ABSTRACT

Title:	MULTIOBJECTIVE OPTIMIZATION MODELS AND SOLUTION METHODS FOR PLANNING LAND DEVELOPMENT USING MINIMUM SPANNING TREES, LAGRANGIAN RELAXATION AND DECOMPOSITION TECHNIQUES
	José Alberto Faria, Doctor of Philosophy, 2005
Directed By:	Professor Steven A. Gabriel
	Department of Civil and Environmental Engineering and Applied Mathematics and Scientific Computation Program University of Maryland

The land development problem is presented as the optimization of a weighted average of the objectives of three or more stakeholders, subject to develop within bounds residential, industrial and commercial areas that meet governmental goals. The work is broken into three main sections. First, a mixed integer formulation of the problem is presented along with an algorithm based on decomposition techniques that numerically has proven to outperform other solution methods. Second, a quadratic mixed integer programming formulation is presented including a compactness measure as applied to land development. Finally, to prevent the proliferation of sprawl a new measure of compactness that involves the use of the minimum spanning tree is embedded into a mixed integer programming formulation. Despite the exponential number of variables and constraints required to define the minimum spanning tree, this problem was solved using a hybrid algorithm developed in this research.

MULTIOBJECTIVE OPTIMIZATION MODELS AND SOLUTION METHODS FOR PLANNING LAND DEVELOPMENT USING MINIMUM SPANNING TREES, LAGRANGIAN RELAXATION AND DECOMPOSITION TECHNIQUES

By

José Alberto Faria

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park, in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2005

Preface

Since 2001 the author of this doctoral dissertation has been working with Dr. Steve Gabriel and Dr. Glenn Moglen on innovative formulations to select a set of parcels for development among a larger pool currently used as farmland or pristine forest. Two papers have been published as a result of these efforts Moglen, Gabriel and Faria 2003; and Gabriel, Faria and Moglen 2005.

The land development problem was approached as a multiobjective optimization using a weighted average of stakeholders' objectives subject to new housing, industrial, and commercial requirements. Upper and lower bounds in the development have been set to model minimum and maximum demand constraints.

With that background and a keen interest in multiobjective and integer programming, the author has developed mixed integer formulations for the land development problem along with efficient algorithms to solve them. A new measure of compactness involving the use of the minimum spanning tree embedded into a mixed integer programming formulation was introduced to prevent the proliferation of sprawl. Despite the exponential number of variables and constraints required to define the embedded minimum spanning tree, this problem can be solved.

Dedication

To my mom, my wife, my children, and my brother Henrique for their support patience and understanding over the past years… and for those years yet to come.

Acknowledgements

I would like to thank Dr. Gregory Baecher for encouraging me to apply to the Ph.D. program and Dr. Glenn Moglen for giving me the opportunity to work with him over the past three years and for his unconditional help.

I would like to thank all committee members for their valuable comments on the dissertation proposal. All of their comments have been incorporated in this work in particular the following points:

a. Expansion on the multiobjective theory to include Tchebycheff method.

b. Explanation of the "duality gap" phenomenon and methods to prevent missing optimal solutions.

c. Support for the weighting method in the binary case.

d. Applicability of the model for the practitioners.

Especially I would like to thank Dr. Benjamin Hobbs who spent considerable time and effort providing valuable comments and improvements to this work.

Finally, I would like to give special thanks to Dr. Steve Gabriel for being an outstanding teacher, mentor, advisor and friend.

Table of Contents

ABSTRACT ii

Preface ii

Dedication iii

Acknowledgements iv

Table of Contents v

List of Tables x

List of Figures xiii

Notation and Abbreviations xvi

Chapter 1 Findings, Contributions and Literature Review 1

- 1.1. Introduction 1
- 1.2. Research Objectives 4
  - 1.2.1. Alternative Solution Method to Moglen, Gabriel and Faria (2003) 4
  - 1.2.2. Alternative Formulatio n to Moglen, Gabriel and Faria (2003) 4
  - 1.2.3. Create a New Compactness Measure to Prevent Sprawl 5
- 1.3. Research Organization 6
- 1.4. Difference from Previous Work 7
- 1.5. Usability of the Models 8
- 1.6. Literature Review 9
  - 1.6.1. Definition and Measures of Compactness and Sprawl 10
  - 1.6.2. Predictive Land Development Models 14
  - 1.6.3. Normative Land Development Models 17
  - 1.6.4. Work by Moglen, Gabriel and Faria (2003) 23
  - 1.6.5. Work by Gabriel, Faria and Moglen (2005) 26
- 1.7. Chapter Conclusions 29

Chapter 2 Land Development Mixed Integer Formulation 31

- 2.1. Objective Functions 33
- 2.2. Constraints 33
- 2.3. Formulation Using the Weighting Method 34
- Evaluation of the Nine Original Cases 38
  - 2.3.1. Branch and Bound 40

2.3.2. Dantzig-Wolfe Decomposition Technique for Land Development 45
2.3.3. Lagrangian Relaxation Technique for Land Development 62
2.4. Application to Moglen, Gabriel and Faria (2003) 65
2.4.1. Evaluation of Original Weights 65
2.4.2. Evaluation of New Cases: Additional Weights 70
2.5. Algorithm Implementation... 88
2.6. Formulation Using the Constraint Method 91
2.7. Previous Work on Decomposition Heuristics 96
2.8. Chapter Conclusions 97
Chapter 3 Land Development Quadratic Mixed Integer Formulation 99
3.1. Objective Functions 99
3.1.1. The Government Planner 99
3.1.2. The Hydrologist 101
3.1.3. The Conservationist 101
3.1.4. The Land Developer.. 101
3.2. Constraints 101
3.2.1. Formulation Using the Weighting Method 106
3.2.2. Evaluation of the Nine Original Cases.. 106
3.2.3. Problem's Structure... 109
3.3. Nested Decomposition Strategies 110
3.4. Chapter Conclusions 110
Chapter 4 Embedded Minimum Spanning Tree Formulation 112
4.1. Selected Works About Minimum Spanning Trees 112
4.2. The Minimum Spanning Tree in the Land Development Problem........... 114
4.3. The Minimum Spanning Tree as a Compactness Measure 117
4.3.1. Formulation of the Land Development Problem 118
4.3.2. Solution Approach 127
4.3.3. Solution Algorithm 131
4.3.4. Example with 100 Nodes W=(1,1,1) 135
4.3.5. Example with 100 Nodes, 40 Previously Developed and 60 Available for Development Using Other Weight Combinations 141
4.3.6. Example with parcels used in Gabriel, Faria and Moglen 2005 143
4.3.7. Comparison of Results Between Models 152

4.3.8. Improvements to the Algorithm 154
4.3.9. Future Work with MST 155
4.4. Chapter Conclusions 159
Appendix 1 Numerical Results for Models 161
A 1.1. Solution of Difficult Weights 161
A 1.1.1. Weight 964- Original Bounds 161
A 1.1.2. Lagrangian Relaxation 161
A 1.1.3. Dantzig-Wolfe Decomposition 161
A 1.1.4. Weight 662- Original Bounds 162
A 1.1.5. Weight 178- Original Bounds 163
A 1.1.6. Weight 921- Original Bounds 163
A 1.1.7. Weight 389 – Tightened Bounds Case 1 164
A 1.1.8. Weight 176 – Tightened Bounds Case 1 165
A 1.1.9. Weight 178 – Tightened Bounds Case 1 168
A 1.1.10. Weight 459 – Tightened Bounds Case 1 168
A 1.1.11. Weight 176- Tightened Bounds Case 2 169
A 1.1.12. Weight 643- Tightened Bounds Case 2 170
A 1.1.13. Weight 724- Tightened Bounds Case 2 171
A 1.2. List of Weights Used to Find New Cases 174
A 1.3. Formulation to Find u in the Lagrangian Relaxation Example 175
A 1.4. Extract of the Solutions for the 1000 Weights Using Original Bounds 176
A 1.5. Extract of the Pareto Optimal Solutions Using Original Bounds 177
A 1.6. Value Path Graph for Pareto Optimal Points 178
A 1.7. Extract of the Solutions Using Tighter Bounds Case 1 179
A 1.8. Extract of the Pareto Optimal Set Using Tighter Bounds Case 1 180
A 1.9. Extract of the Solutions Using Tighter Bounds Case 2 181
A 1.10. Extract of the Pareto Optimal Set Tighter Bounds Case 2 182
A 1.11. Data Collected Iteration by Iteration Solving Embedded Minimum Spanning Tree Large Scale Parcel Set 183
Appendix 2 Optimization Methods 187
A 2.1. Single Objective Optimization 189
A 2.2. Multiobjective Optimization and Solution Methods 191
A 2.2.1. The Weighted Sum Method 194

A 2.2.2. The Constraint Method 195
A 2.2.3. Weighted Metric Method 196
A 2.2.4. Goal Programming 198
A 2.2.5. Multiobjective Simplex 201
A 2.3. Methods to Solve General Mixed Integer Programming Problems 204
A 2.3.1. Branch and Bound 205
A 2.3.2. Lagrangian Relaxation 212
A 2.3.3. Dantzig-Wolfe Decomposition Method 222
A 2.3.4. Dantzig –Wolfe Algorithm for Integer Programming Problems 229
A 2.3.5. Example 236
A 2.3.6. Benders Decomposition 239
A 2.4. Duality Gap 240
Appendix 3 Introduction to Graph Theory 247
A 3.1. Definitions Related to Graph Theory 247
A 3.2. The Shortest Path Problem 250
A 3.2.1. Graph Scanning Algorithm 251
A 3.2.2. Depth First Search 255
A 3.2.3. Breadth First Search 255
A 3.2.4. Bellman's Principle of Optimality 255
A 3.3. Solutions to the Shortest Path Problem 256
A 3.3.1. Algorithmic Approach 256
A 3.3.2. Modification of the BFS Algorithm 257
A 3.3.3. Dijkstra's Algorithm 257
A 3.3.4. Moore-Bellman-Ford Algorithm 258
A 3.3.5. Floyd-Warshall Algorithm 259
A 3.3.6. Mixed Integer Programming Formulations 259
A 3.4. The Minimum Spanning Tree Problem 261
A 3.5. Solutions to the Minimum Spanning Tree 261
A 3.6. Algorithms 262
A 3.6.1. Prim (1930) 262
A 3.6.2. Kruskal (1956) 262
A 3.6.3. Mixed Integer Programming Formulations 263
A 3.6.4. Polyhedral Description or Packing Formulation 263

A 3.6.5. Cutset Formulation 265
A 3.6.6. Modified Cutset Formulation 267
A 3.6.7. Multi-Cut Formulation 267
A 3.6.8. Single Commodity Flow Formulation 268
A 3.6.9. Directed Multi-commodity Flow Model 270
A 3.6.10. Extended Multi-commodity Flow Model 271
Bibliography 273

List of Tables

Table 1.1 Weight vectors for each stakeholder's objective 28

Table 2.1 Lower and upper bounds for development 36

Table 2.2 Objective function values for cases in Moglen, Gabriel and Faria (2003) . 38

Table 2.3 Normalized objective function values for the nine cases evaluated in Moglen, Gabriel and Faria (2003) and execution times 39

Table 2.4 Branch and bound technique at first level.. 41

Table 2.5 Branch and bound technique at second level... 42

Table 2.6 Structure of the land development problem... 45

Table 2.7 Properties of parcels for reduced example 48

Table 2.8 Block structure of the formulation for the reduced example 49

Table 2.9 List of possible development strategies with three parcels in two zones ... 51

Table 2.10 Possible development strategies with three parcels in two zones............. 52

Table 2.11 Initial solutions obtained from the restricted subproblems by zone 55

Table 2.12 List of feasible solutions from total pool of possible strategies 56

Table 2.13 Simplex tableau for reduced example 56

Table 2.14 Table of coefficients for each parcel for reduced cost computation......... 59

Table 2.15 Lagrangian relaxation for reduced example 64

Table 2.16 Arbitrary coefficients used to find initial solutions 65

Table 2.17 Feasible solutions for zone 1 66

Table 2.18 Feasible solutions for zone 2 66

Table 2.19 Feasible solutions for zone 3 66

Table 2.20 Feasible solutions for zone 4 66

Table 2.21 Feasible solutions for zone 5 66

Table 2.22 New columns generated during the pricing step iteration 1 67

Table 2.23 New columns generated during the pricing step iteration 2 68

Table 2.24 Initial Lagrangian relaxation results with $m_1 = m_2 = 0$ 69

Table 2.25 List of top 15 weights in descending order of solution time 72

Table 2.26 Results from the different methods for different weights original bounds 73

Table 2.27 Results over 50 seconds with original bounds and time limit 120 75

Table 2.28 Results over 40 seconds with original bounds and time limit 60 76

Table 2.29 Summary of objective functions and solution times original bounds....... 76

Table 2.30 Summary of results including the maximum time of 10 seconds on last two columns 77

Table 2.31 Cases evaluated in addition to the original bounds 81

Table 2.32 List of top 10 weights in descending order of solution time case 1 83

Table 2.33 Results from the different methods for different weights case 1 84

Table 2.34 Results for case 1 with one minute time limit 84

Table 2.35 Top 10 results for case 2 listed in descending order of solution time 85

Table 2.36 Comparison of methods case 2 87

Table 2.37 Runs exceeding the time limit original bounds 89

Table 2.38 List of Pareto optimal points exceeding maximum time original bounds 90

Table 2.39 List of Pareto optimal points that exceeded maximum time tightened bounds case 1 90

Table 2.40 List of Pareto optimal points that exceeded maximum time tightened bounds case 2 91

Table 2.41 Bounds on the objectives 93

Table 2.42 Selection of ten bounds for the four objectives 93

Table 2.43 Result for weight 150 case 2 96

Table 2.44 Comparison between the result of weight 150 and the bounds using constraint method 96

Table 3.1 Weights assigned to each stakeholders' objective 107

Table 4.1 Number of constraints as a function of the number of nodes 126

Table 4.2 Data collected per iteration 140

Table 4.3 Cases evaluated with different weights 141

Table 4.4 List of nodes with 10 or more appearances in cycles up to iteration 150 . 147

Table A.1 Lagrangian relaxation result weight 964 original bounds 161

Table A.2 Lagrangian relaxation result weight 662 original bounds 162

Table A.3 Lagrangian relaxation result weight 178 original bounds 163

Table A.4 Lagrangian relaxation result weight 921 original bounds 164

Table A.5 Results of Lagrangian relaxation case 1 weight 389 164

Table A.6 Results of Lagrangian relaxation case 1 weight 176 165

Table A.7 Results of the Lagrangian relaxation case 1 using Bisection 166

Table A.8 Branching for Dantzig-Wolfe decomposition case 1 weight 176 167

Table A.9 Branch and bound result for case 1 weight 178 168

Table A.10 Lagrangian relaxation result for case 1 weight 178 168

Table A.11 Lagrangian relaxation solution case 1 weight 459 169
Table A.12 Lagrangian relaxation results weight 176 case 2 169
Table A.13 Lagrangian relaxation results weight 643 case 2 170
Table A.14 Lagrangian relaxation results weight 724 case 2 171
Table A.15 Results for Dantzig-Wolfe weight 724 case 2 first branch 172
Table A.16 Results for Dantzig-Wolfe weight 724 case 2 second branch 173
Table A.17 Results for Dantzig-Wolfe weight 724 case 2 third branch.................. 173
Table A.18 Extract of the list of weights used to find new cases 174
Table A.19 Extract of the solutions for the 1000 Weights Using Original Bounds . 176
Table A.20 Extract of the Pareto Optimal Solutions for the 1000 Weights Using Original Bounds 177
Table A.21 Extract of the solutions for the 1000 Weights Tighter Bounds Case 1.. 179
Table A.22 Extract of the Pareto optimal solutions for the 1000 weights using tighter bounds case 1 180
Table A.23 Extract of the solutions for 1000 weights tighter bounds case 2 181
Table A.24 Extract of the Pareto optimal solutions for the 1000 weights using tighter bounds case 2 182
Table A.25 Data collected iteration by iteration from solving using all parcels in Moglen, Gabriel, and Faria (2003)... 186
Table A.26 Values of the optimal solution for different iterations 218
Table A.27 Feasible points with objective function values 241
Table A.28 Pareto optimal points obtained by changing weights 242
Table A.29 Solutions to the constraint method for different values of $\underline{z}^2$ 244
Table A.30 Incidence matrix of small undirected graph presented in Figure A.74 .. 252
Table A.31 Incidence matrix of small example presented in Figure A.75 253
Table A.32 Adjacency matrix for the network presented in Figure A.75................. 254
Table A.33 Adjacency list of edges sorted by vertex for Figure A.75 254
Table A.34 Adjacency list of edges per node for network in Figure A.75 254

List of Figures

Figure 1.1 Dissertation organization 6

Figure 1.2 Measure of compactness as the diagonal of the “outer development box” 26

Figure 1.3 Corridor with axes rotated 27

Figure 2.4 Research structure for Chapter 2 32

Figure 2.5 Dantzig-Wolfe decomposition algorithm 47

Figure 2.6 Algorithm to apply Lagrangian relaxation to the integer programming version of the land development problem 62

Figure 2.7 Report for weight 662 71

Figure 2.8 Report for weight 964 72

Figure 2.9 Distribution of IMP solutions original bounds 78

Figure 2.10 Distribution of ENV solutions original bounds 79

Figure 2.11 Distribution of PFA solutions original bounds 79

Figure 2.12 Distribution of PRO solutions original bounds 80

Figure 2.13 Ranges of PFA obtained with original bounds 1000 cases 81

Figure 2.14 Result of branch and bound case 1 weight 176 82

Figure 2.15 Result of branch and bound case 1 weight 389 83

Figure 2.16 Result for weight 176 86

Figure 2.17 Result for weight 643 87

Figure 2.18 Algorithm to solve the integer programming version of the land development problem using Lagrangian relaxation and Dantzig-Wolfe decomposition 88

Figure 3.19 Depiction of the diagonal and the rectangle that encloses all developed parcels 100

Figure 3.20 Division of Montgomery county study into four quadrants 105

Figure 3.21 Parcels assigned to each quadrant 105

Figure 3.22 Value path representation of the nine cases evaluated 108

Figure 3.23 Decomposition structure of the quadratic model for land development 109

Figure 4.24 Algorithm to solve the land development formulation with embedded MST 117

Figure 4.25 Example of four edges connecting five nodes 121

Figure 4.26 Set of one node connected to all other nodes 122

Figure 4.27 Set of two nodes connected to all other nodes 122

Figure 4.28 Graph of five nodes with a table of possible combinations 124
Figure 4.29 Graph of four nodes with a table of possible combinations 125
Figure 4.30 Number of constraints as a function of the number of nodes 127
Figure 4.31 Nodes i and j connected to the existing infrastructure (left using e_{ij}, right using e_{i1} and ej2) 129
Figure 4.32 Two nodes *i,j* available for development connected to previously developed nodes 1 and 2 130
Figure 4.33 Algorithm to solve the land development formulation with embedded MST 132
Figure 4.34 Two undeveloped parcels A, and B and one previously developed parcel 1.. 133
Figure 4.35 Example of a cycle and two disconnected sets..................................... 135
Figure 4.36 Initial set of 40 developed parcels connected with dark edges, and set of 60 parcels available for development (not connected)... 136
Figure 4.37 Solution of first iteration step 1, new edges shown lighter 136
Figure 4.38 Three of the five cycle areas found with the first solution.................... 137
Figure 4.39 Two of the five cycle areas found in the initial solution....................... 137
Figure 4.40 Two of the disconnected components found in the initial solution....... 138
Figure 4.41 Second iteration, solution to the augmented formulation...................... 139
Figure 4.42 Optimal solution found 140
Figure 4.43 Value path graph for the seven cases analyzed 142
Figure 4.44 MST of Case 3 (Left) and Case 4 (right)... 143
Figure 4.45 MST over all existing parcels 144
Figure 4.46 Number of constraints added per iteration using 1990 parcels 145
Figure 4.47 Length of the MST per iteration.. 146
Figure 4.48 Neighborhood of node 1721 147
Figure 4.49 Neighborhood of node 1542 148
Figure 4.50 Potential arcs to connect nodes in the neighborhood of node 1542 149
Figure 4.51 Network of potential arcs 149
Figure 4.52 Nodes near node 1811 150
Figure 4.53 Potential arcs to connect nodes in the neighborhood of node 1811 150
Figure 4.54 Solution obtained using set of all parcels. 151
Figure 4.55 Overlapping of Solutions in Quadrant 3... 153
Figure 4.56 Overlapping of Solutions in Quadrant 1... 154

Figure 4.57 MST of five units on a linear fashion (left) and star fashion (right) 155
Figure 4.58 Existing MST connects nodes 1,2, and 3 other disconnected nodes (left) form a planar graph (right)... 157
Figure A.59 Value Path Graph for Pareto Optimal Points....................................... 178
Figure A.60 Hierarchy of decision support methods 192
Figure A.61 Multiobjective solution methods classification by Cohon (1978)........ 193
Figure A.62 Ideal solution to the two objective case with feasible region in gray... 197
Figure A.63 Tree representation with two branches.. 207
Figure A.64 First branch.. 210
Figure A.65 Second branch.. 210
Figure A.66 Third branch... 211
Figure A.67 Values of the objective function for different values of u.................... 219
Figure A.68 Feasible region.. 220
Figure A.69 Example of feasible region of original problem (shown hatched) as a subset of the feasible region of the relaxed problem with two subsets (shaded gray and unshaded) 226
Figure A.70 Feasible region of example 231
Figure A.71 Two dimensional representation of the example's feasible region...... 232
Figure A.72 Feasible region, and gradients of the weighted objective function...... 243
Figure A.73 Cut set around set of nodes X={1,2,3,4} 249
Figure A.74 Small undirected graph example 252
Figure A.75 Small directed graph example 253
Figure A.76 The cycle between nodes 1, 3 and 5 has three edges............................ 264
Figure A.77 Cut around node 1 (Left) and around node 2 (Right) 266
Figure A.78 Cut around nodes 1 and 2 266
Figure A.79 Example of three cuts 268
Figure A.80 Network with n-1 commodities leaving to each node from node 1...... 270

Notation and Abbreviations

*:	Used as a superscript indicates the optimal value of a solution (e.g., d^*).
1:	Vector of ones.
A:	Matrix of technology coefficients as typically presented in the literature of linear programming (e.g., $Ax \le b$).
a_i:	Area of parcel *i*.
b :	Column vector of the right hand side coefficients (e.g. $Ax \le b$).
B:	Matrix of basic coefficients. This is a square matrix of basic variables that solves the set of constraints when written as $Bx_B = b$.
B^{-1}:	Inverse of the matrix *B*.
c :	Column vector of the coefficients in the objective function of the primal problem as typically presented in the literature.
c_B:	Coefficients of the basic variables.
c_N:	Coefficients of the non-basic variables.
c_0:	Minimum easting of the westernmost developed parcel.
c_1:	Maximum easting of the easternmost developed parcel.
COM:	Commercial zone.
$\underline{COM}$:	Lower bound on the number of acres developed in the commercial zone.
$\overline{COM}$:	Upper bound on the number of acres developed in the commercial zone.
conv(X):	Convex hull of the set *X*.
D:	Matrix of coefficients for the complicating constraints. Also used to name a directed graph.
d_i:	Decision variable to develop or not develop parcel *i*.
$d_{z,t}$:	Decision variable to develop or not develop parcel *t* from zone *z*.
d_m^+:	Deviational variable from goal *m* used in a goal programming formulation
d^-:	Deviational variable from goal *m* used in a goal programming formulation
d:	Decision vector to develop or not develop all the available parcels.
$d(X)$:	Cut around a set of nodes *X*

$d^{+}(X)$:	Set of edges directed into the set of nodes X
$d^{-}(X)$:	Set of edges directed out from the set of nodes X
e:	Edge in a graph
$E(G)$:	The set of edges of the graph G.
e_m:	Limiting value of the mth objective.
f :	Function usually in the objective of the problem.
g :	Function usually in the context of a constraint set with $\geq$ inequalities.
G:	An undirected graph
h:	Function usually in the context of a constraint set of equalities.
GIS:	Geographical information system.
ΔI_i:	Imperviousness change resulting by developing parcel i.
I_Z :	Imperviousness change factor of zone z.
IND:	Industrial zone.
$\underline{IND}$:	Lower bound on the number of acres developed in the industrial zone.
$\overline{IND}$:	Upper bound on the number of acres developed in the industrial zone.
INFORMS:	Institute for operations research and the management sciences.
k:	Number of objective functions. Also used as the number of sets in which a feasible region can be decomposed.
LP:	Linear programming, or linear program.
M:	Number of objectives in a multiobjective optimization problem.
MIP:	Mixed integer programming.
MOP:	Multiobjective optimization problem.
MST:	Minimum spanning tree.
N :	This is a matrix of coefficients of the non-basic variables in the equation $Bx_B + Nx_N = b$.
N_z :	Number of parcels available in zone z.
p_i :	Profit obtained when parcel i is developed.
ParPFA:	Shorthand notation for area in acres developed as a solution of the optimization problem.
PFA:	Set of parcels that belong to the priority funding area.

