group is the group with "small" job sizes. The most time consuming problem set in this group is J33/F3/S/A and its average solution time is less than one minute (24 and 30 seconds respectively). Second group is the "mixed" job sized group and the last one is the group with "large" jobs. This pattern is in parallel with the pattern observed among the sets with 50 arrivals.

In Table 4.6 we can observe a pattern in parallel with the results of Table 4.5. There exist 3 sets, J33/F3/B/F, J33/F3/B/A and J33/F3/B/L, whose total number of nodes are above 1 million. The next 3 sets with relatively high total number of nodes are J33/F3/S/A, J33/F3/M/F and J33/F3/M/L.

Table 4.6: Average of total number of nodes created for a problem with 33 jobs

family	FDLB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
1	691,2	1538,8	1475,7	198,1	431,8	5801,9	190,4	2971,5	2007,3
2	817,8	18011,5	21109	104815	69576,2	118932	2912,1	137734	67503,4
3	2883,5	181842	15119	2000000	1307546	1398772	314318	71905,9	394529
family	FILB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
1	692,9	1633,2	1548,8	191,9	458,9	4295,3	232,6	3121,8	1707,8
2	1512,2	26063,4	28749	120136	104270	112690	3852,7	79369,3	73479,8
3	11343,9	388684	37329	2000000	1597138	1622653	460825	182914	441261

Moreover, except one case, among all the sets, the set with 2 families are dominated by its adjecant sets with 3 families. And for these sets the total number of nodes, that are obtained by the branch and bound method with the consolidated family (i.e. family independent) lower bound procedure, are higher than the that of obtained by branch and bound method with family dependent lower bound procedure.

The ranking made among the groups that are formed according to job sizes is also valid for the total number of nodes. First group is the sets of “ small” jobs, then “mixed” jobs’ sets and last one is the “large” jobs’ sets.

Problems with 25 arrivals

In Table 4.7 there exist four problem sets dominating the other sets in terms of the computation times. These are sets J25/F3/B/F, J25/F4/B/F, J25/F3/B/A and J25/F4/B/A. Compared to the dominating solutions of 50 arrival sets and 33 arrival sets, the computation times are shorter. But with the increasing family number the pattern has changed. In the previous sets, the most time requiring sets were the sets with “large” job sizes and the maximum number of families. On the other hand in this group of sets, J25/F4/B/L is dominated by J25/F3/B/F and J25/F3/B/A although the family counts are fewer than the number of families of set J25/F4/B/L. This can be explained by the increasing importance of the intensity of the arrivals compared to the number of families in the case of fewer numbers of arrivals with higher number of families.

Table 4.7: Average of completion time of a problem with 25 jobs.

family	FDLB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
2	00:00:00	00:00:00	00:00:00	00:00:14	00:00:05	00:00:07	00:00:00	00:00:01	00:00:00
3	00:00:00	00:00:01	00:00:00	00:06:14	00:02:46	00:00:12	00:00:08	00:00:01	00:00:01
4	00:00:00	00:00:02	00:00:00	00:12:27	00:03:51	00:00:36	00:00:15	00:00:05	00:00:03
family	FILB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
2	00:00:00	00:00:00	00:00:00	00:00:16	00:00:06	00:00:07	00:00:00	00:00:01	00:00:00
3	00:00:00	00:00:01	00:00:00	00:10:32	00:02:41	00:00:10	00:00:12	00:00:02	00:00:01
4	00:00:00	00:00:02	00:00:01	00:09:35	00:04:38	00:00:30	00:00:31	00:00:07	00:00:02

In Table 4.7 we can differentiate the decline of time requirement among sets with decreasing arrival rate per time. This decline is demonstarting the relation between the arrival rate and the size of the problem. We can conclude that the problem size increses with the increasing arrival rate of jobs (i.e. shorter the inter arrival times of jobs, harder to solve the problem). Ranking of the groups in terms of the computation time is also same. “Small” jobs are easiest, “mixed” jobs are harder and “large” jobs are the hardest.

Since the pattern of the Table 4.8 is almost the same with Table 4.7 we do not make any comments on it We just notice that the solution times are lower compared to the problems with 50 arrivals or 33 arrivals.

Table 4.8: Average of total number of nodes created for a problem with 25 jobs

family	FDLB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
2	353,4	1929,4	687,9	36599,1	13592,6	31422,7	938,8	3686,3	2239,5
3	223,4	7340,1	2561	774153	366485	55584,1	19069,6	5078,6	7611,8
4	544,4	14306,2	4450,3	1542964	611967	126725	32157,9	26974,6	19166,4
family	FILB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
2	534,1	2924	972,6	40500,9	16059,2	35120,8	1078,4	4687,7	2361,4
3	522,1	19635,8	4497,7	1313164	388666	60284,9	29518,7	10856,7	11148,2
4	1943,6	33813,8	11187,9	1672633	873768	182782	83569,4	72605,7	25574,9

Problems with 20 arrivals

Table 4.9: Average of completion time of a problems with 20 jobs

family	FDLB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
3	00:00:00	00:00:00	00:00:00	00:00:33	00:00:06	00:00:01	00:00:01	00:00:00	00:00:00
4	00:00:00	00:00:00	00:00:00	00:02:35	00:02:20	00:00:01	00:00:01	00:00:01	00:00:00
5	00:00:00	00:00:00	00:00:00	00:06:14	00:02:06	00:00:17	00:00:07	00:00:00	00:00:01
family	FILB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
3	00:00:00	00:00:00	00:00:00	00:00:40	00:00:11	00:00:01	00:00:01	00:00:00	00:00:00
4	00:00:00	00:00:00	00:00:00	00:02:54	00:02:12	00:00:01	00:00:01	00:00:01	00:00:00
5	00:00:00	00:00:01	00:00:00	00:05:12	00:01:46	00:00:28	00:00:12	00:00:00	00:00:01

Table 4.9 show great similarity with Table 4.7; the time requiring job sets or sets with high total numbers are collected in the group of sets including large jobs. J20/F5/B/F is the most time requiring problem type among 20 arrivals problems. The sets J20/F4/B/F, J20/F4/B/A and J20/F5/B/A are other most time requiring problem types. On the other hand, the solution times are much less than the problems with 33 jobs or 25 jobs. Here, we can conclude that the number of arrivals is an important factor affecting the solution time.

Table 4.10: Average of total number of nodes created for a problem with 20 jobs

family	FDLB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
3	148,3	476,1	526,7	110174	22861,4	7642,1	4879,8	1567	933
4	249	472,2	1342,8	576077	565558	9139,5	4330,8	3687,8	1259,6
5	110,1	2123,4	1694,3	1590161	529782	90690	33831,9	903,2	11143,5
family	FILB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
3	391,7	1413,8	953,5	134219	46510	10831,9	6248,6	2126,2	1534
4	819,3	1793,3	3436,2	682209	670071	12801,4	9274,2	9121,5	2883,1
5	982,4	14334	9403,4	1801356	585486	238208	72745,6	3602,7	30355,1

Table 4.10 is in parallel with the solutions of Table 4.9.

Problems with 16 arrivals

The final group of sets in this study is composed of 16 job problems. The distribution of sets in terms of intensity or job sizes is same as in the previous groups of sets. Whereas the family numbers are increased; they are 6,8 and 10. Besides, unlike the previous data sets the result of multiplication of number of jobs and number of families reach 160, much longer than its value of 100 for the other data sets.

Table 4.11: Average of completion time of a problems with 16 jobs

family	FDLB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
6	00:00:00	00:00:00	00:00:00	00:02:57	00:01:34	00:00:01	00:00:00	00:00:00	00:00:00
8	00:00:00	00:00:00	00:00:00	00:04:01	00:00:16	00:00:02	00:00:00	00:00:00	00:00:00
10	00:00:00	00:00:00	00:00:00	00:04:05	00:00:34	00:00:01	00:00:01	00:00:00	00:00:00
family	FILB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
6	00:00:00	00:00:00	00:00:00	00:01:57	00:00:57	00:00:01	00:00:00	00:00:00	00:00:00
8	00:00:00	00:00:00	00:00:00	00:01:55	00:00:26	00:00:01	00:00:01	00:00:00	00:00:00
10	00:00:01	00:00:00	00:00:01	00:02:08	00:00:51	00:00:03	00:00:04	00:00:00	00:00:00

In Table 4.11 we observe that set J16/F6/B/F, J16/F8/B/F and J16/F10/B/F dominates all sets in terms of the completion time. In Table 4.7 we had noticed that set J25/F3/B/F dominate the completion time of set J25/F4/B/A, although its family number is less than set J25/F4/B/A. This situation repeated also for the sets J20/F4/B/F and J20/F5/B/A. Finally, we observed above that not only set J16/F8/B/F but also set J16/F6/B/F dominates set J16/F10/B/A. Here we can conclude that as the number of familes increases and the number of jobs per family decreases, the intensity of arrivals play more important role in computation time than the number of families.

For sets with 16 arrivals, the limit of multiplication of number of jobs and number of families reaches to 160; whereas the maximum completion time is still less than that of sets with 20 arrivals and other sets with higher number of arrivals.

Moreover for all sets, the completion time of branch and bound method using family dependent lower bound procedure is higher than the completion time of branch and bound method using family independent lower bound procedure.

Table 4.12: Average of total number of nodes created for a problem with 16 jobs

family	FDLB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
6	107	501,1	164,4	1187636	527036	7246,4	762,3	413,3	101,1
8	59,8	150,4	811,4	1429939	88752,2	13100,1	1503,8	490,6	799
10	42,5	174,1	2662,8	1508816	176913	4929,3	4611,2	323	182,6
family	FILB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
6	839,4	4365,7	1090,5	1440952	608091	25272,8	1848,2	2178,3	441,9
8	1845,7	7012,5	5835,6	1750469	327662	22671,3	19706,5	3615,5	3135,4
10	20327,1	12160,3	23713,9	2000000	796441	50659,7	69820,9	9231,2	3602,2

The results of Table 4.12 are in contrast of the result of Table 4.11. Because, as the number of families increases the quality of the results obtained by the family dependent lower bound become much better than the results obtained by the family dependent lower bound. Whereas, for the family dependent lower bound procedure, the number of comparisons and calculations also increases with the increasing number of families. Therefore the solution time obtained is higher than the solution time obtained by the family independent lower bound procedure.

4.3.2 Terminated problems

Among 126 sets there exists 6 sets, whose average completion time exceed 10 minutes and the number of the problems terminated in the set are at least 5 (the complete data on terminated problems can be found in Appendix B). These sets are J50/F2/B/F, J50/F2/B/A, J50/F2/B/L, J50/F2/M/A, J33/F3/B/F, and J33/F3/B/A. Besides there exist 7 sets, whose average completion time is less than 10 minutes but the number of problems terminated in the set are at least 5. These sets are J50/F2/M/L, J33/F3/B/L, J25/F4/B/F, J20/F5/B/F, J16/F6/B/F, J16/F8/B/F and J16/F10/B/F. The proposed branch and bound methods were not successful for these sets (i.e. solving the problems under given conditions).

When we look at these sets closer the first group is composed of the problems with high number of jobs arrived. Moreover 5 of the 6 sets are composed of large volume jobs, the only exception of this group is J50/F2/M/A, which is composed of mixed volume jobs; And 4 of these 5 sets are composed of problems with tight interarrival time. The second group contains seven jobs and there exist one set from each problem type with different job arrivals. As in the first group, these jobs are mainly composed of large volume jobs; only J50/F2/M/L is a set with mixed volume jobs. 5 of these 7 sets again have tight interarrival times. Hence tight arrivals with large volumes relative to the reactor capacity make it harder to solve.

The difficulty of these types of problems originated from the nature of the branch and bound algorithm. When the jobs are large in volume and arrivals of these

jobs are tight, the problem size (i.e. total number of nodes) increases. There exist two possible reasons of this result. The first one is the length of a branch; in this case length of the branch, going in depth becomes longer than that of composed by small volume jobs. Two jobs (non-divided jobs) cannot be combined in a batch and finished at once. Therefore the sum of the volumes of the jobs from family f necessitate minimum $\lceil n_f /2 \rceil$ batches to be completed throughout the solution of a problem, where n_f represents the number of jobs from family f. On the other hand for the problems with low volume jobs the number of batches decreses because they can be combined in a batch and processed at once.
The second reason of the growth of problem size (i.e. total number of nodes) is the intensity of arrivals. Increase in the intensity of arrivals brings up alternative branching routes, especially in case of existence of multiple families. The increased number of job arrivals brings the high variety of families of existing jobs at a node. Therefore for that node f+1 new branches should be created, where f represents the count of existing families. If the intensity of arrivals is lower than one job per processing time of a batch, then the existence of multiple families will not be effective.

4.3.3 Performances of Upper and Lower Bound Procedures

At this part of our study we will focus on the success of the upper bound procedure and lower bound procedures.

Evaluation of Upper Bound Procedure

For the upper bound procedure we have two success criteria. First one is the number of problems in a set, whose optimal value is estimated by the upper bound procedure. The relevant figures can be found in Appendix C. The second one is the deviation of the upper bound from the optimal solution or last incumbent solution. The relevant figures can be found in Appendix A.

When we examine the figures of Appendix C we see that the success of upper bound procedure is fairly low for 50 arrivals case. There are only 5 problems out of 180, whose optimal solutions were found by the upper bound procedure and they are allocated to the different problem sets. 4 of these 5 sets are single family problem sets.

For the 33 arrivals sets the success of the upper bound procedure increases to 20 out of 270. Again majority of the sets are single family problems but we should note that there exist a small grouping relative to the other sets at sets 27,28 and 29. These are sets with small job volumes and loose interarrival times.

For 25 arrivals sets the number of problems, whose optimal solutions were found by the upper bound procedure, is the same 20 out of 270. On the other hand the problems are allocated in small groups. Therefore groupings can not be determined easily. Whereas we observe that 13 of 20 problems are from the problem sets with loose interarrival times.

For 20 arrivals sets the success of the upper bound procedure increases to 29 out of 270.
17 of them are from the problem sets with loose interarrival times. 9 are from the problem sets with moderate interarrival times and 3 of them are from the sets with tight interarrival times. We should note that there exists no problem at sets with tight interarrival times and large volume jobs.

Finally for 16 arrivals there exist 60 problems out of 270, whose optimal solution were found by the upper bound procedure. The allocations of number of the problems are as follows:

Table 4.13: Allocation of the problems, whose optimal solutions and upper bound values are equal, according to the volumes and interarrival times of jobs.

	small	large	mixed
tight	8	1	8
moderate	2	10	5
loose	2	20	4

For the evaluation of the performance of upper bound procedure, percent deviation of the upper bound value from the optimal solution is more reliable and proper way. Because we can expect that there exist one or two irregular problems in each set and these sets cannot represent the whole set. From this point on we will study the average percentage deviation of the upper bound from the optimal solution for each problem type.

Table 4.14:Average percent deviation of upper bound from the optimal solution for a 50 arrivals problem

family	FDLB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
2	7,4	10,7	3,2	1,9	11,5	7,3	10,5	13,9	5,6
1	11,9	6,5	5,6	0,6	4,8	2,5	3,5	9,6	7,5
family	FILB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
2	7,4	10,1	2,9	1,9	12,3	7,2	10,5	13,6	5,9
1	11,9	6,5	5,6	0,6	4,8	2,5	3,5	9,6	7,5

According to Table 4.14, among the 50 arrivals problem sets it is not possible to observe any trend; on the other hand we can say that the problems with moderate interarrival times have the largest deviation from the optimum solution. The largest deviation from the optimal is 13.88%. It is the set of problems with moderate interarrival times, mixed volume jobs and 2 families. The minimum is 0.63% and it is a single family set with tight arrivals and large volume jobs.

In Table 4.15 the deviations are again scattered such that we cannot conclude any trends in parallel with family number or intensity of interarrival times. But it would be not so wrong if we say that the deviations of the sets with loose arrivals are relatively low and the deviations of the sets with average arrival rates are high. Moreover the problems with tight interarrival times and large volumes have fairly low deviations.

Table 4.15:Average percent deviation of upper bound from the optimal solution for a 33 arrivals problem

family	FDLB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
1	7,8	8,2	2,7	0,6	2,5	2,8	7,0	9,5	3,9
2	10,3	7,9	3,7	3,2	12,8	6,8	9,1	12,7	4,3
3	12,2	8,6	7,6	3,0	12,2	7,3	6,7	15,5	5,8
family	FILB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
1	7,8	8,2	2,7	0,6	2,5	2,8	7,0	9,5	3,9
2	10,3	7,9	3,7	3,2	12,8	6,8	9,1	12,7	4,3
3	12,2	8,6	7,6	3,0	12,2	6,6	6,7	15,5	5,8

The largest deviation from the optimal is 15.48%. It is the set of problems with moderate interarrival times, mixed volume jobs and 3 families. The minimum is 0.64% and it is a single family set with tight arrivals and large volume jobs. This in parallel with the determination that we made for the same problem type using the number of problems, whose upper bound is equal to the optimal solution.

For the 25 arrivals problems and 20 arrivals problems we can generally say that there exist no siginificant change in these sets in terms of deviations and allocation of the most deviating and least deviating problems. The most deviating problems are again the problems with average arrival rates for all groups. Therefore we skip those to avoid repeating the same comments for those problem sets.

Table 4.16:Average percent deviation of upper bound from the optimal solution for a 16 arrivals problem

family	FDLB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
6	6,3	6,0	5,9	1,5	5,5	1,1	2,1	8,2	15,8
8	6,7	8,6	6,3	1,1	3,9	0,4	6,6	6,5	5,6
10	6,1	9,6	9,0	0,1	0,7	0,7	5,2	4,3	3,6
family	FILB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
6	6,3	6,0	5,9	1,4	5,5	1,1	2,1	8,2	15,8
8	6,7	8,6	6,3	1,0	3,9	0,4	6,6	6,5	5,6
10	6,1	9,6	9,0	0,0	0,7	0,7	5,2	4,3	3,6

Finally we investigate the sets with 16 arrivals. When we examine the Table 4.16 we see that all deviation except one, has decreased below 10%. On the other hand the maximum deviation rise to 15.79%. The deviations of the sets with large volume jobs are fairly low; in contrast the deviations of the sets with small volume jobs are relatively high.

At this point we conclude that the upper bound cause relatively less error for loose arrival problems and tight arrival problems with large volume jobs. On the other hand, we cannot point out any problem type easily for causing large deviations of upper bound from the optimal solution. Fixing a problem type as causing trouble is fairly difficult. But we can say that the upper bound procedure deviates more from the optimal solution for the problems with moderate interarrival times. In addition the small volume jobs have relatively higher deviations than the large volume jobs in general.

Evaluation of Lower Bound Procedures

After discussing the upper bound performance we want to discuss the performances of the lower bound procedures. This evaluation was made using the deviation of the lower bound value from the optimal solution.

Table 4.17:Average percent deviation of the minimum lower bound at the root node from the optimal solution for a 50 arrivals problem

family	FDLB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
2	38,5	31,0	23,2	3,9	28,7	45,7	8,8	45,0	32,3
1	33,1	19,9	15,7	3,5	19,1	36,2	8,2	32,8	28,1
family	FILB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
2	37,2	30,9	24,3	3,9	18,8	37,9	9,1	35,3	28,5
1	29,0	19,1	15,4	3,5	15,0	27,5	8,4	27,4	24,1

For 50 arrivals case (Table 4.17) the deviation of the lower bound from the optimal is rather high, but for the case with tight arrivals and large volume jobs it is fairly low, about 4%. Similarly it is relatively low (~8.5%) for the problems with tight arrivals and mixed volume jobs. Maximum deviation is 45.71% for FDLB procedure and 37.9% for the FILB procedure.

According to the Table 4.18 again the problems with tight arrivals and large and mixed volume jobs have relatively low percent deviations (on average ~6% for large volume problems and ~13% for mixed volume jobs.). On the other hand for those problems the deviations have increased in general. It may be a sign of that the deviation of the lower bound increases with the increasing number of families and decreasing number of arrivals. For 33 arrivals case the family independent lower bound procedure deviates less than the family dependent lower bound in general. Maximum deviation is 45.71% for family dependent LB procedure and 39.3% for the family independent LB procedure.

Table 4.18: Average percent deviation of the minimum lower bound at the root node from the optimal solution for a 33 arrivals problem

family	FDLB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
1	29,8	20,3	16,6	5,7	23,3	36,6	14,9	30,1	28,0
2	38,4	26,5	25,5	4,5	27,7	40,3	11,9	39,0	32,5
3	44,1	31,3	25,3	6,6	30,0	36,2	14,5	41,0	34,2
family	FILB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
1	25,4	19,1	16,3	5,6	18,5	28,0	12,5	28,4	22,1
2	38,3	27,5	26,1	4,6	24,0	33,2	10,9	33,8	27,7
3	44,6	34,8	26,9	7,0	25,0	30,8	15,5	38,9	31,2

Another observation is, for the problems with small volumes the deviations from the optimal value decreases with the decreasing arrival rate. In contrast, for the problems with large volumes the deviations from the optimal increase with decreasing arrival rate. There is no such a linear trend for mixed jobs.

Table 4.19: Average percent deviation of the minimum lower bound at the root node from the optimal solution for a 25 arrivals problem

family	FDLB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
2	38,0	25,6	19,2	6,8	23,1	38,2	14,7	31,9	28,8
3	36,3	28,8	25,4	7,7	23,9	33,8	15,4	38,0	28,2
4	36,2	29,2	27,6	9,0	34,7	39,7	15,2	43,8	34,1
family	FILB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
2	37,0	26,6	20,1	6,9	21,9	31,8	16,4	30,2	25,5
3	41,9	33,1	25,8	8,2	25,3	31,5	15,9	38,9	28,8
4	45,7	33,4	30,7	10,3	33,0	33,6	17,2	41,1	35,2

As in the upper bound, the results of the 25 arrivals problems and 20 arrivals problems have no siginificant change. Therefore we give only the deviation tables and skip the comments to avoid repeating the same comments for those problem sets. But we should note that for 25 jobs case and especially for 20 jobs case the family dependent lower bound procedure is superior to the family independent lower bound procedure in terms of deviation. The cause of this domination is the increasing number of families. We can expect that for 16 case problems it will be more significant.

Table 4.20: Average percent deviation of the minimum lower bound at the root node from the optimal solution for a 20 arrivals problem

family	FDLB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
3	33,2	30,2	22,3	8,1	25,2	34,5	17,1	34,2	27,7
4	35,1	28,7	24,6	10,4	24,9	34,9	21,0	25,6	30,6
5	33,9	37,5	26,9	11,3	23,4	34,0	18,7	28,1	34,4
family	FILB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
3	42,2	33,5	24,6	8,5	26,5	33,0	20,9	34,3	26,1
4	46,7	38,0	26,9	11,5	29,7	35,5	23,9	31,5	29,6
5	48,8	41,6	30,1	12,7	29,2	33,5	22,0	40,3	34,0

Table 4.21: Average percent deviation of the minimum lower bound at the root node from the optimal solution for a 16 arrivals problem

family	FDLB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
6	29,7	32,6	24,2	13,0	22,6	29,8	13,2	25,7	19,7
8	21,1	33,0	26,7	12,0	20,0	26,2	17,9	29,1	25,2
10	14,8	30,4	20,6	10,1	17,6	21,8	10,4	23,6	17,8
family	FILB								
	Small			Big			Mixed		
	F	A	L	F	A	L	F	A	L
6	50,4	42,7	29,2	16,0	31,7	37,1	24,6	34,1	25,2
8	52,0	49,9	37,6	19,7	32,5	35,5	39,9	45,3	36,1
10	60,3	48,6	33,3	18,3	39,0	41,1	33,0	46,1	26,3

The last group of problems is the problems with 16 arrivals. When we observe the deviations from the optimum solution in Table 4.21, we can easily distinguish that the family independent lower bound procedure is not suitable for the problems with high number of families. On the other hand, we can observe a slight decrease in deviations for problems solved by using the family dependent lower bound scheme.

At the end of the examination of deviations of the results obtained by upper bound scheme and lower bound schemes, we can conclude that lower bound schemes are effective for the tight arrivals in large volume job problems. Considering the solution times and total number of nodes created these problems are difficult to solve. In that respect we can find the lower bound procedures successfull. On the other hand there exists very high deviations from the optimal solution for the rest of the problems. In that respect the success of the lower bounds are not so good. By looking at the figures it is not possible to relate the difficult problems with the deviations from the optimum solution. Most problems with relatively high deviations from the optimum are solved within seconds, for example set J25/F3/M/A or set J16/F6/M/L. On the other hand the problems such as set J50/F2/B/F, and J33/F3/B/F are terminated at the 2 millionth node although the lower bound and upper bound deviations are low. It is obvious that with better performing upper and lower bound procedures the problems could be solved more efficiently but the main difficulty of the problem is originated from its combinatorial nature. As in the set J50/F2/B/F or J33/F3/B/F as the number of arrivals increases the problem size increases very high although the gap between the upper bound value and lower bound value is relatively small. In contrast when the number of arrivals is low, as in J25/F3/M/A or J16/F6/M/L, the number of total nodes become small although the gap between the lower bound value and upper bound value is large.

On the other hand, upper bound procedure and lower bound procedures have also many weaknesses. Unfortunately we could only notice two of them for the upper bound procedure. The first weakness of the upper bound algorithm is related with its focus length. It focuses only to an interval whose length is limited with

the batch processing time. But, as it is stated before, the nature of the problem is combinatorial. Therefore it cannot capture the flow time decreases, which can be possible by a wider view horizon. To make the statement more clear we will show it by an example.
Let the processing time of a batch be 10 and reactor capacity be 100. There exist three jobs with volumes 80, 79, 20 and with release times 0, 0, and 10 respectively. Finally the families of these jobs are 1, 2 and 1 respectively. According to the upper bound algorithm the sequence should be as follows: 80 of family 1 is processed in interval [0,10), 79 of family 2 in interval [10,20) and finally 20 of family 1 in interval [20,30). By this sequence total flow time would be 80*10 + 79 * 20 + 20 * 20 = 2780. But with an alternative sequence, which is not considered by the upper bound procedure, total flow time could be reduced. Let 79 of family 2 be processed in interval [0,10). Then in interval [10,20) the jobs from family 1 is processed simultaneously (i.e 80+20). By this sequence total flow time would be: 79 * 10 + 80 * 20 + 20*10 = 2590.

Another weakness of the upper bound algorithm is related with its selection method of jobs into a batch. It evaluates the jobs according to their job numbers and if it is worth the job is taken into the batch. Then the next job is evaluated. On the other hand if the remaining capacity is not sufficient to take the job it is skipped and the next one is evaluated. This is also another reason of deviation. Its predecessors hinder the jobs that can complete the remaining capacity. For example, with the same conditions given above, there exist a job with 60 unit volume at the beginning of the time interval, say 0; then 10 unit-volume job is released at time 1 and 35 unit-volume job is released at time 2. Upper bound algorithm will take the 10 unit-volume job and let the 35 unit-volume job aside due to the remaining capacity constraint. But taking the 35 unit-volume job would be more efficient in terms of capacity utilization and unit time utilization.

Unlike the upper bound procedure, for the lower bound procedures the sources of deviation are apperant. These are the relaxed constraints to obtain a lower bound procedure. Therefore these constraints will not be discussed. But we can say using our observations that the "rearrangement of the release times" or "pulling the future arrivals" causes most of the deviation in two ways. Since we do not move from the interval start, flow time that should be charged due to the movement in time axis, is not charged. Moreover for the family dependent lower bound procedure we can pull the jobs from the distant future intervals. This relaxation causes high deviations especially for the problems with small volumes and loose interarrival times.

Finally we want to discuss why the problems with small job sizes are easy to solve although the performance of the lower bound procedures are quite bad for these types of problems.

As we mentioned before, for the problems with high arrival rates and small volumes we can easily combine the jobs in a batch. Moreover the alternative branches are limited for those types of problems. This was the reason of short solution times. On the other hand this small volumes also causes an increase in the deviation of the lower bound value, when it is considered with the pulling action. Since the jobs are small we can combine several jobs in a batch by pulling them to an interval start. In fact the combination of jobs may not be possible for a feasible solution. But it is not a problem for the lower bound procedure since we don't wait these jobs. As a result we may count only the processing times of the future jobs, which causes a great deviation from the optimal solution.

The problems with high volumes and frequent arrivals do not face with such a problem since they fill the reactor capacity more efficiently than the small jobs. Therefore the deviations are smaller compared to the small volume problems.

4.3.4 Performance of Proposed Branch and Bound Algorithm

To investigate the efficiency of the proposed branch and bound algorithm, we solve the sample problem using LINDO linear optimization package. To limit the solution time we set the iteration limit as 1000000.

The solution time of the problem, using our branch and bound algorithm, was less than 0.5 seconds. The solution obtained using the proposed branch and bound algorithm is given in section 3.4.

The general purpose linear optimization package (LINDO) made 1000000 iterations in 9.01 minutes and obtained the following schedule with the objective function value 33464.

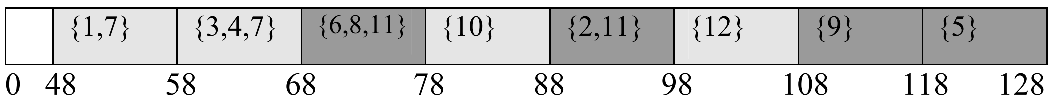

Figure 4.1: Schedule obtained for the sample problem using LINDO optimization package.

By this comparison we can conclude that the proposed branch and bound algorithm is superior to a general purpose optimization package.

CHAPTER 5

CONCLUSION

In this study we developed a branch and bound algorithm for single machine, dynamic arrival, batch scheduling problem with incompatible families of job. Our objective was minimizing total volume weighted flow time of the jobs measured at entire job completion. To the best of our knowledge this study is the first attempt to solve the problem by a branch and bound algorithm. To solve the problems more efficiently, we proposed a heuristic method to set an upper bound to our problem and two alternative lower bound procedures to limit the branching of the problem.

The main idea of the upper bound was determining an interval, whose length is equal to the processing time of a batch and maximizing the volume of processed jobs per unit time within this interval.

Both of the lower bound procedures have the same idea in general. At a point, where available jobs exist and machine is idle, we can have a batch start and we can maximize the volume of processed jobs by pulling the jobs from the future arrivals to the present time disregarding their actual release times (i.e. to the node described above). We called the first proposed lower bound procedure as "family independent lower bound procedure" or " consolidated family lower bound procedure". This procedure consolidates all jobs in a single family. Then start to pull these jobs to the node up to its capacity is full or until the jobs, that will be released in a processing time length interval, will be finished. The second lower bound procedure does not consolidate the jobs. It determines an interval, and then decides on the family for processing. Then start to pull the jobs from that family to the node up to its capacity is full. Unlike the first lower bound it is not limited with the length of the interval.

Using these upper bound and lower bound procedures we performed extensive computational experimentation. For this experiment we solved 126 different problem sets each containing 10 randomly generated problems. These problems are prepared for 3 different job arrival intensities, 3 different ranges of job volume, 5 different total number of job arrivals and 3 different total number of families for each of the five different number of job arrivals. Based on the experimental results we can conclude the following:

For any problem group with given number of jobs, problems with small jobs are simpler than problems with relatively large jobs. Mixing large and small jobs in a problem reduces the solution times compared to problems with large jobs. Given that 10 minutes solution time is reasonable for any problem, the problems with small jobs can be solved effectively by the branch and bound method using either one of the lower bound procedures. For the problems with mixed jobs in terms of size, the proposed branch and bound algorithm can also be used. But we should consider the set 15, which has an exceptionally long solution time by the branch and bound method using the family independent lower bound procedure and fairly long solution time for the branch and bound method using the family dependent lower bound procedure.

The most difficult problems are those with large job sizes. Especially the problems with the highest number of families and the tightest job arrivals are the most difficult problems. Therefore most of these problems are terminated before reaching a guaranteed optimal solution.

The number of job arrivals is the most important parameter of the problem. As the number of job arrivals increase, solution time increases exponentially. Considering the solutions obtained, the number of job limit should be less than 33 for the most difficult problems.

Keeping all other conditions constant, solution time increases with the decreasing interarrival times of the jobs.

Number of families and intensity of the arrivals are two factors increasing the solution time. The family number is effective for problems with small family number and high job arrivals whereas the intensity of the arrivals is more important for the problems with higher family number and relatively lower job arrivals.

When we take 5% deviation from the optimal value as a limit, the performances of upper bound and lower bound procedures are quite bad. Especially lower bound procedures have very poor performances. Therefore, we should state that the solution qualities are low for these procedures. On the other hand, the lower bound procedures have relatively less deviations from the optimal for most of the difficult problems.

The proposed branch and bound algorithm became unsuccessful for some of the problems. Two main characteristics of these problems are large job volumes with frequent interarrival times and high number of job arrivals. A study focusing on these types of problems may be one of the near future research area. This study is conducted on relying on the property that all the jobs have equal processing times. A study assuming different processing times would be another future research area. In this study weights of jobs are directly proportional to their volumes; a study with arbitrary weights of jobs is another future research area. Another extension of this study may include different objective functions as total (weighted) lateness or tardiness. Beside these a further study may include multiple parallel machines or multi stage processing. Of course a study, which is a combination of these areas, is also possible.

REFERENCES

[1] Ahmadi J.H., Ahmadi R.H., Dasu S., Tang C.S. (1992); Batching and scheduling jobs on batch and discrete processors; Operations Research Vol.39 No. 4, pp. 750-763

[2] Azizoğlu M.,Webster S.(2001); Scheduling a batch processing machine with incompatible job families; Computers and Industrial Engineering 39, pp.325-335

[3] Blazewicz J., Ecker K.H., Pesch E., Schmidt G. Weglarz J.; Scheduling computer and manufacturing processes; Springer Verlag, Berlin, (1996).

[4] Brucker P.,Gladky A., Hoogeveen H.,Kovalyov M.Y.,Potts C.N.,Tautenhahn T.,Van de Velde S.L.,(1998); Scheduling a batching machine; Journal of Scheduling 1, pp.31-54

[5] Chandru V.,Lee C.Y.,Uzsoy R.(1993); Minimizing total completion time on batch processing machines; International Journal of Production Research 31, pp.2097-2121

[6] Cheng T.C.E.,Chen Z.L.Oğuz C.(1994); One machine batching and sequencing of multiple type items; Computers and Operations Research 21, pp.717-721

[7] Cheng T.C.E.,Kovalyov M.Y.(2001);Single machine batch scheduling with sequential job processing; IIE Transactions 33, pp.413-429

[8] Dobson G.,Karmarkar U.S.,Rummel J.L.(1987);Batching to minimize flow times on one machine; Management Science 33, pp.784-799

[9] Dobson G., Nambimadom R.S.(1992);The batch loading and scheduling problem; Working paper QM 92-03 University of Rochester

[10] DuPont L, Ghazvini F.J. (1997); A branch and bound algorithm for minimizing mean flow time on a single batch processing machine; International Journal of Industrial Engineering 4(3), pp. 197-203

[11] DuPont L., Flipo C.D. (2002); Minimizing the makespan on a batch machine with non identical job sizes: an exact algorithm; Computers and Operations Research 29, pp. 807-819

[12] Ghazvini J.G., Dupont L. (1998); Minimizing mean flow times criteria on a single batch processing machine with non-identical job sizes; International Journal of Production Economics 55, pp. 273-280

[13] Hochbaum D.S.,Landy D.,(1997); Scheduling semiconductor burn-in operations to minimize total flow time;Operations Research 45, pp. 874-885

[14] Ikura Y., Gimple M. (1986);Efficient scheduling algorithms for a single batch processing machine;Operations Research Letters 5, pp. 61-65

[15] Mehta S.V., Uzsoy R. (1998); Minimizing total tardiness on abatch processing machine with incompatible job families; IIE Transactions 30, pp. 165-178

[16] Potts C.N., Kovalyov M.Y. (2000); Scheduling with batching: A review ; European Journal of Operational Research 120, pp. 228-249

[17] Santos C., Magazine M.(1985);Batching in single machine manufacturing systems; Operations Research Letters 4, pp 99-103

[18] Sung C.S., Choung Y.I., Hong J.M., Kim Y.H. (2002); Minimizing makespan on a single burn-in oven with job families and dynamic job arrivals; Computers and Operations Research 29, pp. 995-1007

[19] Sung C.S.,Joo.U.G.(1997); Batching to minimize weighted mean flow time on a single machine with batch size restrictions; Computers and Industrial Engineering 32, pp 333-340

[20] Sung C.S., Yoon S.H. (1997); Minimizing maximum completion time in a two batch processing machine flowshop with dynamic arrivals allowed; Engineering Optimization Vol.28 pp.231-243

[21] Uzsoy R.(1995); Scheduling batch processing machines with incompatible job families; International Journal of Production Research 33, pp 2685-2708

[22] Webster S., Baker K.R. (1995); Scheduling groups of jobs on a single machine; Operations Research vol. 44 No. 4, pp. 692-703

APPENDIX A

COMPUTATIONAL RESULTS OF BRANCH AND BOUND ALGORITHM

Table A 1 through Table A 126 show the computational results of the branch and bound algorithm, percent deviations of upper bound values from the optimal solution, percent deviations of lower bound values from the optimal solution and percent deviations of first incumbent solution from the optimal solution.

Table A.1: Detailed performance measures of branch and bound algorithm for problem set J50/F2/S/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	400,6	4449,7	23726,7	7,4	38,5	4,0	00:00:11
Max	3197,0	18024,0	63175,0	17,7	46,4	7,6	00:00:31
Min	0,0	0,0	1543,0	0,0	25,3	0,0	00:00:01
St.Dev.	990,5	6339,5	19583,7	4,8	6,9	2,5	00:00:11
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	1140,6	13134,3	45289,7	7,4	37,2	4,5	00:00:18
Max	10329,0	100119,0	119318,0	17,7	42,3	10,7	00:00:57
Min	0,0	0,0	2342,0	0,0	29,1	0,0	00:00:01
St.Dev.	3235,1	31164,0	41856,1	4,8	4,3	3,1	00:00:19

Table A.2: Detailed performance measures of branch and bound algorithm for problem set J50/F1/S/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	42,7	1379,3	4982,3	11,9	33,1	2,7	00:00:02
Max	135,0	3946,0	25137,0	22,4	41,8	11,1	00:00:09
Min	22,0	48,0	324,0	3,6	26,2	0,4	00:00:00
St.Dev.	39,3	1648,7	7347,4	6,1	5,3	3,2	00:00:02
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	112,7	2549,7	5117,8	11,9	29,0	4,1	00:00:02
Max	840,0	14861,0	23495,0	22,4	33,2	8,0	00:00:08
Min	22,0	62,0	433,0	3,6	24,7	1,1	00:00:00
St.Dev.	256,1	4400,5	6729,8	6,1	2,7	2,0	00:00:02

Table A.3: Detailed performance measures of branch and bound algorithm for problem set J50/F2/S/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	44,9	222076,3	1222970,4	10,7	31,0	4,5	00:04:00
Max	49,0	770011,0	2000000,0	16,1	36,1	8,2	00:08:29
Min	43,0	5399,0	74677,0	6,4	28,0	2,5	00:00:19
St.Dev.	2,4	274227,0	823555,7	3,7	2,4	1,8	00:02:51
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	17208,4	588325,1	1404221,2	10,1	30,9	5,8	00:03:24
Max	166185,0	1670818,0	2000000,0	16,1	36,0	13,8	00:06:34
Min	41,0	265,0	120702,0	3,1	28,0	0,0	00:00:19
St.Dev.	52373,5	570159,2	766985,2	4,4	2,3	3,9	00:02:02

Table A.4: Detailed performance measures of branch and bound algorithm for problem set J50/F1/S/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	41890,3	50773,2	114349,4	6,5	19,9	2,1	00:00:11
Max	417619,0	417619,0	690192,0	14,6	24,3	6,5	00:01:03
Min	42,0	49,0	1682,0	0,0	14,7	0,0	00:00:00
St.Dev.	132017,8	131620,7	217557,3	4,8	3,1	2,3	00:00:20
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	4258,2	12015,7	93105,2	6,5	19,1	2,2	00:00:08
Max	40939,0	70511,0	546303,0	14,6	22,9	6,3	00:00:44
Min	42,0	49,0	1671,0	0,0	14,5	0,0	00:00:00
St.Dev.	12894,0	24084,4	171396,1	4,8	2,7	2,4	00:00:14

Table A.5: Detailed performance measures of branch and bound algorithm for problem set J50/F2/S/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	386,6	258703,8	496207,6	3,2	23,2	1,6	00:01:34
Max	3360,0	1612563,0	2000000,0	5,8	28,0	4,0	00:06:18
Min	54,0	173,0	26167,0	0,7	18,1	0,3	00:00:08
St.Dev.	1044,7	494573,2	589794,1	1,7	2,8	1,2	00:01:46
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	83766,8	534069,5	762545,8	2,9	24,3	2,6	00:01:29
Max	327892,0	1828056,0	2000000,0	5,1	27,8	4,8	00:03:42
Min	54,0	43929,0	74556,0	0,7	19,7	0,5	00:00:12
St.Dev.	111056,4	632455,5	705611,6	1,6	2,9	1,6	00:01:13

Table A.6: Detailed performance measures of branch and bound algorithm for problem set J50/F1/S/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	1214,6	6227,5	11620,3	5,6	15,7	0,8	00:00:01
Max	11629,0	15380,0	23418,0	16,5	21,3	2,4	00:00:04
Min	50,0	50,0	635,0	0,3	11,6	0,0	00:00:00
St.Dev.	3659,2	6135,1	8138,0	4,9	2,5	0,8	00:00:01
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	57,1	3770,5	10762,8	5,6	15,4	1,5	00:00:01
Max	62,0	17241,0	21792,0	16,5	21,3	4,1	00:00:03
Min	49,0	55,0	650,0	0,3	11,6	0,0	00:00:00
St.Dev.	3,7	5620,9	7254,9	4,9	2,6	1,4	00:00:01

Table A.7: Detailed performance measures of branch and bound algorithm for problem set J50/F2/B/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	1048,2	445387,0	2000000,0	1,9	3,9	1,3	00:56:34
Max	9292,0	1792945,0	2000000,0	4,5	6,7	4,0	01:01:20
Min	38,0	3254,0	2000000,0	0,6	2,4	0,2	00:45:37
St.Dev.	2902,5	628669,1	0,0	1,2	1,3	1,1	00:04:19
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	1129,6	494656,4	2000000,0	1,9	3,9	1,3	02:20:44
Max	10099,0	1727831,0	2000000,0	4,5	6,8	4,0	02:37:10
Min	38,0	3347,0	2000000,0	0,6	2,4	0,2	01:55:24
St.Dev.	3157,0	663623,7	0,0	1,2	1,4	1,1	00:14:02

Table A.8: Detailed performance measures of branch and bound algorithm for problem set J50/F1/B/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	89,2	89,2	531,8	0,6	3,5	0,0	00:00:01
Max	574,0	574,0	998,0	2,6	4,8	0,0	00:00:01
Min	0,0	0,0	239,0	0,0	2,3	0,0	00:00:00
St.Dev.	170,8	170,8	229,6	0,9	0,7	0,0	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	89,2	89,2	531,8	0,6	3,5	0,0	00:00:01
Max	574,0	574,0	998,0	2,6	4,8	0,0	00:00:01
Min	0,0	0,0	239,0	0,0	2,3	0,0	00:00:00
St.Dev.	170,8	170,8	229,6	0,9	0,7	0,0	00:00:00

Table A.9: Detailed performance measures of branch and bound algorithm for problem set J50/F2/B/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	92,8	1097865,8	1826197,7	11,5	28,7	6,8	01:02:32
Max	507,0	1973721,0	2000000,0	23,8	47,8	12,4	01:41:44
Min	41,0	69991,0	624461,0	5,1	6,2	3,3	00:17:03
St.Dev.	145,8	653410,0	429073,5	5,5	14,1	3,7	00:26:05
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	43,0	979792,4	1654186,4	12,3	18,8	5,9	00:42:48
Max	47,0	1896772,0	2000000,0	24,2	35,8	9,7	00:58:20
Min	41,0	287368,0	463480,0	5,5	6,3	3,7	00:11:51
St.Dev.	1,8	586759,1	589180,9	5,1	8,6	2,1	00:16:02

Table A.10: Detailed performance measures of branch and bound algorithm for problem set J50/F1/B/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	124,0	7419,0	15262,4	4,8	19,1	0,5	00:00:12
Max	899,0	70947,0	139020,0	15,8	29,1	3,0	00:01:46
Min	0,0	0,0	331,0	0,0	9,3	0,0	00:00:00
St.Dev.	272,6	22325,4	43493,4	5,2	6,7	0,9	00:00:33
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	277,1	4381,2	11313,0	4,8	15,0	0,4	00:00:08
Max	2431,0	38484,0	99492,0	15,8	20,4	2,8	00:01:11
Min	0,0	0,0	331,0	0,0	9,3	0,0	00:00:00
St.Dev.	756,9	12010,9	30997,2	5,2	3,5	0,9	00:00:22

Table A.11: Detailed performance measures of branch and bound algorithm for problem set J50/F2/B/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	316988,4	695628,7	1819328,3	7,3	45,7	3,0	00:26:36
Max	1638853,0	1842868,0	2000000,0	23,8	51,7	7,8	01:21:53
Min	0,0	0,0	193283,0	0,0	40,8	0,0	00:06:08
St.Dev.	558284,1	648337,8	571334,1	9,3	3,2	3,2	00:21:04
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	374036,9	707780,1	1827787,9	7,2	37,9	4,4	00:15:30
Max	1972331,0	1973754,0	2000000,0	23,8	45,0	18,9	00:42:36
Min	0,0	0,0	277879,0	0,0	32,7	0,0	00:04:08
St.Dev.	694683,7	697910,5	544582,5	9,4	4,1	6,8	00:10:26

Table A.12: Detailed performance measures of branch and bound algorithm for problem set J50/F1/B/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	23664,2	186923,6	294748,6	2,5	36,2	2,2	00:03:08
Max	167922,0	729933,0	894765,0	3,9	45,8	3,6	00:09:43
Min	49,0	49,0	824,0	0,3	28,2	0,0	00:00:01
St.Dev.	51816,6	271612,7	303585,1	1,1	6,0	1,2	00:03:42
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	27843,4	129874,3	182858,8	2,5	27,5	2,1	00:01:53
Max	190170,0	481099,0	515377,0	3,9	40,3	3,8	00:07:13
Min	49,0	49,0	2979,0	0,3	17,5	0,0	00:00:02
St.Dev.	60558,1	160014,8	170746,7	1,1	6,5	1,1	00:02:11

Table A.13: Detailed performance measures of branch and bound algorithm for problem set J50/F2/M/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	28,9	71520,1	101346,1	10,5	8,8	4,3	00:02:49
Max	36,0	209846,0	354349,0	18,9	13,6	13,0	00:10:17
Min	26,0	11285,0	16477,0	3,5	3,8	0,2	00:00:29
St.Dev.	3,0	76240,8	112407,7	5,8	3,2	3,4	00:03:13
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	28,9	79904,2	114642,0	10,5	9,1	4,4	00:03:08
Max	36,0	227619,0	400229,0	18,9	14,2	13,0	00:11:26
Min	26,0	11340,0	22752,0	3,5	3,8	0,8	00:00:39
St.Dev.	3,0	83834,6	125827,9	5,8	3,4	3,4	00:03:33

Table A.14: Detailed performance measures of branch and bound algorithm for problem set J50/F1/M/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	25,6	32,7	265,5	3,5	8,2	0,0	00:00:00
Max	31,0	96,0	606,0	12,6	21,9	0,2	00:00:01
Min	0,0	0,0	70,0	0,0	3,6	0,0	00:00:00
St.Dev.	9,2	24,1	169,3	3,8	5,3	0,1	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	25,6	32,7	265,5	3,5	8,4	0,0	00:00:00
Max	31,0	96,0	606,0	12,6	24,5	0,2	00:00:01
Min	0,0	0,0	70,0	0,0	3,6	0,0	00:00:00
St.Dev.	9,2	24,1	169,3	3,8	6,1	0,1	00:00:00

Table A.15: Detailed performance measures of branch and bound algorithm for problem set J50/F2/M/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	78,5	402024,0	1456817,0	13,9	45,0	6,6	00:16:24
Max	433,0	1315138,0	2000000,0	45,5	51,8	16,9	00:38:20
Min	0,0	0,0	51360,0	0,0	38,4	0,0	00:00:44
St.Dev.	125,4	521873,7	757909,2	13,5	4,8	5,4	00:11:06
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	215561,2	932134,3	1704968,7	13,6	35,3	6,6	02:37:06
Max	1814109,0	1992927,0	2000000,0	45,5	43,4	16,7	07:19:19
Min	42,0	1261,0	74269,0	0,4	27,0	0,3	00:09:39
St.Dev.	571738,0	715074,7	611116,1	13,7	4,9	4,7	02:17:06

Table A.16: Detailed performance measures of branch and bound algorithm for problem set J50/F1/M/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	88,9	20057,1	118162,6	9,6	32,8	4,7	00:00:36
Max	567,0	130091,0	675693,0	16,8	44,0	13,3	00:02:36
Min	0,0	0,0	1379,0	0,0	23,0	0,0	00:00:01
St.Dev.	168,5	41049,6	212444,4	5,3	7,1	4,6	00:00:54
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	83,5	14702,5	96415,3	9,6	27,4	4,3	00:00:30
Max	515,0	49851,0	470932,0	16,8	40,2	12,6	00:01:50
Min	0,0	0,0	1321,0	0,0	19,4	0,0	00:00:01
St.Dev.	152,2	18981,6	153944,7	5,3	6,7	3,9	00:00:40

Table A.17: Detailed performance measures of branch and bound algorithm for problem set J50/F2/M/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	18823,4	392468,4	1490971,0	5,6	32,3	3,4	00:08:04
Max	151357,0	1590655,0	2000000,0	9,8	40,3	7,5	00:14:28
Min	0,0	0,0	119957,0	0,0	25,0	0,0	00:00:31
St.Dev.	47501,6	601635,4	815262,9	3,6	5,7	2,7	00:05:02
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	25435,8	796251,6	1434487,5	5,9	28,5	3,9	00:06:27
Max	118059,0	1894818,0	2000000,0	10,4	34,3	7,0	00:11:54
Min	0,0	0,0	84498,0	0,0	23,1	0,0	00:00:14
St.Dev.	48974,2	668881,4	801982,9	3,6	4,3	2,5	00:03:57

Table A.18: Detailed performance measures of branch and bound algorithm for problem set J50/F1/M/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	2218,3	9906,6	38345,1	7,5	28,1	2,2	00:00:11
Max	21616,0	31186,0	100879,0	14,2	38,8	5,7	00:00:33
Min	50,0	82,0	3640,0	0,3	19,0	0,2	00:00:01
St.Dev.	6815,7	12831,4	32553,5	5,2	6,4	1,8	00:00:10
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	1621,8	14331,7	34387,1	7,5	24,1	2,5	00:00:09
Max	15229,0	74202,0	91074,0	14,2	37,6	6,8	00:00:27
Min	49,0	88,0	3416,0	0,3	18,8	0,0	00:00:01
St.Dev.	4782,6	22399,5	29600,8	5,2	6,4	2,1	00:00:09