$\underline{PFA}$:	Lower bound on the area of parcels developed in the priority funding area.
$\overline{PFA}$:	Upper bound on the area of parcels developed in the priority funding area.
r_0:	Minimum northing of the southernmost developed parcel.
r_1:	Maximum northing of the northernmost developed parcel.
$\mathbb{R}^n$:	The *n*-dimensional real numbers.
$\mathbb{R}^n_+$:	The *n*-dimensional nonnegative real numbers.
RHD:	Residential high density zone.
$\underline{RHD}$:	Lower bound on the number of units developed in the residential high density zone.
$\overline{RHD}$:	Upper bound on the number of units developed in the residential high density zone.
RLD:	Residential low density zone.
$\underline{RLD}$:	Lower bound on the number of units developed in the residential low density zone.
$\overline{RLD}$:	Upper bound on the number of units developed in the residential low density zone.
RMD:	Residential medium density zone.
$\underline{RMD}$:	Lower bound on the number of units developed in the residential medium density zone.
$\overline{RMD}$:	Upper bound on the number of units developed in the residential medium density zone
S:	Set of nodes or feasible points, depending on the context.
Sc:	Set of parcels included in the environmentally sensitive set.
LDPP:	Land Development Planning Problem.
T:	When used as a superscript means transposed. i.e. c^T is the row vector resulting from taking the transpose of the column vector *c*.
T_z :	Number of parcels available for development in zone z
TImpCh:	Total value of imperviousness change resulting from the development of parcels selected when solving the optimization problem
u_i :	Units available for development of parcel *i*. In the case of the three residential zones these are dwelling units per parcel, in the commercial and industrial cases this value corresponds to the area in acres of the parcel.

V(G):	The set of nodes (vertices) of the graph G.
x :	Column vector of decision variables typically used in most of the literature to describe the variables of the primal problem in linear programming problems (e.g., $Ax \le b$).
x_i :	The *i*th component of vector x.
x_i^L :	Lower bound of the *i*th component of vector x.
x_i^U :	Upper bound of the *i*th component of vector x.
x_B :	Column vector of basic decision variables obtained when solving the equation $Bx_B + Nx_N = b$.
x_N :	Column vector of non-basic decision variables obtained when solving the equation $Bx_B + Nx_N = b$. These variables take values of zero to solve the system of equalities.
y :	Column vector of decision variables typically used in most of the literature to describe the variables of the dual problem in linear programming problems.
w:	Measure of the objective function of the dual problem. i.e. *w = min: y*
w_k :	Weight of *k*th objective function.
w:	Vector of weights = $\{w_1, w_2, .., w_k\}$.
W:	Set of weight vectors.
z:	Each of the development zones, sometimes used as measurement of the objective function i.e. *z = max: x*.
Z:	Set of all development zones = {RLD, RMD, RHD, COM, IND}.
$\mathbb{Z}^n$:	The *n*-dimensional vector of integer numbers.
$\lceil \bullet \rceil$:	Ceiling of $\bullet$.
$\lfloor \bullet \rfloor$:	Floor of $\bullet$.

Note: Whenever the work of other authors is presented in this dissertation, the original notation will be used.

Chapter 1 Findings, Contributions and Literature Review

This chapter provides background information required to understand the work that has been done before in the area of optimization as applied to planning in land development. Previous compactness measurements are reviewed as they were used to reduce sprawl providing the motivation for new measurements.

1.1. <u>Introduction</u>

In general a land development solution combines two decisions: choosing the land to be selected for development, in our case this means to choose among the available parcels those where the development will take place, and deciding the type of development that should take place on the selected parcels. The different possibilities considered by Moglen, Gabriel and Faria (2003) and Gabriel, Faria and Moglen (2005) included three types of residential housing as well as commercial and industrial use. The difference between these works is that in the former the parcels had a predetermined zone type so the decision was limited to develop or not to develop a parcel and the planner's objective function involved the maximization of development in Priority Funded Areas (PFA's). The PFA's are zones in which the government is interested in promoting development.

In Gabriel, Faria and Moglen (2005), a subset of parcels did not have a predetermined zone, so there was another group of decision variables associated with selecting the zone type for each of those parcels besides the decision whether to develop or not the parcel. The planner's objective was a quadratic mixed integer description of a compactness measurement.

The "Land Development Planning Problem" (LDPP) can be stated as which parcels should be developed and for what purpose. To be a solution, a development plan needs to accommodate the growth of the new and existing residents and businesses. To be optimal it needs to provide stakeholders benefits that cannot be improved by another solution without deteriorating at least one of the stakeholders' objectives. This solution concept is known as "Pareto optimal"; in general there is more than one Pareto optimal solution which forms a Pareto optimal set.

To find if a solution is Pareto optimal, one evaluates the objective functions in objective space (as opposed to decision space). The measurement of the stakeholders' objectives can be arranged as a vector, each solution to the land development problem can be associated with a vector that measures the objectives of all the stakeholders under the proposed solution. Typically these vectors are called criterion vectors (Steuer 2002). A criterion vector is said to dominate the criterion vector of a second solution if for all objectives the first criterion vector provides at least[1] the same values for all the stakeholders as compared with the second, and at least one of the stakeholders gets a strictly better value of their objective function. These "nondominated" vectors identify the Pareto optimal solutions from all the feasible solutions to the problem. The Pareto optimal solutions in decision space are also known as efficient solutions to the multiobjective problem. A solution is efficient if its criterion vector is not dominated by the criterion vector of any other solution.

This notion of tradeoffs between solutions implies that between two Pareto optimal solutions one provides more benefit to a particular individual, group, or

[1] Assuming maximization of the objectives.

organization only at the expense of reducing benefits to at least one other individual or group. There are a large number of potential combinations of parcels to develop, in fact the number is exponential with respect to the number of available parcels. The combination of parcels to develop will be on the order of 2^n since each parcel that belongs to the set of n "developable" parcels can be developed or not developed (two possible states), not all those combinations are feasible solutions. To get a quick idea on the number of potential solutions, Moglen, Gabriel and Faria (2003) used 810 parcels in their work. This represents 2^{810} possible choices of development. Later Gabriel, Faria and Moglen (2005) used 401 parcels whose zones were fixed with an additional set of 512 parcels to be developed in any of the five zone categories (or not developed at all), the resulting number of possible combinations is then $\left(2^{401}\right)\left(6^{512}\right)$.

A complete enumeration of possible solutions, their evaluation for feasibility, and objective function is impractical. Therefore, a multiobjective mixed integer programming formulation was required which was solved by the traditional branch and bound method (Wolsey, 1998; Winston, 2004). However, some cases were difficult to solve this way[2]. Because of these difficult cases, other methods are required to decompose the problem into subproblems that can be solved faster, or to relax the problem for easier computations at the expense of accepting suboptimal solutions. These methods however should find "acceptable solutions" in a "reasonable time". The definition of acceptable and reasonable may vary among both the users and critics of the methods.

[2] One of the cases evaluated in Gabriel, Faria and Moglen (2005) took over 20 hours to solve.

1.2. <u>Research Objectives</u>

1.2.1. Alternative Solution Method to Moglen, Gabriel and Faria (2003)

One of the objectives of this work is to find alternative solution methods to solve the formulation presented in Moglen, Gabriel and Faria (2003). Some of the cases considered in the multiobjective formulation for land development of Moglen, Gabriel and Faria (2003), required considerable computing time to solve. This was the motivation to look for alternative solution methods to solve this particular mixed integer programming formulation. An algorithm based on solving the problem using initially the branch and bound method with a time limit of one minute[3], followed by an application of the Lagrangian relaxation and for the cases where the solution is not optimal, Dantzig-Wolfe decomposition is applied. This hybrid algorithm finds a large set of Pareto optimal solutions in a relatively short time.

1.2.2. Alternative Formulation to Moglen, Gabriel and Faria (2003)

The work in Moglen, Gabriel and Faria (2003) is based on the weighting method to solve the development problem. This method is known to have a potential pitfall in finding Pareto optimal solutions when searching for the complete Pareto optimal set since some points might not be found due to the duality gap (see Appendix 2, page 240). This dissertation work expands that work by presenting a formulation using the constraint method with an example, and a brief description of how this problem can be solved using relaxation and decomposition techniques.

[3] The one minute limit was determined based on numerical evidence with different values tried.

1.2.3. Create a New Compactness Measure to Prevent Sprawl

The work by Gabriel, Moglen and Faria (2005) presents a quadratic objective function to minimize the compactness of a land development plan. This measurement was found to change only by the development of only few critical parcels, a new measurement was conceived that depends on all parcels selected for development.

This work uses the classical concept of a minimum spanning tree (MST) as a measurement for compactness in land development. A mixed integer programming formulation was created for this multiobjective land development problem embedding a minimum spanning tree.

1.2.3.1. Create an Algorithm to Solve the Embedded MST Formulation

Despite the exponential number of constraints involved in the formulations found in the literature, an algorithm was created to solve the land development problem using the minimum spanning tree without explicitly including the exponential number of constraints.

1.3. ***Research Organization***

The material presented in this dissertation is organized as presented in Figure 1.1. The boxes with thicker outline represent original contributions of the author for this dissertation.

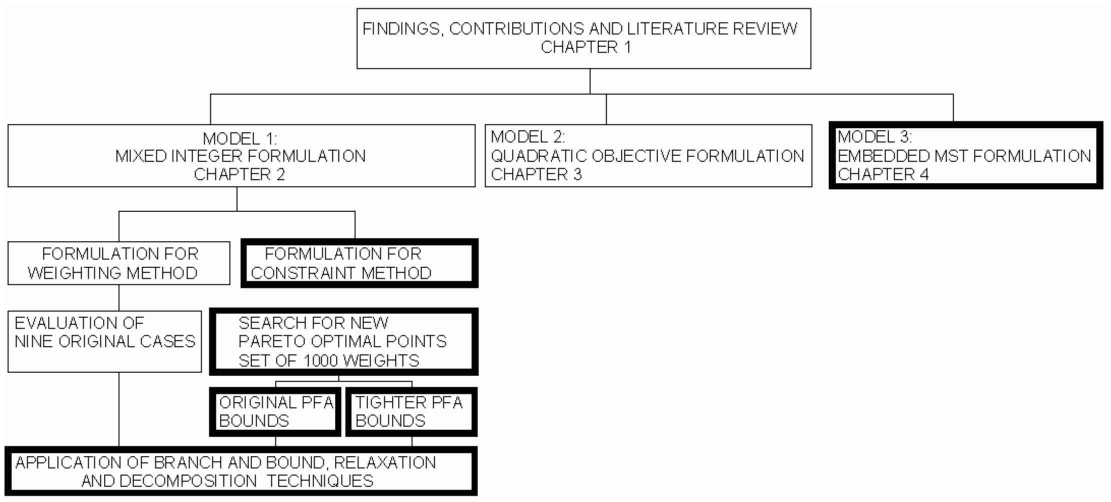

FINDINGS, CONTRIBUTIONS AND LITERATURE REVIEW CHAPTER 1
MODEL 1: MIXED INTEGER FORMULATION CHAPTER 2
MODEL 2: QUADRATIC OBJECTIVE FORMULATION CHAPTER 3
MODEL 3: EMBEDDED MST FORMULATION CHAPTER 4
FORMULATION FOR WEIGHTING METHOD
FORMULATION FOR CONSTRAINT METHOD
EVALUATION OF NINE ORIGINAL CASES
SEARCH FOR NEW PARETO OPTIMAL POINTS SET OF 1000 WEIGHTS
ORIGINAL PFA BOUNDS
TIGHTER PFA BOUNDS
APPLICATION OF BRANCH AND BOUND, RELAXATION AND DECOMPOSITION TECHNIQUES

Figure 1.1 Dissertation organization

Model 1 includes methods to solve the cases (weight assignments for objectives) with long solution time as identified in Moglen, Gabriel and Faria (2003). New random weights were generated creating cases that proved even more difficult to solve in terms of computing time. Also, this dissertation expands Moglen, Gabriel and Faria (2003) by including a formulation using the constraint method instead of the weighting method originally published.

Model 2 includes methods to solve the challenging cases presented in the paper by Gabriel, Faria, and Moglen (2005). The parcels in that work were divided into quadrants where a certain compactness measure was applied related to an outer rectangle encompassing all developed parcels. This research looks into a decomposition of the formulation by those quadrants and also by zones and quadrants proposing a possible decomposition strategy based on the structure of the formulation. We did not applied decomposition or relaxation methods to solve this model, rather a high level view of the possible decomposition strategy was presented.

Model 3 deals with an innovative application of the Minimum Spanning Tree problem used to measure the compactness of the developed area. This work was presented by Faria and Gabriel during the INFORMS conference in Denver 2004.

1.4. <u>Difference from Previous Work</u>

This research differentiates itself from previous work in the following aspects:

1. Innovative use of the minimum spanning tree (MST) as a measurement for compactness in a multiobjective optimization problem applied to land development. To the best of our knowledge no one has set up an embedded minimum spanning tree within this setting.

2. Combination of well-known techniques such as minimum spanning trees, shortest path method, relaxation and decomposition methods, combined in a novel way with graphical and spatial geographical information systems (GIS) in a multiobjective setting for land development.

4. Implementation of decomposition and relaxation techniques for the weighting method for the land development multiobjective optimization in a new way.

5. The formulation mentioned in literal 1 has an exponential number of variables and constraints. An approach to reduce the number of variables and constraints was used, which allows the problem to be solved without explicitly use all constraints.

1.5. ***Usability of the Models***

We expect that the three models could be used in a wide variety of practical and academic settings.

Perhaps the first model, due to its mixed integer linear formulation and decomposition approach, would be attractive to land development stakeholders. We envision this tool as an aid in negotiations over the impact of different policies for zoning, development, and even future transportation initiatives. For academics this model presents an example of mixed integer programming and techniques to solve large scale models by using decomposition, relaxation, column generation and constraints generation strategies.

The other two models, due to their higher level of complexity might be less appealing to practitioners as compared to the first one, although they consider minimizing sprawl as one of the objectives which would be of interest to the planning community.

For the advocates of compact land development, we hope these models will be a useful tool to understand the long term implications of land development decisions. Moreover, the results of relaxation of zone restrictions can be readily applied to support current tendencies of mixed zoning to contain sprawl and encourage walkable communities.

We also envision the models to be used as an example of what can be done for multiobjective optimization applications and decomposition techniques in mathematical and engineering settings both in practical and academic settings.

1.6. Literature Review

This section is divided into five areas starting with a discussion on compactness measurements and sprawl, followed by a brief review of neoclassical economics that predict land development, and prescriptive approaches aimed to plan the development. The last two sections are dedicated to present two land development models used in subsequent chapters.

The development of land is a necessity for human kind, as the population grows, the need for housing grows too. As the economy expands, businesses extend their operations and as a result new facilities are built. Depending on the locations selected, these new developments could cause damage to the environment or to the community itself. “Sprawl growth” is by definition unplanned, randomly selected, scattered and typically outside of traditional development areas. Take for example the definition presented in Gillham (2002):

> “Sprawl is a form of urbanization distinguished by leapfrog patterns of development, commercial strips, low density, separated land uses, automobile dominance, and minimum of public open space.”

Bammi and Bammi (1979) mention this form of sprawl, “linear leapfrog fashion”, as the historical tendency of development in Du Page County near Chicago, Illinois.

This chaotic development translates into the need for infrastructure in the form of water and sewer lines, power lines, sidewalks, communication networks, school districts, etc., required to support small development at relatively remote locations. Gilbert et al. (1985) recognized the desirable effects of compactness of the developed area as a lowering factor in the cost of land development. There is an economic cost associated with building such an infrastructure financed from collection of taxes. There is an environmental cost measured in increased pollution, and there is also an ecological cost measured in reduction of natural resources. The undesirable effects of sprawl have prompted federal and state agencies such as the U.S. Environmental Protection Agency (EPA), the American Association of State Highway Transportation Officials (AASHTO), the University of Maryland, and the Governor of Maryland office among others to create rationalized plans for land development. Such plans are focused on preventing sprawl in order to reduce or "minimize" the impact on cost and the environment, while still promoting economic growth and providing the required housing for the community.

1.6.1. Definition and Measures of Compactness and Sprawl

> "Urban Sprawl is low density, automobile dependent development beyond the edge of service and employment areas. It is ubiquitous and its effects are impacting the quality of life in every region of America, in our large cities and small towns" (EPA, 2005)

Measuring the compactness of a land development project is not as simple as it seems. Knaap, Song and Nedovic-Budic (2004) wrote:

> "Despite the release of several new sprawl indexes, the measurement of sprawl remains an illusive task".

Wolman et al. (2004) agrees, defining what is meant by sprawl, how it should be measured, and what geographical area and type of land should be considered are key factors required to understand the problem and although many researchers have tried no consensus has been reached. The difficulty might be in the multidimensional aspect of the problem where each dimension requires a different measure such as the ones proposed by Torrens and Alberti (2000): density, scatter, leapfrogging, interspersion, and accessibility, or the ones proposed by Galster et al. (2001): density, continuity, concentration, clustering, centrality, nuclearity, mixed uses and proximity. Gillham (2002) claims that sprawl can be seen from an airplane, the main characteristic is the presence of a pattern of developed conglomerates of land surrounded by forests and farms.

Perhaps few of the common descriptions of sprawl include that the sprawl phenomenon is found outside the cities where the new developments are taking place. It is identifiable by the predominance of low density developments (Wolman et al. 2004, Galster et al. 2001), with service and commercial areas reached by vehicle and not primarily by walking due to their separation from the residential areas. These developments are associated with an ad hoc or unplanned fashion (Gillham, 2002). Another term cited in the literature of sprawl is leapfrog development (Bammi and Bammi, 1975; Heim, 2001; Gillhamm, 2002) which refers to the scattered development of land with forests or farm land in between.

Density can be measured as the number of people per unit of area, or as the number of dwelling units per unit of area (Gillham, 2002). Density is the most widely used indicator of sprawl (Burchell and Listokin, 1991; Black, 1996; Torrens and Alberti, 2000; Galster et al., 2001). Galster et al. (2002) proposed the measurement of density as the number of residential units per available area for development, this measure eliminates the commercial and industrial dwellings that are more likely to be clustered together, and also eliminates the parks and areas not available for development.

1.6.1.1. Cost of Sprawl

Almost all the literature about sprawl associates the negative impact of sprawl with the higher cost of infrastructure.

> "Sprawl is creating a hidden debt of unfunded infrastructure and services, social dysfunction, urban decay and environmental degradation". (EPA, 2005)

It seems clear that every unit developed either for commercial, industrial or residential use requires basic services such as electricity, telephone, and some sort of road access, besides service support such as schools, police stations, firefighting stations, etc. If we analyze the cost of connecting the new development to the existing network, we will find a variable cost which is proportional to the distance from the new development to the point of connection, plus some associated fixed cost. So we would expect that the further away a new development is from the interconnection points the more expensive it is to provide the service. Since one of the characteristics of sprawl is having a "leapfrog" type of development with pristine land in between, it is expected to have higher costs as compared to a development plan that maintains a tight distance to the connection points (cities, hubs, highways, etc).

The negative effects of sprawl can be a measure of the cost to society, those unwanted factors are multiple. They cover a large spectrum of social, environmental and technical areas. Some of these factors are traffic congestion with the associated noise and air pollution (Black 1996, Downs 1999), environmental contamination and the destruction of ecosystems (Rees, 1991, Sierra Club 1998), conversion of farmland to urban uses (Bryant and Johnson, 1992, U.S. General Accounting Office 1999). There is also a monetary cost to the public as raised taxes are needed to pay for services such as police and fire departments, infrastructure such as new schools, and roads. Utilities such as water and sewer (Burchell et al. 2000) must be stretched much further to serve the same number of people than they do in the city (Sierra Club 1998). A quick search of “negative effect of sprawl” using Google reported about 118,000 links (April 2005) this provides an idea of how controversial is this topic and how many sources have looked into the sprawl phenomenon.

1.6.1.2. Economic Reasons for Sprawl

The land use in the United States is mostly driven by economic factors rather than by state and regional legislation (Lewis 2001; Gillham 2002). It is a matter of a simple check of the available residential units for sale, to find out that there are many properties located in rural areas whose cost are considerably lower than those in the city or around its immediate borders. This cost factor has made many corporations to move from the city to the rural area in a search for lower costs, better lifestyle and less congestion (Heenan 1999, Gillham 2002). Some homeowners are willing to take longer trips from their residences to their workplace as a tradeoff to be close to open spaces (Wu and Plantinga, 2003).

Gillham (2002) identifies four factors that promote or make sprawl. Land ownership and use is the first. Based on the rights of the owner, the land can be sold, divided, built on, etc. These decisions depend (in most cases) on the land owner and would be driven by the economies of the region and the market. Since 70% of the U.S. land is privately owned, there is a big portion of the decision of land use left to the land owner. The second factor identified as a cause of sprawl is the transportation patterns. It seems rather clear that without a transportation network to tie together the places where people and industries settle it would be impossible to have sprawl. Telecommunications has earned a third place on the list. Since in today's world it is easy to communicate with almost anyone from anywhere, now the geographical location of the business and individuals is less important than ever before. Initially it was the telephone that permitted businesses to be far from each other and still communicate. Today, the computers, internet, cellular technology and satellite communication reduce even more the geographical barriers to do business. Finally the regulations and standards are the ultimate factor that defines the development pattern. How the land is developed and for what use is mostly determined by the economic factors. Although there are some zoning and density regulations to prevent or reduce the arbitrariness of the development decision the final use of the land is mostly dictated by the owner.

1.6.2. Predictive Land Development Models

These models are based on the "first principles" or "neoclassical" tradition of economics focused on explaining and predicting land development. This theory of land development describes the uses of land as they change from one type to another, brings an explanation of why these changes occur, what causes these changes, and what are the mechanisms of change. Consistent with the neoclassical approach, they proceed from fundamental assumptions about consumer utility maximization and producer profit maximization. There is a large literature of land development based on these principles, most of it built upon von Thünen's results.

In 1842 Johann-Heinrich von Thünen published a model for land development based on a central market which is isolated from any external influences and surrounded by unobstructed and unoccupied land with identical climate conditions (Thünen, 1826). The value of the land, given all other factors being equal, is highest in the close proximity to this market place, and reduces as the distance from the marketplace increases.

> "The basic principle was that each piece of land should be devoted to the use in which it would yield the highest rent" (Hoover and Giarratani, 1984).

This model considers the land development as concentric rings with the marketplace in the center. The most profitable perishable goods with high transport costs would be located close to the market, while those less profitable would be found in the outer rings. The original simplistic assumptions used to explain this land rent theory were relaxed by von Thünen himself and by other researchers (Romanos, 1976; Wheeler and Muller, 1981, Hoover and Giarratani 1984).

As a more recent example of the neoclassical approach to land economics, Alonso (1964) describes and explains the residential location of individual households and the resulting spatial structure of an urban area as a function of the rent paid. The rent paid is in time a function of the distance to the market place (city business district).

A different theoretical framework focus on the agents operating in urban contexts and the interactions among them, they take into account the market structure of the urban setting (Christaller, 1966; Pred, 1966; Myrdal, 1957).

For a review of land use change models that include social drivers see Agarwal et al. (2000). For a brief review of land development modeling and economics since von Thünen, see Briassoulis (2000).

A distinct approach to predictive land use model is the agent-based method. In this approach, the decisions of individual agents are modeled, and their actions are the product of heuristics applied under limited foresight, unlike the perfect knowledge and rationality assumptions of the neoclassical model. An example of an agent-based model is Costanza and Wainger (1993) who discuss the extremely complex and nonlinearities of the relations between economics and ecology. This level of complexity influences the predictability of the models, and therefore their use. These relationships require more complex models with larger number of inputs that are sometimes difficult to estimate.

The models presented in this dissertation consider, as one objective, the maximization of profits obtained by a development strategy (among other objectives). In this sense it is broadly consistent with the neoclassical economic concepts. However, the models of this thesis are not meant to be predictive or explanatory of land development in the way the above models are. Rather the models proposed here are normative, suggesting alternative desirable patterns of development for consideration by planners. A further difference with some of the neoclassical models is that the compactness measurements presented in this work are not focused on development around a central market, rather two measurements are used. The first one minimizes the rectangle that encloses all developed parcels (previously developed and selected for development) while the second measurement minimizes the distance required to connect the parcels selected for development to the infrastructure of previously developed parcels (existing infrastructure). This infrastructure is modeled as a network where the parcels are nodes and the arcs are the distances between parcels. However, it would be possible to change the last measurement to minimize the distance to highly populated areas or markets (cities).

1.6.3. Normative Land Development Models

One of the early works that analyzed land development and optimization problems was by Garfinkel and Nemhauser (1970) who developed an algorithm to optimize political district areas. They used the notion of compactness to solve a political redistricting problem which seeks to distribute the population into districts based on contiguity of geographical regions and density of population. They identify at least two possible measures of compactness, one related to geographical compactness and a second one related to population compactness. They depict a compact region as “somewhat circular or squared in shape rather than long and thin”.

Their two-phase method first generates feasible solutions for contiguous districts, compactness and limited population deviation. Then, this procedure finds a set of districts that cover each population exactly once, minimizing the maximum deviation of any district population from the mean. Their model measures compactness using two factors “distance” and “shape”. These definitions are derived from continuity of the units that make up the district and the distance between them. A district is defined as a connected graph with nodes where each node is a district unit and each arc exists only if the nodes are contiguous. Two nodes are said to be contiguous if the border between them is greater than a single point. Then the compactness measure is achieved by computing a dimensionless factor $c = \frac{d_{ij}^2}{A_j}$ where d_{ij} measures the distance between the two farthest apart units of the district called i

and j, and where A_j is the area of the district j. The distance between districts is determined by the shortest path from one point to another since the districts are contiguous. Their model assigns units to districts with compactness and contiguous considerations. The work presented in this dissertation employs similar concepts but in different ways, the shortest path is used to detect cycles in a minimum spanning tree created as a measurement of compactness, we do not impose the continuity requirement but we do promote compactness developments as the tree connecting developed parcels is minimized.

Later, Bammi and Bammi (1975, 1979) developed a multiobjective optimization model with five objectives. The first one minimized a measure of "conflict" between adjacent land uses. They assigned a conflict value for adjacent land uses by creating a table and assigning values to the different possible combinations of adjacent land use, the criteria for weighting these were based on aesthetic factors, noise, pollution, density, transportation, and social and psychological concerns. The second one minimized travel time measuring the distance traveled on trips between the existing developments, and the new land allocated for development by their model. The third objective, minimization of tax cost, included the cost of providing services such as schools to the new communities but also the revenues from taxation to commercial districts searching for a balanced development between costs and revenue sources. The fourth objective, minimization of adverse environmental impact, used an environmental weighting matrix for which a committee of experts in several fields provided their assessments using a scale from zero to twenty for the development. Finally the fifth objective was minimization of cost to build community facilities such as schools, parks, sewage plants, etc. The constraints of the problem were growth requirements per type of use. Their model assigned acreage to each land use type and they considered seven different use types, residential (three sub categories low, medium and high density), commercial, office, research and development, manufacturing, institutional, and open space (two sub categories local and regional). The compactness of the development can be seen as addressed by the minimization of traveling time between existing developments and new developments. This is in a sense a compactness measure but the net result might allow for high density zones to be located near existing population centers (high accessibility) while the low density zones, mostly residential in nature, assigned to the surroundings (low accessibility). In our work by using the minimum spanning tree we are minimizing the total interconnecting distance among developed parcels, this seems to simplify the calculations required to compute the total traveling distance of new owners.

Wright, ReVelle, and Cohon (1983) presented a multiobjective integer programming formulation for land development with an efficient algorithm to find Pareto optimal solutions. They looked at a weighted combination of three objectives: maximize compactness, maximize area, and minimize cost subject to budget constraints, contiguity constraints, number of cells, inclusion of cells in the solution, area constraints regularity of the grid and expansion. The decision variables were to acquire or not acquire individual parcels. Here the constraints included a requirement for contiguity meaning that the land developed should be contiguous, the more general problem of maintaining a compact design without the contiguity requirement requires a different approach. They measured the external border of the development as definition for compactness in their weighted combination of three objectives. This work was extended later by Benabdallah, and Wright (1992). Our work does not impose the contiguity requirement and does not require the land to match a grid pattern required by Wright, ReVelle, and Cohon (1983).

Gilbert, Holmes and Rosenthal (1985) created a four-objective model to minimize acquisition cost, minimize distance to an amenity, which is a place or cell designated as desirable to be close in distance, maximize distance to a detractor or undesirable cell, and minimize a shape measure. Their definition of compactness was the product of the perimeter and the diameter of the set selected. In this setting the perimeter was the number of outside edges and the diameter was the maximum distance between any two cells in the shape. Outside edges were those that divided a cell selected from others that were not selected. An edge between two contiguous selected cells was internal. They used an equally-sized grid to measure the cells so that the number of edges could be used as a measure of distance. They presented a method to generate points that belonged to the Pareto optimal set. They claimed that pursuing the complete set of Pareto optimal points was too large of a task from the computational point of view and cumbersome from the managerial point of view.

The book edited by Beinat, and Nijkamp (1998) includes a set of papers which combine multiobjective land use along with GIS components, and Pullar (1999) who presented a methodology to include spatial iterations as constraints into a multicriteria decision-making process for land allocation with a geographical information system (GIS). The logic of the algorithm was based on computing a weighted average of factors for each location, then ranking them to arrive at the best solution. The algorithm was applied to find the optimal location for timber harvesting evaluating an environmental factor (related to environmentally sensitive areas), a yield indicator, a cost indicator and an allocation indicator related to mill capacity.

Balling et al. 1999 created a multiobjective optimization formulation for urban planning where they minimized traffic congestion, cost related to the maintenance of the network and taxes associated with the land development and zoning considerations, the third objective considered was the minimization of change including rezoning, upgrades to the system, etc. They used genetic algorithms to find a set of Pareto optimal developments.

Similarly Vatalis and Manoliadis (2002) used a multicriteria decision system to find the best location for a waste disposal site. They employed a weighted function of environmental, technical and cost factors to evaluate and rank a list of possible sites. By judging for the number of references obtained in this subject, there are many other researchers who have done multicriteria decision and land development but they seem to be focused on the same principles of the last two references mentioned which are a weighted sum of factors and ranking of possible solutions rather than modeling and solving an optimization problem.

Wu and Plantinga (2003) analyze the impacts of open space that produce leapfrog development they measured the distance between residences and the city business district as the primary amenity for the resident.

Wolman et al. (2005) use the definition of proximity as:

> "the degree to which cross-area observations of a particular land use or pair of land uses are close to each other, relative to the

distribution of all land comprising the study area. Proximity is maximized when all locations with the highest densities of the given land uses are closest together. Low levels of proximity are more sprawl-like."

Gabriel, Faria and Moglen (2005) approached the measure of compactness as the square of the diagonal of a rectangle that encloses all developed parcels. This measurement was proven to have some shortcomings despite the nice mathematical properties of convexity. If the solution proposed to develop a group of parcels falls inside the rectangle formed by the parcels already selected then the compactness measure does not change. Any solution within that rectangle will be considered equivalent. Moreover, this measurement is dependent on the orientation of the axis used to take the measure. The authors worked around the shortcomings of such compactness measurement by dividing the area under study into quadrants, some quadrants resulted with more potential for changes in compactness than others due to the location of the parcels defining the borders of the quadrants.

Aerts et al. (2005) have proposed a goal programming model that contains three spatial compactness objectives using a GIS database and a multicriteria decision making process they search for the lowest cost and also at maximum compactness for land allocation. These spatial compactness objectives are based on size, perimeter and area of a cluster of the same land use. They have divided the land space with a grid with rows and columns, each cell is then assigned to a land use by a model that minimizes cost. Other models are presented considering contiguity of the land selected with same land use. This non linear combinatorial optimization problem was solved using simulated annealing and genetic algorithms.

1.6.4. Work by Moglen, Gabriel and Faria (2003)

The authors presented a model to choose the best development strategy as measured by a weighted sum of the benefits obtained by the stakeholders involved in the land development process. The decision variables were to develop/not develop each parcel in an available set and the constraints were related to the growth of the community. The following stakeholders were considered:

a. The Hydrologist: This stakeholder group has an interest in preserving the environmental conditions of the land as measured by the detriment in the capacity of the soil to permit the absorption of water due to the development of the land (i.e. imperviousness).

b. The Conservationist: A stakeholder group focused on the preservation of the natural resources required by the different species of flora and fauna found in the undeveloped land. The goal of this group is to maintain certain pristine areas undeveloped.

c. The Government Planner: This stakeholder class seeks to maintain an orderly development of communities and is responsible for insuring the existence of sufficient schools, roads, and supportive services such as sewer, fire and police stations, etc.

d. The Land Developer: The collection of individuals and enterprises whose main goal is to obtain an economic benefit from building houses, commercial sites, governmental and industrial facilities.

The local government has established different zone categories for the land to be developed, for example in the state of Maryland the following land uses can be found : residential (low, medium and high density), commercial, institutional, industrial, extractive, open urban land, crop land, pasture, deciduous forest, mixed forest, bush and bare ground.

Developers must adhere to the government's stated land use. These different zone categories can be seen as a mechanism used by the government to maintain groups of similar use within the communities. Consequently, residential land use parcels are expected to be surrounded by other residential parcels, similarly parcels designated for commercial and industrial uses would be contained in commercial areas and industrial parks, respectively. In spite of what we just stated, the modern land development tendency to prevent sprawl seems to integrate commercial and residential use as to minimize automobile usage rather than develop them in different in zones.

Based on the research of Arnold and Gibbons (1996) and Schueler (1994) where imperviousness is used as an index of urban impact, the objective of this group is to minimize the imperviousness of the land as a result of the development. In Moglen, Gabriel and Faria (2003) the Hydrologists' objective function presented sought to minimize the total change of imperviousness defined as

$$\min \sum_{i=1}^{n} (a_i \Delta I_i d_i) \tag{1.1}$$

where n was the number of total parcels under consideration, d_i was a land development variable for parcel i equal to 1 if parcel i was developed, and 0 otherwise, ΔI_i was the change in imperviousness associated with developing parcel i, and a_i was the area of parcel *i*. The Conservationist objective function minimized the development in areas defined as environmentally sensitive expressed as

$$\min \sum_{i \in Sc} a_i d_i \tag{1.2}$$

where the set Sc was the subset of (restricted) parcels that the Conservationist wanted to stop from being developed. The Government Planner sought to steer development inside the zones defined as Priority Funding Areas or PFA's for short. These were regions where the local government provided economic benefits if the development took place, the objective function can be written as

$$\max \sum_{i \in PFA} a_i d_i \tag{1.3}$$

where PFA was the set of parcels designated as Priority Funding Areas. Lastly the land Developer's objective was to maximize net profit computed as:

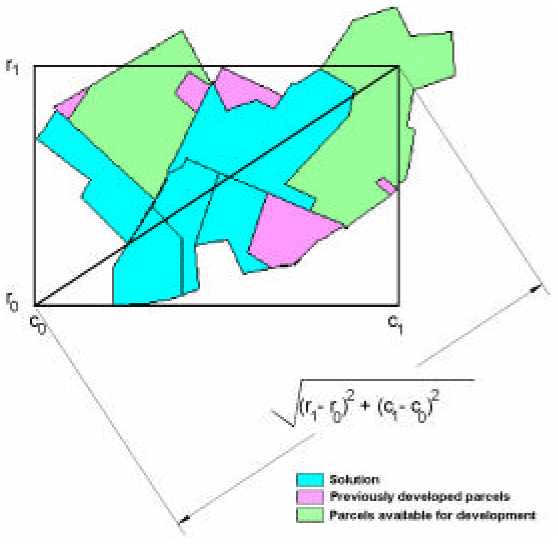
$$\max \sum_{i=1}^{n} p_i d_i \tag{1.4}$$

where p_i was a measure of the economic profit of the parcel i if it was developed, and was statistically determined from actual data.

The land use of each parcel (zone type) was fixed beforehand using a heuristic that considered the distance to existing industrial parks, residential zoning and mayor highways. The problem constraints included minimum and maximum requirements of new housing in terms of units, commercial area and industrial area. The type of development permissible was fixed for each parcel and the development of the parcel was either complete or none meaning that partial development of a parcel was not considered valid. The authors presented a picture of the tradeoffs between stakeholders, and a novel framework for decision making in the land development area.

1.6.5. Work by Gabriel, Faria and Moglen (2005)

Later Gabriel, Faria and Moglen (2005) extended Moglen, Gabriel and Faria (2003) by including in the decision variables the zone type for each parcel, and by changing the planner's objective function to maximize the compactness of the development. This was accomplished by computing the square of the diagonal of an outer rectangle drawn by tracing horizontal lines over the northernmost and southernmost points, and vertical lines over the westernmost and easternmost points as presented in Figure 1.2.



r_1
r_0
c_0
c_1
$\sqrt{(r_1 - r_0)^2 + (c_1 - c_0)^2}$
Solution
Previously developed parcels
Parcels available for development

Figure 1.2 Measure of compactness as the diagonal of the "outer development box"

The actual measure of compactness used was the square of the length of the diagonal (for computationally attractive reasons) so the objective function for the planner became:

$$\min : (r_1 - r_0)^2 + (c_1 - c_0)^2 \tag{1.5}$$

where r_0 was the southernmost coordinate, r_1 was the northernmost coordinate, c_0 was the westernmost and c_1 the easternmost.

This measure of compactness was then evaluated under a rotated axis creating a concept of optimization of land development along a path or corridor as presented in Figure 1.3.

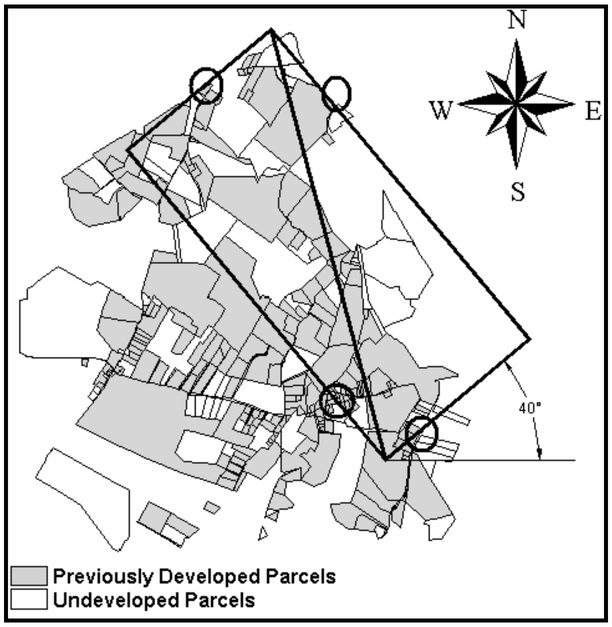


Figure 1.3 Corridor with axes rotated

Besides a different objective function for the Planner in Gabriel, Faria and Moglen (2005), the constraints of the problem were further enhanced by a requirement to assign land use to the parcels. Some parcels were fixed in the type of use available, some others were free to be selected. However, a planning rule applied was that before any of the free zoned parcels could be assigned to any land use, all the available parcels in that land use must be chosen first. For example if there were 15 parcels available for development in the residential high density zoning, all of those should be included in the developed solution before any parcel in the “free” zone set could be selected for development for residential high density purposes.

The weighting method was the approach used to solve the multiobjective optimization problem. Nine different weight vectors were used for the stakeholders’ objectives, and the results of the optimization were analyzed. The nine vectors used were selected to find extreme development points of view mixed with equally weighted consensus as presented in Table 1.1. For example in the case “Planner Alone” the Planners’ objective (compactness) got a weight of 1 while all other objectives got a weight of 0.

Case		Planner (Compactness)	Hydrologist (Imp. Change)	Conservationist (Env. Sensitive Area)	Developer (Profit)	Relative Gap
1	Planner Alone	1	0	0	0	5e-005
2	Planner Pareto	1	0.001	0.001	0.001	5e-005
3	Hydrologist Alone	0	1	0	0	5e-005
4	Hydrologist Pareto	0.001	1	0.001	0.001	5e-005
5	Conservationist Alone	0	0	1	0	5e-005
6	Conservationist Pareto	0.001	0.001	1	0.001	5e-005
7	Developer Alone	0	0	0	1	5e-005
8	Developer Pareto[4]	0.001	0.001	0.001	1	5e-004
9	All Perspectives	1	1	1	1	5e-005

Table 1.1 Weight vectors for each stakeholder’s objective

Note that the rightmost column of Table 1.1 contains a value for the relative gap defined as

[4] A relative gap of 5e-005 was not achievable within a reasonable amount of time. The authors thus slightly relaxed the problem and it solved with a relative gap of 5e-004 instead.

$$\text{Relative Gap} = \frac{\left|\text{Best Solution - Best Bound}\right|}{\text{Best Bound}} \qquad (1.6)$$

This value was required because the solution of the problem for some of the weight vectors would require extremely long times (over 20 hours in some cases). The tolerance chosen is really extremely small, it is possible that in fact the best solution has been found but the solver keeps trying to achieve a bound that cannot be achieved. The relative gap tolerance not necessarily implies that the solver accepts a suboptimal solution in lieu of the optimal solution, but that case is still possible. This tolerance is called relative because the value of the gap is divided by the best bound thus creating a relative value.

1.7. Chapter Conclusions

Finding the best land development plan is not a new problem, many researchers have provided formulations and economic theories aimed to provide an answer to this dilemma, some by rigorous math models, and some by envisioning the ideal community based on environmental requirements and then proposing guidelines to achieve the ideal solution. From the early work of von Thünen the concept of compactness can be seen as develop the land around a central point (could be a city) outwards, facilitating the communication between those individual living in the community. The model developed Gabriel, Faria and Moglen (2005) also looks to maximize compactness but this measure is determined by the parcels that define the outer rectangle, inside the rectangle the selection of the parcels does not affects the compactness of the general development. Other compactness measurements that consider the perimeter of the development have been used on a small scale. Since the compactness measure seems to be an elusive task, a new compactness measure proposed in this dissertation comes to increase the general body of knowledge on this area. This particular aspect of the dissertation can be applied to solve large scale problems of minimum spanning trees in embedded in optimization problems, not necessarily related to land development.

Chapter 2 Land Development Mixed Integer Formulation

In this section we present an algorithm to solve the multiobjective formulation for the land development problem presented in the paper of Moglen, Gabriel and Faria (2003). This formulation has the following considerations:

-All parcels belong to one predetermined zone category, the parcels can be developed only in that zone.

-The Hydrologist is concerned with the minimization of the imperviousness change suffered by the soil by effect of the development of the parcel.

- The Conservationist is concerned with steering the development away from some environmentally sensitive areas.

-The Government Planner is concerned with the maximization of the development in the so called Priority Funding Areas (PFA's).

-The Developer is concerned with the maximization of the profit obtained by the development of the parcels.

The objective of this section is to expand the work of Moglen, Gabriel and Faria (2003) by:

a) Presenting a larger set of "Pareto optimal" solutions.
b) Using the constraint method to solve the multiobjective problem.
c) Presenting an algorithm to derive an approximation of the Pareto optimal set.
d) Analyze the effectiveness of Lagrangian relaxation, branch and bound and Dantzig-Wolfe decomposition when solving the problem at hand.

Previously in Figure 1.1, we presented the roadmap of the dissertation goals. Now we present the branch for this chapter in Figure 2.4.

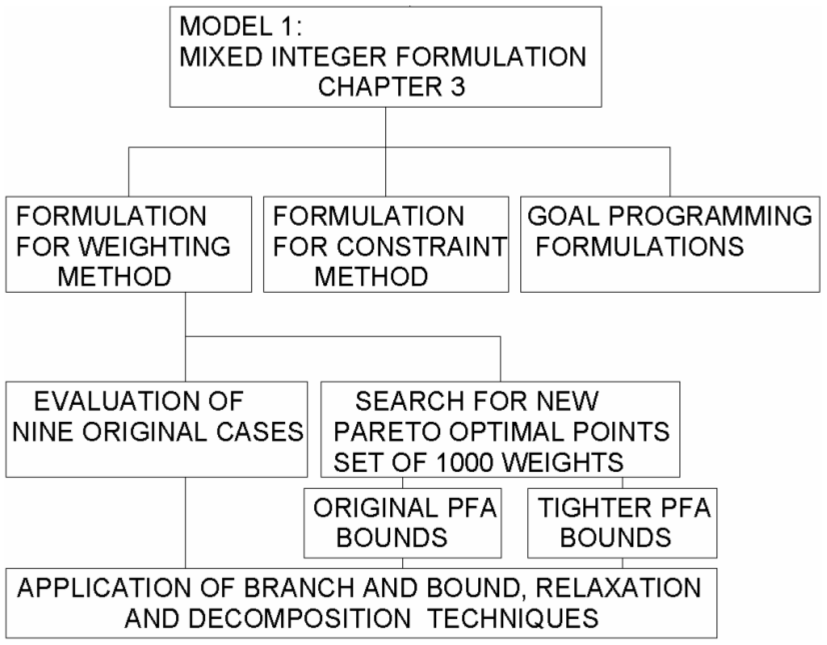

MODEL 1:
MIXED INTEGER FORMULATION
CHAPTER 3
FORMULATION FOR WEIGHTING METHOD
FORMULATION FOR CONSTRAINT METHOD
GOAL PROGRAMMING FORMULATIONS
EVALUATION OF NINE ORIGINAL CASES
SEARCH FOR NEW PARETO OPTIMAL POINTS SET OF 1000 WEIGHTS
ORIGINAL PFA BOUNDS
TIGHTER PFA BOUNDS
APPLICATION OF BRANCH AND BOUND, RELAXATION AND DECOMPOSITION TECHNIQUES

Figure 2.4 Research structure for Chapter 2

The original paper described before under 1.6.4 Work by Moglen, Gabriel and Faria (2003) was conceived as a mixed integer programming problem with multiple linear objectives and linear constraints. The solution approach was to use the weighting method and a set of nine cases were evaluated.

This chapter expands that work in two areas: First it looks into the nine cases evaluated and identifies a case for which the solution took relatively longer time as compared with the rest. Then an algorithm based on relaxation and decomposition methods is applied to solve the problem. Further, a set of 1000 new weights were evaluated using three different bounds on development for the priority funding areas to test the efficiency of the proposed algorithm. The second area focuses on setting up a framework to solve the land development problem using the constraint method.

2.1. ***Objective Functions***

As previously introduced, the Land Development Planning Problem can be set up as a mixed integer programming problem considering four stakeholders. The stakeholders and objectives used in Moglen, Gabriel, and Faria (2003) were:

The Hydrologist: $\min \sum_{i=1}^{n} a_i \Delta I_i d_i$ (2.1)

The Conservationist: $\min \sum_{i \in Sc} a_i d_i$ (2.2)

The Government Planner: $\max \sum_{i \in PFA} a_i d_i$ (2.3)

The Developer: $\max \sum_{i=1}^{n} p_i d_i$ (2.4)

where Sc is the set of environmentally sensitive parcels, PFA is the set of preferred funded parcels and n is the total number of parcels available for development.

2.2. ***Constraints***

The constraints for this problem were to reach a level of development to cover the requirements of residential housing and areas for commercial and industrial growth. These constraints are required to depict market requirements, there is a minimum number of units required to accommodate the growth of the population, but there is also a maximum number of units that would be bought during the period.

Additionally, there are minimum and maximum requirements in terms of area developed in the priority funding areas. This constraint is considered a complicating constraint. The reason for this characterization is that if this constraint weren't present, then the problem could be easily decomposed into five subproblems, one for each zone, as described in a later section.