Table A.19: Detailed performance measures of branch and bound algorithm for problem set J33/F1/S/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	135,4	489,3	691,2	7,79	29,78	3,24	00:00:00
Max	1193	2104	2623	25,43	43,58	13,22	00:00:00
Min	0	0	22	0,00	16,88	0,00	00:00:00
St.Dev.	371,6806	686,9783	875,7334	7,52	8,54	4,22	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	62,6	461,9	692,9	7,79	25,36	3,01	00:00:00
Max	462	2746	2899	25,43	35,36	13,22	00:00:00
Min	0	0	22	0,00	16,21	0,00	00:00:00
St.Dev.	140,5435	834,9545	922,3825	7,52	6,31	4,25	00:00:00

Table A.20: Detailed performance measures of branch and bound algorithm for problem set J33/F2/S/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	165,2	498	817,8	10,27	38,37	4,28	00:00:00
Max	1317	2067	2156	19,01	46,29	10,24	00:00:01
Min	0	0	52	0,00	30,47	0,00	00:00:00
St.Dev.	408,7213	598,7776	738,2893	6,31	4,66	2,96	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	369,1	1139,4	1512,2	10,27	38,27	5,36	00:00:00
Max	2996	4673	4939	19,01	45,31	11,03	00:00:01
Min	0	0	61	0,00	31,07	0,00	00:00:00
St.Dev.	937,2549	1434,364	1546,977	6,31	3,57	3,38	00:00:00

Table A.21: Detailed performance measures of branch and bound algorithm for problem set J33/F3/S/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	84,1	2274,4	2883,5	12,20	44,07	7,45	00:00:01
Max	287	11696	13711	29,85	52,50	19,22	00:00:02
Min	17	75	568	0,57	28,22	0,00	00:00:00
St.Dev.	101,9754	3495,101	4015,566	9,68	6,75	6,92	00:00:01
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	212,8	8886,5	11343,9	12,20	44,63	9,62	00:00:02
Max	1043	69098	78453	29,85	53,48	23,29	00:00:11
Min	19	591	1089	0,57	35,93	0,36	00:00:00
St.Dev.	344,1001	21221,18	23853,42	9,68	4,59	7,76	00:00:03

Table A.22: Detailed performance measures of branch and bound algorithm for problem set J33/F1/S/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	24,6	555	1538,8	8,21	20,28	0,86	00:00:00
Max	37	3211	4263	23,05	30,53	5,10	00:00:00
Min	0	0	234	0,00	13,94	0,00	00:00:00
St.Dev.	13,54991	999,8537	1198,487	7,31	4,94	1,56	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	66,6	824,5	1633,2	8,21	19,10	2,16	00:00:00
Max	450	3281	4306	23,05	27,13	5,10	00:00:00
Min	0	0	230	0,00	13,94	0,00	00:00:00
St.Dev.	135,3836	1095,769	1239,462	7,31	3,58	2,12	00:00:00

Table A.23: Detailed performance measures of branch and bound algorithm for problem set J33/F2/S/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	42,4	5558,2	18011,5	7,88	26,53	2,65	00:00:02
Max	116	23070	77785	14,62	32,29	10,90	00:00:06
Min	0	0	418	0,00	21,07	0,00	00:00:00
St.Dev.	32,2876	8447,979	23705,03	5,00	4,32	3,71	00:00:02
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	619,2	9611,8	26063,4	7,88	27,52	5,70	00:00:02
Max	3317	34371	99743	14,62	32,11	12,71	00:00:06
Min	0	0	772	0,00	22,06	0,00	00:00:00
St.Dev.	1237,618	13445,31	32024,8	5,00	3,38	3,96	00:00:02

Table A.24: Detailed performance measures of branch and bound algorithm for problem set J33/F3/S/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	3255,4	137249,5	181842,2	8,63	31,31	4,74	00:00:24
Max	27435	850185	925521	20,09	37,63	11,61	00:02:02
Min	27	1250	3480	1,44	25,66	0,03	00:00:01
St.Dev.	8543,375	280042,2	310285,3	7,21	3,63	3,55	00:00:41
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	5863,3	230717,2	388684,3	8,63	34,78	7,01	00:00:30
Max	46022	903621	2000000	20,09	46,44	19,47	00:02:33
Min	31	775	5839	1,44	26,55	1,34	00:00:00
St.Dev.	14208,6	305799,1	625928,1	7,21	5,61	5,87	00:00:48

Table A.25: Detailed performance measures of branch and bound algorithm for problem set J33/F1/S/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	26,7	59,5	1475,7	2,70	16,60	0,28	00:00:00
Max	42	258	4898	11,07	19,44	1,61	00:00:00
Min	0	0	141	0,00	13,89	0,00	00:00:00
St.Dev.	18,66101	78,00748	1465,661	3,71	2,04	0,54	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	34,5	510,2	1548,8	2,70	16,32	0,58	00:00:00
Max	113	3424	5551	11,07	19,44	1,67	00:00:00
Min	0	0	141	0,00	13,89	0,00	00:00:00
St.Dev.	33,17378	1082,414	1659,006	3,71	1,97	0,71	00:00:00

Table A.26: Detailed performance measures of branch and bound algorithm for problem set J33/F2/S/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	593,8	7413	21108,8	3,75	25,48	1,31	00:00:02
Max	5675	68083	85828	9,74	33,45	5,50	00:00:10
Min	0	0	253	0,00	18,85	0,00	00:00:00
St.Dev.	1785,423	21390,56	30262,67	3,16	4,96	2,17	00:00:03
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	1048,7	12124,4	28748,7	3,75	26,14	2,32	00:00:02
Max	6611	88083	109369	9,74	34,06	7,00	00:00:08
Min	0	0	327	0,00	19,19	0,00	00:00:00
St.Dev.	2041,573	27169,89	37842,31	3,16	5,43	2,59	00:00:03

Table A.27: Detailed performance measures of branch and bound algorithm for problem set J33/F3/S/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	35	3285	15119,1	7,55	25,34	1,95	00:00:02
Max	41	16090	59257	16,58	30,06	4,76	00:00:07
Min	0	0	2497	0,00	19,24	0,00	00:00:00
St.Dev.	12,37381	5783,195	17450,74	5,34	3,59	1,75	00:00:02
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	1263,3	18279	37329,1	7,55	26,89	5,07	00:00:03
Max	6780	107379	209250	16,58	29,66	10,81	00:00:13
Min	0	0	4616	0,00	21,98	0,00	00:00:00
St.Dev.	2521,542	32706,06	61842,84	5,34	2,51	4,07	00:00:04

Table A.28: Detailed performance measures of branch and bound algorithm for problem set J33/F1/B/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	45,4	45,4	198,1	0,64	5,73	0,00	00:00:00
Max	187	187	469	3,00	14,07	0,00	00:00:00
Min	0	0	84	0,00	3,10	0,00	00:00:00
St.Dev.	57,66031	57,66031	110,8296	1,08	3,24	0,00	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	43	43	191,9	0,64	5,64	0,00	00:00:00
Max	163	163	445	3,00	14,07	0,00	00:00:00
Min	0	0	79	0,00	3,08	0,00	00:00:00
St.Dev.	51,25535	51,25535	105,969	1,08	3,30	0,00	00:00:00

Table A.29: Detailed performance measures of branch and bound algorithm for problem set J33/F2/B/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	934,7	38539,9	104815,3	3,24	4,49	1,88	00:01:19
Max	9090	100823	255785	6,21	6,94	5,24	00:03:17
Min	26	325	32965	0,55	3,41	0,09	00:00:24
St.Dev.	2865,482	36901,13	72840,48	2,13	1,04	1,66	00:00:54
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	1160,7	43492,4	120136,4	3,24	4,56	1,88	00:01:40
Max	11350	108006	306566	6,21	7,02	5,24	00:04:37
Min	26	325	40526	0,55	3,20	0,09	00:00:29
St.Dev.	3580,156	41456,39	84043,82	2,13	1,07	1,66	00:01:23

Table A.30: Detailed performance measures of branch and bound algorithm for problem set J33/F3/B/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	142,6	852314,1	2000000	2,97	6,56	2,22	00:29:40
Max	1072	1590441	2000000	5,57	11,13	4,01	00:33:58
Min	0	0	2000000	0,00	3,20	0,00	00:20:36
St.Dev.	327,6421	526727,9	0	1,71	2,31	1,40	00:03:44
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	244,9	860398,2	2000000	2,79	7,00	2,04	00:25:58
Max	1194	1585732	2000000	5,31	12,70	4,01	00:29:46
Min	0	0	2000000	0,00	3,35	0,00	00:19:12
St.Dev.	439,6105	659901,4	0	1,66	2,69	1,27	00:02:55

Table A.31: Detailed performance measures of branch and bound algorithm for problem set J33/F1/B/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	44,7	174,7	431,8	2,50	23,28	0,42	00:00:00
Max	249	1060	1522	9,07	48,91	2,93	00:00:01
Min	0	0	83	0,00	9,21	0,00	00:00:00
St.Dev.	72,75232	328,8259	408,1105	2,99	13,09	0,97	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	44,7	210,1	458,9	2,50	18,52	0,67	00:00:00
Max	249	1060	1522	9,07	33,75	2,93	00:00:01
Min	0	0	83	0,00	9,21	0,00	00:00:00
St.Dev.	72,75232	330,6866	428,844	2,99	8,76	1,16	00:00:00

Table A.32: Detailed performance measures of branch and bound algorithm for problem set J33/F2/B/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	45,5	28654,3	69576,2	12,77	27,73	6,75	00:00:43
Max	184	110763	200106	24,37	40,94	15,95	00:01:36
Min	29	658	3551	3,42	13,48	3,00	00:00:02
St.Dev.	48,68093	31844,95	65171,29	7,92	10,41	3,84	00:00:38
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	933,4	68163,9	104269,6	12,77	24,00	6,46	00:01:02
Max	8012	307927	331437	24,37	36,75	13,31	00:03:38
Min	29	692	3861	3,42	11,93	3,00	00:00:02
St.Dev.	2497,872	104045,1	115442,8	7,92	8,69	3,14	00:01:11

Table A.33: Detailed performance measures of branch and bound algorithm for problem set J33/F3/B/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	55,6	761209	1307546	12,17	30,03	7,67	00:18:57
Max	275	1883686	2000000	21,86	51,50	15,47	00:31:25
Min	26	57732	218936	5,82	9,44	3,72	00:02:51
St.Dev.	77,14373	651760,2	790278,8	4,77	14,86	3,60	00:12:24
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	65,8	878196	1597138	12,17	25,03	7,85	00:20:53
Max	285	1959395	2000000	21,86	51,98	21,60	00:28:00
Min	26	64062	249860	5,82	10,94	3,72	00:03:04
St.Dev.	81,92924	569318,9	640459,9	5,16	13,24	5,33	00:09:09

Table A.34: Detailed performance measures of branch and bound algorithm for problem set J33/F1/B/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	547,7	2880,5	5801,9	2,85	36,59	1,91	00:00:02
Max	1621	15491	19874	9,45	41,98	9,23	00:00:06
Min	31	31	590	0,28	25,54	0,00	00:00:00
St.Dev.	637,8361	4779,95	5940,713	2,71	4,85	2,86	00:00:02
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	587,9	2199,3	4295,3	2,85	27,99	1,65	00:00:01
Max	2033	7001	11387	9,45	35,14	7,12	00:00:03
Min	31	31	562	0,28	21,11	0,00	00:00:00
St.Dev.	790,8147	2230,348	3778,224	2,71	5,15	2,15	00:00:01

Table A.35: Detailed performance measures of branch and bound algorithm for problem set J33/F2/B/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	3500,7	27190,4	118932,2	6,77	40,32	2,95	00:00:46
Max	31928	157532	572370	23,47	52,11	9,00	00:03:02
Min	35	35	4705	0,59	20,82	0,00	00:00:02
St.Dev.	9996,695	49586,73	187409,5	7,02	9,73	3,10	00:01:06
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	8886,5	36954,9	112689,8	6,77	33,22	3,79	00:00:41
Max	47842	193963	394014	23,47	58,54	10,19	00:02:51
Min	34	185	5221	0,59	16,17	0,52	00:00:02
St.Dev.	15068,98	57357,87	147000,2	7,02	11,52	3,55	00:00:54

Table A.36: Detailed performance measures of branch and bound algorithm for problem set J33/F3/B/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	6242,9	454731,3	1398772	7,32	36,24	3,29	00:07:36
Max	30202	1871285	2000000	33,66	51,60	10,01	00:16:05
Min	31	16426	69225	0,02	25,51	0,00	00:00:27
St.Dev.	10030,43	579480,7	797360,7	10,58	9,20	3,41	00:04:55
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	113013,4	712465,5	1622653	6,56	30,83	3,03	00:07:47
Max	846538	1697874	2000000	33,66	40,28	9,64	00:17:07
Min	0	0	157051	0,00	25,21	0,00	00:00:48
St.Dev.	262256,6	582010,4	571842,9	10,51	4,90	3,41	00:05:44

Table A.37: Detailed performance measures of branch and bound algorithm for problem set J33/F1/M/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	17	43,3	190,4	7,03	14,92	0,24	00:00:00
Max	20	92	768	20,92	34,33	0,68	00:00:00
Min	0	0	67	0,00	6,49	0,00	00:00:00
St.Dev.	6,073622	32,09032	215,9306	7,06	10,08	0,31	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	22,4	155,9	232,6	7,03	12,52	0,64	00:00:00
Max	73	1079	1154	20,92	21,18	2,42	00:00:00
Min	0	0	67	0,00	6,49	0,00	00:00:00
St.Dev.	18,7806	327,0995	334,6408	7,06	6,07	0,80	00:00:00

Table A.38: Detailed performance measures of branch and bound algorithm for problem set J33/F2/M/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	20,6	1731,8	2912,1	9,13	11,92	3,70	00:00:02
Max	23	6892	9189	17,06	24,51	9,87	00:00:06
Min	18	50	507	5,22	3,42	0,21	00:00:00
St.Dev.	1,95505	2054,899	2792,318	3,68	6,87	3,05	00:00:02
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	20,5	2301,5	3852,7	9,13	10,91	3,91	00:00:02
Max	23	7058	9592	17,06	18,32	9,87	00:00:06
Min	18	50	588	5,22	3,51	0,21	00:00:00
St.Dev.	1,840894	2621,492	3669,722	3,68	5,29	3,05	00:00:02

Table A.39: Detailed performance measures of branch and bound algorithm for problem set J33/F3/M/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	3050,5	291233,3	314318	6,72	14,47	5,50	00:04:44
Max	19760	1898934	2000000	17,42	32,26	10,75	00:29:31
Min	19	5520	6343	1,44	6,13	1,30	00:00:04
St.Dev.	6077,924	606709,3	639053,4	4,70	8,10	3,11	00:09:36
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	4493,3	326846,7	460824,5	6,58	15,51	5,35	00:06:26
Max	30394	1926240	2000000	17,42	31,18	10,55	00:29:48
Min	25	10162	11120	1,44	7,48	1,30	00:00:06
St.Dev.	9338,178	617262,8	816946,1	4,70	7,57	3,14	00:11:41

Table A.40: Detailed performance measures of branch and bound algorithm for problem set J33/F1/M/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	33,8	2057	2971,5	9,52	30,12	2,08	00:00:01
Max	123	18271	19174	26,43	39,85	6,93	00:00:05
Min	0	0	43	0,00	20,05	0,00	00:00:00
St.Dev.	32,67619	5701,822	5795,214	7,29	6,89	2,29	00:00:02
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	50,4	1174	3121,8	9,52	28,36	4,53	00:00:01
Max	220	6475	19532	26,43	40,23	10,94	00:00:05
Min	0	0	43	0,00	18,29	0,00	00:00:00
St.Dev.	62,44678	2044,934	5878,307	7,29	6,85	3,90	00:00:01

Table A.41: Detailed performance measures of branch and bound algorithm for problem set J33/F2/M/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	2703,2	85916,3	137733,6	12,67	39,02	6,48	00:00:34
Max	25559	637285	850999	22,92	44,04	11,33	00:03:09
Min	25	1051	4582	5,48	32,58	4,04	00:00:01
St.Dev.	8035,326	197069,3	263692,5	6,59	4,18	2,39	00:00:58
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	289,7	58358,4	79369,3	12,67	33,79	7,35	00:00:22
Max	2071	227295	287389	22,92	38,54	17,06	00:01:29
Min	28	749	5469	5,48	28,08	1,20	00:00:03
St.Dev.	642,5571	90120,63	111340,9	6,59	3,39	4,70	00:00:29

Table A.42: Detailed performance measures of branch and bound algorithm for problem set J33/F3/M/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	62,4	17177,8	71905,9	15,48	41,03	5,19	00:00:29
Max	307	80387	221123	28,45	48,29	12,30	00:01:26
Min	0	0	8082	0,00	34,70	0,00	00:00:02
St.Dev.	89,19915	23941,72	73291,78	11,26	4,31	4,35	00:00:32
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	63,7	24594,7	182914,1	15,48	38,89	6,88	00:00:55
Max	412	68705	877031	28,45	58,29	17,63	00:04:21
Min	0	0	12320	0,00	16,91	0,00	00:00:03
St.Dev.	122,7256	26070,84	256431,4	11,26	10,44	5,82	00:01:18

Table A.43: Detailed performance measures of branch and bound algorithm for problem set J33/F1/M/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	162,4	531	2007,3	3,93	27,99	1,97	00:00:00
Max	1243	1243	4053	9,49	41,86	6,48	00:00:01
Min	34	42	422	0,33	18,36	0,00	00:00:00
St.Dev.	379,7678	427,7125	1223,596	3,44	6,40	2,52	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	303,7	809,4	1707,8	3,93	22,06	2,03	00:00:00
Max	1910	2228	3052	9,49	26,34	6,48	00:00:00
Min	33	60	425	0,33	16,94	0,00	00:00:00
St.Dev.	606,0116	832,996	926,8378	3,44	3,03	2,37	00:00:00

Table A.44: Detailed performance measures of branch and bound algorithm for problem set J33/F2/M/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	2927,8	30574,8	67503,4	4,32	32,47	1,93	00:00:11
Max	28167	294599	317668	11,86	42,27	4,57	00:00:55
Min	0	0	1093	0,00	21,69	0,00	00:00:00
St.Dev.	8869,604	92787,45	131734,6	4,30	7,20	1,63	00:00:21
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	571,2	57865,2	73479,8	4,32	27,65	1,98	00:00:10
Max	4547	308469	359365	11,86	39,75	4,81	00:00:48
Min	0	0	961	0,00	18,83	0,00	00:00:00
St.Dev.	1410,533	119769,5	143348,3	4,30	6,94	1,54	00:00:18

Table A.45: Detailed performance measures of branch and bound algorithm for problem set J33/F3/M/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	804	197898	394528,7	5,85	34,25	2,46	00:01:22
Max	7255	1039354	2000000	19,27	40,29	7,46	00:06:52
Min	0	0	8861	0,00	27,22	0,00	00:00:01
St.Dev.	2268,86	361724,3	658038,8	6,05	4,92	2,27	00:02:08
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	7931,9	155086,9	441261	5,85	31,19	4,26	00:01:06
Max	40075	692107	2000000	19,27	40,32	15,73	00:04:57
Min	0	0	31024	0,00	24,01	0,00	00:00:03
St.Dev.	13391,47	224621,3	613421,6	6,05	5,03	4,77	00:01:32

Table A.46: Detailed performance measures of branch and bound algorithm for problem set J25/F2/S/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	48,9	186,2	353,4	6,58	37,97	3,87	00:00:00
Max	214	469	1316	14,11	45,97	7,21	00:00:00
Min	0	0	45	0,00	28,80	0,00	00:00:00
St.Dev.	66,52226	166,3536	396,6524	4,29	5,57	2,78	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	46,8	265,3	534,1	6,58	36,96	4,83	00:00:00
Max	252	799	1956	14,11	41,75	11,51	00:00:00
Min	0	0	50	0,00	30,97	0,00	00:00:00
St.Dev.	73,43145	276,0862	596,9782	4,29	3,13	3,33	00:00:00

Table A.47: Detailed performance measures of branch and bound algorithm for problem set J25/F3/S/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	51,8	133,1	223,4	6,69	36,34	3,29	00:00:00
Max	246	462	523	21,37	46,85	9,61	00:00:00
Min	0	0	83	0,00	27,59	0,00	00:00:00
St.Dev.	75,67445	150,3126	141,7252	6,81	6,98	3,89	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	97,2	304,8	522,1	6,69	41,95	5,29	00:00:00
Max	451	868	1044	21,37	47,31	20,33	00:00:00
Min	0	0	201	0,00	35,23	0,00	00:00:00
St.Dev.	141,873	319,1074	349,4438	6,81	4,63	6,32	00:00:00

Table A.48: Detailed performance measures of branch and bound algorithm for problem set J25/F4/S/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	57,1	283,2	544,4	7,40	36,15	4,80	00:00:00
Max	195	1242	1957	19,50	56,65	18,11	00:00:00
Min	0	0	13	0,00	16,32	0,00	00:00:00
St.Dev.	70,32378	362,9548	615,3577	6,43	10,64	6,00	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	100,5	810,1	1943,6	7,40	45,66	4,88	00:00:00
Max	523	2923	5739	19,50	51,91	18,11	00:00:01
Min	0	0	25	0,00	29,96	0,00	00:00:00
St.Dev.	153,605	1043,353	2156,589	6,43	6,64	5,96	00:00:00

Table A.49: Detailed performance measures of branch and bound algorithm for problem set J25/F2/S/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	475	933	1929,4	9,58	25,62	4,15	00:00:00
Max	4445	5105	10177	19,32	32,25	9,60	00:00:01
Min	22	22	231	0,37	19,71	0,00	00:00:00
St.Dev.	1395,102	1730,854	3060,257	6,98	4,57	3,40	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	662,2	1501,6	2924	9,58	26,58	5,43	00:00:00
Max	6345	7275	14619	19,32	33,62	14,74	00:00:01
Min	21	33	388	0,37	18,78	0,14	00:00:00
St.Dev.	1996,752	2416,516	4335,525	6,98	4,83	4,50	00:00:00

Table A.50: Detailed performance measures of branch and bound algorithm for problem set J25/F3/S/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	41,4	1003,8	7340,1	12,39	28,84	5,95	00:00:01
Max	192	6338	26246	26,45	36,63	10,86	00:00:03
Min	21	51	194	4,42	22,21	1,00	00:00:00
St.Dev.	53,07688	1985,37	10130,36	7,95	4,54	2,90	00:00:01
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	994,5	8259,9	19635,8	12,39	33,08	8,96	00:00:01
Max	8986	37058	61771	26,45	39,46	18,15	00:00:04
Min	18	46	521	4,42	25,18	4,06	00:00:00
St.Dev.	2814,303	12562,72	24812,42	7,95	4,64	4,84	00:00:01

Table A.51: Detailed performance measures of branch and bound algorithm for problem set J25/F4/S/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	426,4	11449,1	14306,2	9,46	29,16	4,46	00:00:02
Max	3828	111289	120697	16,36	40,70	13,12	00:00:13
Min	22	77	418	1,03	18,88	0,00	00:00:00
St.Dev.	1196,17	35081,58	37431,78	5,21	6,45	4,19	00:00:04
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	1246,4	24316,3	33813,8	9,46	33,40	6,63	00:00:02
Max	10880	207698	254051	16,36	43,53	13,35	00:00:13
Min	23	103	1967	1,03	24,60	0,00	00:00:00
St.Dev.	3390,33	64569,97	77670,1	5,21	5,87	4,40	00:00:04

Table A.52: Detailed performance measures of branch and bound algorithm for problem set J25/F2/S/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	25	118,2	687,9	7,01	19,20	2,05	00:00:00
Max	42	637	2138	15,07	28,19	6,02	00:00:00
Min	0	0	138	0,00	12,27	0,00	00:00:00
St.Dev.	13,88044	189,1113	744,7249	5,53	4,95	2,07	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	37,1	242,4	972,6	7,01	20,06	4,82	00:00:00
Max	127	1162	3041	15,07	28,23	10,80	00:00:00
Min	0	0	226	0,00	13,41	0,00	00:00:00
St.Dev.	35,45404	356,0853	985,0251	5,53	4,34	4,04	00:00:00

Table A.53: Detailed performance measures of branch and bound algorithm for problem set J25/F3/S/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	25,5	513	2561	4,29	25,41	0,63	00:00:00
Max	36	3707	10185	14,41	32,58	3,49	00:00:01
Min	0	0	287	0,00	21,82	0,00	00:00:00
St.Dev.	13,59126	1133,98	3033,331	4,59	3,45	1,07	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	316,3	1684,4	4497,7	4,29	25,83	1,56	00:00:00
Max	2485	7389	17898	14,41	32,58	5,65	00:00:01
Min	0	0	635	0,00	22,11	0,00	00:00:00
St.Dev.	767,035	2198,732	5306,343	4,59	3,56	1,76	00:00:00