2.3. Formulation Using the Weighting Method

Consider that there are five zones z numbered 1 to 5 representing respectively, residential low density, residential medium density, residential high density, commercial and industrial zones. Also, consider that each zone z has N_z parcels. Each parcel i belongs to a zone type. In order to reveal the structure that facilitates the decomposition of this formulation by zone types, the subscript i previously used has been substituted by a subscript z,n where z is the zone type and n represents the number of the parcels within that zone. $u_{z,n}$ represents the number of dwelling units of the n^{th} parcel in zone z. Then the mathematical formulation used for this model can be written as follows:

$$\text{Min: } \sum_{z=1}^{5}\sum_{n=1}^{N_z} a_{z,n}\Delta I_{z,n} d_{z,n} \tag{2.5}$$

$$\text{Min: } \sum_{z=1}^{5}\sum_{n\in Sc} a_{z,n} d_{z,n} \tag{2.6}$$

$$\text{Max: } \sum_{z=1}^{5}\sum_{n\in PFA} a_{z,n} d_{z,n} \tag{2.7}$$

$$\text{Max: } \sum_{z=1}^{5}\sum_{n=1}^{N_z} p_{z,n} d_{z,n} \tag{2.8}$$

Subject to:

$$\underline{PFA} \le \sum_{z=1}^{5} \sum_{n \in PFA} a_{z,n} d_{z,n} \le \overline{PFA} \tag{2.9}$$

$$\underline{RLD} \le \sum_{n=1}^{N_1} u_{z,n} d_{z,n} \le \overline{RLD}, z = 1 \tag{2.10}$$

$$\underline{RMD} \le \sum_{n=1}^{N_2} u_{z,n} d_{z,n} \le \overline{RMD}, z = 2 \tag{2.11}$$

$$\underline{RHD} \le \sum_{n=1}^{N_3} u_{z,n} d_{z,n} \le \overline{RHD}, z = 3 \tag{2.12}$$

$$\underline{COM} \le \sum_{n=1}^{N_4} u_{z,n} d_{z,n} \le \overline{COM}, z = 4 \tag{2.13}$$

$$\underline{IND} \le \sum_{n=1}^{N_5} u_{z,n} d_{z,n} \le \overline{IND}, z = 5 \tag{2.14}$$

$$d_{z,n} \in \{0,1\}, n \in \{N_z\}, \forall z \in \{1,2,3,4,5\} \tag{2.15}$$

In Moglen, Gabriel, Faria (2003) the weighting method was used to find a set of solutions to the problem. Not all of the points found in Moglen, Gabriel and Faria (2003) are nondominated, because since some of the weights used are zero, they may be “weakly Pareto optimal”. This means that they will outperform or match any other solution for the objective with positive weight, but there might be another solution with the same value on that objective and better value for at least one other objective. For the cases in which all weights were positive, the resulting solutions are Pareto optimal (Cohon 2003). By applying a weight to each objective the formulation can be written as:

$$\text{Min: } w_1 \sum_{z=1}^{5} \sum_{n=1}^{N_z} a_{z,n} \Delta I_{z,n} d_{z,n} + w_2 \sum_{z=1}^{5} \sum_{n \in Sc} a_{z,n} d_{z,n} - w_3 \sum_{z=1}^{5} \sum_{n \in PFA} a_{z,n} d_{z,n} - w_4 \sum_{z=1}^{5} \sum_{n=1}^{N_z} p_{z,n} d_{z,n} \tag{2.16}$$

s.t.

$$\underline{PFA} \le \sum_{z=1}^{5} \sum_{n \in PFA} a_{z,n} d_{z,n} \le \overline{PFA} \tag{2.17}$$

$$\underline{RLD} \le \sum_{n=1}^{N_1} u_{z,n} d_{z,n} \le \overline{RLD}, z = 1 \tag{2.18}$$

$$\underline{RMD} \le \sum_{n=1}^{N_2} u_{z,n} d_{z,n} \le \overline{RMD}, z = 2 \tag{2.19}$$

$$\underline{RHD} \le \sum_{n=1}^{N_3} u_{z,n} d_{z,n} \le \overline{RHD}, z = 3 \tag{2.20}$$

$$\underline{COM} \le \sum_{n=1}^{N_4} u_{z,n} d_{z,n} \le \overline{COM}, z = 4 \tag{2.21}$$

$$\underline{IND} \le \sum_{n=1}^{N_5} u_{z,n} d_{z,n} \le \overline{IND}, z = 5 \tag{2.22}$$

$$d_{z,n} \in \{0,1\}, n \in \{N_z\}, \forall z \in \{1,2,3,4,5\} \tag{2.23}$$

The minus sign on the objective function for the last two objectives forces the solution to be a maximization of those objectives. The bounds used are presented in Table 2.1.

	Lower Bound	Upper Bound	
Development in Preferred Funded Area	2	1,000	Acres
Development in Residential Low Density Zone	1,554	2,331	Units
Development in Residential Medium Density Zone	8,190	12,285	Units
Development in Residential High Density Zone	4,256	6,384	Units
Development in Commercial Zone	270	406	Acres
Development in Industrial Zone	179	268	Acres

Table 2.1 Lower and upper bounds for development

It is convenient to group the terms in objective function by zone, so (2.16) can be rewritten as:

$$\sum_{z=1}^{5}\left(w_1\sum_{n=1}^{N_z} a_{z,n}\Delta I_{z,n} d_{z,n} + w_2\sum_{n\in Sc} a_{z,n} d_{z,n} - w_3\sum_{n\in PFA} a_{z,n} d_{z,n} - w_4\sum_{n=1}^{N_z} p_{z,n} d_{z,n} \right) \qquad (2.24)$$

It is also convenient to group the terms by each parcel so the objective function (2.24) can be written as:

$$\sum_{z=1}^{5}\sum_{n=1}^{Nz}\left(w_1 a_{z,n}\Delta I_{z,n} + w_2 a_{z,n}^{Sc} - w_3 a_{z,n}^{PFA} - w_4 p_{z,n}\right) d_{z,n} \qquad (2.25)$$

where $a_{z,n}^{Sc}$ is the area of the parcel if the parcel is in the environmentally sensitive area or zero otherwise, and $a_{z,n}^{PFA}$ is the area of the parcel if the parcel is in the priority funding area or zero otherwise. The term inside the parenthesis can be computed for each parcel as:

$$c_{z,n} = w_1 a_{z,n}\Delta I_{z,n} + w_2 a_{z,n}^{Sc} - w_3 a_{z,n}^{PFA} - w_4 p_{z,n} \qquad (2.26)$$

So the integer programming formulation for the land development problem can be written as:

LDIP:

Min: $\sum_{z=1}^{5}\sum_{n=1}^{Nz} c_{z,n} d_{z,n}$ (2.27)

s.t.

$$\underline{PFA} \le \sum_{z=1}^{5}\sum_{n=1}^{Nz} a_{z,n}^{PFA} d_{z,n} \le \overline{PFA} \qquad (2.28)$$

$$\underline{u_z} \le \sum_{n=1}^{N_z} u_{z,n} d_{z,n} \le \overline{u_z},\ z = 1,2,..5 \qquad (2.29)$$

$$d_{z,n} \in \{0,1\}, n \in \{N_z\}, \forall z \in \{1,2,3,4,5\} \qquad (2.30)$$

Evaluation of the Nine Original Cases

In Moglen, Gabriel and Faria (2003) the problem was solved using nine cases as presented in Table 2.2.

	Weights				Original Objective Functions			
Case	Hydrologist	Conservationist	Planner	Developer	(Imp. Change) Hydrologist Objective Area (ha)	(Env. Sensitive Area) Conserv. Objective (ha)	(PFA Area) Planner Objective (ha)	(Profit) Developer Objective ($10e6)
1H	**1**	0	0	0	658.38	1,063.72	173.20	1,091.44
1C	0	**1**	0	0	727.13	17.00	153.41	1,207.87
1P	0	0	**1**	0	760.86	1,007.96	343.65	1,229.96
1D	0	0	0	**1**	997.26	1,446.18	206.31	1,583.71
2	1	1	1	1	689.25	77.04	177.18	1,143.28
3H	**2**	1	1	1	661.71	491.82	182.21	1,096.24
3C	1	**2**	1	1	689.25	77.04	177.18	1,143.28
3P	1	1	**2**	1	693.99	77.04	208.53	1,153.50
3D	1	1	1	**2**	846.71	99.67	181.67	1,449.41

Table 2.2 Objective function values for cases in Moglen, Gabriel and Faria (2003)

The notation for the case is the case (first digit) followed by the stakeholder whose objective function has the greatest weight. For example 1H is the first case for which the Hydrologist is optimized with a weight of one while all others are zero, 3H is a case where the Hydrologists' objective function has a weight of 2 while the others have a weight of 1.

The data used to solve the model was normalized in an attempt to reduce the big differences in scale for the measurements of the objectives.

The normalization method used changed the scale of the original objective's measurement for each parcel to a 0-100 scale applying the following formula.

$$\text{Scaled Value} = 100\frac{\text{Original Value - Min}\{\text{All Values}\}}{\text{Max}\{\text{All Values}\}\text{-Min}\{\text{All Values}\}} \qquad (2.31)$$

The numbers presented in Table 2.2 represent the solutions obtained using the original scale of the objectives. For the rest of this work we will be using normalized data rather than real values unless otherwise noted. The normalized results are presented in Table 2.3 along with an additional column to show the time required to achieve the optimal solution.

	Weights				Normalized Objective Functions				
Case	Hydrologist	Conservationist	Planner	Developer	(Imp. Change) Hydrologist Objective Area (ha)	(Env. Sensitive Area) Conserv. Objective (ha)	(PFA Area) Planner Objective (ha)	(Profit) Developer Objective ($10e6)	Time (s)
1H	**1**	0	0	0	1,051.55	372.30	52.71	763.81	0.51
1C	0	**1**	0	0	1,419.38	1.77	45.75	782.00	0.07
1P	0	0	**1**	0	1,547.89	294.34	79.39	800.59	0.07
1D	0	0	0	**1**	1,706.94	451.65	57.39	1,100.99	595.00
2	1	1	1	1	1,201.43	25.09	54.16	795.79	0.18
3H	**2**	1	1	1	1,084.72	173.06	54.51	765.80	1.08
3C	1	**2**	1	1	1,201.43	25.09	54.16	795.79	1.08
3P	1	1	**2**	1	1,221.01	25.09	62.62	802.47	0.24
3D	1	1	1	**2**	1,502.61	31.85	55.79	1,008.64	0.07

Table 2.3 Normalized objective function values for the nine cases evaluated in Moglen, Gabriel and Faria (2003) and execution times

Note how all but one case was solved in less than two seconds. The case for the Developer alone took the longest time at about ten minutes (595 seconds). Now we will explore other solution methods to reduce the computational time of that particular case.

By carefully observing the weights selected for each of the cases one can predict some conditions of the solutions. For example when the developer's objective is optimized alone, we expect the solution to be binding on the maximum number of parcels allowed for development since as the number of parcels developed increases so does the profit. Similarly we expect the hydrologist's solution to be binding to the lower bound of developed parcels when this perspective is considered alone since the lower the number of parcels developed the lower the imperviousness. In the case of the government planner it is not possible to associate the solution to the bounds of available parcels since the objective function tends to select as many parcels in the PFA area as possible so any feasible solution within the bounds of required development per zone can be selected.

2.3.1. Branch and Bound

A traditional approach to finding the optimal solution is by use of the branch and bound technique (for details of this technique see the Appendix section A 2.3.1). Consider the solution to the integer relaxation found earlier. Relaxing the constraint that forced the variables $d_{z,n}$ to be binary variables, we obtained four parcels with fractional values presented in Table 2.4.

We could take any of these parcels and create two problems, one in which the parcel is forced to be developed and another one where the parcel is forced not to be developed. For example take parcel $d_{11,266}$ to create two problems, one with an additional constraint as:

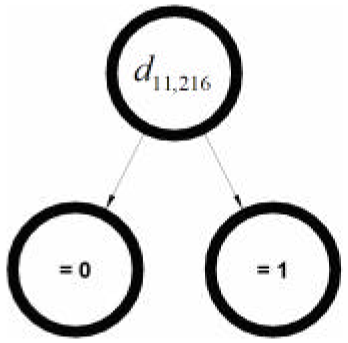
$$d_{11,266} = 0 \tag{2.32}$$

and the other one with

$$d_{11,266} = 1 \tag{2.33}$$

Clearly both cases are mutually exclusive since a parcel cannot be developed and not developed at the same time, and they are also collectively exhaustive since there is no other possible outcome for that particular parcel.

Solving with the relaxed problem including constraint (2.32) provides an optimal solution of z = -1102.43, while solving with constraint (2.33) provides an optimal solution of z = -1102.39 as presented in Table 2.4. At this point we set the lower bound to -1102.43 and since we have no feasible solution, the upper bound is set to infinity. The tree with two branches is presented below with their respective solutions:

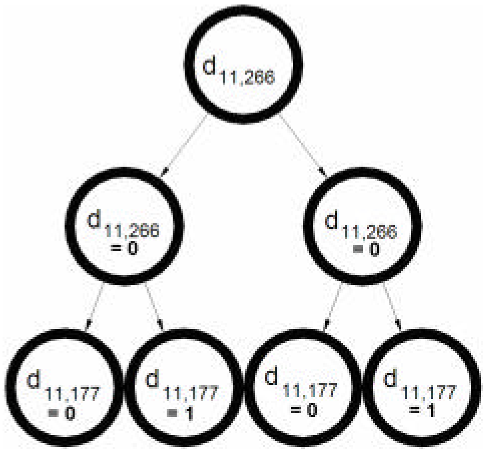


$d_{11,266} = 0$		$d_{11,266} = 1$	
z =-1102.43		z = -1102.39	
Fractional parcels:		Fractional Parcels:	
$d_{11,230}$	0.098413	$d_{11,282}$	0.164835
$d_{12,177}$	0.691223	$d_{12,177}$	0.691223
$d_{14,11}$	0.077997	$d_{14,11}$	0.077997
$d_{15,251}$	0.855685	$d_{15,251}$	0.855685

Table 2.4 Branch and bound technique at first level

These two solutions resulted in more variables being fractional, from each branch we can create other branches by taking each fractional variable and force it binary.

There are some considerations as to which variable should be selected to branch on. Previo usly, we selected $d_{11,266}$ among four possible variables, now we have a choice of four again for each of the two cases resulting from fixing the value of $d_{11,266}$. Some researchers suggest that the best variable to branch on is the one closest to 0.5 In our case that variable would have been $d_{11,177}$. For continuity purposes assume we selected first $d_{11,266}$ and then $d_{12,177}$, the solution would look as presented in Table 2.5.



$d_{11,266} = 0$	$d_{11,266} = 1$
$d_{12,177} = 0$ z = -1102.43	$d_{12,177} = 0$ z = -1102.38
$d_{11,266} = 0$	$d_{11,266} = 1$
$d_{12,177} = 1$ z = -1102.42	$d_{12,177} = 1$ z = -1102.38

Table 2.5 Branch and bound technique at second level

After these nodes have been evaluated we update the lower bound as the minimum of the integer relaxations found, in this case it is -1102.43 (no change). The upper bound is set to infinity since no feasible solution has been found so far (no change).

The branches of the tree will grow until all variables in the solution are binary. The only information known thus far is the upper bound on the objective function. The tree although large does not have to be exhaustively enumerated. As previously presented, there are three cases in which a complete branch of the tree can be eliminated or pruned.

a) Pruned by infeasibility

This case arises when forcing a variable to be binary results in the problem being infeasible. For example, if $d_{15,251}$ which was a fractional variable in Table 2.4 is forced to 1, then the problem becomes infeasible. This happens because the area of that particular parcel combined with others already selected is larger than the upper bound in industrial acres to be developed. Therefore $d_{15,25} = 0$ is required for the problem to be feasible.

b) Pruned by optimality

If we reach a solution in which all variables are binary, then we have obtained a feasible solution which provides an upper bound on the value of the objective function.

Since we are minimizing if we find one feasible solution we obtain an upper bound to the problem. As we find more feasible solutions we could take the smallest of them to be our tightest bound. Once we reach a branch that is optimal there is no need to further evaluate the branch since all variables are already binary. The information obtained in these cases provides valuable insight for the next case since sometimes this solution becomes the best available feasible solution.

c) Pruned by bound

Having an upper bound and a lower bound on z is very valuable since when the relaxed integer problem is solved at a node, if the solution goes over the upper bound (minimization case) then we know that no matter what combination of variables are tried, that branch will not reach an optimal solution because the bound of the best value that can be obtained is already been improved by another feasible solution.

For example if parcel $d_{14,11}$ is forced to zero, then the optimal objective function of the integer relaxation is -1097.59 but this number is worse (higher) than the current best feasible solution of -1099.03. Therefore, regardless of the selection of the rest of the parcels, if this parcel is not developed, then it will be impossible to find a better solution than the one at hand. This leads to cut the branch rooted at $d_{14,11}$=0 by bound, since it will make no improvement to keep following that trajectory.

The solver evaluated over 2.1 million nodes before finally arriving at an optimal solution using the branch and bound method, using a weight of 1 for the Developer and 0 for all other objectives.

Next we will develop the Dantzig-Wolfe decomposition technique and the Lagrangian relaxation methods tailored for the mixed integer land development problem with a small reduced example followed by the proposed algorithm to solve for a list of weighting vectors.

2.3.2. *Dantzig-Wolfe Decomposition Technique for Land Development*

Because each parcel belongs to exactly one of five different parcel sets, the problem's coefficient matrix has the structure presented in Table 2.6 below.

$\sum_{n\in PFA} a_{1,n}d_{1,n}$	$\sum_{n\in PFA} a_{2,n}d_{2,n}$	$\sum_{n\in PFA} a_{3,n}d_{3,n}$	$\sum_{n\in PFA} a_{4,n}d_{4,n}$	$\sum_{n\in PFA} a_{5,n}d_{5,n}$	$\geq$	$\underline{PFA}$
$\sum_{n\in PFA} a_{1,n}d_{1,n}$	$\sum_{n\in PFA} a_{2,n}d_{2,n}$	$\sum_{n\in PFA} a_{3,n}d_{3,n}$	$\sum_{n\in PFA} a_{4,n}d_{4,n}$	$\sum_{n\in PFA} a_{5,n}d_{5,n}$	$\leq$	$\overline{PFA}$
$\sum_{n=1}^{N_{RLD}} u_{1,n}d_{1,n}$					$\geq$	$\underline{RLD}$
$\sum_{n=1}^{N_{RLD}} u_{1,n}d_{1,n}$					$\leq$	$\overline{RLD}$
	$\sum_{n=1}^{N_{RMD}} u_{2,n}d_{2,n}$				$\geq$	$\underline{RMD}$
	$\sum_{n=1}^{N_{RMD}} u_{2,n}d_{2,n}$				$\leq$	$\overline{RMD}$
		$\sum_{n=1}^{N_{RHD}} u_{3,n}d_{3,n}$			$\geq$	$\underline{RHD}$
		$\sum_{n=1}^{N_{RHD}} u_{3,n}d_{3,n}$			$\leq$	$\overline{RHD}$
			$\sum_{n=1}^{N_{COM}} u_{4,n}d_{4,n}$		$\geq$	$\underline{COM}$
			$\sum_{n=1}^{N_{COM}} u_{4,n}d_{4,n}$		$\leq$	$\overline{COM}$
				$\sum_{n=1}^{N_{IND}} u_{5,n}d_{5,n}$	$\geq$	$\underline{IND}$
				$\sum_{n=1}^{N_{IND}} u_{5,n}d_{5,n}$	$\leq$	$\overline{IND}$

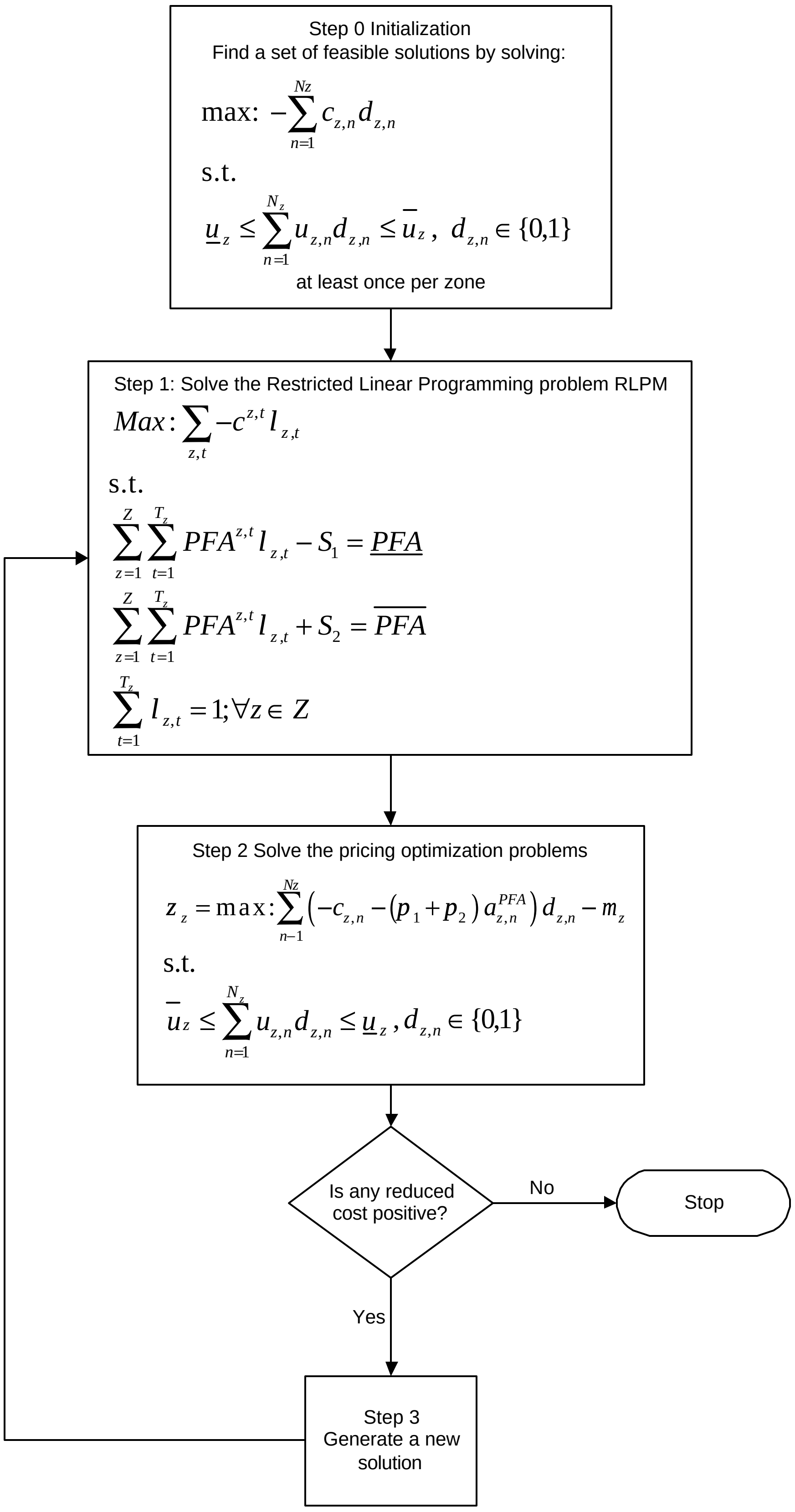
Table 2.6 Structure of the land development problem

Since this is a multiobjective optimization problem, and the proposed method for solving it was the weighting method, then the objective function is a weighted combination of the four objectives as presented in (2.27).

This structure suggests that if the first two rows are eliminated, then the problem can be decomposed into five problems that can be independently solved. However, the solution of such relaxation does not necessarily solve the original problem since the solution might be infeasible due to the elimination of the constraints.

The advantage of this approach is that at a minimum, we have a lower bound on the value of the optimal solution. This information could be used during the execution of the branch and bound optimization strategy to reduce the search space. Also, in general it will be computationally easier to solve five smaller problems and a master problem than one large problem because the number of nodes required will be significantly reduced therefore reducing also the amo unt of memory needed and the execution time. Moreover, the Dantzig-Wolfe decomposition and the Lagrangian relaxation approach provide a mechanism that result in the solution of the original problem by iteratively obtaining better feasible solutions to the original problem via combinations of solutions obtained from the relaxations.

The algorithm (presented in Figure 2.5) starts with a set of feasible solutions, solves a restricted master linear problem and tests for optimality, if the solution is not optimal more feasible solutions are incorporated into the set of feasible solutions, and the restricted master linear problem is solved again.



Step 0 Initialization
Find a set of feasible solutions by solving:
max: $-\sum_{n=1}^{Nz} c_{z,n} d_{z,n}$
s.t.
$\underline{u}_z \leq \sum_{n=1}^{N_z} u_{z,n} d_{z,n} \leq \overline{u}_z, \; d_{z,n} \in \{0,1\}$
at least once per zone
Step 1: Solve the Restricted Linear Programming problem RLPM
$Max: \sum_{z,t} -c^{z,t} l_{z,t}$
s.t.
$\sum_{z=1}^{Z}\sum_{t=1}^{T_z} PFA^{z,t} l_{z,t} - S_1 = \underline{PFA}$
$\sum_{z=1}^{Z}\sum_{t=1}^{T_z} PFA^{z,t} l_{z,t} + S_2 = \overline{PFA}$
$\sum_{t=1}^{T_z} l_{z,t} = 1; \forall z \in Z$
Step 2 Solve the pricing optimization problems
$z_z = \max: \sum_{n-1}^{Nz} \left(-c_{z,n} - (p_1 + p_2)\, a_{z,n}^{PFA}\right) d_{z,n} - m_z$
s.t.
$\overline{u}_z \leq \sum_{n=1}^{N_z} u_{z,n} d_{z,n} \leq \underline{u}_z, d_{z,n} \in \{0,1\}$
Is any reduced cost positive?
No
Stop
Yes
Step 3
Generate a new solution

Figure 2.5 Dantzig-Wolfe decomposition algorithm

The problems mentioned in steps 1, and 2 will be detailed in the next sections along with a small example to detail the procedure, followed by a step-by step description using all parcels from the work of Moglen, Gabriel and Faria (2003).

2.3.2.1. Dantzig-Wolfe Decomposition Reduced Example

To illustrate the decomposition structure of the problem, and the application of the method, let's start with a simple case of land development that has two zones and three parcels in each zone.

Lets suppose that the data from the parcels is as presented in Table 2.7, where the first column (PFA) represents the area of the parcel in the Priority Funding Area, the second column (IMP) represents the change of imperviousness due to the development of the parcel, the third column (ENV) represents the area of the parcel in the environmentally sensitive area, the fourth column (PRO) represents the level of profit obtained by the development of the parcel, the fifth column (U) represents the number of dwelling units that can be built in the parcel.

Zone, Parcel	IMP	ENV	PFA	PROF	U
1,1	7	8	2	2	5
1,2	9	6	3	3	4
1,3	12	4	5	7	2
2,1	8	10	6	4	4
2,2	8	5	8	9	6
2,3	9	5	6	6	8

Table 2.7 Properties of parcels for reduced example

Assuming a weight vector W = (1,1,1,1), the following bounds: $\underline{PFA} = 15$, $\overline{PFA} = 30$, $\underline{u}_1 = 6$, $\overline{u}_1 = 10$, $\underline{u}_2 = 8$ and $\overline{u}_2 = 13$.

The objective function can be computed as:

```
Min  7 d11 +  8 d11 – 2 d11 – 2 d11 +
     9 d12 +  6 d12 - 3 d12 – 3 d12 +
    12 d13 +  4 d13 – 5 d13 – 7 d13 +
     8 d21 + 10 d21 - 6 d21 – 4 d21 +
     8 d22 +  5 d22 - 8 d22 – 9 d22 +
     9 d23 +  5 d23 - 6 d23 – 6 d23
```

Grouping terms the problem can be written as in Table 2:

Min: 11 d11 + 9 d12 + 4 d13	+ 8 d21 - 4 d22 + 2 d23	
s.t.		
2 d11 + 3 d12 + 5 d13	+ 6 d21 + 8 d22 + 6 d23	>= 15
2 d11 + 3 d12 + 5 d13	+ 6 d21 + 8 d22 + 6 d23	<= 30
5 d11 + 4 d12 + 2 d13		>= 6
5 d11 + 4 d12 + 2 d13		<= 10
	4 d21 + 6 d22 + 8 d23	>= 8
	4 d21 + 6 d22 + 8 d23	<= 13

Table 2.8 Block structure of the formulation for the reduced example

It is clear how the formulation when written as in Table 2.8 matches the structure presented in Table 2.6.

The solution to this problem is:

```
        OBJECTIVE FUNCTION VALUE:      17.00000

  VARIABLE        VALUE          REDUCED COST
       D11         0.000000         11.000000
       D12         1.000000          9.000000
       D13         1.000000          4.000000
       D21         1.000000          8.000000
       D22         1.000000         -4.000000
       D23         0.000000          2.000000
```

This solution calls for development of parcels 2 and 3 from zone 1, and parcels 1 and 2 from zone 2 which would yield an objective function of 17.

Consider that each of the zones *z* contains a large but finite set of development strategies *Tz*. Introducing a binary vector of decision variables $l_z = (l_{z,1}, l_{z,2}, ..., l_{z,Tz})$ to decide if a strategy *t* from the set is developed ($l_{z,t} = 1$) or not ($l_{z,t} = 0$), the original problem with the structure presented in Table 2.6 can be re-written as:

$$\min: \sum_{z,t}\left(w_1\sum_{z=1}^{5}\sum_{n=1}^{N_z} a_{z,n}\Delta I_{z,n} d^{t}_{z,n} + w_2\sum_{z=1}^{5}\sum_{n\in Sc} a_{z,n} d^{t}_{z,n} - w_3\sum_{z=1}^{5}\sum_{n\in PFA} a_{z,n} d^{t}_{z,n} - w_4\sum_{z=1}^{5}\sum_{n=1}^{N_z} p_{z,n} d^{t}_{z,n} \right) l_{z,t} \quad (2.34)$$

where $d^{t}_{z,n}$ is the binary decision variable associated to develop under strategy t, the nth parcel in zone z.

To simplify (2.34) we define the following terms:

$$IMP^{z,t} = \sum_{n=1}^{N_z} a_{z,n}\Delta I_{z,n} d^{t}_{z,n} \quad (2.35)$$

$$ENV^{z,t} = \sum_{n\in Sc} a_{z,n} d^{t}_{z,n} \quad (2.36)$$

$$PFA^{z,t} = \sum_{n\in PFA} a_{z,n} d^{t}_{z,n} \quad (2.37)$$

$$PRO^{z,t} = \sum_{t=1}^{N_z} p_{z,n} d^{t}_{z,n} \quad (2.38)$$

$IMP^{z,t}$ accounts for the change of imperviousness resulting from selecting the development strategy *(z,t)* similarly, $ENV^{z,t}$ for the area of environmentally sensitive area, $PFA^{z,t}$ for the area of Priority Funding Area, and $PRO^{z,t}$ for the profit.

So the objective function can be written as:

$$\min: \sum_{z,t}\left(w_1 IMP^{z,t} + w_2 ENV^{z,t} - w_3 PFA^{z,t} - w_4 PRO^{z,t} \right) l_{z,t} \quad (2.39)$$

grouping the terms inside the parenthesis

$$c^{z,t} = w_1 IMP^{z,t} + w_2 ENV^{z,t} - w_3 PFA^{z,t} - w_4 PRO^{z,t} \quad (2.40)$$

the formulation can be written as:

$$\min: \sum_{z,t} c^{z,t} l_{z,t} \tag{2.41}$$

s.t.

$$\underline{PFA} \le \sum_{z,t} PFA^{z,t} l_{z,t} \le \overline{PFA} \tag{2.42}$$

$$\sum_{t=1}^{T_z} l_{z,t} = 1 \text{ for } z = 1,2,3,4,5 \tag{2.43}$$

$$l_{z,t} \in \{0,1\} \tag{2.44}$$

Where each *(z,t)* is a feasible solution or development strategy for zone *z* and there are *Tz* of these feasible solutions in each zone *z*. So the problem is a binary program whose solution is to choose the best development strategy possible.

Since the $l_{z,t}$ are the decision variables, if $l_{z,t}$=1, then development strategy *t* in zone *z* is selected. Constraint (2.43) selects only one development strategy per zone. Going back to the example presented, where we have three 3 parcels per zone, since each parcel can be developed or not, we have $2^3 = 8$ development strategies per zone as presented in Table 2.9. Then there are a total of 16 development strategies divided in two groups of eight. The problem becomes to choose a strategy per zone.

Zone 1	Parcel 1	Parcel 2	Parcel 3	Zone 2	Parcel 1	Parcel 2	Parcel 3
Strategy 1,1	0	0	0	Strategy 2,1	0	0	0
Strategy 1,2	0	0	1	Strategy 2,2	0	0	1
Strategy 1,3	0	1	0	Strategy 2,3	0	1	0
Strategy 1,4	0	1	1	Strategy 2,4	0	1	1
Strategy 1,5	1	0	0	Strategy 2,5	1	0	0
Strategy 1,6	1	0	1	Strategy 2,6	1	0	1
Strategy 1,7	1	1	0	Strategy 2,7	1	1	0
Strategy 1,8	1	1	1	Strategy 2,8	1	1	1

Table 2.9 List of possible development strategies with three parcels in two zones

The result of these calculations using our example are presented in Table 2.10

Zone 1 Strategy	IMP	ENV	PFA	PRO	U	C	Zone 2 Strategy	IMP	ENV	PFA	PRO	U	C
1,1	0	0	0	0	0	0	2,1	0	0	0	0	0	0
1,2	12	4	5	7	2	4	2,2	9	5	6	6	8	2
1,3	9	6	3	3	4	9	2,3	8	5	8	9	6	-4
1,4	21	10	8	10	6	13	2,4	17	10	14	15	14	-2
1,5	7	8	2	2	5	11	2,5	8	10	6	4	4	8
1,6	19	12	7	9	7	15	2,6	17	15	12	10	12	10
1,7	16	14	5	5	9	20	2,7	16	15	14	13	10	4
1,8	28	18	10	12	11	24	2,8	25	20	20	19	18	6

Table 2.10 Possible development strategies with three parcels in two zones

The value of the coefficients C above were computed using a weight vector W = (1,1,1,1). Using the following bounds: $\underline{PFA} = 15$, $\overline{PFA} = 30$, $\underline{u}_1 = 6$, $\overline{u}_1 = 10$, $\underline{u}_2 = 8$ and $\overline{u}_2 = 13$, the formulation can be explicitly written as follows:

$$\text{Min } 0\, l_{1,1} + 4\, l_{1,2} + 9\, l_{1,3} + 13\, l_{1,4} + 11\, l_{1,5} + 15\, l_{1,6} + 20\, l_{1,7} + 24\, l_{1,8} + 0\, l_{21} + 2\, l_{2,2} - 4\, l_{2,3} - 2\, l_{2,4} + 8\, l_{2,5} + 10\, l_{2,6} + 4\, l_{2,7} + 6\, l_{2,8} \tag{2.45}$$

s.t.

$$0\, l_{1,1} + 5\, l_{1,2} + 3\, l_{1,3} + 8\, l_{1,4} + 2\, l_{1,5} + 7\, l_{1,6} + 5\, l_{1,7} + 10\, l_{1,8} + 0\, l_{21} + 6\, l_{2,2} + 8\, l_{2,3} + 14\, l_{2,4} + 6\, l_{2,5} + 12\, l_{2,6} + 14\, l_{2,7} + 20\, l_{2,8} \geq 15 \tag{2.46}$$

$$0\, l_{1,1} + 5\, l_{1,2} + 3\, l_{1,3} + 8\, l_{1,4} + 2\, l_{1,5} + 7\, l_{1,6} + 5\, l_{1,7} + 10\, l_{1,8} + 0\, l_{21} + 6\, l_{2,2} + 8\, l_{2,3} + 14\, l_{2,4} + 6\, l_{2,5} + 12\, l_{2,6} + 14\, l_{2,7} + 20\, l_{2,8} \leq 30 \tag{2.47}$$

$$0\, l_{1,1} + 2\, l_{1,2} + 4\, l_{1,3} + 6\, l_{1,4} + 5\, l_{1,5} + 7\, l_{1,6} + 9\, l_{1,7} + 11\, l_{1,8} \geq 6 \tag{2.48}$$

$$0\, l_{1,1} + 2\, l_{1,2} + 4\, l_{1,3} + 6\, l_{1,4} + 5\, l_{1,5} + 7\, l_{1,6} + 9\, l_{1,7} + 11\, l_{1,8} \leq 10 \tag{2.49}$$

$$l_{1,1} + l_{1,2} + l_{1,3} + l_{1,4} + l_{1,5} + l_{1,6} + l_{1,7} + l_{1,8} = 1 \tag{2.50}$$

$$0\, l_{21} + 8\, l_{2,2} + 6\, l_{2,3} + 14\, l_{2,4} + 4\, l_{2,5} + 12\, l_{2,6} + 10\, l_{2,7} + 18\, l_{2,8} \geq 8 \tag{2.51}$$

$$0\, l_{21} + 8\, l_{2,2} + 6\, l_{2,3} + 14\, l_{2,4} + 4\, l_{2,5} + 12\, l_{2,6} + 10\, l_{2,7} + 18\, l_{2,8} \leq 13 \tag{2.52}$$

$$l_{2,1} + l_{2,2} + l_{2,3} + l_{2,4} + l_{2,5} + l_{2,6} + l_{2,7} + l_{2,8} = 1 \qquad (2.53)$$

$$l_{z,t} \in \{0,1\} \qquad (2.54)$$

The solution to this problem is:

```
        OBJECTIVE FUNCTION VALUE

        1)      17.00000

  VARIABLE        VALUE          REDUCED COST
       L11         0.000000          0.000000
       L12         0.000000          4.000000
       L13         0.000000          9.000000
       L14         1.000000         13.000000
       L15         0.000000         11.000000
       L16         0.000000         15.000000
       L17         0.000000         20.000000
       L18         0.000000         24.000000
       L21         0.000000          0.000000
       L22         0.000000          2.000000
       L23         0.000000         -4.000000
       L24         0.000000         -2.000000
       L25         0.000000          8.000000
       L26         0.000000         10.000000
       L27         1.000000          4.000000
       L28         0.000000          6.000000
```

This solution calls for development strategies 14 and 27. Strategy 14 call for the development of parcels 2 and 3 from zone 1, and strategy 27 calls for the development of parcels 1 and 2 from zone 2. Together they yield an objective function value of 17. As expected this solution is exactly the same as the one obtained by solving the original problem.

This formulation has a decomposable structure since the variables in constraints (2.48) - (2.50) are different from the variables included in constraints (2.51) -(2.53). A procedure to solve this problem would be as follows:

2.3.2.1.1. *Step 0: Initialization:*

Find a set of feasible solution from each zone. To obtain this set we solve a restricted programming subproblem with arbitrary weight coefficients. Suppose we select two weights to obtain two feasible solutions within each zone. Then we would need to solve the following subproblems:

Min: $w_1 IMP^{z,t} + w_2 ENV^{z,t} - w_3 PFA^{z\ t} - w_4 PRO^{z,t}$ (2.55)

s.t.

$$IMP = \sum_{n=1}^{N_z} a_{z,n} \Delta I_{z,n} d_{z,n} \tag{2.56}$$

$$ENV = \sum_{n \in Sc} a_{z,n} d_{z,n} \tag{2.57}$$

$$PFA = \sum_{n \in PFA} a_{z,n} d_{z,n} \tag{2.58}$$

$$PRO = \sum_{n=1}^{N_z} p_{z,n} d_{z,n} \tag{2.59}$$

$$\sum_{n=1}^{N_z} u_{z,n} d_{z,n} \geq \underline{u}_z \tag{2.60}$$

$$\sum_{n=1}^{N_z} u_{z,n} d_{z,n} \leq \bar{u}_z \tag{2.61}$$

$$d_{z,n} \in \{0,1\} \tag{2.62}$$

By solving (2.55) - (2.62) four times, two for each of the zones we obtain the initial feasible solutions. For example, suppose we obtained the feasible solutions presented in Table 2.11

Parcel	Parcel 1	Parcel 2	Parcel 3	IMP	ENV	PFA	PRO	U	C
Zone 1 weight 1	1	0	1	19	12	7	9	7	15
Zone 1 weight 2	1	1	0	16	14	5	5	9	20
Zone 2 weight 1	1	0	1	17	15	12	10	12	10
Zone 2 weight 2	1	1	0	16	15	14	13	10	4

Table 2.11 Initial solutions obtained from the restricted subproblems by zone

2.3.2.1.2. Iteration 1, Step 1: Solve the restricted linear master problem

With at least one solution, we proceed to solve the relaxed restricted master problem:

$$Max : \sum_{z,t} -c^{z,t}\, l_{z,t} \tag{2.63}$$

s.t.

$$\sum_{z=1}^{Z}\sum_{t=1}^{T_z} PFA^{z,t}\, l_{z,t} - S_1 = \underline{PFA} \tag{2.64}$$

$$\sum_{z=1}^{Z}\sum_{t=1}^{T_z} PFA^{z,t}\, l_{z,t} + S_2 = \overline{PFA} \tag{2.65}$$

$$\sum_{t=1}^{T_z} l_{z,t} = 1; \forall z = 1,2 \tag{2.66}$$

where S_1 and S_2 are deviation variables from the PFA bounds. The master problem is set as a maximization problem to mach the notation in the literature. The formulation can be explicitly written as:

$$\text{Max} -15\, l_{1,1} - 20\, l_{1,2} - 10\, l_{2,1} - 4\, l_{2,2} \tag{2.67}$$

s.t.

$$7\, l_{1,1} + 5\, l_{1,2} + 12\, l_{2,1} + 14\, l_{2,2} - S_1 = 15 \tag{2.68}$$

$$7\, l_{1,1} + 5\, l_{1,2} + 12\, l_{2,1} + 14\, l_{2,2} + S_2 = 30 \tag{2.69}$$

$$l_{1,1} + l_{1,2} = 1 \tag{2.70}$$

$$l_{2,1} + l_{2,2} = 1 \tag{2.71}$$

Whose solution is:

$$l_{1,1} = 1,\ l_{1,2} = 0,\ l_{2,1} = 0,\ l_{2,2} = 1,\ p = (0,0) \text{ and } m = (-15,-4)$$

Where p and m are the dual variables corresponding to the first two and last two constraints respectively.

2.3.2.1.3. Iteration 1, Step 2: Optimality check

We need to check whether the set of variables (p, m) is dual feasible for the master problem.

Note that not all the development combinations presented in Table 2.10 are feasible solutions, only the feasible strategies are considered because the solutions to the subproblems are feasible within each zone. They are listed in Table 2.12

Parcel	Parcel 1	Parcel 2	Parcel 3	IMP	ENV	PFA	PRO	U	C
Strategy 1,4	0	1	1	21	10	8	10	6	13
Strategy 1,6	1	0	1	19	12	7	9	7	15
Strategy 1,7	1	1	0	16	14	5	5	9	20
Strategy 2,2	0	0	1	9	5	6	6	8	2
Strategy 2,6	1	0	1	17	15	12	10	12	10
Strategy 2,7	1	1	0	16	15	14	13	10	4

Table 2.12 List of feasible solutions from total pool of possible strategies

The evaluation of each possible feasible solution, as the simplex method would do, can be set up in the form of a typical simplex tableau as presented in Table 2.13.

	$l_{1,1}$	$l_{1,2}$	$l_{1,3}$	$l_{2,1}$	$l_{2,2}$	$l_{2,3}$	S_1	S_2
C	-13	-15	-20	-2	-10	-4	0	0
1)	8	7	5	6	12	14	1	0
2)	8	7	5	6	12	14	0	-1
3)	1	1	1	0	0	0	0	0
4)	0	0	0	1	1	1	0	0
Zj	-15	-15	-15	-4	-4	-4	0	0
Cj-Zj	2	0	-5	2	-6	0	0	0

Table 2.13 Simplex tableau for reduced example

Where $z_j = p^T A^k$

We see that there are some positive reduce reduced costs in particular those corresponding to $l_{1,1}$, and $l_{2,1}$. This result implies that the current solution can be improved by letting any one of those variables with positive reduced cost into the basis.

Rather than evaluate all possible points, we solve an optimization problem.

Let's consider the dual of the Master Problem (2.63) - (2.66) which can be written as:

$$\text{Min } \underline{PFA}p_1 + \overline{PFA}p_2 + \sum_z m_z \tag{2.72}$$

s.t.

$$\sum_{z,t} PFA^{z,t} p_1 + \sum_{z,t} PFA^{z,t} p_2 + E^{'} m \geq -c^{z,t} \tag{2.73}$$

$$-p_1 \geq 0 \tag{2.74}$$

$$p_2 \geq 0 \tag{2.75}$$

Where $E^{'}$ is the transposed matrix of the coefficients for the convex constraints (2.66). This problem has a large number of constraints (one per development strategy) but we have solved the primal problem for a restricted number of development solutions. To find if the solution at hand to the restricted primal problem is dual feasible we can solve the following set of subproblems (one per zone):

$$V_z = \max_t \{-c^{z,t} - \sum_{z,t} PFA^{z,t} p_1 - \sum_{z,t} PFA^{z,t} p_2 - E^{'} m\}, z = 1,2,3,4,5 \tag{2.76}$$

s.t.

$$\bar{u}_z \leq \sum_{n=1}^{N_z} u_{z,n} d_{z,n} \leq \underline{u}_z \tag{2.77}$$

$$d_{z,n} \in \{0,1\} \tag{2.78}$$

In terms of the original decision variables, each one of the subproblems becomes:

$z_z = \max:$

$$\sum_{z,n} -\left(w_1 a_{z,n} \Delta I_{z,n} + w_2 a^{Sc}_{z,n} - w_3 a^{PFA}_{z,n} - w_4 p_{z,n} \right) d_{z,n} - (p_1 + p_2)\, a^{PFA}_{z,n} - m_z \tag{2.79}$$

s.t. (2.77) - (2.78)

where $a^{PFA}_{z,n}$ is the are in the PFA of parcel *z,n*. Using the weighted coefficient calculation for each parcel

$$c_{z,n} = w_1 IMP_{z,n} + w_2 ENV_{z,n} - w_3 PFA_{z,n} - w_4 PRO_{z,n} \tag{2.80}$$

We can write (2.79) as

$$z_z = \max : \left(-c_{z,n} - (p_1 + p_2)\, a^{PFA}_{z,n} \right) d_{z,n} - m_z \tag{2.81}$$

The solution to each subproblem finds the variable with highest reduced cost. If any of those is positive the development vector $d_{z,t} = \left(d_{z,1}, d_{z,2}, ..., d_{z,Nz} \right)$ provides an additional feasible solution for the next iteration.

Going back to our example, the cost coefficients $c_{z,n} = (w_1 IMP_{z,n} + w_2 ENV_{z,n} - w_3 PFA_{z,n} - w_4 PRO_{z,n})$ for each of the parcels, using the weighting vector $w = (1,1,1,1)$ are presented in Table 2.14.

Zone, Parcel	IMP	ENV	PFA	PROF	U	C
1,1	7	8	2	2	5	11
1,2	9	6	3	3	4	9
1,3	12	4	5	7	2	4
2,1	8	10	6	4	4	8
2,2	8	5	8	9	6	-4
2,3	9	5	6	6	8	2

Table 2.14 Table of coefficients for each parcel for reduced cost computation

The terms of (2.76) come from Table 2.14 and from the coefficients of S_1 and S_2 in formulation (2.67) - (2.71).

The pricing subproblems are as follows:

So we have for zone 1:

```
Max (-11 – (0+0)2)d11 +
    (- 9 – (0+0)3)d12 +
    (- 4 – (0+0)5)d13
    - (-15)
s.t.
    - 5 d11 - 4 d12 - 2 d13 <= -6
      5 d11 + 4 d12 + 2 d13 <= 10
End
Int d11
Int d12
Int d13
```

Which is equivalent to:

```
Max -11 d11 - 9 d12 - 4 d13 + 15
s.t.
    - 5 d11 - 4 d12 - 2 d13 <= -6
      5 d11 + 4 d12 + 2 d13 <= 10
End
Int d11
Int d12
Int d13
```

Whose optimal solution is

```
        OBJECTIVE FUNCTION VALUE
        1)     -13.00000

  VARIABLE        VALUE          REDUCED COST
       d11         0.000000         11.000000
       d12         1.000000          9.000000
       d13         1.000000          4.000000
       ROW   SLACK OR SURPLUS     DUAL PRICES
        2)         0.000000          0.000000
        3)         4.000000          0.000000

-13 + 15 = 2
```

And for zone 2 we have:

```
Max (- 8 – (0*6+0*6))d21 +
    (+ 4 – (0*8+0*8))d22 +
    (- 2 – (0*6+0*6))d23
    - (-4)
s.t.
    -4 d21 - 6 d22 - 8 d23 <= -8
     4 d21 + 6 d22 + 8 d23 <= 13
End
Int d21
Int d22
Int d23
```

Which is equivalent to:

```
Max - 8 d21 + 4 d22 - 2 d23 + 4
s.t.
    -4 d21 - 6 d22 - 8 d23 <= -8
     4 d21 + 6 d22 + 8 d23 <= 13
End
Int d21
Int d22
Int d23
```

With optimal solution:

```
        OBJECTIVE FUNCTION VALUE
        1)     -2.000000

  VARIABLE        VALUE          REDUCED COST
       d21         0.000000          8.000000
       d22         0.000000         -4.000000
       d23         1.000000          2.000000

       ROW   SLACK OR SURPLUS     DUAL PRICES
        2)         0.000000          0.000000
        3)         5.000000          0.000000

-2 + 4 = 2
```

These results are as expected equivalent to the results form the simplex method.

2.3.2.1.4. Iteration 1, Step 3: Stopping Criterion

Since there are positive values of the reduced cost, the algorithm moves into the next step.

An alternative stopping criterion is to check if the complicating constraint is met, if so then the solution at hand is optimal.

2.3.2.1.5. Iteration 1, Step 4: Generating a New Column

For our example, feasible solution 1 from zone 1, and feasible solution 3 from zone 2 would enter the group of feasible solutions for consideration on the next iteration.

2.3.2.1.6. Iteration 2, Step 1: Solve the restricted linear master problem

Having now a new solution from each zone, we solve the restricted linear master problem again obtaining: $l_{1,1} = 0$, $l_{1,2}=0$, $l_{1,3}=1$, $l_{2,1}=0$, $l_{2,2}=0.125$, $l_{2,3}=0.875$ $p = (-0.25, 0)$ and $m = (-11, -0.5)$.