Table A.54: Detailed performance measures of branch and bound algorithm for problem set J25/F4/S/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	51,9	522,6	4450,3	7,77	27,61	2,25	00:00:00
Max	204	2554	19801	18,23	37,00	6,97	00:00:02
Min	24	29	415	2,11	20,92	0,00	00:00:00
St.Dev.	55,71844	811,9317	6218,471	5,46	5,28	2,72	00:00:01
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	470,9	7079	11187,9	7,77	30,65	6,92	00:00:01
Max	1298	28712	35212	18,23	40,90	17,76	00:00:02
Min	27	1311	1988	2,11	24,05	0,15	00:00:00
St.Dev.	535,9217	8745,236	11686,77	5,46	5,54	5,78	00:00:01

Table A.55: Detailed performance measures of branch and bound algorithm for problem set J25/F2/B/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	333	22777,8	36599,1	3,33	6,76	2,29	00:00:14
Max	2192	78303	123789	6,88	12,44	4,25	00:00:47
Min	21	393	8499	1,85	3,02	0,03	00:00:03
St.Dev.	667,46	31521,61	37675,13	1,43	2,87	1,24	00:00:14
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	413,7	25192,3	40500,9	3,33	6,94	2,32	00:00:16
Max	2500	87617	131431	6,88	12,70	4,25	00:00:54
Min	21	393	10554	1,85	3,07	0,03	00:00:04
St.Dev.	812,4322	34613,25	40500,88	1,43	2,91	1,23	00:00:16

Table A.56: Detailed performance measures of branch and bound algorithm for problem set J25/F3/B/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	4790,9	292714,4	774152,7	4,92	7,71	3,73	00:06:14
Max	46423	1196772	2000000	8,98	17,43	7,02	00:16:29
Min	19	35095	37951	1,02	3,43	0,47	00:00:16
St.Dev.	14629,98	376885,1	776402,5	3,02	4,17	2,40	00:06:24
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	6923,9	720457,2	1313164	4,95	8,17	3,80	00:10:32
Max	67614	3456661	4441898	8,98	18,42	7,02	00:33:25
Min	19	46809	50640	1,02	3,90	0,47	00:00:20
St.Dev.	21326,02	1079585	1555834	2,98	4,38	2,38	00:12:32

Table A.57: Detailed performance measures of branch and bound algorithm for problem set J25/F4/B/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	10445,2	1115457	1542964	5,89	9,04	5,10	00:12:27
Max	52556	1997380	2000000	13,78	14,45	10,96	00:19:58
Min	22	54900	143888	0,17	4,53	0,12	00:00:58
St.Dev.	17261,46	763357	723424,8	4,35	3,20	3,54	00:06:14
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	19114	1056799	1672633	5,31	10,31	4,52	00:09:35
Max	110193	1930619	2000000	13,78	16,70	10,96	00:13:59
Min	22	22	238100	0,17	5,46	0,00	00:01:12
St.Dev.	36053,8	795890,9	692664,2	4,56	3,35	3,77	00:05:14

Table A.58: Detailed performance measures of branch and bound algorithm for problem set J25/F2/B/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	355,5	7007,9	13592,6	10,44	23,13	6,67	00:00:05
Max	3341	21649	33513	20,36	32,90	10,77	00:00:15
Min	22	631	1091	2,92	9,49	1,60	00:00:00
St.Dev.	1048,999	6926,225	11107,9	5,89	7,44	3,17	00:00:05
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	434,6	8855,2	16059,2	10,44	21,94	6,69	00:00:06
Max	4131	17341	35780	20,36	30,93	12,68	00:00:14
Min	22	281	986	2,92	9,92	1,60	00:00:00
St.Dev.	1298,784	6440,901	11371,4	5,89	7,46	3,64	00:00:05

Table A.59: Detailed performance measures of branch and bound algorithm for problem set J25/F3/B/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	1103,6	335669,8	366484,5	12,68	23,92	7,06	00:02:46
Max	7418	1959547	2000000	26,85	46,33	15,01	00:14:40
Min	23	7839	12862	4,12	8,34	3,35	00:00:04
St.Dev.	2343,405	675362,8	705709,7	7,25	11,83	3,95	00:05:21
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	1478,7	228284,2	388665,9	12,71	25,34	7,28	00:02:41
Max	9277	1334387	1671707	26,85	61,95	15,31	00:12:24
Min	23	23436	35429	4,12	8,43	3,35	00:00:11
St.Dev.	2946,717	417976,5	666812,6	7,22	14,96	4,17	00:04:45

Table A.60: Detailed performance measures of branch and bound algorithm for problem set J25/F4/B/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	35292,1	487889,8	611967,2	10,97	34,70	7,66	00:03:51
Max	262876	1882472	2000000	21,48	49,43	20,50	00:13:56
Min	23	12914	19147	1,16	19,86	0,99	00:00:08
St.Dev.	84040,88	618988,8	765248,9	8,70	10,00	7,42	00:04:38
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	91369,5	635018,3	873768	10,49	33,00	7,54	00:04:38
Max	803291	1727738	2000000	21,48	46,25	20,50	00:12:37
Min	23	35476	43077	1,13	23,02	0,99	00:00:11
St.Dev.	252043	619720,5	845432,7	9,01	8,65	7,06	00:04:34

Table A.61: Detailed performance measures of branch and bound algorithm for problem set J25/F2/B/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	33,6	2856	31422,7	9,84	38,23	4,70	00:00:07
Max	102	8058	139921	19,10	47,82	9,25	00:00:33
Min	0	0	559	0,00	30,86	0,00	00:00:00
St.Dev.	26,08618	2871,292	43156,35	5,74	5,14	3,39	00:00:10
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	1092,9	20757,3	35120,8	9,84	31,81	7,85	00:00:07
Max	10559	101411	107326	19,10	38,54	18,98	00:00:28
Min	0	0	1119	0,00	21,95	0,00	00:00:00
St.Dev.	3326,197	31945,84	32700,57	5,74	4,87	5,18	00:00:08

Table A.62: Detailed performance measures of branch and bound algorithm for problem set J25/F3/B/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	6355,8	8702,8	55584,1	4,31	33,77	3,35	00:00:12
Max	60857	60857	302693	7,23	45,25	6,99	00:01:06
Min	0	0	622	0,00	22,75	0,00	00:00:00
St.Dev.	19156,14	18806,47	100642	3,11	7,02	2,89	00:00:22
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	13922,3	22267,6	60284,9	4,31	31,49	3,45	00:00:10
Max	130072	130072	287128	7,23	38,66	6,72	00:00:55
Min	0	0	1247	0,00	21,34	0,00	00:00:00
St.Dev.	40847,67	40033,69	99741,4	3,11	5,33	2,69	00:00:17

Table A.63: Detailed performance measures of branch and bound algorithm for problem set J25/F4/B/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	19895,9	71961,6	126725,4	3,16	39,69	2,76	00:00:36
Max	179547	529517	581740	13,40	60,95	12,93	00:03:18
Min	0	0	3974	0,00	26,83	0,00	00:00:01
St.Dev.	56245,87	164471,5	179971,3	4,57	10,33	4,42	00:01:01
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	30739,8	98110,8	182781,5	3,16	33,62	1,94	00:00:30
Max	273608	625828	681976	13,40	45,68	8,66	00:02:06
Min	0	0	4572	0,00	26,06	0,00	00:00:00
St.Dev.	85584,35	194113,1	208573	4,57	5,90	2,74	00:00:39

Table A.64: Detailed performance measures of branch and bound algorithm for problem set J25/F2/M/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	48,5	524,2	938,8	7,35	14,65	2,00	00:00:00
Max	318	1614	2475	13,44	33,62	4,88	00:00:01
Min	15	17	76	0,35	5,26	0,00	00:00:00
St.Dev.	94,85808	522,8252	721,2648	5,33	8,61	1,79	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	51,4	596,8	1078,4	7,35	16,39	2,02	00:00:00
Max	366	1819	2950	13,44	47,47	4,88	00:00:01
Min	15	17	79	0,35	5,70	0,00	00:00:00
St.Dev.	110,5473	618,1082	855,3143	5,33	12,25	1,79	00:00:00

Table A.65: Detailed performance measures of branch and bound algorithm for problem set J25/F3/M/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	2402,6	15441,5	19069,6	7,84	15,36	6,25	00:00:08
Max	19444	129163	153173	21,36	34,84	19,57	00:01:09
Min	15	377	429	2,72	3,47	0,75	00:00:00
St.Dev.	6064,31	39986,18	47211,53	6,05	10,19	5,54	00:00:21
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	3599,3	23630,7	29518,7	7,84	15,89	6,26	00:00:12
Max	29091	196496	235314	21,36	31,16	19,57	00:01:38
Min	15	631	692	2,72	4,29	0,75	00:00:00
St.Dev.	9066,354	60775,02	72449,32	6,05	9,29	5,53	00:00:30

Table A.66: Detailed performance measures of branch and bound algorithm for problem set J25/F4/M/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	420	10518	32157,9	8,60	15,21	4,20	00:00:15
Max	2895	32863	153047	20,72	32,14	8,56	00:01:27
Min	18	269	617	0,65	3,33	0,48	00:00:00
St.Dev.	886,8705	12497,33	49880,18	7,01	9,84	2,94	00:00:27
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	344,8	18954,3	83569,4	8,60	17,20	4,46	00:00:31
Max	1215	61119	403347	20,72	40,24	8,56	00:02:50
Min	18	808	2666	0,65	4,05	0,60	00:00:00
St.Dev.	497,9616	22258,15	135076,6	7,01	10,46	2,74	00:00:55

Table A.67: Detailed performance measures of branch and bound algorithm for problem set J25/F2/M/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	962,6	1251,7	3686,3	8,11	31,90	2,83	00:00:01
Max	7888	7914	17312	16,72	49,65	5,85	00:00:04
Min	19	72	220	0,29	18,12	0,17	00:00:00
St.Dev.	2454,488	2393,43	5488,85	5,94	11,80	2,16	00:00:01
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	1363,9	1892,9	4687,7	8,11	30,20	4,08	00:00:01
Max	10057	10271	21313	16,72	54,67	10,69	00:00:04
Min	21	105	325	0,29	15,71	0,17	00:00:00
St.Dev.	3105,309	3089,559	6645,484	5,94	10,48	3,31	00:00:01

Table A.68: Detailed performance measures of branch and bound algorithm for problem set J25/F3/M/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	218,5	1964,1	5078,6	11,31	37,97	6,73	00:00:01
Max	1520	9665	30995	29,58	47,55	20,26	00:00:07
Min	0	0	171	0,00	24,89	0,00	00:00:00
St.Dev.	477,8906	3051,233	9276,561	9,74	7,35	6,47	00:00:02
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	703,4	5013,8	10856,7	11,31	38,86	8,24	00:00:02
Max	3886	32052	65051	29,58	55,17	20,70	00:00:11
Min	0	0	222	0,00	25,99	0,00	00:00:00
St.Dev.	1286,495	9891,218	19428,24	9,74	10,22	7,50	00:00:03

Table A.69: Detailed performance measures of branch and bound algorithm for problem set J25/F4/M/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	350,4	11207,9	26974,6	13,25	43,83	6,93	00:00:05
Max	2472	103821	164962	36,01	48,91	22,74	00:00:28
Min	0	0	885	0,00	36,43	0,00	00:00:00
St.Dev.	783,0055	32566,6	50454,04	10,58	4,68	7,32	00:00:09
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	1934,6	38967,7	72605,7	13,25	41,14	8,20	00:00:07
Max	18244	310346	423255	36,01	48,70	20,31	00:00:30
Min	0	0	4510	0,00	29,21	0,00	00:00:01
St.Dev.	5732,393	96233,52	127393,9	10,58	5,17	6,84	00:00:09

Table A.70: Detailed performance measures of branch and bound algorithm for problem set J25/F2/M/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	135,4	680,5	2239,5	5,18	28,78	1,65	00:00:00
Max	706	2010	5493	14,81	39,89	5,23	00:00:01
Min	27	27	125	0,51	17,43	0,00	00:00:00
St.Dev.	213,2662	734,426	2146,091	4,33	9,05	1,78	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	207,8	657	2361,4	5,18	25,54	2,29	00:00:00
Max	808	1829	5605	14,81	38,79	5,23	00:00:01
Min	27	68	201	0,51	17,02	0,00	00:00:00
St.Dev.	297,9656	690,7526	2203,624	4,33	7,91	1,70	00:00:00

Table A.71: Detailed performance measures of branch and bound algorithm for problem set J25/F3/M/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	31,6	1993,3	7611,8	8,27	28,18	3,00	00:00:01
Max	72	7114	29554	17,02	36,62	13,80	00:00:05
Min	0	0	140	0,00	18,63	0,00	00:00:00
St.Dev.	17,34743	2665,674	10201,53	6,21	5,68	4,07	00:00:01
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	
Average	88,3	3429,1	11148,2	8,27	28,81	4,43	00:00:01
Max	559	12725	54651	17,02	44,64	12,70	00:00:05
Min	0	0	180	0,00	17,62	0,00	00:00:00
St.Dev.	166,6534	4842,835	16895,95	6,21	8,05	3,73	00:00:02

Table A.72: Detailed performance measures of branch and bound algorithm for problem set J25/F4/M/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	537,4	2347,5	19166,4	3,40	34,10	1,62	00:00:03
Max	3812	11511	119679	16,45	42,60	5,61	00:00:16
Min	0	0	519	0,00	22,21	0,00	00:00:00
St.Dev.	1222,32	4476,082	36296,84	5,11	6,31	2,22	00:00:05
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	376,9	6513,1	25574,9	3,40	35,16	2,57	00:00:02
Max	2687	56311	97375	16,45	39,36	11,42	00:00:06
Min	0	0	910	0,00	29,72	0,00	00:00:00
St.Dev.	845,3085	17527,77	31501,17	5,11	3,52	3,68	00:00:02

Table A.73: Detailed performance measures of branch and bound algorithm for problem set J20/F3/S/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	20,1	103,5	148,3	11,99	33,23	5,09	00:00:00
Max	49	343	383	24,91	48,41	18,16	00:00:00
Min	10	10	12	2,17	17,15	0,00	00:00:00
St.Dev.	13,58471	106,7887	112,7002	8,05	8,71	5,35	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	41,4	259,6	391,7	11,99	42,21	6,22	00:00:00
Max	140	876	1014	24,91	51,10	18,16	00:00:00
Min	12	13	34	2,17	33,91	0,00	00:00:00
St.Dev.	48,79253	319,9539	341,857	8,05	5,40	5,96	00:00:00

Table A.74: Detailed performance measures of branch and bound algorithm for problem set J20/F4/S/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	25,6	167,7	249	8,57	35,08	5,96	00:00:00
Max	67	502	722	22,32	43,51	21,98	00:00:00
Min	0	0	64	0,00	25,57	0,00	00:00:00
St.Dev.	21,13554	157,0761	212,9178	8,30	6,29	6,96	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	102,8	539	819,3	8,57	46,73	7,67	00:00:00
Max	244	1261	1968	22,32	53,20	22,15	00:00:00
Min	0	0	258	0,00	36,22	0,00	00:00:00
St.Dev.	74,62916	374,8434	552,8568	8,30	5,32	8,53	00:00:00

Table A.75: Detailed performance measures of branch and bound algorithm for problem set J20/F5/S/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	21	72,8	110,1	10,65	33,94	4,79	00:00:00
Max	72	179	272	23,52	49,52	19,23	00:00:00
Min	0	0	20	0,00	17,11	0,00	00:00:00
St.Dev.	19,44794	68,2541	83,0267	9,77	11,27	7,26	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	164,8	661,1	982,4	10,65	48,81	8,46	00:00:00
Max	860	1828	2476	23,52	57,03	19,64	00:00:00
Min	0	0	119	0,00	35,19	0,00	00:00:00
St.Dev.	257,281	582,9563	731,3712	9,77	6,95	8,35	00:00:00

Table A.76: Detailed performance measures of branch and bound algorithm for problem set J20/F3/S/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	20,2	85,3	476,1	12,82	30,24	6,14	00:00:00
Max	50	224	2352	37,81	34,89	18,34	00:00:00
Min	0	0	78	0,00	19,61	0,00	00:00:00
St.Dev.	14,65757	71,89815	702,9224	12,71	4,98	6,23	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	159,2	657,1	1413,8	12,82	33,45	8,47	00:00:00
Max	1386	4108	7900	37,81	37,11	22,32	00:00:00
Min	0	0	121	0,00	24,07	0,00	00:00:00
St.Dev.	431,4216	1293,341	2424,364	12,71	3,86	8,21	00:00:00

Table A.77: Detailed performance measures of branch and bound algorithm for problem set J20/F4/S/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	43,8	203,9	472,2	7,37	28,71	1,49	00:00:00
Max	259	713	879	16,95	36,68	3,76	00:00:00
Min	0	0	199	0,00	18,85	0,00	00:00:00
St.Dev.	76,58953	223,6279	262,3275	5,92	4,96	1,26	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	320,3	1198,3	1793,3	7,37	37,96	4,36	00:00:00
Max	1800	4144	4403	16,95	47,19	11,17	00:00:00
Min	0	0	564	0,00	26,23	0,00	00:00:00
St.Dev.	545,1347	1291,518	1259,502	5,92	6,19	4,48	00:00:00

Table A.78: Detailed performance measures of branch and bound algorithm for problem set J20/F5/S/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	59,4	814,3	2123,4	8,01	37,48	5,09	00:00:00
Max	439	4734	7156	16,39	44,75	13,67	00:00:01
Min	0	0	37	0,00	28,14	0,00	00:00:00
St.Dev.	133,5209	1437,623	2413,757	5,32	4,92	4,78	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	986,5	8593,5	14333,6	8,01	41,59	6,93	00:00:01
Max	4657	51805	57889	16,39	48,53	15,43	00:00:03
Min	0	0	498	0,00	35,85	0,00	00:00:00
St.Dev.	1500,94	15600,63	20448,4	5,32	3,99	5,36	00:00:01

Table A.79: Detailed performance measures of branch and bound algorithm for problem set J20/F3/S/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	39,6	282,8	526,7	6,02	22,29	2,04	00:00:00
Max	183	726	1646	14,62	32,61	5,05	00:00:00
Min	0	0	89	0,00	15,89	0,00	00:00:00
St.Dev.	51,28829	272,9606	470,7002	4,82	4,74	1,57	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	203,3	696,3	953,5	6,02	24,62	2,62	00:00:00
Max	1162	2159	2625	14,62	32,45	4,84	00:00:00
Min	0	0	148	0,00	20,49	0,00	00:00:00
St.Dev.	377,1799	717,9231	759,3903	4,82	3,90	1,64	00:00:00

Table A.80: Detailed performance measures of branch and bound algorithm for problem set J20/F4/S/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	96,4	434,7	1342,8	6,31	24,56	1,37	00:00:00
Max	723	2433	5885	10,30	42,44	2,74	00:00:01
Min	22	23	101	0,92	15,92	0,00	00:00:00
St.Dev.	220,3211	742,0909	1916,933	3,50	7,13	1,02	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	963,3	1787,8	3436,2	6,31	26,90	3,67	00:00:00
Max	8602	8602	14831	10,30	42,66	8,69	00:00:01
Min	21	21	240	0,92	18,31	0,00	00:00:00
St.Dev.	2692,142	3065,295	5141,242	3,50	6,97	3,14	00:00:00

Table A.81: Detailed performance measures of branch and bound algorithm for problem set J20/F5/S/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	29,4	302,7	1694,3	8,51	26,90	5,40	00:00:00
Max	85	1045	7065	17,21	35,09	14,60	00:00:01
Min	0	0	83	0,00	20,50	0,00	00:00:00
St.Dev.	21,43828	332,3693	2297,917	5,03	4,82	5,51	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	112,6	3699,7	9403,4	8,51	30,14	7,46	00:00:00
Max	689	23366	39187	17,21	37,55	15,46	00:00:02
Min	0	0	226	0,00	21,62	0,00	00:00:00
St.Dev.	209,5838	7030,105	14885,66	5,03	5,73	4,47	00:00:01

Table A.82: Detailed performance measures of branch and bound algorithm for problem set J20/F3/B/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	820	55143,8	110174,2	6,74	8,07	4,85	00:00:33
Max	3512	183780	269982	17,16	11,09	8,82	00:01:21
Min	17	8268	24681	1,59	4,10	1,35	00:00:07
St.Dev.	1317,872	50897,42	79797,14	4,54	2,02	2,09	00:00:25
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	924,3	59833,7	134219,3	6,74	8,48	4,62	00:00:40
Max	3693	216691	331448	17,16	11,77	8,82	00:01:32
Min	17	1251	30618	1,59	4,19	1,35	00:00:10
St.Dev.	1468,727	61276,8	92419,42	4,54	2,20	2,19	00:00:27

Table A.83: Detailed performance measures of branch and bound algorithm for problem set J20/F4/B/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	24102,2	314926	576076,8	6,22	10,42	4,06	00:02:35
Max	229094	1952984	2000000	16,94	15,92	8,72	00:08:28
Min	0	0	30331	0,00	6,14	0,00	00:00:10
St.Dev.	72103,43	592192	769321,4	5,31	3,54	3,03	00:03:05
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	35654,7	274523,6	682208,5	6,21	11,47	4,14	00:02:54
Max	328900	1148286	2000000	16,94	17,54	8,72	00:07:00
Min	0	0	51330	0,00	7,03	0,00	00:00:13
St.Dev.	103370	359840,7	732924,5	5,31	3,64	3,15	00:02:30

Table A.84: Detailed performance measures of branch and bound algorithm for problem set J20/F5/B/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	165551,3	640826,2	1590161	2,79	11,34	2,23	00:06:14
Max	1315165	1589900	2000000	8,78	17,68	6,23	00:10:00
Min	0	0	217734	0,00	4,53	0,00	00:01:33
St.Dev.	407949,1	618335,2	710203,1	2,65	3,57	1,87	00:02:42
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	239472,5	973131,6	1801356	2,79	12,66	2,19	00:05:12
Max	1680940	1968929	2000000	8,78	18,83	6,23	00:10:19
Min	19	0	407165	0,00	6,12	0,00	00:00:00
St.Dev.	512354,4	834328,6	505245,1	2,65	3,57	1,89	00:02:54

Table A.85: Detailed performance measures of branch and bound algorithm for problem set J20/F3/B/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	1127,2	16072,2	22861,4	8,80	25,25	5,42	00:00:06
Max	9079	108628	109842	25,27	45,48	15,59	00:00:29
Min	19	200	299	0,54	9,80	0,25	00:00:00
St.Dev.	2817,273	32981,64	34487,82	7,94	9,12	5,78	00:00:09
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	1963,4	35629,6	46510	8,80	26,49	5,93	00:00:11
Max	12278	269273	284935	25,27	37,42	20,11	00:01:11
Min	19	234	362	0,54	10,25	0,25	00:00:00
St.Dev.	3906,108	82579,85	87085,17	7,94	7,86	6,82	00:00:22

Table A.86: Detailed performance measures of branch and bound algorithm for problem set J20/F4/B/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	55913,1	159485,4	565557,9	3,58	24,86	3,30	00:02:20
Max	529264	723510	2000000	8,91	32,87	8,45	00:09:19
Min	0	0	2399	0,00	13,63	0,00	00:00:01
St.Dev.	166387,4	297123,4	827005,7	2,65	5,00	2,46	00:03:34
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	102911,8	249819,6	670070,9	3,58	29,71	3,29	00:02:12
Max	748229	1057565	2000000	8,91	48,43	8,45	00:07:12
Min	0	0	3190	0,00	13,31	0,00	00:00:01
St.Dev.	239209,8	424627,6	873686,7	2,65	10,16	2,46	00:02:56

Table A.87: Detailed performance measures of branch and bound algorithm for problem set J20/F5/B/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	8962,1	187780,5	529782,2	8,20	23,44	6,87	00:02:06
Max	87826	1042106	2000000	17,65	40,81	12,46	00:08:03
Min	0	0	2075	0,00	9,95	0,00	00:00:01
St.Dev.	27713,58	336778,4	792506,3	5,44	9,96	4,02	00:03:11
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	21733,3	267278,5	585485,8	8,44	29,16	5,80	00:01:46
Max	207511	1297108	2000000	17,65	44,39	12,46	00:06:56
Min	19	942	5000	2,23	13,62	0,31	00:00:01
St.Dev.	65312,2	395632,5	778236,7	5,11	11,83	4,28	00:02:20

Table A.88: Detailed performance measures of branch and bound algorithm for problem set J20/F3/B/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	171,5	2082,3	7642,1	6,14	34,53	3,36	00:00:01
Max	735	6832	16602	18,59	56,06	12,61	00:00:02
Min	0	0	216	0,00	15,09	0,00	00:00:00
St.Dev.	250,8888	2491,413	5228,49	6,96	11,65	4,17	00:00:01
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	607,9	3719,3	10831,9	6,14	32,96	3,71	00:00:01
Max	3860	15677	22202	18,59	46,78	12,61	00:00:03
Min	0	0	305	0,00	18,88	0,00	00:00:00
St.Dev.	1187,804	4981,379	7304,372	6,96	8,55	4,97	00:00:01

Table A.89: Detailed performance measures of branch and bound algorithm for problem set J20/F4/B/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	856,5	2465,7	9139,5	1,48	34,89	1,07	00:00:01
Max	2719	11590	20547	6,27	49,70	5,72	00:00:04
Min	0	0	1584	0,00	18,24	0,00	00:00:00
St.Dev.	1048,985	3610,901	6873,693	2,06	9,02	1,83	00:00:01
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	1337,1	4027,6	12801,4	1,48	35,46	1,22	00:00:01
Max	4118	20515	23773	6,27	41,66	5,80	00:00:04
Min	0	0	1991	0,00	30,71	0,00	00:00:00
St.Dev.	1826,118	6354,048	7181,269	2,06	3,81	1,93	00:00:01