2.3.2.1.7. Iteration 2, Step 2: Optimality check

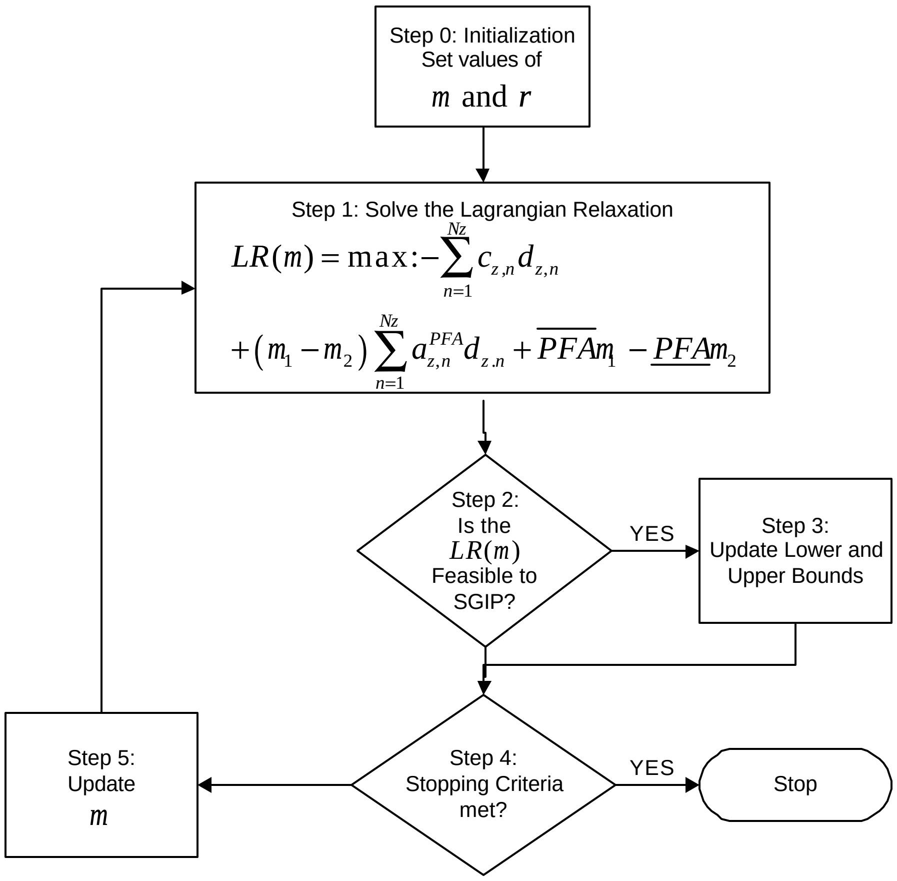
We solved the optimization subproblems (2.79) :

$V_k = \max\{\left(c_k - p\,A^k\right)^T x - m_k : x \in X^k\}$ obtaining an objective function value of 0 for zone 1 and 0 for zone 2.

Since none of the reduced costs is positive, then the solution is optimal. But since the solutions are fractional we need to recourse to branch and bound to find an integer solution. The solution obtained was: $l_{1,1} = 1$, $l_{2,3}=1$ and the objective function is -17 which matches exactly the results previously obtained.

2.3.3. Lagrangian Relaxation Technique for Land Development

From the set of constraints it seems natural to relax the constraints that limit the development in PFA areas. An algorithm is depicted in Figure 2.6.



Step 0: Initialization Set values of m and r
Step 1: Solve the Lagrangian Relaxation
$LR(m) = \max : -\sum_{n=1}^{Nz} c_{z,n} d_{z,n} + (m_1 - m_2)\sum_{n=1}^{Nz} a_{z,n}^{PFA} d_{z,n} + \overline{PFA} m_1 - \underline{PFA} m_2$
Step 2: Is the $LR(m)$ Feasible to SGIP?
YES
Step 3: Update Lower and Upper Bounds
Step 4: Stopping Criteria met?
YES
Stop
Step 5: Update m

Figure 2.6 Algorithm to apply Lagrangian relaxation to the integer programming version of the land development problem

What follows is a description of the steps and the formulations involved in the algorithm including an example using only two zones and three parcels in each zone.

Starting from the integer programming formulation of the land development problem (2.27) - (2.30) we can write the following Lagrangian relaxation:

$$\text{Max:} -\sum_{z=1}^{5}\sum_{n=1}^{Nz} c_{n,z} d_{n,z} + \left(\sum_{z=1}^{5}\sum_{n=1}^{Nz} a_{z,n}^{PFA} d_{z,n} - \underline{PFA}\right) m_1 + \left(\overline{PFA} - \sum_{z=1}^{5}\sum_{n=1}^{Nz} a_{z,n}^{PFA} d_{z,n}\right) m_2 \quad (2.82)$$

The terms in objective function can be grouped by the decision variables so the Lagrangian relaxation of the land development problem can be written as:

$$\text{Max:} -\sum_{z=1}^{5}\sum_{n=1}^{Nz} c_{n,z} d_{n,z} + \left(\sum_{z=1}^{5}\sum_{n=1}^{Nz} a_{z,n}^{PFA} d_{z,n}\right) m_1 - \left(\sum_{z=1}^{5}\sum_{n=1}^{Nz} a_{z,n}^{PFA} d_{z,n}\right) m_2 - \underline{PFA} m_1 + \overline{PFA} m_2 \tag{2.83}$$

s.t.

$$\underline{u}_z \leq \sum_{n=1}^{N_z} u_{z,n} d_{z,n} \leq \overline{u_z}, z = 1,2,..5 \tag{2.84}$$

$$d_{z,n} \in \{0,1\}, n \in \{N_z\}, \forall z \in \{1,2,3,4,5\} \tag{2.85}$$

$$m_1, m_2 \geq 0 \tag{2.86}$$

Once again we see that the formulation can be decomposed since the constraints are independent per zone. The objective function can be broken down per zones also so the final solution is the sum of the subproblems.

The subproblems have the following form:

$$\text{Max: } -\sum_{n=1}^{Nz} c_{n,z} d_{n,z} + \left(\sum_{n=1}^{Nz} a_{z,n}^{PFA} d_{z,n}\right) m_1 - \left(\sum_{n=1}^{Nz} a_{z,n}^{PFA} d_{z,n}\right) m_2 \tag{2.87}$$

s.t.

$$\underline{u}_z \leq \sum_{n=1}^{N_1} u_{z,n} d_{z,n} \leq \overline{u}_z \tag{2.88}$$

$$m_1, m_2 \geq 0 \tag{2.89}$$

There is one subproblem per zone. As previously explained, the Lagrangian relaxation finds an upper bound to the original problem. The challenge is to find values for the multipliers such that we obtain the minimum possible bound.

As an example consider again the reduced case with two zones and three parcels per zone.

Using the explicit formulation as presented before in Table 2.8, we proceed to post the problem as maximization, and to relax the bounds imposed on the PFA obtaining the formulation presented in Table 2.15.

Max: -11 d11 - 9 d12 - 4 d13 - 8 d21 + 4 d22 - 2 d23 (2 d11 + 3 d12 + 5 d13 + 6 d21 + 8 d22 + 6 d23 - 15) m_1 + [30 - (2 d11 + 3 d12 + 5 d13 + 6 d21 + 8 d22 + 6 d23)] m_2 s.t.		
5 d11 + 4 d12 + 2 d13 5 d11 + 4 d12 + 2 d13		>= 6 <= 10
	4 d21 + 6 d22 + 8 d23 4 d21 + 6 d22 + 8 d23	>= 8 <= 13

Table 2.15 Lagrangian relaxation for reduced example

This problem can be decomposed into two problems, one for each zone as follows:

Zone 1

Max: -11 d11 - 9 d12 - 4 d13
(2 d11 + 3 d12 + 5 d13) m_1
- (2 d11 + 3 d12 + 5 d13) m_2
s.t.
5 d11 + 4 d12 + 2 d13 >= 6
5 d11 + 4 d12 + 2 d13 <= 10

Zone 2

Max: - 8 d21 + 4 d22 - 2 d23
(6 d21 + 8 d22 + 6 d23) m_1 -
(6 d21 + 8 d22 + 6 d23)] m_2
s.t.
4 d21 + 6 d22 + 8 d23 >= 8
4 d21 + 6 d22 + 8 d23 <= 13

We started with $m_1 = m_2 = 0$ and obtained the following solution:

$d_{11} = 0, d_{12} = 1, d_{13} = 1, z^1 = -13$ and $d_{21} = 0, d_{22} = 1, d_{23} = 1, z^2 = -2$ so the final objective function $z = z^1 + z^2 = -15$ which coincides with the solution obtained before.

2.4. <u>Application to Moglen, Gabriel and Faria (2003)</u>

2.4.1. Evaluation of Original Weights

We now take the difficult case from Moglen, Gabriel and Faria (2003) and solve it using both Dantzig-Wolfe decomposition and Lagrangian Relaxation,

2.4.1.1. Dantzig-Wolfe Decomposition

The initial solutions required for the initialization step were found by solving the subproblems for each zone with random coefficients in the objective function. This procedure ensures feasibility on the number of units developed per zone, since they meet all constraints. Alternative methods such as heuristics could be used to find initial solutions, but using random coefficients seems easier in this case.

Because two sets of weights could result on the same optimal solution, the initial feasible solutions obtained by this method could be duplicated.

2.4.1.1.1. Step 0: Initialization

To obtain at least one solution per zone, we solved the subproblems for each zone with the following four arbitrary objective function coefficients:

	IMP	ENV	PFA	PRO
1	16.13	2.79	6.75	4.28
2	2.33	12.04	8.95	7.29
3	1.67	0.17	1.3	14.35
4	0	0	1	0

Table 2.16 Arbitrary coefficients used to find initial solutions

For example for zone 1, we solved subproblem (2.55) - (2.62) four times, each time with a different weighting vector from Table 2.16 obtaining four feasible solutions. The four objectives of each solution are listed in Table 2.17. The same logic follows for the other five zones.

Solving the subproblems yielded the following initial solutions per zone:

IMP1	ENV1	PFA1	PRO1
181.37	202.67	0.00	97.90
288.77	0.00	0.00	94.55
286.31	313.50	0.00	146.26
246.85	224.82	4.37	100.02

Table 2.17 Feasible solutions for zone 1

IMP4	ENV4	PFA4	PRO4
147.03	25.09	28.58	120.59
264.13	26.07	29.50	178.95
249.32	32.33	28.58	179.21
233.42	21.26	32.19	136.41

Table 2.20 Feasible solutions for zone 4

IMP2	ENV2	PFA2	PRO2
343.62	39.16	0.00	337.62
582.91	0.00	8.10	503.16
537.68	87.63	0.00	504.97
466.27	87.63	11.67	357.91

Table 2.18 Feasible solutions for zone 2

IMP5	ENV5	PFA5	PRO5
68.41	0.00	0.00	62.67
97.86	0.00	2.13	84.92
97.86	0.00	2.13	84.92
167.41	23.62	5.22	82.29

Table 2.21 Feasible solutions for zone 5

IMP3	ENV3	PFA3	PRO3
314.75	0.00	25.93	148.88
320.38	0.00	25.08	155.42
470.63	0.00	25.93	185.09
470.63	0.00	25.93	185.09

Table 2.19 Feasible solutions for zone 3

2.4.1.1.2. Iteration 1: Step 1: Solve the Restricted Linear Programming Problem

Given these initial solutions we solved the reduced master linear programming (2.63) - (2.66) obtaining the following solution.

Objective function: 1100.45

$l_{1,4} = 1,\ l_{2,4} = 1,\ l_{3,4} = 1,\ l_{4,4} = 1,\ l_{5,3} = 1$

$p_1 = 0,\ p_2 = 0$

$m_1 = 146.263, m_2 = 504.965, m_3 = 185.094, m_4 = 179.211, m_5 = 84.9171$

Note that the l_z are vectors of decision variables to pick one of the feasible development strategies, the vectors have as many components as feasible solutions used in the restricted linear programming master problem, the positive components of the vector are reported here, the rest are zeroes.

2.4.1.1.3. Iteration 1: Step 2: Solve the Pricing Optimization Problems

Using the values of (p, m) obtained in Step 1, we checked for each zone the reduced cost finding the following results:

$z = (0.0002234, 0.124942, -0.000332115, 0.212737, 0.198477)$.

2.4.1.1.4. Iteration 1: Step 3: Is any Reduced Cost Positive?

Yes, there are four positive reduced costs. Therefore the current solution is not optimal and the algorithm goes to the next step.

2.4.1.1.5. Iteration 1: Step 4: Generate a New Column

Each time we solve the pricing problem and we get a positive reduced cost, we also obtain a development strategy that would improve the value of the objective function. We use these results and add them to the initial solutions found in Step 0.

Table 2.22 presents the four solutions that have been added as a result of this step (one per each positive reduced cost).

Zone	IMP	ENV	PFA	PRO
1	316.49	335.03	0.00	146.26
2	537.75	116.40	0.00	505.09
4	309.46	21.75	31.46	179.42
5	102.77	31.07	0.00	85.12

Table 2.22 New columns generated during the pricing step iteration 1

Having these new solutions at hand, we proceed to solve the restricted linear programming problem again.

2.4.1.1.6. Iteration 2: Step 1: Solve the Restricted Linear Programming Problem

Objective function: 1100.986147

$l_{1,4}=1$, $l_{2,6}=1$, $l_{3,4}=1$, $l_{4,6}=1$, $l_{5,6}=1$

$p_1=0$, $p_2=0$

$m_1=146.263$, $m_2=505.09$, $m_3=185.094$, $m_4=179.424$, $m_5=85.1156$

2.4.1.1.7. Iteration 2: Step 2: Solve the Pricing Optimization Problems

$z=(0.0002234, -5.776e-005, -0.000332115, -0.000263231, -2.329e-005)$.

2.4.1.1.8. Iteration 2: Step 3: Is any Reduced Cost Positive?

Yes, there is one positive reduced cost. Therefore the current solution is not optimal and the algorithm goes to the next step.

2.4.1.1.9. Iteration 2: Step 4: Generate a New Column

Table 2.23 presents the solutions that have been added as a result of this step.

Zone	IMP	ENV	PFA	PRO
1	316.49	335.03	0.00	146.26

Table 2.23 New columns generated during the pricing stepiteration 2

2.4.1.1.10. Iteration 3: Step 1: Solve the Restricted Linear Programming Problem

$l_{1,4}=1$, $l_{2,6}=1$, $l_{3,4}=1$, $l_{4,6}=1$, $l_{5,6}=1$

$p_1=0$, $p_2=0$

$m_1=146.263$, $m_2=505.09$, $m_3=185.094$, $m_4=179.424$, $m_5=85.1156$

We have obtained the same solution as in the previous iteration. Therefore the algorithm has to stop due to the lack of progress.

We set the upper bound for this problem as

$$z^{LPM} \leq c^T \tilde{l}^* = \sum_{i=1}^{m} p_i b_i + \sum_{k=1}^{K} m_k + \sum_{k=1}^{K} V_k \qquad (2.90)$$

Since the solution to the restricted linear programming problem is feasible to the original integer programming problem we have that

$z^{LPM} \leq 1100.986147 \leq z^{\mathrm{IP}}$

This bound is very close to the optimal solution of the problem, which is

$z^{IP} = 1100.994233$

Even though we had to solve many linear programming problems (20 for the initial solutions, 5 each time step 1 is executed, 1 for each time step 2 is executed, and 5 each time step 2 is executed) each execution can be solved in about a second so the bound was obtained in less than a minute.

2.4.1.2. Lagrangian Relaxation

Starting with $m_1 = m_2 = 0$ we obtained the first solution for the weight *W*=(0,0,0,1) as follows

	IMP	ENV	PFA	PRO	Objective Function z(u)	Units
Zone 1	316.41	334.91	0.00	146.26	146.26	2331.00
Zone 2	537.75	116.40	0.00	505.09	505.09	12285.00
Zone 3	470.63	0.00	25.93	185.09	185.09	5384.00
Zone 4	309.46	21.75	31.46	179.42	179.42	405.93
Zone 5	102.77	31.07	0.00	85.12	85.12	267.98
Total	1737.02	504.12	57.39	1100.99	1100.99	

Table 2.24 Initial Lagrangian relaxation results with $m_1 = m_2 = 0$

We observe that for the first solution the LHS of the complicating constraints $\left(\overline{PFA} - \sum_{n \in PFA} a_{z,n} d_{z,n} \right)$ and $\left(\sum_{n \in PFA} a_{z,n} d_{z,n} - \underline{PFA} \right)$ are 942.60 and 55.39 respectively. Since both terms are positive then this solution is feasible to the original problem therefore this solution is optimal.

2.4.2. Evaluation of New Cases: Additional Weights

The fact that one of the previously evaluated weight vectors was difficult to solve created the motivation to apply the decomposition techniques previously used. The question remains on how efficient these techniques are when a large set of weights is used.

To answer this question we generated a set of 1000 random weights[5] and solved the problem for each one of them using the traditional branch and bound technique. We found that if we solve those 1000 cases with the branch and bound algorithm but with a stopping criterion of one minute, then we can look into those cases that needed extra time and apply Lagrangian relaxation. If the solution found when solving the Lagrangian relaxation problem is the same as the one obtained by the branch and bound procedure then we stop because we would know that the branch and bound has found an optimal solution but keeps searching because there is still a gap to the best bound, otherwise we proceed to apply Dantzig-Wolfe decomposition. If still the solution is not optimal then we conclude that the case cannot be solved in short amount of time using alternative methods and the full branch and bound algorithm needs to be used.

[5] An extract of the list of weights used can be found in Table A.18 in the appendix.

To reach this conclusion we ran a series of tests obtaining the results summarized in the next sections. Starting with the original bounds we found that weight sets (662), and (921) resulted in tremendous computational effort as seen in Figure 2.7 and Figure 2.8. The detailed computations for each method and each weight set evaluated can be found in the appendix.

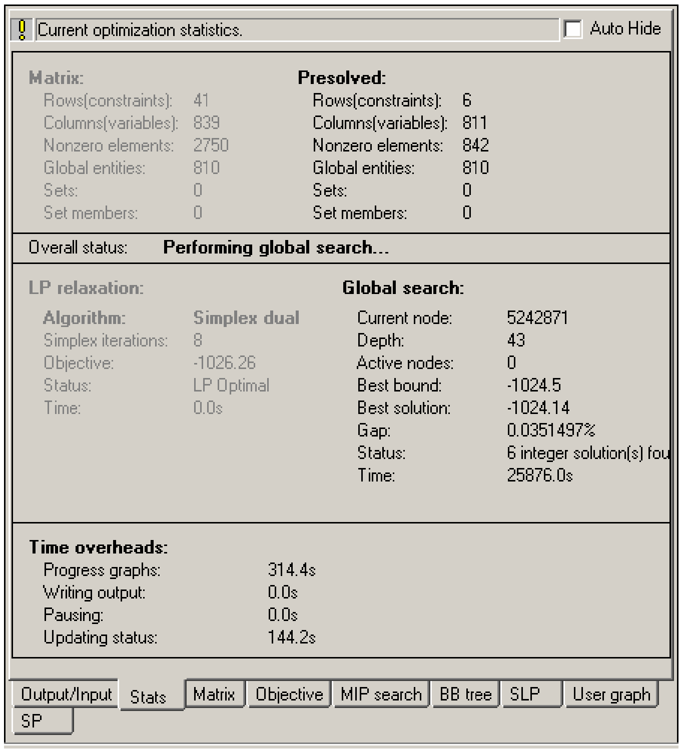
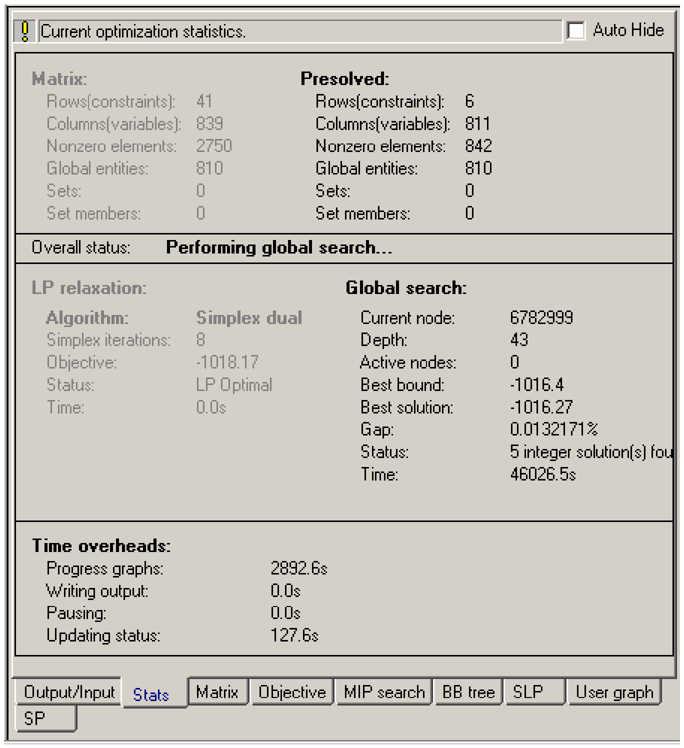

```
Current optimization statistics.                          Auto Hide

Matrix:                                Presolved:
  Rows(constraints):     41              Rows(constraints):     6
  Columns(variables):    839             Columns(variables):    811
  Nonzero elements:      2750            Nonzero elements:      842
  Global entities:       810             Global entities:       810
  Sets:                  0               Sets:                  0
  Set members:           0               Set members:           0

Overall status:     Performing global search...

LP relaxation:                           Global search:
  Algorithm:             Simplex dual      Current node:      5242871
  Simplex iterations:    8                 Depth:             43
  Objective:             -1026.26          Active nodes:      0
  Status:                LP Optimal        Best bound:        -1024.5
  Time:                  0.0s              Best solution:     -1024.14
                                           Gap:               0.0351497%
                                           Status:            6 integer solution(s) fou
                                           Time:              25876.0s

Time overheads:
  Progress graphs:            314.4s
  Writing output:             0.0s
  Pausing:                    0.0s
  Updating status:            144.2s
```

Figure 2.7 Report for weight 662


Current optimization statistics.

Auto Hide

Matrix:		Presolved:	
Rows(constraints):	41	Rows(constraints):	6
Columns(variables):	839	Columns(variables):	811
Nonzero elements:	2750	Nonzero elements:	842
Global entities:	810	Global entities:	810
Sets:	0	Sets:	0
Set members:	0	Set members:	0

Overall status: **Performing global search...**

LP relaxation:		Global search:	
Algorithm:	Simplex dual	Current node:	6782999
Simplex iterations:	8	Depth:	43
Objective:	-1018.17	Active nodes:	0
Status:	LP Optimal	Best bound:	-1016.4
Time:	0.0s	Best solution:	-1016.27
		Gap:	0.0132171%
		Status:	5 integer solution(s) fou
		Time:	46026.5s

Time overheads:	
Progress graphs:	2892.6s
Writing output:	0.0s
Pausing:	0.0s
Updating status:	127.6s

Figure 2.8 Report for weight 964

Note that the solver had to evaluate over 5.2 million nodes for over 7 hours (weight 662) and over 6.7 million nodes for over 12 hours (weight 964) until finally the solver halted. The top 15 weights in descending order of solution time were:

Solution	Weight ID	IMP	ENV	PFA	PRO	Obj. F.	Time in seconds
1	964	2044.91	21.74	78.39	1086.46	-1016.27	46,026.50
2	662	2030.18	21.07	77.65	1086.90	-1024.14	25,876.00
3	178	2030.14	21.07	77.65	1086.90	-640.73	7,749.32
4	921	2059.86	21.75	79.01	1085.82	-387.98	6,857.84
5	389	2045.17	21.07	78.27	1086.25	-412.21	4,495.65
6	802	2030.12	21.07	77.65	1086.90	-762.24	2,158.55
7	459	2015.20	21.07	76.84	1087.54	-883.56	2,123.08
8	339	2030.10	21.07	77.65	1086.90	-467.82	291.44
9	952	2029.34	20.63	77.76	1085.76	-203.14	50.76
10	361	2014.23	20.63	76.95	1086.40	-369.64	49.55
11	48	2015.06	21.07	76.84	1087.54	-687.43	39.45
12	55	2015.05	21.07	76.84	1087.54	-657.70	17.87
13	592	1054.16	267.89	52.75	766.25	1025.22	9.78
14	150	1128.85	94.83	52.75	762.40	909.84	6.28
15	179	1428.18	19.30	63.90	948.52	-24.68	5.28

Table 2.25 List of top 15 weights in descending order of solution time

From the other 998 weights we observed three cases where it took about two hours to find the solution and other two cases where the solution was found over thirty minutes.

We will now proceeded to compute the solution of these difficult cases (those taking more then 30 minutes to solve) by using relaxation and decomposition methods, and then we will compare our results with the obtained above.

The data from the "difficult to solve" weights is presented in Table 2.26

Weight ID		Branch and Bound	Lagrangian Relaxation	Dantzig-Wolfe	Best Method
964	Best solution	-1016.27	-1016.27	-1016.27	
	Effort	12 hours, 8.78 Million nodes	Few seconds One iteration	One minute Two iterations	LR
662	Best solution	-1024.14	-1023.88	- 1024.14	
	Effort	7 hours 5.2 Million nodes	Few seconds One iteration	One minute Two iterations	DW
178	Best solution	-640.73	-640.727	- 640.727	
	Effort	2.15 hours	Few seconds One iteration	One minute Two iterations	LR
921	Best solution	-387.98	-387.977	- 387.977	
	Effort	1.90 hours	Few seconds One iteration	One minute Two iterations	LR
389	Best solution	-412.21	-412.208	- 412.208	
	Effort	1.25 hours	Few seconds One iteration	One minute Two iterations	LR
802	Best solution	-762.24	-762.239	- 762.239	
	Effort	0.60 hours	Few seconds One iteration	One minute Two iterations	LR
459	Best solution	-883.56	-883.565	- 883.565	
	Effort	0.60 hours	Few seconds One iteration	One minute Two iterations	LR

Table 2.26 Results from the different methods for different weights original bounds

Clearly the Lagrangian relaxation and the Dantzig-Wolfe decomposition methods are much faster than the traditional branch and bound when solving these complicated cases. Although they require more effort because the computation of the multiple subproblems, they do not need to solve a relaxation on each node as the branch and bound method does. The downside is that neither the Lagrangian relaxation method nor the Dantzig-Wolfe decomposition will always achieve the optimal solution. It is possible that the solution obtained by the Lagrangian relaxation is no better than the solution obtained by relaxing the integer requirement of the decision variables (integer relaxation) of the problem (Wolsey, 1998).