Table A.90: Detailed performance measures of branch and bound algorithm for problem set J20/F5/B/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	1286,4	2406,9	90690	2,16	33,99	1,53	00:00:17
Max	12648	20570	792835	7,65	51,58	5,41	00:02:36
Min	0	0	383	0,00	14,07	0,00	00:00:00
St.Dev.	3992,189	6398,87	247188,8	2,67	10,46	2,08	00:00:49
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	2662,8	29393	238207,8	2,16	33,51	1,79	00:00:28
Max	17959	259092	2000000	7,65	46,40	4,78	00:04:16
Min	0	0	1689	0,00	16,24	0,00	00:00:00
St.Dev.	5886,346	81274,32	622519,2	2,67	8,23	2,08	00:01:20

Table A.91: Detailed performance measures of branch and bound algorithm for problem set J20/F3/M/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	78,5	3722,8	4879,8	11,14	17,11	5,46	00:00:01
Max	309	27534	31294	23,86	34,29	13,82	00:00:07
Min	0	0	252	0,00	8,97	0,00	00:00:00
St.Dev.	112,2599	8502,896	9434,084	8,50	8,31	4,10	00:00:02
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	102,1	4718	6248,6	11,14	20,92	6,28	00:00:01
Max	378	32918	37187	23,86	44,21	17,39	00:00:08
Min	0	0	392	0,00	9,55	0,00	00:00:00
St.Dev.	142,5408	10090,83	11076,08	8,50	10,85	5,50	00:00:02

Table A.92: Detailed performance measures of branch and bound algorithm for problem set J20/F4/M/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	516,1	3055,5	4330,8	10,04	21,00	3,92	00:00:01
Max	4075	14208	15049	18,51	40,59	11,11	00:00:02
Min	11	79	162	1,73	5,51	0,79	00:00:00
St.Dev.	1271,196	4326,998	4551,862	5,21	13,39	3,32	00:00:01
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	1018,6	5943,4	9274,2	10,04	23,93	3,93	00:00:01
Max	7856	23385	25612	18,51	41,28	11,11	00:00:04
Min	11	84	241	1,73	6,83	0,82	00:00:00
St.Dev.	2477,632	8669,441	9712,542	5,21	14,07	3,30	00:00:01

Table A.93: Detailed performance measures of branch and bound algorithm for problem set J20/F5/M/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	9906,9	24135,9	33831,9	4,76	18,74	3,37	00:00:07
Max	34333	110906	116849	10,86	44,32	10,82	00:00:24
Min	12	124	680	0,16	8,37	0,09	00:00:00
St.Dev.	12486,8	34252,35	43222,75	3,35	10,19	3,42	00:00:09
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	19804,3	54762,9	72745,6	4,76	21,99	3,54	00:00:12
Max	69733	288569	305942	10,86	36,05	10,82	00:00:47
Min	14	254	1641	0,16	10,54	0,09	00:00:00
St.Dev.	26668,93	90096,35	98123,63	3,35	7,87	3,37	00:00:16

Table A.94: Detailed performance measures of branch and bound algorithm for problem set J20/F3/M/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	554,7	1219,3	1567	8,09	34,20	3,10	00:00:00
Max	3801	8179	8956	24,89	47,31	10,36	00:00:01
Min	0	0	165	0,00	19,92	0,00	00:00:00
St.Dev.	1190,002	2474,293	2652,608	8,09	8,61	3,63	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	524,3	694,9	2126,2	8,09	34,34	4,25	00:00:00
Max	2461	2554	8986	24,89	46,03	10,93	00:00:01
Min	0	0	285	0,00	25,79	0,00	00:00:00
St.Dev.	851,4576	842,2699	2678,786	8,09	6,80	4,31	00:00:00

Table A.95: Detailed performance measures of branch and bound algorithm for problem set J20/F4/M/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	47,2	1228,6	3687,8	10,24	25,63	2,12	00:00:01
Max	222	8797	26987	30,28	45,26	9,66	00:00:06
Min	0	0	91	0,00	12,53	0,00	00:00:00
St.Dev.	70,78732	2715,315	8267,034	9,08	10,30	3,16	00:00:02
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	70,1	3506,9	9121,5	10,24	31,52	3,13	00:00:01
Max	454	18215	64531	30,28	44,85	8,88	00:00:10
Min	0	0	268	0,00	22,21	0,00	00:00:00
St.Dev.	138,1396	6024,983	19770,29	9,08	8,76	2,86	00:00:03

Table A.96: Detailed performance measures of branch and bound algorithm for problem set J20/F5/M/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	73,2	263,6	903,2	7,10	28,10	3,65	00:00:00
Max	293	807	3089	39,03	49,85	20,41	00:00:00
Min	0	0	40	0,00	11,15	0,00	00:00:00
St.Dev.	106,0113	253,3562	981,6075	11,84	13,15	6,11	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	648	1327	3602,7	7,10	40,34	2,20	00:00:00
Max	3752	5078	9950	39,03	62,86	5,84	00:00:01
Min	0	0	312	0,00	21,45	0,00	00:00:00
St.Dev.	1175,932	1647,314	3550,116	11,84	14,01	1,96	00:00:00

Table A.97: Detailed performance measures of branch and bound algorithm for problem set J20/F3/M/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	22,7	458	933	8,97	27,69	3,67	00:00:00
Max	46	2928	4984	19,55	40,30	15,99	00:00:00
Min	0	0	268	0,00	17,67	0,00	00:00:00
St.Dev.	11,06596	897,5396	1441,726	7,51	8,33	4,87	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	52,9	1075,9	1534	8,97	26,11	3,73	00:00:00
Max	187	6621	7129	19,55	33,20	15,89	00:00:00
Min	0	0	432	0,00	17,09	0,00	00:00:00
St.Dev.	53,41754	2027,778	2047,738	7,51	5,40	5,12	00:00:00

Table A.98: Detailed performance measures of branch and bound algorithm for problem set J20/F4/M/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	30,5	54,4	1259,6	5,04	30,62	0,59	00:00:00
Max	132	225	4790	13,65	45,20	1,93	00:00:01
Min	0	0	186	0,00	11,74	0,00	00:00:00
St.Dev.	37,13414	74,80077	1515,728	5,39	10,93	0,81	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	232,5	588,5	2883,1	5,04	29,61	2,33	00:00:00
Max	1485	2098	10692	13,65	39,15	11,79	00:00:01
Min	0	0	245	0,00	13,88	0,00	00:00:00
St.Dev.	453,6037	809,0125	3353,276	5,39	8,07	3,85	00:00:00

Table A.99: Detailed performance measures of branch and bound algorithm for problem set J20/F5/M/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	67,7	376,5	11143,5	6,22	34,43	3,61	00:00:01
Max	364	2104	90295	17,23	49,35	12,94	00:00:11
Min	0	0	98	0,00	21,89	0,00	00:00:00
St.Dev.	113,2922	647,9393	28113,98	6,71	9,73	4,37	00:00:04
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	6989,1	7614,2	30355,1	6,22	33,98	2,91	00:00:01
Max	68072	71852	267460	17,23	42,24	11,29	00:00:12
Min	0	0	347	0,00	25,23	0,00	00:00:00
St.Dev.	21464,96	22577,53	83588,48	6,71	6,22	3,58	00:00:04

Table A.100: Detailed performance measures of branch and bound algorithm for problem set J16/F6/S/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	11,8	85,4	107	6,34	29,70	2,25	00:00:00
Max	36	539	576	19,17	49,87	8,38	00:00:00
Min	0	0	10	0,00	15,00	0,00	00:00:00
St.Dev.	12,43472	164,1505	169,684	7,35	9,29	2,93	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	135,8	578,7	839,4	6,34	50,40	4,95	00:00:00
Max	809	1927	2283	19,17	56,01	17,27	00:00:00
Min	0	0	200	0,00	42,43	0,00	00:00:00
St.Dev.	271,3349	686,7259	727,5432	7,35	5,02	6,61	00:00:00

Table A.101: Detailed performance measures of branch and bound algorithm for problem set J16/F8/S/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	14,9	44,3	59,8	6,73	21,07	2,63	00:00:00
Max	21	230	236	16,88	35,14	14,34	00:00:00
Min	12	12	17	0,94	3,79	0,00	00:00:00
St.Dev.	3,142893	66,77832	68,84088	5,03	10,53	4,45	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	498	993,4	1845,7	6,73	51,99	3,99	00:00:00
Max	1052	2880	4964	16,88	64,78	16,51	00:00:00
Min	19	159	428	0,94	42,40	0,00	00:00:00
St.Dev.	393,4938	863,1461	1505,734	5,03	7,18	4,79	00:00:00

Table A.102: Detailed performance measures of branch and bound algorithm for problem set J16/F10/S/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	9,5	31,9	42,5	6,06	14,81	1,41	00:00:00
Max	18	143	148	18,21	23,83	6,24	00:00:00
Min	0	0	5	0,00	10,55	0,00	00:00:00
St.Dev.	8,289887	43,36781	43,01486	7,68	4,65	2,44	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	3032,7	16290,3	20327,1	6,06	60,34	5,23	00:00:01
Max	17287	65460	78400	18,21	67,90	16,99	00:00:03
Min	0	0	1342	0,00	51,85	0,00	00:00:00
St.Dev.	5361,153	24050,14	26368,53	7,68	4,58	6,56	00:00:01

Table A.103: Detailed performance measures of branch and bound algorithm for problem set J16/F6/S/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	16,7	55,5	501,1	6,02	32,65	1,26	00:00:00
Max	35	179	3866	16,20	45,35	5,12	00:00:00
Min	0	0	40	0,00	17,55	0,00	00:00:00
St.Dev.	8,641631	59,85028	1185,604	4,55	8,81	1,70	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	2175,4	2729,3	4365,7	6,02	42,67	5,26	00:00:00
Max	19335	19335	30569	16,20	53,25	14,92	00:00:01
Min	0	0	357	0,00	29,33	0,00	00:00:00
St.Dev.	6035,689	5907,946	9293,792	4,55	6,99	4,30	00:00:00

Table A.104: Detailed performance measures of branch and bound algorithm for problem set J16/F8/S/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	19,1	60	150,4	8,57	33,00	2,17	00:00:00
Max	55	177	261	21,36	48,57	8,51	00:00:00
Min	0	0	54	0,00	18,78	0,00	00:00:00
St.Dev.	13,7554	53,59104	77,10195	6,77	9,03	2,83	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	1069,8	3700,7	7012,5	8,57	49,94	7,02	00:00:00
Max	5103	15095	20454	21,36	59,20	13,33	00:00:01
Min	0	0	1835	0,00	35,73	0,00	00:00:00
St.Dev.	1554,679	5124,106	6503,508	6,77	7,87	5,05	00:00:00

Table A.105: Detailed performance measures of branch and bound algorithm for problem set J16/F10/S/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	18	116,4	174,1	9,64	30,35	3,13	00:00:00
Max	26	435	482	21,24	47,85	7,48	00:00:00
Min	14	17	24	2,80	0,00	0,00	00:00:00
St.Dev.	3,126944	143,0044	159,6966	4,90	14,03	2,42	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	940,6	5992,1	12160,3	9,64	48,65	8,02	00:00:00
Max	4386	19646	31040	21,24	59,78	16,65	00:00:01
Min	17	260	561	2,80	29,93	1,83	00:00:00
St.Dev.	1528,809	6133,96	10026,96	4,90	8,74	3,93	00:00:00

Table A.106: Detailed performance measures of branch and bound algorithm for problem set J16/F6/S/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	17,2	60,2	164,4	5,89	24,22	1,28	00:00:00
Max	31	182	357	16,32	39,09	5,62	00:00:01
Min	0	0	28	0,00	17,89	0,00	00:00:00
St.Dev.	9,874771	62,33565	100,8477	5,51	6,63	1,81	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	130,1	738,2	1090,5	5,89	29,23	4,59	00:00:00
Max	643	4028	4676	16,32	44,78	11,83	00:00:00
Min	0	0	45	0,00	17,43	0,00	00:00:00
St.Dev.	202,042	1232,15	1377,738	5,51	9,05	4,09	00:00:00

Table A.107: Detailed performance measures of branch and bound algorithm for problem set J16/F8/S/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	38,8	163	811,4	6,26	26,70	2,02	00:00:00
Max	183	714	3042	18,28	39,30	9,02	00:00:00
Min	19	19	82	0,59	4,50	0,00	00:00:00
St.Dev.	50,93962	233,9611	930,7469	4,91	10,45	2,93	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	824,1	2146,5	5835,6	6,26	37,57	4,38	00:00:00
Max	3054	6263	17878	18,28	46,76	14,32	00:00:01
Min	22	35	307	0,59	20,79	0,00	00:00:00
St.Dev.	1077,05	2079,973	6401,178	4,91	8,80	4,16	00:00:00

Table A.108: Detailed performance measures of branch and bound algorithm for problem set J16/F10/S/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	23,3	295,4	2662,8	9,01	20,59	0,93	00:00:00
Max	35	1827	21498	22,47	24,54	3,08	00:00:03
Min	18	18	41	0,16	15,19	0,00	00:00:00
St.Dev.	6,092801	552,8456	6676,908	7,31	3,26	1,01	00:00:01
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	2820	17671	23713,9	9,01	33,30	5,59	00:00:01
Max	21207	121205	143651	22,47	48,76	14,86	00:00:07
Min	20	61	94	0,16	22,98	0,00	00:00:00
St.Dev.	6611,329	39436,08	48573,68	7,31	7,71	4,85	00:00:02

Table A.109: Detailed performance measures of branch and bound algorithm for problem set J16/F6/B/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	175275,8	651459,1	1187636	1,51	12,99	1,31	00:02:57
Max	647334	1988212	2000000	3,62	16,87	3,28	00:05:23
Min	0	0	20147	0,00	8,89	0,00	00:00:05
St.Dev.	249837,8	733069,1	900888,1	1,33	3,03	1,22	00:02:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	224432,4	660644,8	1440952	1,45	16,03	1,25	00:01:57
Max	806584	1909851	2000000	3,62	19,96	2,70	00:03:07
Min	0	0	45254	0,00	11,30	0,00	00:00:07
St.Dev.	296904,4	624016,1	797568,8	1,25	2,87	1,12	00:01:00

Table A.110: Detailed performance measures of branch and bound algorithm for problem set J16/F8/B/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	90252,5	289716,3	1429939	1,11	11,98	0,96	00:04:01
Max	838537	1781665	2000000	3,24	17,27	3,17	00:08:22
Min	0	0	58937	0,00	7,36	0,00	00:00:10
St.Dev.	263605,1	598702,2	789391,6	1,46	2,94	1,26	00:02:29
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	239492,8	452084,8	1750469	0,97	19,74	0,81	00:01:55
Max	1800731	1940594	2000000	3,24	26,40	3,17	00:02:38
Min	0	0	280088	0,00	11,75	0,00	00:00:21
St.Dev.	569744,3	724659	571225	1,44	3,88	1,22	00:00:39

Table A.111: Detailed performance measures of branch and bound algorithm for problem set J16/F10/B/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	2782,3	73107,9	1508816	0,12	10,13	0,11	00:04:05
Max	21738	638239	2000000	0,78	22,57	0,77	00:08:39
Min	0	0	127805	0,00	5,13	0,00	00:00:15
St.Dev.	6929,454	200699	801198,4	0,26	4,97	0,25	00:02:32
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	103647	117744,7	2000000	0,04	18,33	0,03	00:02:08
Max	1036470	1177447	2000000	0,37	30,31	0,34	00:03:15
Min	0	0	2000000	0,00	12,64	0,00	00:01:34
St.Dev.	327760,6	372341,4	0	0,12	5,50	0,11	00:00:32

Table A.112: Detailed performance measures of branch and bound algorithm for problem set J16/F6/B/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	25602,1	111426,2	527036,3	5,51	22,58	2,75	00:01:34
Max	101303	553068	2000000	15,02	32,91	12,11	00:06:45
Min	0	0	352	0,00	11,83	0,00	00:00:00
St.Dev.	38856,32	171969,3	833100,5	5,83	6,60	3,67	00:02:37
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	39707,1	169680,8	608091,1	5,51	31,66	4,10	00:00:57
Max	154185	531601	2000000	15,02	50,99	12,35	00:03:57
Min	0	0	4019	0,00	16,46	0,00	00:00:00
St.Dev.	53765,86	189405,9	845253,1	5,83	11,00	4,56	00:01:24

Table A.113: Detailed performance measures of branch and bound algorithm for problem set J16/F8/B/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	18173	46407,4	88752,2	3,92	19,96	1,79	00:00:16
Max	106414	176720	276887	16,70	35,40	5,88	00:00:53
Min	0	0	985	0,00	7,81	0,00	00:00:00
St.Dev.	34580,26	72282,76	102690,8	5,53	8,80	2,25	00:00:19
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	99764,2	167657,4	327662,3	3,92	32,53	1,98	00:00:26
Max	537516	538732	808908	16,70	54,55	7,76	00:01:07
Min	0	0	4387	0,00	18,80	0,00	00:00:00
St.Dev.	173378,9	213100,8	299959,8	5,53	10,82	2,67	00:00:25

Table A.114: Detailed performance measures of branch and bound algorithm for problem set J16/F10/B/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	39302,3	70219,6	176912,5	0,66	17,63	0,25	00:00:34
Max	372211	460560	754473	2,73	38,76	1,69	00:02:14
Min	0	0	639	0,00	5,48	0,00	00:00:00
St.Dev.	117046,1	152110,7	274901	0,98	12,09	0,53	00:00:52
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	155017,1	360035,5	796440,7	0,66	39,01	0,38	00:00:51
Max	1259298	1937280	2000000	2,73	73,60	1,69	00:02:22
Min	0	0	8205	0,00	23,37	0,00	00:00:00
St.Dev.	392913,5	719476,2	912746,4	0,98	15,52	0,57	00:00:58

Table A.115: Detailed performance measures of branch and bound algorithm for problem set J16/F6/B/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	290	339,2	7246,4	1,06	29,84	0,56	00:00:01
Max	1649	1649	30532	9,07	40,33	5,55	00:00:05
Min	0	0	375	0,00	17,83	0,00	00:00:00
St.Dev.	516,8791	550,3185	10815,23	2,82	8,92	1,76	00:00:02
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	531,3	840,2	25272,8	1,06	37,10	0,76	00:00:01
Max	2428	2980	146941	9,07	46,15	7,42	00:00:08
Min	0	0	764	0,00	28,22	0,00	00:00:00
St.Dev.	880,5703	1145,786	45503,04	2,82	6,30	2,34	00:00:02

Table A.116: Detailed performance measures of branch and bound algorithm for problem set J16/F8/B/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	6203,9	8691,6	13100,1	0,36	26,15	0,33	00:00:02
Max	61924	86725	120288	1,75	37,48	1,67	00:00:15
Min	0	0	49	0,00	17,43	0,00	00:00:00
St.Dev.	19578,07	27418,19	37709,87	0,71	6,21	0,69	00:00:05
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	8273,2	11216,2	22671,3	0,36	35,55	0,33	00:00:01
Max	82434	111428	169207	1,75	51,28	1,67	00:00:11
Min	0	0	257	0,00	24,65	0,00	00:00:00
St.Dev.	26057,58	35211,52	52545,71	0,71	8,45	0,69	00:00:04

Table A.117: Detailed performance measures of branch and bound algorithm for problem set J16/F10/B/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	31,7	31,7	4929,3	0,73	21,84	0,00	00:00:01
Max	299	299	19958	7,14	33,36	0,00	00:00:03
Min	0	0	59	0,00	11,05	0,00	00:00:00
St.Dev.	94,08985	94,08985	7026,775	2,25	7,38	0,00	00:00:01
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	588,6	920,7	50659,7	0,73	41,08	0,62	00:00:03
Max	5275	5275	296434	7,14	57,42	6,20	00:00:17
Min	0	0	259	0,00	24,71	0,00	00:00:00
St.Dev.	1657,791	1966,649	93248,84	2,25	9,91	1,96	00:00:05

Table A.118: Detailed performance measures of branch and bound algorithm for problem set J16/F6/M/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	221,7	261,4	762,3	2,09	13,16	1,16	00:00:00
Max	1229	1324	1740	6,13	26,02	5,22	00:00:00
Min	0	0	18	0,00	6,56	0,00	00:00:00
St.Dev.	408,4472	423,5585	701,7605	2,35	6,53	1,76	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	512,4	647,3	1848,2	2,09	24,58	1,22	00:00:00
Max	2437	2437	4718	6,13	39,24	5,22	00:00:00
Min	0	0	52	0,00	6,33	0,00	00:00:00
St.Dev.	936,2998	918,9449	1596,988	2,35	11,44	1,80	00:00:00

Table A.119: Detailed performance measures of branch and bound algorithm for problem set J16/F8/M/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	819,7	1066,4	1503,8	6,63	17,92	3,36	00:00:00
Max	4756	4784	6717	18,89	30,40	17,45	00:00:02
Min	0	0	49	0,00	6,87	0,00	00:00:00
St.Dev.	1715,593	1751,49	2418,804	6,19	8,28	5,23	00:00:01
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	10500,2	11989,8	19706,5	6,63	39,86	4,87	00:00:01
Max	63251	63301	89549	18,89	55,32	18,34	00:00:07
Min	0	0	500	0,00	25,37	0,00	00:00:00
St.Dev.	22487,99	22124,98	33054,08	6,19	11,16	5,66	00:00:02

Table A.120: Detailed performance measures of branch and bound algorithm for problem set J16/F10/M/F

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	132	990,1	4611,2	5,23	10,42	1,45	00:00:01
Max	573	5884	35096	13,39	16,41	6,17	00:00:09
Min	0	0	13	0,00	1,33	0,00	00:00:00
St.Dev.	216,2493	1864,266	10833,3	5,53	4,62	1,98	00:00:03
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	2487,1	31282,7	69820,9	5,23	32,98	3,45	00:00:04
Max	17847	185178	263592	13,39	55,03	11,75	00:00:17
Min	0	0	513	0,00	12,11	0,00	00:00:00
St.Dev.	5540,811	58244,87	99936,08	5,53	12,74	3,99	00:00:06

Table A.121: Detailed performance measures of branch and bound algorithm for problem set J16/F6/M/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	63,4	124,6	413,3	8,22	25,74	1,75	00:00:00
Max	379	502	1171	22,78	40,39	11,75	00:00:00
Min	0	0	46	0,00	12,37	0,00	00:00:00
St.Dev.	119,684	161,8142	430,2403	8,56	8,71	3,59	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	261,5	655,8	2178,3	8,22	34,07	3,85	00:00:00
Max	1704	2153	6179	22,78	41,69	17,94	00:00:01
Min	0	0	375	0,00	18,42	0,00	00:00:00
St.Dev.	546,0574	808,0484	1993,114	8,56	6,83	5,95	00:00:00

Table A.122: Detailed performance measures of branch and bound algorithm for problem set J16/F8/M/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	21,7	334,1	490,6	6,49	29,06	2,80	00:00:00
Max	50	2816	3405	22,63	53,95	14,60	00:00:01
Min	0	0	40	0,00	13,09	0,00	00:00:00
St.Dev.	12,84134	873,1309	1033,596	7,52	12,06	4,34	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	596,9	1886,7	3615,5	6,49	45,27	2,73	00:00:00
Max	3064	12115	19219	22,63	64,80	12,48	00:00:01
Min	0	0	187	0,00	27,77	0,00	00:00:00
St.Dev.	1038,501	3793,598	5745,607	7,52	10,93	4,24	00:00:00

Table A.123: Detailed performance measures of branch and bound algorithm for problem set J16/F10/M/A

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	35,9	134,3	323	4,27	23,62	1,01	00:00:00
Max	186	426	1081	16,80	34,10	2,11	00:00:00
Min	0	0	21	0,00	7,04	0,00	00:00:00
St.Dev.	54,36185	147,2353	340,1621	5,63	8,51	0,86	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	2154,4	3327,5	9231,2	4,27	46,08	2,53	00:00:00
Max	20137	22264	37962	16,80	52,95	13,31	00:00:02
Min	0	0	1843	0,00	39,90	0,00	00:00:00
St.Dev.	6321,801	6890,339	11467,65	5,63	5,45	4,60	00:00:01

Table A.124: Detailed performance measures of branch and bound algorithm for problem set J16/F6/M/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	21,9	48,6	101,1	15,79	19,75	1,88	00:00:00
Max	35	186	265	39,22	33,47	11,07	00:00:01
Min	17	17	30	0,08	9,99	0,00	00:00:00
St.Dev.	5,466057	52,55938	85,58615	15,05	7,35	3,66	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	26,8	285,3	441,9	15,79	25,24	7,43	00:00:00
Max	59	1456	1703	39,22	35,39	23,58	00:00:00
Min	18	19	88	0,08	16,71	0,00	00:00:00
St.Dev.	13,7905	439,5617	475,476	15,05	6,05	8,20	00:00:00

Table A.125: Detailed performance measures of branch and bound algorithm for problem set J16/F8/M/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	77,4	223,9	799	5,64	25,20	2,77	00:00:00
Max	595	596	2615	18,53	39,32	17,41	00:00:00
Min	0	0	103	0,00	11,09	0,00	00:00:00
St.Dev.	182,3234	248,1113	743,9131	7,10	8,66	5,32	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	395,1	1283,4	3135,4	5,64	36,13	3,10	00:00:00
Max	2396	4211	8743	18,53	45,27	17,23	00:00:00
Min	0	0	370	0,00	26,40	0,00	00:00:00
St.Dev.	730,7533	1653,669	2728,022	7,10	5,31	5,32	00:00:00

Table A.126: Detailed performance measures of branch and bound algorithm for problem set J16/F10/M/L

FDLB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	35,2	44,1	182,6	3,64	17,82	1,43	00:00:00
Max	122	175	687	11,51	29,45	8,97	00:00:00
Min	0	0	34	0,00	6,64	0,00	00:00:00
St.Dev.	38,53656	51,74607	203,8759	4,20	8,27	2,80	00:00:00
FILB							
	NodeofF.S.	NodeofInc.	TotalNode	UB-Opt.	LB-Opt.	F.S.Opt.	Elap.time
Average	992,1	1152,2	3602,2	3,64	26,30	2,13	00:00:00
Max	9404	9404	19950	11,51	40,91	10,75	00:00:01
Min	0	0	167	0,00	12,77	0,00	00:00:00
St.Dev.	2956,343	2925,151	6592,124	4,20	9,63	3,33	00:00:00

APPENDIX B

NUMBER OF PROBLEMS TERMINATED AT THE NODE LIMIT

Table B 1 shows the number of the problems terminated at node limit, which is 2000000 nodes, before completion.

Table B.1: Number of problems exceed node limit in problem sets

a. job number = 50

set #	Fam.Dep.	Fam.Ind.	set #	Fam.Dep.	Fam.Ind.	set #	Fam.Dep.	Fam.Ind.
1	0	0	**7**	10	10	**13**	0	0
2	0	0	**8**	0	0	**14**	0	0
3	3	5	**9**	7	7	**15**	5	6
4	0	0	**10**	0	0	**16**	0	0
5	1	2	**11**	9	9	**17**	6	6
6	0	0	**12**	0	0	**18**	0	0

b. jobnumber = 33

set #	Fam.Dep.	Fam.Ind.	set #	Fam.Dep.	Fam.Ind.	set #	Fam.Dep.	Fam.Ind.
19	0	0	**28**	0	0	**37**	0	0
20	0	0	**29**	0	0	**38**	0	0
21	0	0	**30**	10	10	**39**	1	2
22	0	0	**31**	0	0	**40**	0	0
23	0	0	**32**	0	0	**41**	0	0
24	0	1	**33**	5	5	**42**	0	0
25	0	0	**34**	0	0	**43**	0	0
26	0	0	**35**	0	0	**44**	0	0
27	0	0	**36**	6	5	**45**	1	1

c. jobnumber = 25

set #	Fam.Dep.	Fam.Ind.	set #	Fam.Dep.	Fam.Ind.	set #	Fam.Dep.	Fam.Ind.
46	0	0	**55**	0	0	**64**	0	0
47	0	0	**56**	1	0	**65**	0	0
48	0	0	**57**	6	8	**66**	0	0
49	0	0	**58**	0	0	**67**	0	0
50	0	0	**59**	1	0	**68**	0	0
51	0	0	**60**	2	3	**69**	0	0
52	0	0	**61**	0	0	**70**	0	0
53	0	0	**62**	0	0	**71**	0	0
54	0	0	**63**	0	0	**72**	0	0

d. jobnumber = 20 (Table B1 cont.)

set #	Fam.Dep.	Fam.Ind.	set #	Fam.Dep.	Fam.Ind.	set #	Fam.Dep.	Fam.Ind.
73	0	0	**82**	0	0	**91**	0	0
74	0	0	**83**	2	2	**92**	0	0
75	0	0	**84**	7	8	**93**	0	0
75	0	0	**85**	0	0	**94**	0	0
77	0	0	**86**	2	2	**95**	0	0
78	0	0	**87**	2	2	**96**	0	0
79	0	0	**88**	0	0	**97**	0	0
80	0	0	**89**	0	0	**98**	0	0
81	0	0	**90**	0	1	**99**	0	0

e. jobnumber = 16

set #	Fam.Dep.	Fam.Ind.	set #	Fam.Dep.	Fam.Ind.	set #	Fam.Dep.	Fam.Ind.
100	0	0	**109**	5	6	**118**	0	0
101	0	0	**110**	5	8	**119**	0	0
102	0	0	**111**	7	10	**120**	0	0
103	0	0	**112**	2	2	**121**	0	0
104	0	0	**113**	0	0	**122**	0	0
105	0	0	**114**	0	3	**123**	0	0
106	0	0	**115**	0	0	**124**	0	0
107	0	0	**116**	0	0	**125**	0	0
108	0	0	**117**	0	0	**126**	0	0

APPENDIX C

PERFORMANCE OF THE UPPER BOUND PROCEDURE

Table C 1 shows the number of the problems, at which the upper bound value is equal to the optimal solution.

Table C.1: Number of problems, at which the optimal solution is equal to the upper bound value, in a set.

a. job number = 50

set #	ub=opt	set #	ub=opt	set #	ub=opt
1	1	**7**	0	**13**	0
2	0	**8**	1	**14**	1
3	0	**9**	0	**15**	0
4	0	**10**	1	**16**	1
5	0	**11**	0	**17**	0
6	0	**12**	0	**18**	0

b. job number = 33

set #	ub=opt	set #	ub=opt	set #	ub=opt
19	1	**28**	2	**37**	1
20	1	**29**	0	**38**	0
21	0	**30**	0	**39**	0
22	2	**31**	2	**40**	1
23	1	**32**	0	**41**	0
24	0	**33**	0	**42**	1
25	3	**34**	0	**43**	0
26	2	**35**	0	**44**	1
27	1	**36**	0	**45**	1

c. job number = 25

set #	ub=opt	set #	ub=opt	set #	ub=opt
46	1	**55**	0	**64**	0
47	3	**56**	0	**65**	0
48	1	**57**	0	**66**	0
49	0	**58**	0	**67**	0
50	0	**59**	0	**68**	1
51	0	**60**	0	**69**	1
52	2	**61**	1	**70**	0
53	2	**62**	1	**71**	1
54	0	**63**	2	**72**	4

d. job number = 20 (Table C1 cont.)

set #	ub=opt	set #	ub=opt	set #	ub=opt
73	0	**82**	0	**91**	1
74	1	**83**	0	**92**	0
75	1	**84**	0	**93**	0
75	2	**85**	0	**94**	1
77	1	**86**	1	**95**	1
78	1	**87**	0	**96**	2
79	1	**88**	2	**97**	1
80	0	**89**	4	**98**	2
81	1	**90**	4	**99**	2

e. job number = 16

set #	ub=opt	set #	ub=opt	set #	ub=opt
100	4	**109**	1	**118**	3
101	0	**110**	0	**119**	2
102	4	**111**	0	**120**	3
103	1	**112**	1	**121**	2
104	1	**113**	4	**122**	1
105	0	**114**	5	**123**	2
106	2	**115**	5	**124**	0
107	0	**116**	7	**125**	2
108	0	**117**	8	**126**	2

APPENDIX D

PSEUDO CODE OF UPPER BOUND HEURISTIC

```
1. Set the standard interval, which is equal to the processing time of a batch.
   (The first interval will be [$R_1$, $R_1+p$) i.e. IntervalStart = $R_1$, IntervalEnd =
   $R_1+p$; at any finish time ,t, of a batch, if there exist available job(s) waiting
   to be processed, the interval will be [t, t+p); finally at any finish time of a
   batch ,t, if there exists no available jobs and next release of any job is at t+n,
   then the interval will be [t+n, t+n+p).)

2. FOR every family f
   Determine all available jobs from family f and their respective volumes;

3. FOR every family f
   Sum all available jobs from family f, $\sum_{x=0}^{m^r(f,i)} V^f{}_{xi}$
   IF $\sum_{x=0}^{m^r(f,i)} V^f{}_{xi}$ < reactor capacity THEN
   Take all available jobs of family f into the batch, then calculate the capacity
   utilization of family f $= \frac{\sum_{x=0}^{m^r(f,i)} V^f{}_{xi}}{p}$.
   Starting from the job $m^r(f,i)$ to $m^*(f,i)$ check
   If $V^f_{xi} \leq$ remaining batch capacity and
```

$\dfrac{V^f{}_{xi} + \sum_{x=0}^{m^r(f,i)} V^f{}_{xi}}{R^f{}_{xi} + p - R^f{}_{0i}} \geq \dfrac{\sum_{x=0}^{m^r(f,i)} V^f{}_{xi}}{p}$ then take the job into the batch; decrease the remaining capacity; check the next job.

IF $\sum_{x=0}^{m^r(f,i)} V^f{}_{xi} \geq$ reactor capacity THEN

Select the jobs among family f using the bin packing algorithm, establish a batch.

4. Select the family f with f = argmax $\{V_f\}$ where V_f is the volume of the batch of family f per unit time.

   Look for the last job,x, selected from family f ,

   BatchStart = max $\{R^f_{xi}$,IntervalStart$\}$, BatchFinish = BatchStart+p

5. Calculate total weighted flow time of jobs throughout the interval;

   $$\sum_{z=1}^{f} \sum_{x=0}^{m^*(z,i)} V^z_{xi} * (R^z_{0i} + p - R^z_{xi})$$

   If jobs are finished, Stop, sum all weighted flow times calculated throughout the intervals, UB is found;

   Else Goto Step 1.

APPENDIX E

PSEUDO CODE OF FAMILY INDEPENDENT LOWER BOUND PROCEDURE

```
1.  Determine the flow time charged up to the time t;
    FOR every family f
2.  Determine the sum of available jobs, $\sum_{x=0}^{m^r(f,i)} V^f_{x1}$, from family f;
    IF there is no available jobs from family f, $\sum_{x=0}^{m^r(f,i)} V^f_{x1} = 0$, THEN
    Label the lower bound of the family f as "do not branch" (i.e. LB[j,f] = BIG),
    f= f+1, GOTO 2;
    IF there is no available jobs, $\sum_{z=1}^{f} \sum_{x=0}^{m^r(z,i)} V^z_{x1} = 0$, THEN
        IF the time of the node < Release time of the last job, i.e. for $\exists$z that
    tnode<$R^z_{m*(z,i)i*}$, THEN
        Go forward until the next arrival of a job
        ELSE
        End of the Lower bound procedure, Exit the Procedure.
3a. IF $\sum_{x=0}^{m^r(f,i)} V^f_{x1}$ > reactor capacity THEN
    Determine the interval with length p (i.e. Interval Start = node time ,
    Interval End = Interval Start + p);
```

Find the total weighted flowtime of jobs in the interval (Total weighted flowtime in the interval = $\sum_{z=1}^{f}\sum_{x=0}^{m^*(z,i)}$ V^z_{x1}* (R^z_{01} + p – R^z_{x1});

Determine the volume of jobs that will be available in the next interval (Available in the next Interval = $\sum_{z=1}^{f}\sum_{x=0}^{m*(z,i)}$ V^z_{x1} + $\sum_{z=1}^{f}\sum_{x=1}^{m^r(z,i)}$ V^z_{x2} – reactor capacity),

3b. IF $\sum_{x=0}^{m^r(f,i)}$ V^f_{x1}<= reactor capacity AND $\sum_{x=0}^{m^r(f,i)}$ V^f_{x1}>0 THEN

Determine the interval with length p (i.e. Interval Start = node time , Interval End = Interval Start + p);

Find the total weighted flowtime of jobs in the interval (Total weighted flowtime in the interval = $\sum_{z=1}^{f}\sum_{x=1}^{m^r(z,i)}$ V^z_{x1}* (R^f_{01} + p - R^f_{x1});

Determine the volume of jobs that will be available in the next interval (Available in the next Interval = $\sum_{z=1}^{f}\sum_{x=0}^{m*(z,i)}$ V^z_{x1} + $\sum_{z=1}^{f}\sum_{x=1}^{m^r(z,i)}$ V^z_{x2} – $\sum_{x=0}^{m^r(f,i)}$ V^f_{x1}),

4. Determine the interval i with length p (IF Available in the Next Interval ≥ 0 and t < Relaese of last Job, $R^z_{m*(z)i*}$, THEN

Interval Start = Previous Interval End, Interval End = Interval Start + p

IF Available in the Next Interval = 0 and t > Relaese of last Job, , $R^z_{m*(z)i*}$, THEN

Stop, LB[j,f] = Flowtime[j] + Total Flow of Jobs,);

Find the Sum of all available jobs at the Interval Start ($\sum_{z=1}^{f}\sum_{x=0}^{m^r(z,i)}$ V^z_{xi} =

$\sum_{z=1}^{f}\sum_{x=0}^{m*(z,i)}$ V^z_{xi-1} + $\sum_{z=1}^{f}\sum_{x=1}^{m^r(z,i)}$ V^z_{xi} – reactor capacity);

```
5. IF Sum of All Available Jobs, $\sum_{z=1}^{f} \sum_{x=0}^{m^r(z,i)}$ $V^z_{xi}$, =< reactor capacity THEN
    Find the remaining capacity (The Remaining Capacity = reactor capacity – $\sum_{z=1}^{f} \sum_{x=0}^{m^r(z,i)}$ $V^z_{xi}$);
   Determine the Next Job, x+1, its volume, $V^z_{x+1i}$, and its release time $R^z_{x+1i}$,
   5a. IF $V^z_{x+1i}$ > Remaining capacity AND $R^z_{x+1i}$, < Interval End THEN
       Divide the job x+1 into two portion ($V^z_{x+1i}$ – Remaining Capacity , Remaining Capacity);
       Take the "Remaining Capacity" portion into the batch (i.e. $\sum_{z=1}^{f} \sum_{x=0}^{m^r(z,i)}$ $V^z_{xi}$ + Remaining Capacity), which starts at interval start;
       Rename the "Volume of Next Job – Remaining Capacity" portion as job x+1;
       Calculate the total weighted flowtime of jobs in the Interval (Total Weighted Flowtime of Jobs in the Interval = $\sum_{z=1}^{f} \sum_{x=0}^{m^r(z,i)}$ $V^z_{xi}$ + Remaining Capacity * p + $\sum_{z=1}^{f} \sum_{x=m^r(z,i)+1}^{m^*(z,i)}$ $V^z_{xi}$ ($R^z_{0i}$ + p - $R^z_{xi}$ );
       Determine the remaining jobs that will be available in the next interval (Available in the next Interval = ($\sum_{z=1}^{f} \sum_{x=0}^{m^r(z,i)}$ $V^z_{xi+1}$ = $\sum_{z=1}^{f} \sum_{x=0}^{m^*(z,i)}$ $V^z_{xi}$ + $\sum_{z=1}^{f} \sum_{x=1}^{m^r(z,i)}$ $V^z_{xi+1}$ – reactor capacity);
   5.b IF $V^z_{x+1i}$ =< Remaining capacity AND $R^z_{x+1i}$, < Interval End THEN
       Take the job ,x+1, into the batch by changing the Release of the next job as Interval start (i.e. $R^z_{x+1i}$ = Interval Start), GOTO 5.

6. IF Sum of All Available Jobs, $\sum_{z=1}^{f} \sum_{x=0}^{m^r(z,i)}$ $V^z_{xi}$, > reactor capacity THEN
```

```
   Find the total weighted flowtime of jobs in the interval (Total weighted
   flowtime of jobs in the interval = $\sum_{z=1}^{f}\sum_{x=0}^{m*(z,i)} V^{z}_{xi} * (R^{z}_{0i} + p - R^{z}_{xi})$;
   Determine the volume of jobs that will be available in the next interval
   (Available in the next Interval = $\sum_{z=1}^{f}\sum_{x=0}^{m*(z,i)} V^{z}_{xi} + \sum_{z=1}^{f}\sum_{x=1}^{m^{r}(z,i)} V^{z}_{xi+1}$ – reactor
   capacity), GOTO 4
7. Find the Cumulative Delay of the job at node j (Cumulative Delay[j] =
   Cumulative Delay[j-1] + TNODE[j] – TNODE[j-1] );
8. FOR family number + 1 DO
   IF Cumulative Delay>p THEN
   Do not Branch (i.e. LB[j,f] = BIG for f= family number +1), Exit the lower
   bound Procedure.;
   IF Cumulative Delay<p THEN
   GOTO 6
9. Determine the interval with length less than p
   (i.e. Interval Start = node time , Interval End = Next Event time);
   Find the total flow of jobs in the interval
   (Total weighted flowtime of jobs in the interval = $\sum_{z=1}^{f}\sum_{x=0}^{m^{*}(z,i)} V^{z}_{x1} * (R^{z}_{02} -$
   $R^{z}_{01})$;
   Determine the volume of remaining jobs that will be available in the next
   interval (Available in the next Interval = $\sum_{z=1}^{f}\sum_{x=0}^{m*(z,i)} V^{z}_{x1} + \sum_{z=1}^{f}\sum_{x=1}^{m^{r}(z,i)} V^{z}_{x2}$);
10. Determine the interval i with length p (IF Available in the Next Interval $\geq 0$
   and t < Relaese of last Job, $R^{z}_{m*(z)i*}$ , THEN
   Interval Start = Previous Interval End, Interval End = Interval Start + p
   IF Available in the Next Interval = 0 and t > Relaese of last Job, , $R^{z}_{m*(z)i*}$ ,
   THEN
   Stop, LB[j,f] = Flowtime[j] + Total Flow of Jobs,);
```

Find the Sum of all available jobs at the Interval Start ($\sum_{z=1}^{f}\sum_{x=0}^{m^r(z,i)} V^z_{xi}$ = $\sum_{z=1}^{f}\sum_{x=0}^{m^*(z,i)} V^z_{xi-1} + \sum_{z=1}^{f}\sum_{x=1}^{m^r(z,i)} V^z_{xi}$ – reactor capacity);

11. IF Sum of All Available Jobs, $\sum_{z=1}^{f}\sum_{x=0}^{m^r(z,i)} V^z_{xi}$, =< reactor capacity THEN

Find the remaining capacity (The Remaining Capacity = reactor capacity – $\sum_{z=1}^{f}\sum_{x=0}^{m^r(z,i)} V^z_{xi}$);

Determine the Next Job, x+1, its volume, V^z_{x+1i}, and its release time R^z_{x+1i},

11a. IF V^z_{x+1i} > Remaining capacity AND R^z_{x+1i}, < Interval End THEN

Divide the job x+1 into two portion (V^z_{x+1i} – Remaining Capacity , Remaining Capacity);

Take the "Remaining Capacity" portion into the batch (i.e. $\sum_{z=1}^{f}\sum_{x=0}^{m^r(z,i)} V^z_{xi}$ + Remaining Capacity), which starts at interval start;

Rename the "Volume of Next Job – Remaining Capacity" portion as job x+1;

Calculate the total weighted flowtime of jobs in the Interval (Total Weighted Flowtime of Jobs in the Interval = $\sum_{z=1}^{f}\sum_{x=0}^{m^r(z,i)} V^z_{xi}$ + Remaining Capacity * p + $\sum_{z=1}^{f}\sum_{x=m^r(z,i)+1}^{m^*(z,i)} V^z_{xi} (R^z_{0i} + p - R^z_{xi})$;

Determine the remaining jobs that will be available in the next interval (Available in the next Interval = ($\sum_{z=1}^{f}\sum_{x=0}^{m^r(z,i)} V^z_{xi+1} = \sum_{z=1}^{f}\sum_{x=0}^{m^*(z,i)} V^z_{xi} + \sum_{z=1}^{f}\sum_{x=1}^{m^r(z,i)} V^z_{xi+1}$ – reactor capacity);

11b IF V^z_{x+1i} =< Remaining capacity AND R^z_{x+1i}, < Interval End THEN

Take the job ,x+1, into the batch by changing the Release of the next job as Interval start (i.e. R^{z}_{x+1i} = Interval Start), GOTO 11.

12. IF Sum of All Available Jobs, $\sum_{z=1}^{f}\sum_{x=0}^{m^{r}(z,i)}$ V^{z}_{xi}, > reactor capacity THEN

Find the total weighted flowtime of jobs in the interval (Total weighted flowtime of jobs in the interval = $\sum_{z=1}^{f}\sum_{x=0}^{m^{*}(z,i)}$ V^{z}_{xi}* (R^{z}_{0i} + p – R^{z}_{xi});

Determine the volume of jobs that will be available in the next interval (Available in the next Interval = $\sum_{z=1}^{f}\sum_{x=0}^{m^{*}(z,i)}$ V^{z}_{xi} + $\sum_{z=1}^{f}\sum_{x=1}^{m^{r}(z,i)}$ V^{z}_{xi+1} – reactor capacity), GOTO 10.

APPENDIX F

PSEUDO CODE OF FAMILY DEPENDENT LOWER BOUND PROCEDURE

```
1.) Determine the total weighted flowtime charged up to the time t;
    FOR every family f
2.) Determine the sum of available jobs, $\sum_{x=0}^{m^r(f,i)} V^f_{x1}$, for family f;
    IF there is no available jobs from family f, $\sum_{x=0}^{m^r(f,i)} V^f_{x1}= 0$, THEN
    Label the lower bound of the family f as “do not branch” (i.e. LB[j,f] = BIG),
        f= f+1, GOTO 2;
    IF there is no available jobs, $\sum_{z=1}^{f} \sum_{x=0}^{m^r(z,i)} V^z_{x1} = 0$, THEN
    IF the time of the node < Release time of the last job, i.e. for $\exists$z that
    tnode<$R^z_{m*(z)i*}$ , THEN
    Go forward until the next arrival of a job
    ELSE
    End of the Lower bound procedure, Exit the Procedure.
3a.) IF $\sum_{x=0}^{m^r(f,i)} V^f_{x1}$ > reactor capacity THEN
    Determine the interval with length p   (i.e. Interval Start = node time ,
    Interval End = Interval Start + p);
    Find the total weighted flowtime of jobs from famil f in the interval
    (Total weighted flowtime in the interval[f] = $\sum_{x=0}^{m^*(f,i)} V^f_{x1} * (R^f_{01} + p - R^f_{x1})$;
    Determine the volume of jobs, that will be available in the next interval for
    family f.
    (Available jobs in the next interval[f]= $\sum_{x=0}^{m^*(f,1)} V^f_{x1} + \sum_{x=1}^{m^r(f,2)} V^f_{x2}$ – reactor
       capacity)
3b.) IF $\sum_{x=0}^{m^r(f,i)} V^f_{x1} <=$ reactor capacity AND $\sum_{x=0}^{m^r(f,i)} V^f_{x1} > 0$ THEN
    Determine the interval with length p   (i.e. Interval Start = node time ,
    Interval End = Interval Start + p);
    Determine the total flow of jobs from family f in the interval.
    (Flow in the interval[f] = $\sum_{x=0}^{m^*(f,i)} V^f_{x1} * (R^f_{01} + p - R^f_{x1})$;
    Determine the volume of remaining jobs from family f, determine the
    volume of jobs, that will be available in the next interval for family f.
    (Available jobs in the next interval[f]= $\sum_{x=0}^{m^*(f,1)} V^f_{x1} + \sum_{x=1}^{m^r(f,2)} V^f_{x2} - \sum_{x=0}^{m^r(f,1)} V^f_{x1}$)
4.) FOR each family $g \neq f$ DO
    Determine the total weighted flowtime of jobs from family g in the interval.
    (Total weighted flowtime in the Interval[g]= $\sum_{x=0}^{m^*(g,i)} V^g_{x1} * (R^g_{01} + p - R^g_{x1})$;
    Determine the volume of jobs that will be available in the next interval for
    family g.
    (Available jobs in the next interval[g]= $\sum_{x=0}^{m^*(g,1)} V^g_{x1} + \sum_{x=1}^{m^r(g,2)} V^g_{x2}$;

5.) Calculate the total flow of jobs in the interval
    (Total weighted flowtime of all Jobs = Total weighted flowtime of all Jobs +
    Total weighted flowtime in the Interval[f])
```

6.) Determine the interval i with length p (i.e. Interval Start = Interval End , Interval End = Interval Start + p

7.) Determine the family f, with maximum volume of jobs available at interval start

7a.) IF $\sum_{x=0}^{m^r(f,i)} V^f_{xi} \geq$ reactor capacity THEN

Determine the total weighted flowtime of jobs from family f in the interval.

(Total weighted flowtime in the Interval[f]= $\sum_{x=0}^{m^*(f,i)} V^f_{xi} * (R^f_{0i} + p - R^f_{xi})$;

Determine the total volume of jobs, that will be available in the next interval for family f.

(Available jobs in the next interval[f]= $\sum_{x=0}^{m^*(f,i)} V^f_{xi} + \sum_{x=1}^{m^r(f,i+1)} V^f_{xi+1}$ – reactor capacity);

7b.) IF $\sum_{x=0}^{m^r(f,i)} V^f_{xi} <$ reactor capacity THEN

Determine the total volume of jobs from family f that are available and will be released throughout the interval [t,t+p).

Determine the total weighted flowtime of jobs from family f in the interval.

(Total weighted flowtime in the Interval[f]= $\sum_{x=0}^{m^*(f,i)} V^f_{xi} * (R^f_{0i} + p - R^f_{xi})$;

Find the effective weighted flowtime of family f in the interval.

(IF Total Volume of jobs in the Interval[f] ≤ reactor capacity THEN

Effective weighted flowtime [f] = Total weighted flowtime in the Interval[f]

ELSE

Effective weighted flowtime [f] = $\sum_{x=0}^{m(f,i)} V^f_{xi} * (R^f_{0i} + p - R^f_{xi})$+(reactor capacity - $\sum_{x=0}^{m(f,i)} V^f_{xi}) * (R^f_{0i} + p - R^f_{m(f,i)+1i}))$

Choose the family f with maximum effective weighted flowtime[f].