This particular case was relatively easy to solve because the complicating constraints are easily met, the requirements for area in PFA was to be above two and below 1000. Therefore, the use of a simpleton Lagrangian vector (0, 0) was all that we needed to reach a solution. Perhaps a narrower range could shed more light on the efficiency of these methods. That is the main goal of section 2.4.2.1 Evaluation of New Cases: Tighter PFA Bounds.

A good strategy to find the solution in general seems to be as follows:

1. Set up a maximum execution time on the branch and bound procedure of about two minutes.

2. Solve all weights keeping track of those instances where the solution was not met due to time constraints.

3. Solve the cases identified in step 2 using Lagrangian relaxation. If the Lagrangian bound is too far away from the best solution found by the branch and bound method, then proceed to apply the Dantzig-Wolfe decomposition technique.

As a test for this strategy we proceeded to set a maximum time of two minutes for evaluating the weights using the branch and bound procedure with the same PFA bounds used in the work by Moglen, Gabriel and Faria (2003) obtaining seven results over 120 seconds followed by the next result obtained in almost 52 seconds.

Solution	Weight ID	IMP	ENV	PFA	PRO	Obj. F.	Time
1	964	2044.84	21.75	78.39	1086.46	-1016.27	133.21
2	662	2030.15	21.07	77.65	1086.90	-1024.14	133.18
3	921	2059.88	21.75	79.01	1085.82	-387.98	131.68
4	459	2015.20	21.07	76.84	1087.54	-883.56	131.57
5	178	2030.14	21.07	77.65	1086.90	-640.73	130.80
6	389	2045.20	21.07	78.27	1086.25	-412.21	129.82
7	802	2030.23	21.07	77.65	1086.90	-762.24	102.84
8	361	2014.23	20.63	76.95	1086.40	-369.64	51.53

Table 2.27 Results over 50 seconds with original bounds and time limit 120

Note that Table 2.27 presents some values over the maximum allotted time, we believe that the reason is that the solver reports the total time including some overhead to save the data.

We can expect that the top nine cases were terminated by exceeding the limited time allowed while the other 991 cases were solved to optimality. The objective functions of the top nine cases were exactly the same as the values presented in Table 2.25. Given the accuracy of the procedure one could think about reducing the time limit since the worst case solved that did not exceed the time limit was almost 56 seconds. We decided to run the 1000 weights again with a one minute time limit obtaining the results presented in Table 2.28.

Solution	Weight ID	IMP	ENV	PFA	PRO	Obj. F.	Time
1	535	2015.12	21.07	76.84	1087.54	-893.68	67.89
2	389	2045.17	21.07	78.27	1086.25	-412.21	67.22
3	339	2030.13	21.07	77.65	1086.85	-467.80	66.71
4	662	2030.18	21.07	77.65	1086.90	-1024.14	66.48
5	921	2059.89	21.75	79.01	1085.82	-387.98	65.96
6	802	2045.19	21.07	78.27	1086.25	-762.16	65.70
7	459	2015.11	21.07	76.84	1087.54	-883.56	64.64
8	964	2044.91	21.75	78.39	1086.46	-1016.27	64.23
9	178	2030.14	21.07	77.65	1086.90	-640.73	61.02
10	361	2014.23	20.63	76.95	1086.40	-369.64	47.87

Table 2.28 Results over 40 seconds with original bounds and time limit 60

Table 2.29 shows the objective functions and their respective times for different weights obtained with the original bounds by setting different values of maximum time to compute (no maximum, 120 seconds and 60 seconds). These results indicate that the branch and bound method is able to find the solution within a minute.

		Max time 60 seconds		Max time 120 seconds		No max time	
Solution	Weight ID	Obj. F.	Time	Obj. F.	Time	Obj. F.	Time
	178	-640.73	61.02	-640.73	130.80	-640.73	7749.32
	339	-467.80	66.71	-467.80	89.14	-467.82	91.44
	361	-369.64	47.87	-369.64	51.53	-369.64	55.93
	389	-412.21	67.22	-412.21	129.82	-412.21	4495.65
	459	-883.56	64.64	-883.56	131.57	-883.56	2123.08
	662	-1024.14	66.48	-1024.14	133.18		
	802	-762.16	65.70	-762.24	102.84	-762.24	2158.55
	921	-387.98	65.96	-387.98	131.68	-387.98	6857.84
	964	-1016.27	64.23	-1016.27	133.21		

Table 2.29 Summary of objective functions and solution times original bounds

Since the objective value obtained after a minute is the same as the one obtained after two minutes (all cases except 802) then it seems safe to use one minute as time limit instead of two minutes. If we consider that 98.80% of the 1000 cases were solved in less than ten seconds then it even makes sense to push down the time limit and analyze the results. Appending the data for such a test to Table 2.29 we obtain the data presented in Table 2.30.

	Max time 60 seconds		Max time 120 seconds		No max time		Max time 10 seconds	
Weight ID	Obj. F.	Time	Obj. F.	Time	Obj. F.	Time	Obj. F.	Time
178	-640.73	61.02	-640.73	130.80	-640.73	7749.32	-640.73	11.086
339	-467.80	66.71	-467.80	89.14	-467.82	291.44	-467.82	10.785
361	-369.64	47.87	-369.64	51.53	-369.64	55.93	-369.64	11.106
389	-412.21	67.22	-412.21	129.82	-412.21	4495.65	-412.21	10.525
459	-883.56	64.64	-883.56	131.57	-883.56	2123.08	-883.56	10.875
662	-1024.14	66.48	-1024.14	133.18			-1023.66	11.106
802	-762.16	65.70	-762.24	102.84	-762.24	2158.55	-761.66	10.946
921	-387.98	65.96	-387.98	131.68	-387.98	6857.84	-387.98	11.086
964	-1016.27	64.23	-1016.27	133.21			-1016.27	10.846

Table 2.30 Summary of results including the maximum time of 10 seconds on last two columns

We see that there are only one case (lightly shaded) where the objective function is different from the previous calculations. This makes us believe that within the first ten seconds the branch and bound technique is able to very accurately find the optimal solution. The drawback with ten seconds time is that there will be 11 solutions solved in ten seconds or more so to verify the accuracy of the solutions we would need to apply the Lagrangian relaxation those cases. Therefore we think that using one minute as time limit provides a good tradeoff between the number of cases that needs to be checked and the total tome to solve the problems.

Consider that to reach the results presented in Table 2.25 we spent almost 27 hours of computing time over a period of about four days. The 1000 runs with a two minute limit took only half of an hour and the 1000 runs with one minute limit took twenty minutes. Since the evaluation of the Lagrangian relaxation took about one minute then the savings in time are considerable.

The distribution of the objective function values for the four stakeholders has been presented in Figure 2.9 - Figure 2.12. These figures could be used by the decision makers to visualize the ranges in which the solutions tend to be located. For example the Hydrologist could argue that there are about 140 solutions with low level of imperviousness, so those should be evaluated first. The problem is that the solutions are not related to other objectives as they are in a Value Path graph, but due to the number of Pareto points the Value Paths do not add much information (see Figure A.59 in the appendix).

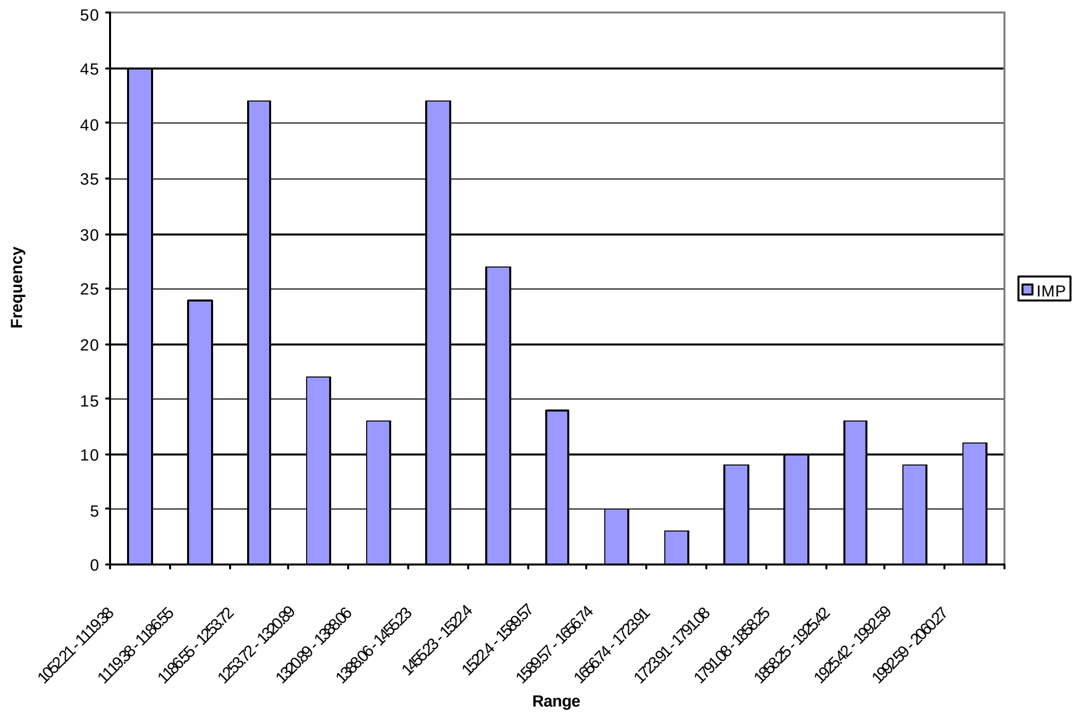


Figure 2.9 Distribution of IMP solutions original bounds

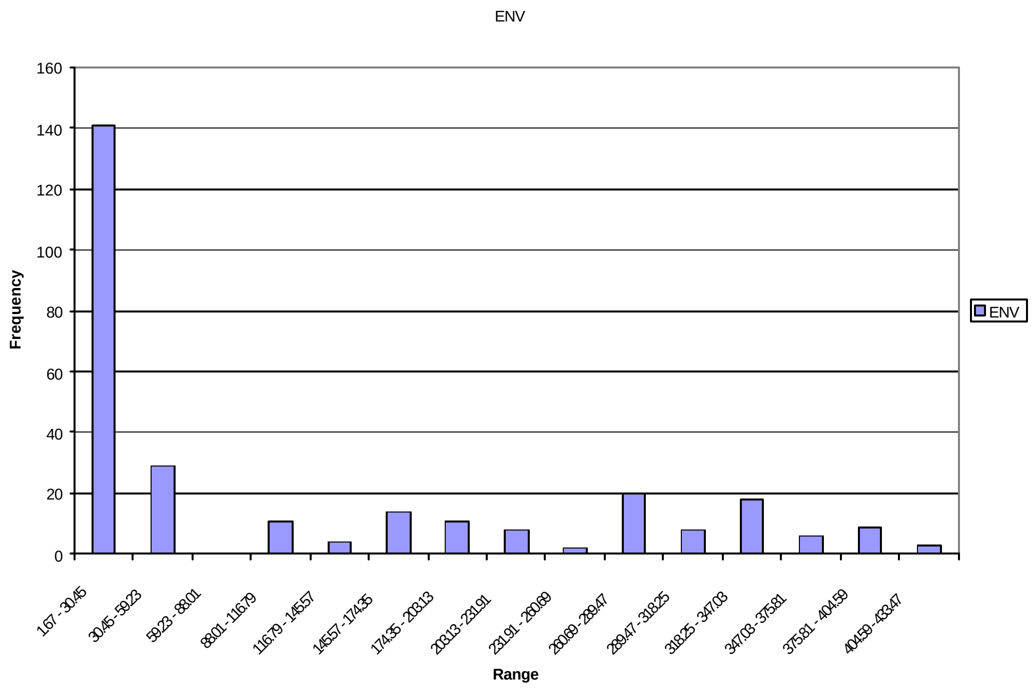


Figure 2.10 Distribution of ENV solutions original bounds

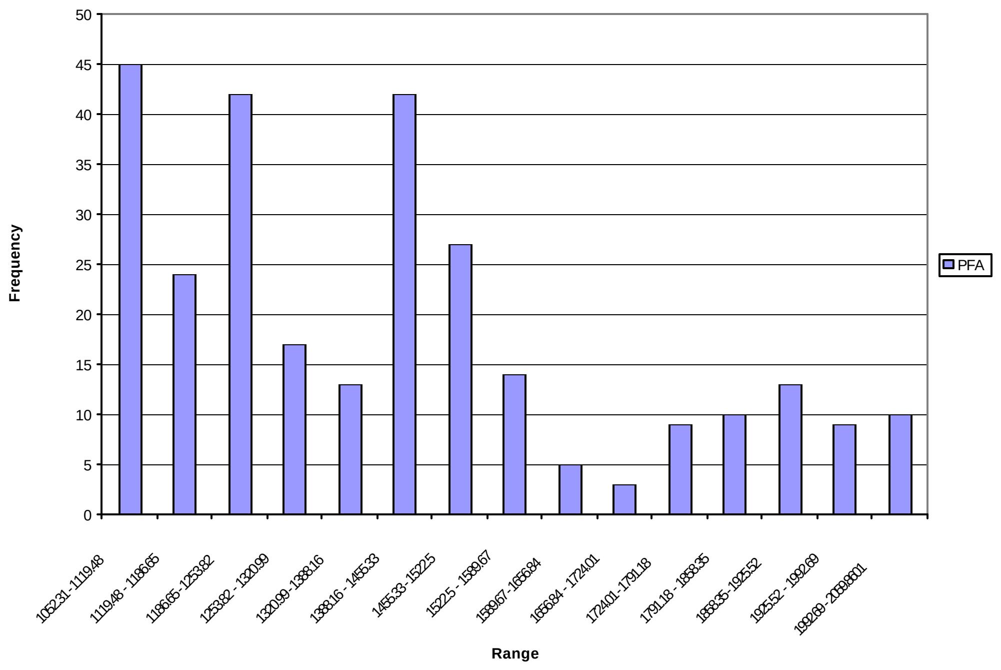


Figure 2.11 Distribution of PFA solutions original bounds

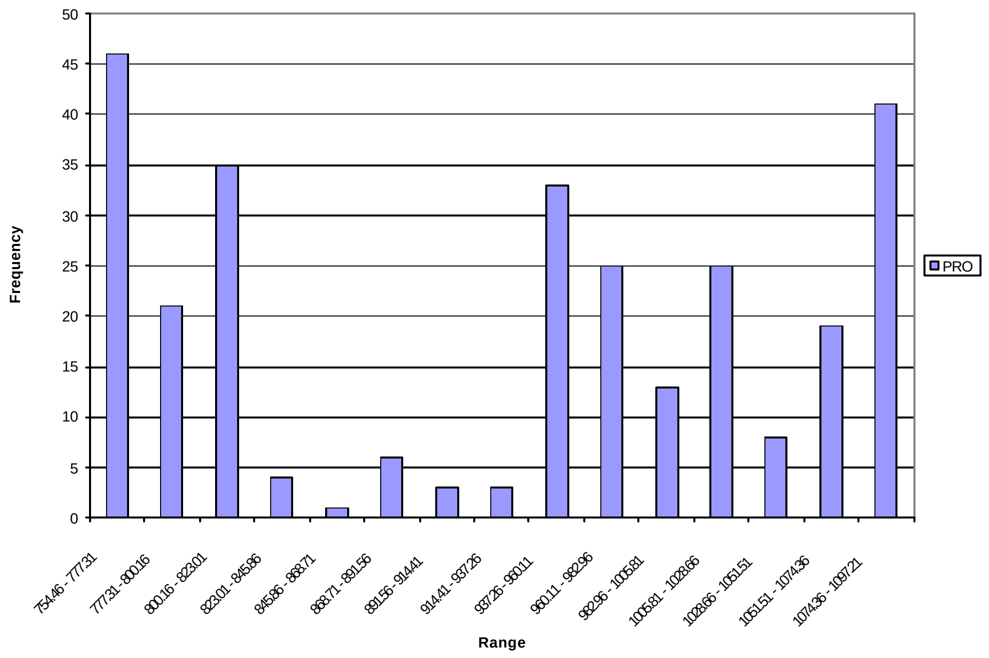


Figure 2.12 Distribution of PRO solutions original bounds

2.4.2.1. Evaluation of New Cases: Tighter PFA Bounds

Having the somewhat wide range defined by the previous bounds $\underline{PFA} = 2$, $\overline{PFA} = 1000$, we now seek to find new solutions within a much tighter range for PFA.

The ranges of PFA values obtained from the evaluation of the 1000 weights with the original bounds were divided in 15 intervals, the results are presented in Figure 2.13.

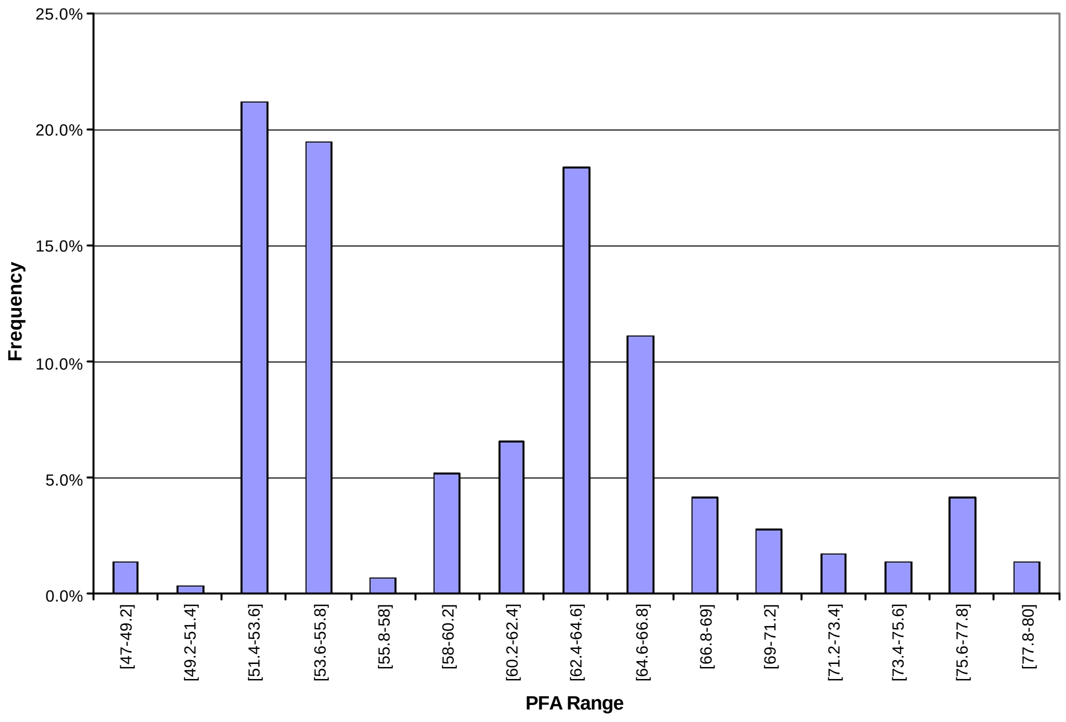


Figure 2.13 Ranges of PFA obtained with original bounds 1000 cases

The minimum value of PFA obtained was 47.52 and the maximum was 79.01, with those numbers at hand, we selected two ranges for the bounds, one in which the lower bound would be restrictive, and another one where both bounds would be restrictive. The selected ranges are presented in Table 2.31.

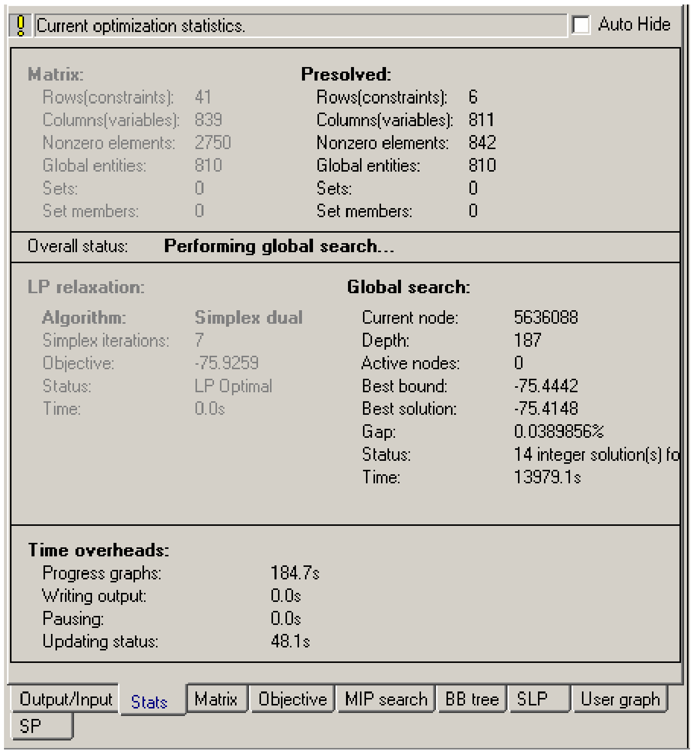
Case	$\underline{PFA}$	$\overline{PFA}$	Number of solutions obtained
1	60	1000	150
2	60	70	125

Table 2.31 Cases evaluated in addition to the original bounds

The last column represents the number of solutions obtained within the bounds using the original weights, we expect to obtain feasible solutions for the problem with the tighter bounds.

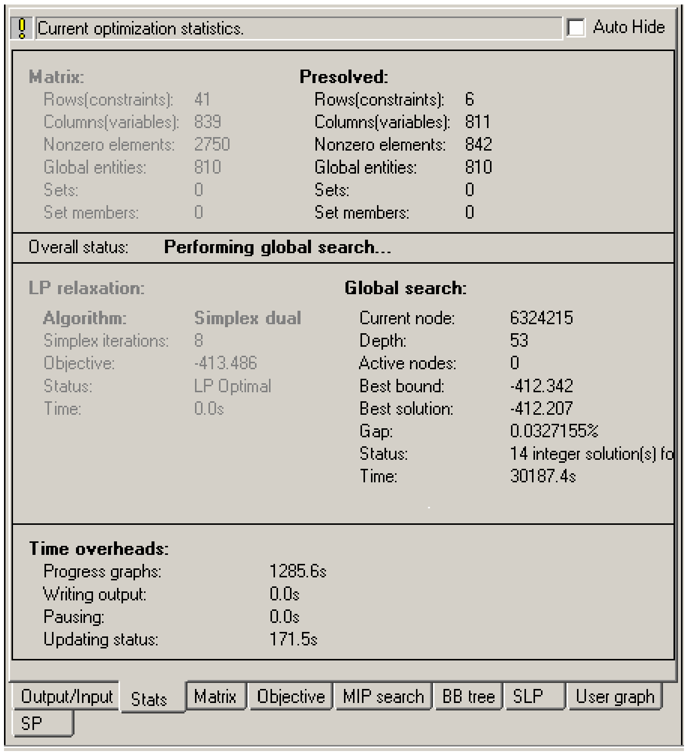
2.4.2.1.1. Case 1

We ran the same 1000 weights again obtaining two cases (weights 176, and 389) where the computer ran out of memory before producing the optimal solution(see Figure 2.14 and Figure 2.15). There is one case (weight 178) that took over two hours to solve. The following one took over half an hour (weight 459).



```
Current optimization statistics.                          Auto Hide

Matrix:                               Presolved:
  Rows(constraints):    41              Rows(constraints):    6
  Columns(variables):   839             Columns(variables):   811
  Nonzero elements:     2750            Nonzero elements:     842
  Global entities:      810             Global entities:      810
  Sets:                 0               Sets:                 0
  Set members:          0               Set members:          0

Overall status:   Performing global search...

LP relaxation:                           Global search:
  Algorithm:            Simplex dual       Current node:     5636088
  Simplex iterations:   7                  Depth:            187
  Objective:            -75.9259           Active nodes:     0
  Status:               LP Optimal         Best bound:       -75.4442
  Time:                 0.0s               Best solution:    -75.4148
                                           Gap:              0.0389856%
                                           Status:           14 integer solution(s) fo
                                           Time:             13979.1s

Time overheads:
  Progress graphs:          184.7s
  Writing output:           0.0s
  Pausing:                  0.0s
  Updating status:          48.1s
```

Figure 2.14 Result of branch and bound case 1 weight 176



```
Matrix:                                   Presolved:
  Rows(constraints):     41                 Rows(constraints):    6
  Columns(variables):    839                Columns(variables):   811
  Nonzero elements:      2750               Nonzero elements:     842
  Global entities:       810                Global entities:      810
  Sets:                  0                  Sets:                 0
  Set members:           0                  Set members:          0

Overall status:   Performing global search...

LP relaxation:                              Global search:
  Algorithm:          Simplex dual            Current node:     6324215
  Simplex iterations: 8                       Depth:            53
  Objective:          -413.486                Active nodes:     0
  Status:             LP Optimal              Best bound:       -412.342
  Time:               0.0s                    Best solution:    -412.207
                                              Gap:              0.0327155%
                                              Status:           14 integer solution(s) fo
                                              Time:             30187.4s

Time overheads:
  Progress graphs:        1285.6s
  Writing output:         0.0s
  Pausing:                0.0s
  Updating status:        171.5s
```

Figure 2.15 Result of branch and bound case 1 weight 389

For the other weights we were able to obtain a solution as presented in Table 2.32.