7bi.) IF $\sum_{x=0}^{m^*(f,i)} V^f_{xi}$ > reactor capacity THEN

Pull the jobs to the interval start until the total volume of jobs at the interval start = reactor capacity (i.e. $\sum_{x=0}^{m^r(f,i)} V^f_{xi} = \sum_{x=0}^{m(f,i)} V^f_{xi}$ + (reactor capacity - $\sum_{x=0}^{m(f,i)} V^f_{xi}$))

Determine the total weighted flowtime of jobs from family f in the interval.

(Total weighted flowtime in the Interval[f]= $\sum_{x=0}^{m^*(f,i)} V^f_{xi} * (R^f_{0i} + p - R^f_{xi})$);

Determine the total volume of jobs, that will be available in the next interval for family f.

(Available jobs in the next interval[f]= $\sum_{x=0}^{m^*(f,i)} V^f_{xi} + \sum_{x=1}^{m^r(f,i+1)} V^f_{xi+1}$ – reactor capacity);

7bii.) IF $\sum_{x=0}^{m^*(f,i)} V^f_{xi} \leq$ reactor capacity THEN

Pull the job(s) and/or job portion to the interval start until the total volume of jobs at the interval start = reactor capacity. (i.e. $\sum_{x=0}^{m^r(f,i)} V^f_{xi} = \sum_{x=0}^{m^*(f,i)} V^f_{xi}$ + min { reactor capacity - $\sum_{x=0}^{m^*(f,i)} V^f_{xi}$; $\sum_{y=i+1}^{i^*} \sum_{x=1}^{m^*(f,y)} V^f_{xy}$})

Determine the total weighted flowtime of jobs from family f in the interval.

(Total weighted flowtime in the Interval[f]= $\sum_{x=0}^{m^r(f,i)} V^f_{xi}$ * processing time)

Determine the volume of jobs, that will be available in the next interval for family f.

(Available jobs in the next interval[f]= max {0 , Sum of available jobs [f] + Jobs Released at the Interval End[f] – reactor capacity});

```
8.)     FOR every family $g \neq f$
    Determine the total weighted flowtime of jobs from family g in the interval.
    (Total Weighted flowtime in the Interval[g]= $\sum_{x=0}^{m^*(g,i)} V^g_{xi} * (R^g_{0i} + p - R^g_{xi})$);
    Determine the volume of jobs that will be available in the next interval for
    family g.
    (Available jobs in the next interval[g]= $\sum_{x=0}^{m^*(g,i)} V^g_{xi} + \sum_{x=1}^{m^r(g,i+1)} V^g_{xi}$);

9.)     Calculate the total weighted flowtime of all jobs in the interval
    (Total weighted flowtime of all Jobs = Total weighted flowtime of all Jobs +
    Total weighted flowtime in the Interval[f])
    IF all jobs are finished THEN
    Calculate LB[j,f] , f = f+1 , GOTO 2.

10.)    Find the Cumulative delay of the job at node j;
    (Cumulative Delay[j]= Cumulative Delay[j-1] + TNODE[j] – TNODE[j-1] );

11.)  FOR familynumber +1
    IF Cumulative delay + node time[j+1] - node time[j] $\geq$ processing time
        THEN
    Do not branch (i.e. LB [j,f] = BIG), exit the lower bound procedure
    IF Cumulative delay + node time[j+1] - node time[j] $<$ processing time
    THEN
    GOTO 12.

12.)  Determine the interval with length $< p$   (i.e. Interval Start = node time ,
    Interval End = Next Event time);
13.)    FOR every family f
    Determine the total weighted flowtime of jobs from family f in the interval.
```

(Total weighted flowtime in the Interval[f]= $\sum_{x=0}^{m^r(f,i)}$ V^f_{x1} * (R^f_{02} – R^f_{01}))

Determine the volume of jobs that will be available in the next interval for family f.

(Available jobs in the next interval[f]= Sum of available jobs [f]+ Jobs Released at the Interval End[f]);

14.) Calculate the total weighted flowtime of all jobs in the interval

(Total weighted flowtime of all Jobs = Total weighted flowtime of all Jobs + Total weighted flowtime in the Interval[f])

15.) Determine the interval i with length p (i.e. Interval Start = Interval End , Interval End = Interval Start + p

16.) Determine the family f, with maximum volume of jobs available at interval start

16a.) IF $\sum_{x=0}^{m^r(f,i)}$ $V^f_{xi} \geq$ reactor capacity THEN

Determine the total weighted flowtime of jobs from family f in the interval.

(Flow in the Interval[f]= $\sum_{x=0}^{m^*(f,i)}$ V^f_{xi}* (R^f_{0i} + p - R^f_{xi});

Determine the total volume of jobs, that will be available in the next interval for family f.

(Available jobs in the next interval[f]= $\sum_{x=0}^{m^*(f,i)}$ V^f_{xi} + $\sum_{x=1}^{m^r(f,i+1)}$ V^f_{xi+1} – reactor capacity);

16b.) IF $\sum_{x=0}^{m^r(f,i)}$ V^f_{xi} < reactor capacity THEN

Determine the total volume of jobs from family f that are available and will be released throughout the interval [t,t+p).

Determine the total weighted flowtime of jobs from family f in the interval.

(Total weighted flowtime in the Interval[f]= $\sum_{x=0}^{m^*(f,i)} V^f_{xi} * (R^f_{0i} + p - R^f_{xi})$;

Find the effective weighted flowtime of family f in the interval.

(IF Total Volume of jobs in the Interval[f] ≤ reactor capacity THEN

Effective weighted flowtime [f] = Total weighted flowtime in the Interval[f]

ELSE

Effective weighted flowtime [f] = $\sum_{x=0}^{m(f,i)} V^f_{xi} * (R^f_{0i} + p - R^f_{xi})$+(reactor capacity - $\sum_{x=0}^{m(f,i)} V^f_{xi}) * (R^f_{0i} + p - R^f_{m(f,i)+1i}))$

Choose the family f with maximum Effective weighted flowtime [f].

16bi.) IF $\sum_{x=0}^{m^*(f,i)} V^f_{xi}$ > reactor capacity THEN

Pull the jobs to the interval start until the total volume of jobs at the interval start = reactor capacity (i.e. $\sum_{x=0}^{m^r(f,i)} V^f_{xi} = \sum_{x=0}^{m(f,i)} V^f_{xi}$ + (reactor capacity - $\sum_{x=0}^{m(f,i)} V^f_{xi}$))

Determine the total weighted flowtime of jobs from family f in the interval.

(Flow in the Interval[f]= $\sum_{x=0}^{m^*(f,i)} V^f_{xi} * (R^f_{0i} + p - R^f_{xi})$);

Determine the total volume of jobs, that will be available in the next interval for family f.

(Available jobs in the next interval[f]= $\sum_{x=0}^{m^*(f,i)} V^f_{xi} + \sum_{x=1}^{m^r(f,i+1)} V^f_{xi+1}$ – reactor capacity);

16bii.) IF $\sum_{x=0}^{m^*(f,i)} V^f_{xi}$ ≤ reactor capacity THEN

Pull the job(s) and/or job portion to the interval start until the total volume of jobs at the interval start = reactor capacity. (i.e. $\sum_{x=0}^{m^r(f,i)} V^f_{xi} = \sum_{x=0}^{m^*(f,i)} V^f_{xi} + \min\{$ reactor capacity - $\sum_{x=0}^{m^*(f,i)} V^f_{xi}; \sum_{y=i+1}^{i^*} \sum_{x=1}^{m^*(f,y)} V^f_{xy}\}$)

Determine the total weighted flowtime of jobs from family f in the interval.

(Total weighted flowtime in the Interval[f]= $\sum_{x=0}^{m^r(f,i)} V^f_{xi}$ * processing time)

Determine the volume of jobs, that will be available in the next interval for family f.

(Available jobs in the next interval[f]= max {0 , Sum of available jobs [f] + Jobs Released at the Interval End[f] – reactor capacity});

17.) FOR every family $g \neq f$

Determine the total weighted flowtime of jobs from family g in the interval.

(Total weighted flowtime in the Interval[g]= $\sum_{x=0}^{m^*(g,i)} V^g_{xi} * (R^g_{0i} + p - R^g_{xi})$);Determine the volume of jobs that will be available in the next interval for family g.

(Available jobs in the next interval[g]= $\sum_{x=0}^{m^*(g,i)} V^g_{xi} + \sum_{x=1}^{m^r(g,i+1)} V^g_{xi}$);

18.) Calculate the total weighted flowtime of jobs in the interval

(Total weighted flowtime of all Jobs = Total weighted flowtime of all Jobs + Total weighted flowtime in the Interval[f])

IF all jobs are finished THEN

Calculate LB[j,familynumber+1],exit Lower bound procedure.

APPENDIX G

PROOF OF FAMILY INDEPENDENT LOWER BOUND PROCEDURE

Let $BS_{i,opt}$ be the total volume of jobs completed up to the beginning of interval i under optimizing procedure. And let $AS_{i,opt}$ be the total volume of jobs that are released but not processed yet and the jobs that are not released yet at the beginning of interval i under optimizing procedure.

For the optimizing procedure

$$BS_{i,opt} = \sum_{z=1}^{f}\sum_{y=1}^{i-1}\sum_{x=1}^{m^*(z,y)} V^{z}_{xy} - \sum_{z=1}^{f} V^{z}_{0i,opt}$$

and

$$AS_{i,opt} = \sum_{z=1}^{f}\sum_{y=1}^{i-1}\sum_{x=1}^{m^*(z,y)} V^{z}_{xy} + \sum_{z=1}^{f} V^{z}_{0i,opt} ;$$

Similarly for the lower bound procedure

$$AS_{i,LB} = \sum_{z=1}^{f}\sum_{y=i}^{i^*}\sum_{x=1}^{m^*} V^{z}_{xy} + \sum_{z=1}^{f} V^{z}_{0i,LB} .$$

$$AS_{i,LB} - AS_{i,opt} = \sum_{z=1}^{f} V^{z}_{0i,LB} - \sum_{z=1}^{f} V^{z}_{0i,opt} .$$

For the optimal solution

$$\sum_{z=1}^{f} V^{z}_{0i+1,opt} = \sum_{z=1}^{f} V^{z}_{0i,opt} + \sum_{z=1}^{f}\sum_{x=1}^{m*(f,i)} V^{z}_{xi} - \min \{\text{reactor capacity}, V^{f}_{0i,opt} + \sum_{x=0}^{m(f,i)} V^{f}_{xi}\} \quad (1)$$

For the lower bound procedure

$$\sum_{z=1}^{f} V^{z}_{0i+1,LB} = \sum_{z=1}^{f} V^{z}_{0i,LB} + \sum_{z=1}^{f}\sum_{x=1}^{m*(f,i)} V^{z}_{xi} - \min \{\text{reactor capacity}, \sum_{z=1}^{f} V^{z}_{0i,LB} + \sum_{z=1}^{f}\sum_{x=1}^{m*(f,i)} V^{z}_{xi} \} \quad (2)$$

Let for the interval i, $BS_{i,opt} = BS_{i,LB}$ then $\sum_{z=1}^{f} V^{z}_{0i,LB} = \sum_{z=1}^{f} V^{z}_{0i,opt}$.

Then there exist three cases

Case 1:

If $V^{f}_{0i,opt} + \sum_{x=0}^{m(f,i)} V^{f}_{xi} > C$ and $\sum_{z=1}^{f} V^{z}_{0i,LB} + \sum_{z=1}^{f}\sum_{x=1}^{m*(f,i)} V^{z}_{xi} > C$ then

(1)–(2) gives

$$\sum_{z=1}^{f} V^{z}_{0i+1,opt} - \sum_{z=1}^{f} V^{z}_{0i+1,LB} = \sum_{z=1}^{f} V^{z}_{0i,opt} + \sum_{z=1}^{f}\sum_{x=1}^{m*(f,i)} V^{z}_{xi} - \text{reactor capacity} - (\sum_{z=1}^{f} V^{z}_{0i,LB} + \sum_{z=1}^{f}\sum_{x=1}^{m*(f,i)} V^{z}_{xi} - \text{reactor capacity}) = 0.$$

If $V^{f}_{0i,opt} + \sum_{x=0}^{m(f,i)} V^{f}_{xi} < C$ and $\sum_{z=1}^{f} V^{z}_{0i,LB} + \sum_{z=1}^{f}\sum_{x=1}^{m*(f,i)} V^{z}_{xi} > C$ then

(1)–(2) gives

$$\sum_{z=1}^{f} V^{z}_{0i+1,opt} - \sum_{z=1}^{f} V^{z}_{0i+1,LB} = \sum_{z=1}^{f} V^{z}_{0i,opt} + \sum_{z=1}^{f}\sum_{x=1}^{m*(f,i)} V^{z}_{xi} - V^{f}_{0i,opt} + \sum_{x=0}^{m(f,i)} V^{f}_{xi} - (\sum_{z=1}^{f} V^{z}_{0i,LB} + \sum_{z=1}^{f}\sum_{x=1}^{m*(f,i)} V^{z}_{xi} - \text{reactor capacity}) > 0.$$

If $V^{f}_{0i,opt} + \sum_{x=0}^{m(f,i)} V^{f}_{xi} < C$ and $\sum_{z=1}^{f} V^{z}_{0i,LB} + \sum_{z=1}^{f}\sum_{x=1}^{m*(f,i)} V^{z}_{xi} < C$ then

(1)–(2) gives

$$\sum_{z=1}^{f} V^{z}_{0i+1,opt} - \sum_{z=1}^{f} V^{z}_{0i+1,LB} = \sum_{z=1}^{f} V^{z}_{0i,opt} + \sum_{z=1}^{f}\sum_{x=1}^{m^*(f,i)} V^{z}_{xi} - V^{f}_{0i,opt} + \sum_{x=0}^{m(f,i)} V^{f}_{xi} -$$

$$(\sum_{z=1}^{f} V^{z}_{0i,LB} + \sum_{z=1}^{f}\sum_{x=1}^{m^*(f,i)} V^{z}_{xi} - V^{z}_{0i,LB} + \sum_{z=1}^{f}\sum_{x=1}^{m^*(f,i)} V^{z}_{xi}) > 0$$

For all cases discussed above $\sum_{z=1}^{f} V^{z}_{0i+1,opt} \geq \sum_{z=1}^{f} V^{z}_{0i+1,LB}$. As the consequence of that we see $AS_{i+1,opt} \geq AS_{i+1,LB.}$ At the start of the interval i+1, since the total volume of jobs that should be processed by the optimizing procedure is greater than the total volume of jobs that should be processed by the consolidated family lower bound procedure, we can conclude that the total weighted flow time of the optimizing procedure will be greater than the lower bound procedure. We should note that in the comparison above we compared the cases where the processing start time is the same (i.e. $R^{f}_{m(f,i)i} = R^{f}_{0i}$). This is the most competing case of the optimizing procedure. If the processing start time is later than the interval start time (i.e. $R^{f}_{m(f,i)i} > R^{f}_{0i}$) then there will be a rightward shift in completion time. This shift charges even more additional flow time to the optimizing procedure. Therefore we will only compare the cases without shifts.

Now for the interval i let $BS_{i,opt} < BS_{i,LB}$ then $\sum_{z=1}^{f} V^{z}_{0i,LB} < \sum_{z=1}^{f} V^{z}_{0i,opt}$.

Here we can identify four different cases of available job volumes to the reactor capacity. Let the reactor capacity be C.

Case1:

if $V^{f}_{0i,opt} + \sum_{x=0}^{m(f,i)} V^{f}_{xi} \geq C$ and $\sum_{z=1}^{f} V^{z}_{0i,LB} + \sum_{z=1}^{f}\sum_{x=1}^{m^*(f,i)} V^{z}_{xi} \geq C$ then

(1) – (2) gives the following

$\sum_{z=1}^{f} V^{z}_{0i+1,opt} - \sum_{z=1}^{f} V^{z}_{0i+1,LB} = \sum_{z=1}^{f} V^{z}_{0i,opt} - \sum_{z=1}^{f} V^{z}_{0i,LB} > 0$, so the remaining total volume at the beginning of the interval i+1 is larger in optimizing procedure;

Case 2:

if $V^f_{0i,opt} + \sum_{x=0}^{m(f,i)} V^f_{xi} > R$ and $R > \sum_{z=1}^{f} V^z_{0i,LB} + \sum_{z=1}^{f}\sum_{x=1}^{m^*(f,i)} V^z_{xi}$ then

the result of (1) – (2) becomes

$\sum_{z=1}^{f} V^z_{0i+1,opt} - \sum_{z=1}^{f} V^z_{0i+1,LB} = \sum_{z=1}^{f} V^z_{0i,opt} + \sum_{z=1}^{f}\sum_{x=1}^{m^*(f,i)} V^z_{xi}$ – reactor capacity – 0 > 0, as in the first case it means the remaining total volume at the beginning of the interval i+1 is larger in optimizing procedure;

Case 3:

if $\sum_{z=1}^{f} V^z_{0i,LB} + \sum_{z=1}^{f}\sum_{x=1}^{m^*(f,i)} V^z_{xi} > R$ and $R > V^f_{0i,opt} + \sum_{x=0}^{m(f,i)} V^f_{xi}$ then

(1) – (2) is

$\sum_{z=1}^{f} V^z_{0i+1,opt} - \sum_{z=1}^{f} V^z_{0i+1,LB} = \sum_{z=1}^{f} V^z_{0i,opt} + \sum_{z=1}^{f}\sum_{x=1}^{m^*(f,i)} V^z_{xi} - V^f_{0i,opt} - \sum_{x=0}^{m(f,i)} V^f_{xi} - \sum_{z=1}^{f} V^z_{0i,LB} - \sum_{z=1}^{f}\sum_{x=1}^{m^*(f,i)} V^z_{xi}$ + reactor capacity,

$= \sum_{z=1}^{f} V^z_{0i,opt} - V^f_{0i,opt} - \sum_{x=0}^{m(f,i)} V^f_{xi} - \sum_{z=1}^{f} V^z_{0i,LB}$ + reactor capacity

$= (\sum_{z=1}^{f} V^z_{0i,opt} - \sum_{z=1}^{f} V^z_{0i,LB})$ + (reactor capacity – $(V^f_{0i,opt} + \sum_{x=0}^{m(f,i)} V^f_{xi})) > 0$, the remaining total volume at the beginning of the interval i+1 is larger in optimizing procedure;

Case 4:

if $R > \sum_{z=1}^{f} V^z_{0i,LB} + \sum_{z=1}^{f}\sum_{x=1}^{m^*(f,i)} V^z_{xi}$ and $R > V^f_{0i,opt} + \sum_{x=0}^{m(f,i)} V^f_{xi}$ then

(1) – (2) becomes

$\sum_{z=1}^{f} V^z_{0i+1,opt} - \sum_{z=1}^{f} V^z_{0i+1,LB} = \sum_{z=1}^{f} V^z_{0i,opt} + \sum_{z=1}^{f}\sum_{x=1}^{m^*(f,i)} V^z_{xi} - V^f_{0i,opt} - \sum_{x=0}^{m(f,i)} V^f_{xi} - 0 \geq 0$;

as it can be seen $\sum_{z=1}^{f} V^z_{0i+1,opt} \geq \sum_{z=1}^{f} V^z_{0i+1,LB}$ for the condition $BS_{i,opt} < BS_{i,LB}$, .

Hence $AS_{i+1,opt} \geq AS_{i+1,LB}$;

It is apparent that for all cases $AS_{i+1,opt} \geq AS_{i+1,LB}$. This means that the lower bound procedure finishes the processing on or earlier than in the optimal solution. Therefore for the lower bound procedure the weighted total flow time of the jobs are necessarily less than that of the optimal solution.

APPENDIX H

PROOF OF FAMILY DEPENDENT LOWER BOUND PROCEDURE

Suppose at interval i $BS^z_{i,opt} = BS^z_{i,LB}$ for $\forall$ z and lower bound procedure starts to process a batch from family f whereas optimal solution procedure also starts to process a batch from family f.

Then at the interval i+1, the total volume of jobs that are not processed from family f is

$$AS^f_{i+1,opt} = V^f_{0i,opt} + \sum_{y=i}^{i*} \sum_{x=1}^{m*(f,y)} V^f_{xy,opt} \text{ - } min\{reactor\ capacity\ ,\ \sum_{x=0}^{m(f,i)} V^f_{xi,opt}\} \tag{3}$$

for optimizing procedure and

$$AS^f_{i+1,LB} = V^f_{0i,LB} + \sum_{y=i}^{i*} \sum_{x=1}^{m*(f,y)} V^f_{xy,LB} \text{ - } min\{reactor\ capacity\ ,\ V^f_{0i,LB} + \sum_{y=i}^{i*} \sum_{x=1}^{m*(f,y)} V^f_{xy,LB}\} \tag{4}$$

for family dependent lower bound procedure.

In fact this part is very similar to the family independent lower bound procedure. For FILB amount of processed job in interval i is indicated by

min {reactor capacity, $\sum_{z=1}^{f} V^z_{0i,LB} + \sum_{z=1}^{f} \sum_{x=1}^{m*(f,i)} V^z_{xi}$ }

Whereas for FDLB procedure this is given by min{reactor capacity , $V^{f}_{0i,LB} + \sum_{y=i}^{i^*} \sum_{x=1}^{m^*(f,y)} V^{f}_{xy,LB}$ }.

We can identify three cases of available job volume of family f relative to reactor capacity, C.

Case 1:

If $C < \sum_{x=0}^{m(f,i)} V^{f}_{xi,opt}$ and $C < V^{f}_{0i,LB} + \sum_{y=i}^{i^*} \sum_{x=1}^{m^*(f,y)} V^{f}_{xy,LB}$ then

(3) – (4) gives

$AS^{f}_{i+1,opt}$ - $AS^{f}_{i+1,LB}$ = $V^{f}_{0i,opt} + \sum_{y=i}^{i^*} \sum_{x=1}^{m^*(f,y)} V^{f}_{xy,opt}$ *-reactor capacity –* ($V^{f}_{0i,LB}$ + $\sum_{y=i}^{i^*} \sum_{x=1}^{m^*(f,y)} V^{f}_{xy,LB}$ *- reactor capacity) = 0.*

We conclude that for Case 1 the lower bound procedure and optimizing procedure performs the same under even the best condition for the optimal solution.

Case 2:

If $\sum_{x=0}^{m(f,i)} V^{f}_{xi,opt} < C$ and $C < V^{f}_{0i,LB} + \sum_{y=i}^{i^*} \sum_{x=1}^{m^*(f,y)} V^{f}_{xy,LB}$ then

(3) – (4) gives

$AS^{f}_{i+1,opt}$ - $AS^{f}_{i+1,LB}$ = $V^{f}_{0i,opt} + \sum_{y=i}^{i^*} \sum_{x=1}^{m^*(f,y)} V^{f}_{xy,opt} - \sum_{x=0}^{m(f,i)} V^{f}_{xi,opt}$ – ($V^{f}_{0i,LB}$ + $\sum_{y=i}^{i^*} \sum_{x=1}^{m^*(f,y)} V^{f}_{xy,LB}$ *- reactor capacity)*

= *(reactor capacity -* $\sum_{x=0}^{m(f,i)} V^{f}_{xi,opt}$ *) > 0*, so lower bound finishes more jobs than the optimal procedure on time or earlier than the optimal procedure.

Case 3:

If $\sum_{x=0}^{m(f,i)} V^{f}_{xi,opt} < C$ and $V^{f}_{0i,LB} + \sum_{y=i}^{i^*} \sum_{x=1}^{m^*(f,y)} V^{f}_{xy,LB} < C$ then

(3) – (4) gives

$$AS^{f}_{i+1,opt} - AS^{f}_{i+1,LB} = V^{f}_{0i,opt} + \sum_{y=i}^{i^*} \sum_{x=1}^{m^*(f,y)} V^{f}_{xy,opt} - \sum_{x=0}^{m(f,i)} V^{f}_{xi,opt} - (V^{f}_{0i,LB} + \sum_{y=i}^{i^*} \sum_{x=1}^{m^*(f,y)} V^{f}_{xy,LB} - V^{f}_{0i,LB} + \sum_{y=i}^{i^*} \sum_{x=1}^{m^*(f,y)} V^{f}_{xy,LB})$$

$$= V^{f}_{0i,opt} + \sum_{y=i}^{i^*} \sum_{x=1}^{m^*(f,y)} V^{f}_{xy,opt} - \sum_{x=0}^{m(f,i)} V^{f}_{xi,opt} - 0 > 0.$$

For all three cases we conclude that the lower bound procedure completes atleast same amount of jobs on time or earlier than the optimal procedure. Therefore we can conclude that at the start of the interval i we can establish the optimal sequence of batches without any extra unit volumes of jobs or flow times.

In second part we cannot use the remaining volumes for comparison because the optimizing procedure and lower bound procedure may select different families for processing. In that case we cannot divide the problem into intervals as easy as the problems in which the optimal procedure and lower bound procedure. In this part we will compare the case of $BS^{z}_{i,opt} = BS^{z}_{i,LB}$ for $\forall z$ when optimizing procedure starts to process a batch of family f and lower bound procedure starts to process a batch of family g .

There exist numerous cases that can be compared; here we will compare the case $\sum_{x=0}^{m^*(f,i)} V^{f}_{xi} > \sum_{x=0}^{m^*(g,i)} V^{g}_{xi}$ and $\sum_{x=0}^{m^*(g,i)} V^{g}_{xi} = \sum_{y=i}^{i^*} \sum_{x=0}^{m^*(g,i)} V^{g}_{xy}$,which is the most critical case.

This is critical since in all other cases lower bound procedure completes at least equal volume of jobs that the optimizing procedure does. Therefore it has dominance in terms of total weighted flow time in the interval and total volume of completed jobs in the interval. But for this case the dominance in terms of total volume of completed jobs in the interval is violated by the optimizing procedure. Therefore this is the most advantageous case of optimizing procedure.

Case 1:

Let $\sum_{x=0}^{m(f,i)} V^{f}_{xi}$ = reactor capacity and $\sum_{x=0}^{m^*(f,i)} V^{f}_{xi}$ = (n * reactor capacity) in interval i and let in interval i+1 exist a single release of job with volume V^{f}_{1i+1} with release time R^f_{1i+1}. For the lower bound procedure the total weighted flow time can be written as

LB:

$$\sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * p + \sum_{x=0}^{m^*(f,i)} V^f{}_{xi} * (R^f{}_{0i} + p - R^f{}_{xi}) +$$

$$\sum_{x=0}^{m^*(f,i)} V^f{}_{xi} * p + V^f{}_{1i+1} * (R^f{}_{0i} + 2p - R^g{}_{1i+1}) +$$

$$(\sum_{x=0}^{m^*(f,i)} V^f{}_{xi} \text{ - reactor capacity}) * p + V^f{}_{1i+1} * (R^f{}_{0i} + 3p - (R^f{}_{0i} + 2p)) +$$

.

.

$$(\sum_{x=0}^{m^*(f,i)} V^f{}_{xi} -(n-1)* \text{ reactor capacity})*p + V^f{}_{1i+1} * (R^f{}_{0i} + (n+1)*p - (R^f{}_{0i} + n*p))$$

$+$

$$V^f{}_{1i+1} * (R^f{}_{0i} + (n+1)*p - (R^f{}_{0i} + n*p))$$

For the optimal solution procedure the total weighted flow time is given below

OPT:

$$\sum_{x=0}^{m(g,i)} V^g{}_{xi} * (R^f{}_{m(f,i)i} - R^f{}_{0i}) + \sum_{x=0}^{m^*(f,i)} V^f{}_{xi} * (R^f{}_{m(f,i)i} - R^f{}_{0i}) +$$

$$\sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi} *(R^f{}_{m(f,i)i} + p - R^f{}_{xi}) + \sum_{x=0}^{m(f,i)} V^f{}_{xi} * p + \sum_{x=0}^{m(g,i)} V^g{}_{xi} * (R^f{}_{m(f,i)i} + p - R^f{}_{m(f,i)i})$$

$$+ \sum_{x=m(g,i)+1}^{m^*(g,i)} V^g{}_{xi} * (R^f{}_{m(f,i)i} + p - R^f{}_{xi}) +$$

$$\sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi} * p + \sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * (R^f{}_{m(f,i)i} + 2p - (R^f{}_{m(f,i)i} + p)) + V^f{}_{1i+1} * (R^f{}_{m(f,i)i} +$$

$$2*p - R^f{}_{1i+1})+$$

$$(\sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi} \text{ - reactor capacity})*p + \sum_{x=0}^{m^*(g,i)} V^g{}_{xi} *(R^f{}_{m(f,i)i} + 3*p - (R^f{}_{m(f,i)i} + 2*p)) +$$

$$V^f{}_{1i+1}*(R^f{}_{m(f,i)i} + 3*p - (R^f{}_{m(f,i)i} + 2*p)) +$$

$(\sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi}$ *- 2*reactor capacity) * p +* $\sum_{x=0}^{m^*(g,i)} V^g{}_{xi}$ $* (R^f{}_{m(f,i)I} + 4*p - (R^f{}_{m(f,i)I} + 3*p)) + V^f{}_{1i+1}*(R^f{}_{m(f,i)I} + 4*p - (R^f{}_{m(f,i)I} + 3*p)) +$

.

.

$(\sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi}$ *-(n-2)*reactor capacity)* p+* $\sum_{x=0}^{m^*(g,i)} V^g{}_{xi}$ $*(R^f{}_{m(f,i)i}+n*p-(R^f{}_{m(f,i)i}+(n-1)*p)) + V^f{}_{1i+1}*(R^f{}_{m(f,i)i} + 2p - (R^f{}_{m(f,i)i} + p)) +$

$\sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * p + V^f{}_{1i+1} * p +$

$min \{ \sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * p , V^f{}_{1i+1} * p\};$

let $\sum_{x=0}^{m^*(g,i)} V^g{}_{xi} < V^f{}_{1i+1}$ and $R^f{}_{m(f,i)i} + p > R^f{}_{1i+1}$. After the elimination of the terms with equal flow times the problem can be reduced to the following form

LB:

$\sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * p + \sum_{x=0}^{m^*(f,i)} V^f{}_{xi} * (R^f{}_{0i} + p - R^f{}_{xi}) +$

$\sum_{x=0}^{m^*(f,i)} V^f{}_{xi} * p + V^f{}_{1i+1} * (R^f{}_{0i} + 2p - R^g{}_{1i+1}) +$

$V^f{}_{1i+1} * (R^f{}_{0i} + (n+1)*p - (R^f{}_{0i} + n*p))$

OPT:

$\sum_{x=0}^{m(g,i)} V^g{}_{xi} * (R^f{}_{m(f,i)i} - R^f{}_{0i}) + \sum_{x=0}^{m(f,i)} V^f{}_{xi} * (R^f{}_{m(f,i)i} - R^f{}_{0i}) +$

$\sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi} *(R^f{}_{m(f,i)i} + p - R^f{}_{xi}) + \sum_{x=0}^{m^*(f,i)} V^f{}_{xi} * p + \sum_{x=0}^{m(g,i)} V^g{}_{xi} * (R^f{}_{m(f,i)i} + p - R^f{}_{m(f,i)i})$

$+ \sum_{x=m(g,i)+1}^{m^*(g,i)} V^g{}_{xi} * (R^f{}_{m(f,i)i} + p - R^f{}_{xi}) +$

$V^f{}_{1i+1} * (R^f{}_{m(f,i)i} + 2*p - R^f{}_{1i+1}) + n * \sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * (R^f{}_{m(f,i)i} + 2p - (R^f{}_{m(f,i)i} + p)) +$

$\sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * p,$

Note that $\sum_{x=0}^{m^*(f,i)} V^f{}_{xi}$ * (R^f_{0i} + p – R^f_{xi}) = $\sum_{x=0}^{m(f,i)} V^f{}_{xi}$ * (R^f_{0i} + p – R^f_{xi}) + $\sum_{x=m(f)+1}^{m^*(f,i)} V^f{}_{xi}$ * (R^f_{0i} + p – R^f_{xi}) and $\sum_{x=0}^{m^*(f,i)} V^f{}_{xi}$ * p = $\sum_{x=0}^{m(f,i)} V^f{}_{xi}$ * p + $\sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi}$ * p and V^f_{1i+1} * (R^f_{0i} + (n+1)*p – (R^f_{0i} + n*p)) = V^f_{1i+1} * (R^f_{0i} + 2*p – ($R^f_{m(f,i)i}$ + p)) + V^f_{1i+1} * ($R^f_{m(f,i)i}$ + p – (R^f_{0i} +p))

By using these information we can re-write the lower bound and optimal solution procedures as

LB:

$\sum_{x=0}^{m(f,i)} V^f{}_{xi}$ * (R^f_{0i} + p – R^f_{xi}) + $\sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi}$ * (R^f_{0i} + p – R^f_{xi}) + $\sum_{x=0}^{m(f,i)} V^f{}_{xi}$ * p + $\sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi}$ * p + V^f_{1i+1} * (R^f_{0i} + 2*p – ($R^f_{m(f,i)i}$ + p)) + V^f_{1i+1} * ($R^f_{m(f,i)i}$ + p – R^f_{1i+1}) + V^f_{1i+1} * p;

OPT:

$\sum_{x=0}^{m(g,i)} V^g{}_{xi}$ * ($R^f_{m(f,i)i}$ – R^f_{xi}) + $\sum_{x=0}^{m(f,i)} V^f{}_{xi}$ * ($R^f_{m(f,i)i}$ – R^f_{xi}) + $\sum_{x=0}^{m(f,i)} V^f{}_{xi}$ * ($R^f_{m(f,i)i}$ + p – (R^f_{0i} + p)) + $\sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi}$ * (R^f_{0i} + p – R^f_{xi}) + $\sum_{x=0}^{m(f,i)} V^f{}_{xi}$ * p + $\sum_{x=0}^{m(g,i)} V^g{}_{xi}$ * p + $\sum_{x=m(g,i)+1}^{m^*(g,i)} V^g{}_{xi}$ * ($R^f_{m(f,i)i}$ + p – (R^f_{0i} + p)) + $\sum_{x=m(g,i)+1}^{m^*(g,i)} V^g{}_{xi}$ * (R^f_{0i} + p – R^f_{xi}) +n * $\sum_{x=0}^{m(g,i)} V^g{}_{xi}$ * p + V^f_{1i+1} * ($R^f_{m(f,i)i}$ + p–R^f_{1i+1}) + V^f_{1i+1} * p;

after this elimination step :

LB:

$\sum_{x=0}^{m^*(f,i)} V^f{}_{xi}$ * (R^f_{0i} + p – $R^f_{m(f,i)i}$) + V^f_{1i+1} * (R^f_{0i} + p – $R^f_{m(f,i)i}$)

OPT:

$$n * \sum_{x=0}^{m(g,i)} V^g{}_{xi} * p + \sum_{x=0}^{m(g,i)} V^g{}_{xi} * (R^f{}_{m(f,i)i} + p - R^f{}_{xi}) + \sum_{x=m(g,i)+1}^{m^*(g,i)} V^g{}_{xi} * (R^f{}_{m(f,i)i} + p - R^f{}_{xi})$$

or

$$n * \sum_{x=0}^{m(g,i)} V^g{}_{xi} * p + \sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * (R^f{}_{m(f,i)i} - R^f{}_{0i}) + \sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * (R^f{}_{0i} + p - R^f{}_{xi})$$

by the definition of the lower bound procedure (from step 4) we know that

$$\sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * (R^f{}_{0i} + p - R^f{}_{xi}) > \sum_{x=0}^{m(f,i)} V^f{}_{xi} * (R^f{}_{0i} + p - R^f{}_{xi})$$

then

$$\sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * (R^f{}_{0i} + p - R^f{}_{xi}) > \sum_{x=0}^{m(f,i)} V^f{}_{xi} * (R^f{}_{0i} + p - R^f{}_{m(f,i)i})$$

Since $\sum_{x=0}^{m(f,i)} V^f{}_{xi}$ = reactor capacity $\geq V^f{}_{1i+1}$

$$\sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * (R^f{}_{0i} + p - R^f{}_{xi}) > V^f{}_{1i+1} * (R^f{}_{0i} + p - R^f{}_{m(f,i)i});$$

$$\sum_{x=0}^{m(g,i)} V^g{}_{xi} * p > \sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * (R^f{}_{0i} + p - R^f{}_{xi}) > \sum_{x=0}^{m(f,i)} V^f{}_{xi} * (R^f{}_{0i} + p - R^f{}_{m(f,i)i}),$$

$\sum_{x=0}^{m^*(f,i)} V^f{}_{xi}$ = n * reactor capacity , $\sum_{x=0}^{m(f,i)} V^f{}_{xi}$ = reactor capacity then

$\sum_{x=0}^{m(f,i)} V^f{}_{xi} \; (R^f{}_{0i} + p - R^f{}_{xi}) > \sum_{x=0}^{m(f,i)} V^f{}_{xi} * (R^f{}_{0i} + p - R^f{}_{m(f,i)i})$ consequently

$n * \sum_{x=0}^{m(g,i)} V^g{}_{xi} * p > \sum_{x=0}^{m^*(f,i)} V^f{}_{xi} * (R^f{}_{0i} + p - R^f{}_{m(f,i)i})$; flowtime charged by the optimal solution procedure is greater than that of lower bound procedure charged.

Case2:

Let $\sum_{x=0}^{m(f,i)} V^f{}_{xi}$ = reactor capacity and $\sum_{x=0}^{m^*(f,i)} V^f{}_{xi}$ < 2* reactor capacity in interval i and let in interval i+1 be a single release with volume $V^f{}_{1i+1}$ with release time $R^f{}_{1i+1}$

LB:

$$\sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * p + \sum_{x=0}^{m^*(f,i)} V^f{}_{xi} * (R^f{}_{0i} + p - R^f{}_{xi}) +$$

$$\sum_{x=0}^{m^*(f,i)} V^f{}_{xi} * p + V^f{}_{1i+1} * (R^f{}_{0i} + 2p - R^f{}_{1i+1}) +$$

$$(\sum_{x=0}^{m^*(f,i)} V^f{}_{xi} \text{ - reactor capacity}) * p + V^f{}_{1i+1} * (R^f{}_{0i} + 3p - (R^f{}_{0i} + 2p)) +$$

$$(\sum_{x=0}^{m^*(f,i)} V^f{}_{xi} + V^f{}_{1i+1} - 2 * \text{ reactor capacity}) * (R^f{}_{0i} + 4p - (R^f{}_{0i} + 3p))$$

OPT:

$$\sum_{x=0}^{m(g,i)} V^g{}_{xi} * (R^f{}_{m(f,i)i} - R^f{}_{xi}) + \sum_{x=0}^{m(f,i)} V^f{}_{xi} * (R^f{}_{m(f,i)i} - R^f{}_{xi}) +$$

$$\sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi} *(R^f{}_{m(f,i)i} + p - R^f{}_{xi}) + \sum_{x=0}^{m(f,i)} V^f{}_{xi} * p + \sum_{x=0}^{m(g,i)} V^g{}_{xi} * (R^f{}_{m(f,i)i} + p - R^f{}_{m(f,i)i})$$

$$+ \sum_{x=m(g,i)+1}^{m^*(g,i)} V^g{}_{xi} * (R^f{}_{m(f,i)i} + p - R^f{}_{xi}) +$$

$$(\sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi} + V^f{}_{1i+1})* p + \sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * (R^f{}_{m(f,i)i} + 2p - (R^f{}_{m(f,i)i} + p)) +$$

$$(\sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi} + V^f{}_{1i+1} - \text{reactor capacity})* p + \sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * p +$$

$$\min \{(\sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi} + V^f{}_{1i+1} - \text{reactor capacity})* p \,;\, \sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * p \}$$

Define intervals with length $R^f{}_{m(f,i)i} - R^f{}_{0i} = a$ and $R^f{}_{0i} + p - R^f{}_{m(f,i)i} = b$, then we can write $(R^f{}_{m(f,i)i} - R^f{}_{xi})$ as a^- if $R^f{}_{0i} < R^f{}_{xi} < R^f{}_{m(f,i)i}$. Similarly we can write $(R^f{}_{0i} + p - R^f{}_{xi})$ as b^- if $R^f{}_{0i} + p > R^f{}_{xi} > R^f{}_{m(f,i)i}$.

After canceling the terms with equal flow times we can write the remaining terms as follows:

LB:

$$\sum_{x=0}^{m(f,i)} V^f{}_{xi} * b + \sum_{x=0}^{m(f,i)} V^f{}_{xi} * a^- + \sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi} * b^- +$$

$$\sum_{x=0}^{m(f,i)} V^f{}_{xi} * \mathrm{a} + \sum_{x=0}^{m(f,i)} V^f{}_{xi} * \mathrm{b} + \sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi} * \mathrm{a} + \sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi} * \mathrm{b} + V^f{}_{1i+1} * \mathrm{b}$$

OPT:

$$\sum_{x=0}^{m(g,i)} V^g{}_{xi} * \mathrm{a}^- + \sum_{x=0}^{m(f,i)} V^f{}_{xi} * \mathrm{a}^- +$$

$$\sum_{x=0}^{m(f,i)} V^f{}_{xi} * \mathrm{a} + \sum_{x=0}^{m(f,i)} V^f{}_{xi} * \mathrm{b} + \sum_{x=0}^{m(g,i)} V^g{}_{xi} * \mathrm{a} + \sum_{x=0}^{m(g,i)} V^g{}_{xi} * \mathrm{b} + \sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi} * \mathrm{a} +$$

$$\sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi} * \mathrm{b}^- + \sum_{x=m(g,i)+1}^{m^*(g,i)} V^g{}_{xi} * \mathrm{a} + \sum_{x=m(g,i)+1}^{m^*(g,i)} V^g{}_{xi} * \mathrm{b}^- +$$

$$\sum_{x=0}^{m(g,i)} V^g{}_{xi} * \mathrm{a} + \sum_{x=0}^{m(g,i)} V^g{}_{xi} * \mathrm{b} + \sum_{x=m(g,i)+1}^{m^*(g,i)} V^g{}_{xi} * \mathrm{a} + \sum_{x=m(g,i)+1}^{m^*(g,i)} V^g{}_{xi} * \mathrm{b} +$$

$$\min \{ (\sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi} + V^f{}_{1i+1} - \text{reactor capacity}) * (\mathrm{a}+\mathrm{b}) \; ; \; \sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * (\mathrm{a}+\mathrm{b}) \}.$$

Note that

$$\sum_{x=0}^{m(g,i)} V^g{}_{xi} * \mathrm{b} + \sum_{x=0}^{m(g,i)} V^g{}_{xi} * \mathrm{a}^- + \sum_{x=m(g,i)+1}^{m^*(g,i)} V^g{}_{xi} * \mathrm{b}^- > \sum_{x=0}^{m(f,i)} V^f{}_{xi} * \mathrm{a}^- + \sum_{x=0}^{m(f,i)} V^f{}_{xi} * \mathrm{b} \text{ and}$$

$\sum_{x=0}^{m(f,i)} V^f{}_{xi} =$ reactor capacity , $\sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi} <$ reactor capacity and $V^f{}_{1i+1} \leq$ reactor capacity,

LB:

$$\sum_{x=0}^{m(f,i)} V^f{}_{xi} * \mathrm{b} + \sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi} * \mathrm{b} + V^f{}_{1i+1} * \mathrm{b}$$

OPT:

$$\sum_{x=0}^{m(g,i)} V^g{}_{xi} * \mathrm{a}^- + \sum_{x=0}^{m(g,i)} V^g{}_{xi} * \mathrm{a} + \sum_{x=0}^{m(g,i)} V^g{}_{xi} * \mathrm{b} + \sum_{x=m(g,i)+1}^{m^*(g,i)} V^g{}_{xi} * \mathrm{a} + \sum_{x=m(g,i)+1}^{m^*(g,i)} V^g{}_{xi} * \mathrm{b}^- +$$

$$\sum_{x=0}^{m(g,i)} V^g{}_{xi} * \mathrm{a} + \sum_{x=0}^{m(g,i)} V^g{}_{xi} * \mathrm{b} + \sum_{x=m(g,i)+1}^{m^*(g,i)} V^g{}_{xi} * \mathrm{a} + \sum_{x=m(g,i)+1}^{m^*(g,i)} V^g{}_{xi} * \mathrm{b} +$$

$$\min \{ (\sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi} + V^f{}_{1i+1} - \text{reactor capacity}) * (\mathrm{a}+\mathrm{b}) \; ; \; \sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * (\mathrm{a}+\mathrm{b}) \}.$$

If $\sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi}$ + $V^f{}_{1i+1}$ – reactor capacity < $\sum_{x=0}^{m^*(g,i)} V^g{}_{xi}$ then

$\sum_{x=0}^{m(g,i)} V^g{}_{xi}$ * b + $\sum_{x=0}^{m(g,i)} V^g{}_{xi}$ * a^- + $\sum_{x=m(g,i)+1}^{m^*(g,i)} V^g{}_{xi}$ * b^- > $\sum_{x=0}^{m(f,i)} V^f{}_{xi}$ * b and

($\sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi}$ + $V^f{}_{1i+1}$ – reactor capacity)* (a+b) + $\sum_{x=0}^{m^*(g,i)} V^g{}_{xi}$ * (a+b) > $\sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi}$ * b + $V^f{}_{1i+1}$ * b;

If $\sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi}$ + $V^f{}_{1i+1}$ – reactor capacity > $\sum_{x=0}^{m^*(g,i)} V^g{}_{xi}$ then

$\sum_{x=0}^{m(g,i)} V^g{}_{xi}$ * b + $\sum_{x=0}^{m(g,i)} V^g{}_{xi}$ * a^- + $\sum_{x=m(g,i)+1}^{m^*(g,i)} V^g{}_{xi}$ * b^- > $\sum_{x=0}^{m(f,i)} V^f{}_{xi}$ * b and

2 * $\sum_{x=0}^{m^*(g,i)} V^g{}_{xi}$ * (a+b) > $\sum_{x=m(f,i)+1}^{m^*(f,i)} V^f{}_{xi}$ * b + $V^f{}_{1i+1}$ * b;

i.e flowtime charged by the optimal solution procedure is greater than that of lower bound procedure charged.

Case 3:

Let $\sum_{x=0}^{m(f,i)} V^f{}_{xi}$ = $\sum_{x=0}^{m^*(f,i)} V^f{}_{xi}$ ≤ reactor capacity in interval i and in interval i+1, there exist single release with volume $V^f{}_{1i+1}$ and release time $R^f{}_{1i+1}$

LB:

$\sum_{x=0}^{m^*(g,i)} V^g{}_{xi}$ * p + $\sum_{x=0}^{m^*(f,i)} V^f{}_{xi}$ * ($R^f{}_{0i}$ + p – $R^f{}_{xi}$) +

($\sum_{x=0}^{m^*(f,i)} V^f{}_{xi}$ + $\Delta^f{}_{1i+1}$)* p + ($V^f{}_{1i+1}$ -$\Delta^f{}_{1i+1}$)* ($R^f{}_{0i}$ + 2p – $R^f{}_{1i+1}$) +

($V^f{}_{1i+1}$ -$\Delta^f{}_{1i+1}$)* p, where $\Delta^f{}_{1i+1}$ = reactor capacity - $\sum_{x=0}^{m^*(f,i)} V^f{}_{xi}$.

OPT:

$\sum_{x=0}^{m(g,i)} V^g{}_{xi}$ * ($R^f{}_{m(f,i)i}$ – $R^f{}_{xi}$) + $\sum_{x=0}^{m^*(f,i)} V^f{}_{xi}$ * ($R^f{}_{m(f,i)i}$ – $R^f{}_{xi}$) +

$$\sum_{x=0}^{m(f,i)} V^f{}_{xi} * p + \sum_{x=0}^{m(g,i)} V^g{}_{xi} * (R^f{}_{m(f,i)i} + p - R^f{}_{m(f,i)i}) + \sum_{x=m(g,i)+1}^{m^*(g,i)} V^g{}_{xi} * (R^f{}_{m(f,i)i} + p - R^f{}_{xi})$$

+

$$\sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * p + V^f{}_{1i+1} * p + \min \{ \sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * p; V^f{}_{1i+1} * (R^f{}_{m(f,i)i} - R^f{}_{0i}) \}.$$

LB:

$$\sum_{x=0}^{m^*(f,i)} V^f{}_{xi} * b + (V^f{}_{1i+1} - \Delta^f{}_{1i+1}) * b;$$

OPT:

$$\sum_{x=0}^{m(g,i)} V^g{}_{xi} * a^- + \sum_{x=0}^{m(g,i)} V^g{}_{xi} * a + \sum_{x=0}^{m(g,i)} V^g{}_{xi} * b + \sum_{x=m(g,i)+1}^{m^*(g,i)} V^g{}_{xi} * b^- + \sum_{x=m(g,i)+1}^{m^*(g,i)} V^g{}_{xi} * a +$$

$$\min\{ \sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * (a+b); V^f{}_{1i+1} * b \}.$$

Remember that

$$\sum_{x=0}^{m(g,i)} V^g{}_{xi} * b + \sum_{x=0}^{m(g,i)} V^g{}_{xi} * a^- + \sum_{x=m(g,i)+1}^{m^*(g,i)} V^g{}_{xi} * b^- > \sum_{x=0}^{m(f,i)} V^f{}_{xi} * a^- + \sum_{x=0}^{m(f,i)} V^f{}_{xi} * b \text{ then}$$

$$\sum_{x=0}^{m(g,i)} V^g{}_{xi} * b + \sum_{x=0}^{m(g,i)} V^g{}_{xi} * a^- + \sum_{x=m(g,i)+1}^{m^*(g,i)} V^g{}_{xi} * b^- > \sum_{x=0}^{m(f,i)} V^f{}_{xi} * b$$

if $\sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * (a+b) > V^f{}_{1i+1} * b$ then

$V^f{}_{1i+1} * b > (V^f{}_{1i+1} - \Delta^f{}_{1i+1}) * b;$

if $\sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * (a+b) < V^f{}_{1i+1} * b$ then

for the case

reactor capacity - $\sum_{x=0}^{m^*(f,i)} V^f{}_{xi} \geq V^f{}_{1i+1}$ $\Delta^f{}_{1i+1}$ becomes equal to $V^f{}_{1i+1}$

$$\sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * (a+b) > (V^f{}_{1i+1} - \Delta^f{}_{1i+1}) * b;$$

for the case

reactor capacity - $\sum_{x=0}^{m^*(f,i)} V^f{}_{xi} < V^f{}_{1i+1}$ $\Delta^f{}_{1i+1}$ is equal to reactor capacity- $\sum_{x=0}^{m^*(f,i)} V^f{}_{xi}$

$$\sum_{x=0}^{m^*(f,i)} V^f{}_{xi} + \Delta^f{}_{1i+1} \geq V^f{}_{1i+1}$$

$$\sum_{x=0}^{m^*(g,i)} V^g{}_{xi} * (a+b) > (V^f{}_{1i+1} - \Delta^f{}_{1i+1}) * b;$$

Thus total weighted flow time charged by the optimal solution procedure is necessarily greater than that the lower bound procedure charged.