Weight ID	IMP	ENV	PRO	PFA	Obj	Time in seconds
389	2045.20	21.07	1086.25	78.27	-412.20	30187.40
176	2004.05	1.95	1029.29	59.70	-75.86	13979.10
178	2030.14	21.07	1086.90	77.65	-640.73	7465.24
459	2015.20	21.07	1087.54	76.84	-883.56	2256.82
361	2014.24	20.63	1086.40	76.95	-369.64	149.56
339	2030.11	21.07	1086.90	77.65	-467.82	32.06
39	2015.08	21.07	1087.54	76.84	-948.09	27.07
48	2015.05	21.07	1087.54	76.84	-687.43	9.05
55	2015.06	21.07	1087.54	76.84	-657.70	8.64
582	1189.60	25.09	761.45	62.26	624.96	6.75
150	1143.54	94.83	762.09	60.85	919.90	5.76
166	1189.69	25.09	761.56	62.62	1042.15	5.52

Table 2.32 List of top 10 weights in descending order of solution time case 1

Here there are only two cases where the solution time took more than thirty minutes. They are evaluated below along with the two cases for which we couldn't find a solution.

Some of the weight vectors required effort to achieve the optimal solution, we evaluated them to compared the performance of the solution methods. The solutions are presented in the appendix, the results of the tests are summarized in Table 2.33.

Weight ID		**Branch and Bound**	**Lagrangian Relaxation**	**Dantzig-Wolfe**	**Best Method**
176	Best solution	-75.41	-69.15	-69.15	B&B
	Effort	Stopped after almost 2 hrs	9 iterations 5 minutes	4 iterations 6 minutes	
389	Best solution	-412.20	-412.21	-412.208	LR
	Effort	Stopped after 8 hrs	1 iteration	3 iterations	
178	Best solution	-640.73	-640.73	-640.73	LR
	Effort	Obtained after 2 hours	1 iteration	3 iterations	
459	Best solution	-883.56	-883.56	-883.56	LR
	Effort	Obtained after half an hour	1 iteration	3 iterations	

Table 2.33 Results from the different methods for different weights case 1

Once again it seems like the Lagrangian relaxation would be a preferred method over Dantzig-Wolfe. We tried again reducing the time to compute down to one minute obtaining the results presented in Table 2.34.

Weight	PFA	IMP	ENV	PRO	Obj	Time in seconds
389	78.27	2045.20	21.07	1086.25	-412.21	69.52
176	60.01	2092.69	2.05	1026.86	-75.41	67.61
361	76.95	2014.24	20.63	1086.40	-369.64	67.29
662	77.65	2030.15	21.07	1086.90	-1024.14	66.80
964	78.39	2044.84	21.75	1086.46	-1016.27	66.39
952	77.76	2029.33	20.63	1085.76	-203.14	66.18
802	78.27	2045.25	21.07	1086.25	-762.16	66.02
178	77.65	2030.14	21.07	1086.90	-640.73	65.52
921	79.01	2059.88	21.75	1085.82	-387.98	62.20
459	76.84	2015.20	21.07	1087.54	-883.56	61.62
724	60.03	1492.83	2.05	750.93	44.00	61.32
339	77.65	2030.11	21.07	1086.90	-467.82	32.19

Table 2.34 Results for case 1 with one minute time limit

We found eleven cases where the time to compute the optimal solution took more than sixty seconds. Those cases then would need to be evaluated using Lagrangian Relaxation. Again all 1000 cases where solved in about twenty two minutes which is much better than the fifteen hours that took to solve the same cases with the same bounds but without time limits.

2.4.2.1.2. Case 2

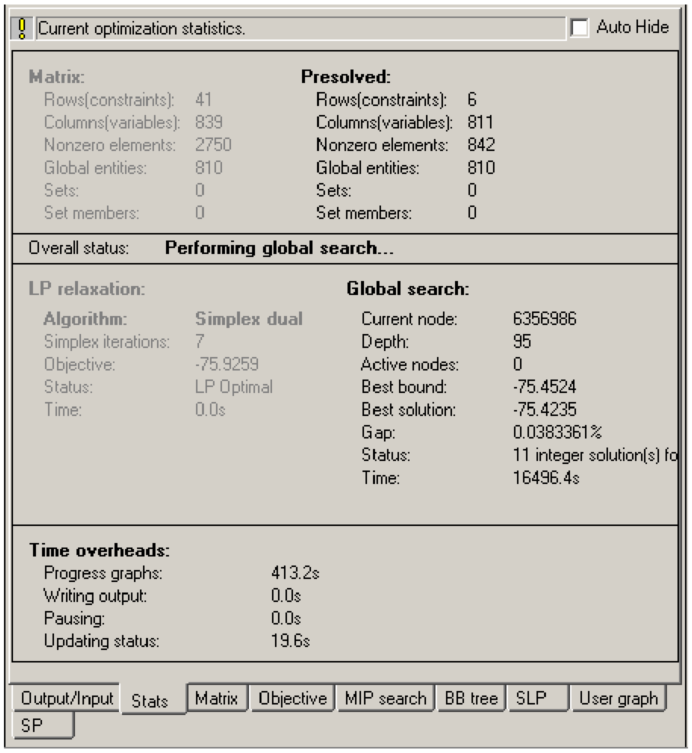
We ran the same 1000 weights again obtaining two solutions (weights 176 and 643) where the computer ran out of memory before producing the optimal solution (See Figure 2.16 and Figure 2.17). Besides those two cases, the branch and bound procedure provided the results in less than four minutes on the worst case. We expected this case with tighter bounds on the PFA requirements to be faster because the reduction of the bounds implies a reduction on the feasible region, making the trees smaller in size.

Only one other weight (724) took a very long time to complete (over 3hrs). The top ten results in descending computation time are summarized in Table 2.35

Weight	PFA	IMP	ENV	PRO	Obj	Time in seconds
176					-75.4235	16496.4
643					-110.172	12817.1
724	60.0263	1492.67	2.04793	750.893	43.9923	12530.5
964	69.9165	1943.18	21.0726	1090.46	-1013.24	208.1
535	69.9165	1943.1	21.0726	1090.46	-890.983	185.2
459	69.9165	1943.19	21.0726	1090.46	-882.556	92.5
578	60.0067	2077.71	2.53492	1030.1	-133.97	11.3
430	69.9165	1943.07	21.0726	1090.46	-634.369	8.3
952	69.9798	1942.15	20.5856	1089.2	-202.456	7.8
955	62.6185	1872.85	31.8475	1091.11	-986.22	7.4

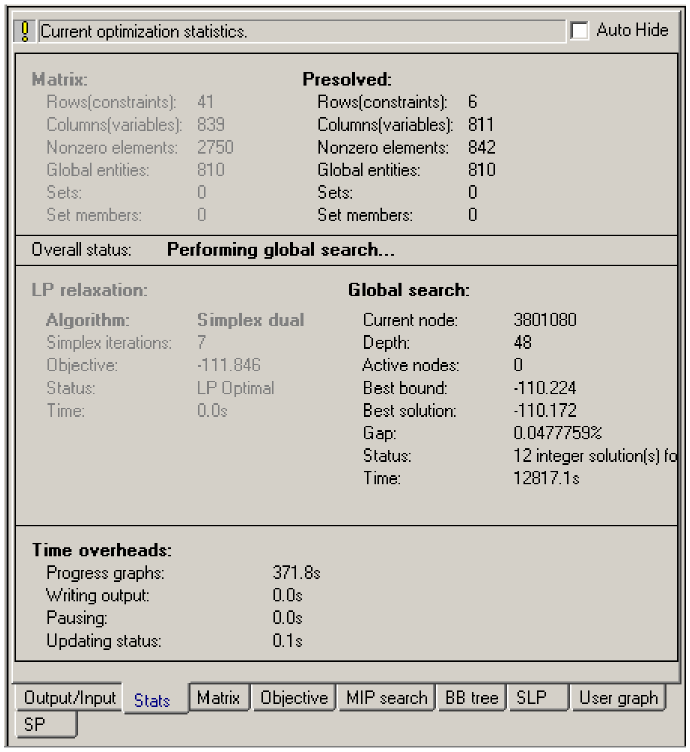
Table 2.35 Top 10 results for case 2 listed in descending order of solution time

We will next analyze the one case that took long time, along with the two cases (shown below) that were not solved.



```
Current optimization statistics.                    Auto Hide

Matrix:                                Presolved:
  Rows(constraints):     41              Rows(constraints):     6
  Columns(variables):    839             Columns(variables):    811
  Nonzero elements:      2750            Nonzero elements:      842
  Global entities:       810             Global entities:       810
  Sets:                  0               Sets:                  0
  Set members:           0               Set members:           0

Overall status:     Performing global search...

LP relaxation:                          Global search:
  Algorithm:             Simplex dual      Current node:      6356986
  Simplex iterations:    7                 Depth:             95
  Objective:             -75.9259          Active nodes:      0
  Status:                LP Optimal        Best bound:        -75.4524
  Time:                  0.0s              Best solution:     -75.4235
                                           Gap:               0.0383361%
                                           Status:            11 integer solution(s) fo
                                           Time:              16496.4s

Time overheads:
  Progress graphs:           413.2s
  Writing output:            0.0s
  Pausing:                   0.0s
  Updating status:           19.6s
```

Figure 2.16 Result for weight 176



```
Current optimization statistics.                         Auto Hide

Matrix:                                  Presolved:
  Rows(constraints):    41                 Rows(constraints):   6
  Columns(variables):   839                Columns(variables):  811
  Nonzero elements:     2750               Nonzero elements:    842
  Global entities:      810                Global entities:     810
  Sets:                 0                  Sets:                0
  Set members:          0                  Set members:         0

Overall status:    Performing global search...

LP relaxation:                              Global search:
  Algorithm:           Simplex dual           Current node:    3801080
  Simplex iterations:  7                      Depth:           48
  Objective:           -111.846               Active nodes:    0
  Status:              LP Optimal             Best bound:      -110.224
  Time:                0.0s                   Best solution:   -110.172
                                              Gap:             0.0477759%
                                              Status:          12 integer solution(s) fo
                                              Time:            12817.1s

Time overheads:
  Progress graphs:          371.8s
  Writing output:           0.0s
  Pausing:                  0.0s
  Updating status:          0.1s
```

Figure 2.17 Result for weight 643

We proceed to apply the decomposition techniques finding that some of the weights required a great computational effort to achieve the optimal solution, we compared the performance of the solution methods and the results have been summarized in Table 2.36. Dantzig-Wolfe outperformed all others in this case.

Weight ID		Branch and Bound	Lagrangian Relaxation	Dantzig-Wolfe	Best Method
176	Best solution	-75.4235	Best bound -75.86	-75.754	
	Effort	over 6 million nodes over four and a half hours	No feasible solution found 9 iterations	Four iterations	DW
643	Best solution	-110.172	Best Bound -112.327	-111.52	
	Effort	over 3.8 million nodes almost 4 hours	No feasible solution found 11 iterations	Four iterations	DW
724	Best solution	43.9923	Best bound 49.66	47.0496	
	Effort	Over three and a half hours	No feasible solution found 11 iterations	Several iterations and B&B	Unclear

Table 2.36 Comparison of methods case 2

2.5. Algorithm Implementation

The solution algorithm that we proposed to solve the integer programming land development model is presented below in Figure 2.18.

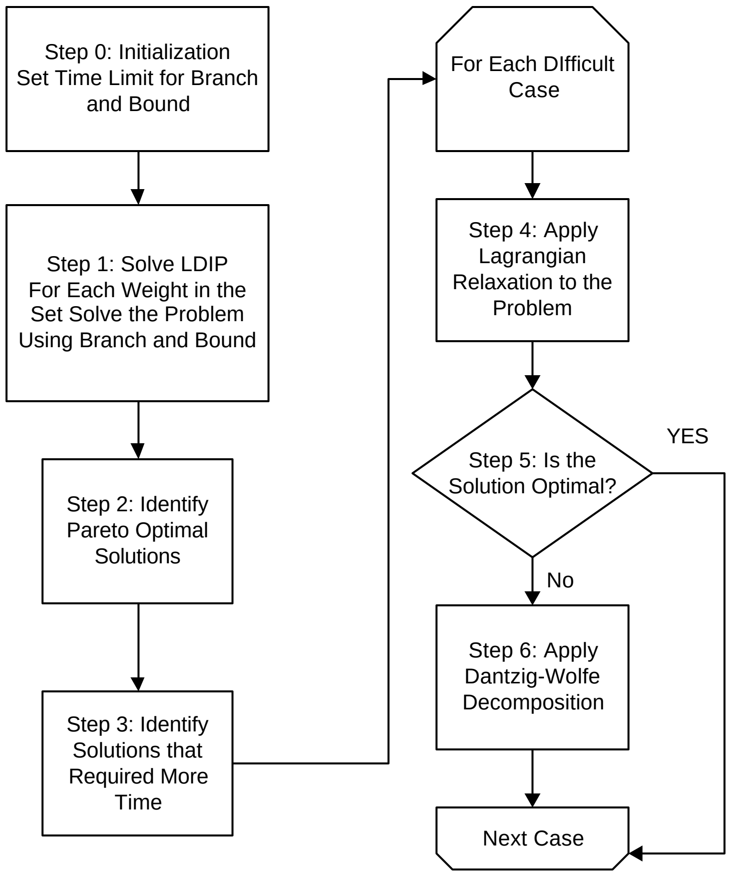

Step 0: Initialization
Set Time Limit for Branch and Bound
Step 1: Solve LDIP
For Each Weight in the Set Solve the Problem Using Branch and Bound
Step 2: Identify Pareto Optimal Solutions
Step 3: Identify Solutions that Required More Time
For Each DIfficult Case
Step 4: Apply Lagrangian Relaxation to the Problem
Step 5: Is the Solution Optimal?
YES
No
Step 6: Apply Dantzig-Wolfe Decomposition
Next Case

Figure 2.18 Algorithm to solve the integer programming version of the land development problem using Lagrangian relaxation and Dantzig-Wolfe decomposition

This algorithm begins by setting a time limit on the branch and bound search, we have found by experimentation that for our problem a one minute limit works quite well since the total time to evaluate 1000 weights was less than an hour. After all weights have been solved using branch and bound the algorithm checks all those cases in which the algorithm stopped the branch and bound search due to time limitations. For those cases we then solve a Lagrangian relaxation, if the result of the

Lagrangian relaxation is the same as the one obtained by branch and bound then we consider the solution to be optimal, if it was not then we proceed to apply Dantzig-Wolfe decomposition.

2.5.1. Original Bounds

To test our proposed algorithm we ran the 1000 cases with the case 2 bound using two minutes max time, and obtained in about 25 minutes the results presented in the appendix.

These results are consistent with what we have found without time limit. We identified that from the 1000 runs, seven of them exceeded the two minutes limit, those solutions are listed in Table 2.37. The bounds obtained after two minutes are extremely close to those found after hours of computation. This method would reduce the total time to compute the all the solutions, since apparently only seven cases would need to be evaluated further.

Solution	Weight ID	IMP	ENV	PFA	PRO	Obj. F.	Time in seconds
1	178	2030.14	21.0726	77.6513	1086.90	-640.727	126.813
2	389	2045.20	21.0726	78.2727	1086.25	-412.207	125.931
3	459	2015.20	21.0726	76.8444	1087.54	-883.564	127.153
4	662	2030.15	21.0726	77.6513	1086.90	-1024.140	133.993
5	802	2030.23	21.0726	77.6513	1086.90	-762.239	128.145
6	921	2059.88	21.7481	79.0115	1085.82	-387.977	134.404
7	964	2044.84	21.7481	78.3901	1086.46	-1016.270	131.198

Table 2.37 Runs exceeding the time limit original bounds

The solutions listed in Table 2.37 are candidates to further revision by the Lagrangian and Dantzig-Wolfe decomposition techniques. However, when we checked for Pareto optimality we found that from the 1000 solutions there are 285 unique are Pareto optimal points. The list of those unique Pareto optimal points can be found in the appendix. Since all weights used are positive, all runs should be Pareto optimal, but since more than one weight can result on the same solution then the set of Pareto optimal points is a subset of all runs. The interesting finding is that from the list of Pareto optimal points only three points exceeded the solution time. This means that some of the solutions found by the branch and bound procedure are being mapped to another weight combination that took less time and solved to optimality, so there is no need to evaluate them again.

The solutions in the Pareto optimal set that need to be evaluated are:

Weight ID	IMP	ENV	PFA	PRO	Obj. F.	Seconds
389	2045.2	21.0726	78.2727	1086.25	-412.207	125.931
921	2059.88	21.7481	79.0115	1085.82	-387.977	134.404
964	2044.84	21.7481	78.3901	1086.46	-1016.27	131.198

Table 2.38 List of Pareto optimal points exceeding maximum time original bounds

2.5.2. Tightened Bounds Case 1

The 1000 cases were solved using the algorithm with a two minutes maximum solution time. It took 36 minutes to find all solutions. The list of the solutions obtained is presented in the appendix. We found that 11 of the solutions exceeded the allotted time to solve. Also, from the set of 1000 runs only 250 points were Pareto optimal. It was expected to find a smaller number of points as compared to the original bounds since the feasible region have been reduced. From those 250 Pareto optimal points, the 7cases are presented in Table 2.39 need to be evaluated further.

Solution	Weight ID	IMP	ENV	PFA	PRO	Obj. F	Seconds
1	176	2,092.69	2.05	60.01	1,026.86	-75.41	133.3020
2	361	2,014.24	20.63	76.95	1,086.40	-369.64	133.9530
3	389	2,045.20	21.07	78.27	1,086.25	-412.21	135.8250
4	724	1,492.83	2.05	60.03	750.93	44.00	132.8610
5	921	2,059.88	21.75	79.01	1,085.82	-387.98	134.3630
6	952	2,029.33	20.63	77.76	1,085.76	-203.14	123.2270
7	964	2,044.84	21.75	78.39	1,086.46	-1,016.27	123.1970

Table 2.39 List of Pareto optimal points that exceeded maximum time tightened bounds case 1

2.5.3. Tightened Bounds Case 2

The 1000 cases were solved using the algorithm with a two minutes maximum solution time. It took 20 minutes to find all solutions. The list of the solutions obtained is presented in the appendix. We found that 5 of the solutions exceeded the allotted time to solve. Also, from the set of 1000 runs only 236 points were Pareto optimal. It was expected to find a smaller number of points as compared to the original bounds since the feasible region have been reduced. From those 236 Pareto optimal points, only two need to be evaluated further. Those cases are presented in Table 2.40.

Solution	Weight ID	IMP	ENV	PFA	PRO	Obj. F.	Seconds
1	176	2,092.66	2.05	60.03	1,026.85	-75.42	128.8250
2	724	1,492.77	2.05	60.03	750.93	44.00	122.2860

Table 2.40 List of Pareto optimal points that exceeded maximum time tightened bounds case 2

2.6. <u>Formulation Using the Constraint Method</u>

Consider again the same settings as for the weighted method. The difference is that now, we are interested in optimizing one of the objectives while we set a bound on the others.

2.6.1. Optimizing the PFA's

We could set up the problem as to maximize the area of PFA developed, while at the same time maintain a total imperviousness change that does not exceed a certain upper bound $\left(\overline{IMP}\right)$, develop environmentally sensitive areas up to certain upper bound $\left(\overline{ENV}\right)$, while making at least a minimum profit $\left(\underline{PRO}\right)$.

Then the mathematical formulation used for this model can now be written as follows:

Max: $\sum_{z=1}^{5} \sum_{n \in PFA} a_{z,n} d_{z,n}$ (2.91)

Subject to:

$$\sum_{z=1}^{5} \sum_{n=1}^{N_z} a_{z,n} \Delta I_{z,n} d_{z,n} \le \overline{IMP} \quad (2.92)$$

$$\sum_{z=1}^{5} \sum_{n \in Sc} a_{z,n} d_{z,n} \le \overline{ENV} \quad (2.93)$$

$$\sum_{z=1}^{5} \sum_{n=1}^{N_z} p_{z,n} d_{z,n} \ge \underline{PRO} \quad (2.94)$$

$$\underline{PFA} \le \sum_{z=1}^{5} \sum_{n \in PFA} a_{z,n} d_{z,n} \le \overline{PFA} \quad (2.95)$$

$$\underline{RLD} \le \sum_{n=1}^{N_1} u_{z,n} d_{z,n} \le \overline{RLD}, z = 1 \quad (2.96)$$

$$\underline{RMD} \le \sum_{n=1}^{N_2} u_{z,n} d_{z,n} \le \overline{RMD}, z = 2 \quad (2.97)$$

$$\underline{RHD} \le \sum_{n=1}^{N_3} u_{z,n} d_{z,n} \le \overline{RHD}, z = 3 \quad (2.98)$$

$$\underline{COM} \le \sum_{n=1}^{N_4} u_{z,n} d_{z,n} \le \overline{COM}, z = 4 \quad (2.99)$$

$$\underline{IND} \le \sum_{n=1}^{N_5} u_{z,n} d_{z,n} \le \overline{IND}, z = 5 \quad (2.100)$$

$$d_{z,n} \in \{0,1\}, n \in \{N_z\}, \forall z \in \{1,2,3,4,5\} \quad (2.101)$$

This formulation would avoid missing Pareto optimal points due to the duality gap as explained before in page 240. The bounds can be determined by optimizing each one of the objectives first as a minimization (to obtain a lower bound) followed by a maximization (to obtain an upper bound). After the bounds are set then one can determine how many runs are desired to run and then divide the range of the bounds in suitable intervals. Note that the number of runs can be determined as $N_{IMP} * N_{ENV} * N_{PRO}$ which can result in a large number of runs. For example if each bound is broken in 10 ranges then there would be 1000 runs.

We proceeded to optimize the four objectives twice as described above (one for minimization and one for maximization) obtaining the Table 2.41.

Objective	PFA	IMP	ENV	PRO
Min	13.2145	1051.55	1.76926	613.33
Max	79.3887	5967.03	678.063	1100.99

Table 2.41 Bounds on the objectives

Having set the bounds for all objectives we could now divide the ranges in eleven intervals and use the intermediate points (ten points) as bounds for the constraints as presented in Table 2.42.

Objective	PFA	IMP	ENV	PRO
Min	13.2145	1051.55	1.76926	613.33
Max	79.3887	5967.03	678.063	1100.99
0	13.21	1051.55	1.77	613.33
1	19.23	1498.41	63.25	657.66
2	25.25	1945.27	124.73	702.00
3	31.26	2392.14	186.21	746.33
4	37.28	2839.00	247.69	790.66
5	43.29	3285.86	309.18	834.99
6	49.31	3732.72	370.66	879.33
7	55.33	4179.58	432.14	923.66
8	61.34	4626.44	493.62	967.99
9	67.36	5073.31	555.10	1012.32
10	73.37	5520.17	616.58	1056.66
11	79.39	5967.03	678.06	1100.99

Table 2.42 Selection of ten bounds for the four objectives

The points start at zero with the lower bound and end at 11 with the upper bound, we would take the ten intermediate points as bounds for the optimization.

Example, the first run would be to maximize the PFA area subject to the Imperviousness not to exceed a value of 1498.41, the environmentally sensitive area not to exceed 63.25, and the profit to be at least 657.66.

2.6.2. *Lagrangian Relaxation*

To solve the above problem using Lagrangian relaxation we would proceed to relax the first three constraints and include in the objective function the positive slack of each constraint multiplied by a Lagrangian multiplier similarly as previously done with the weighting method. The formulation would be as follows:

$$\text{Max: } \sum_{z=1}^{5}\sum_{n\in PFA} a_{z,n}d_{z,n} + m_1\left(\overline{IMP} - \sum_{z=1}^{5}\sum_{n=1}^{N_z} a_{z,n}\Delta I_{z,n}d_{z,n}\right) + m_2\left(\overline{ENV} - \sum_{z=1}^{5}\sum_{n\in Sc} a_{z,n}d_{z,n}\right) +$$

$$m_3\left(\sum_{z=1}^{5}\sum_{n=1}^{N_z} p_{z,n}d_{z,n} - \underline{PRO}\right) \tag{2.102}$$

Subject to:

$$\underline{RLD} \le \sum_{n=1}^{N_1} u_{z,n}d_{z,n} \le \overline{RLD}, z = 1 \tag{2.103}$$

$$\underline{RMD} \le \sum_{n=1}^{N_2} u_{z,n}d_{z,n} \le \overline{RMD}, z = 2 \tag{2.104}$$

$$\underline{RHD} \le \sum_{n=1}^{N_3} u_{z,n}d_{z,n} \le \overline{RHD}, z = 3 \tag{2.105}$$

$$\underline{COM} \le \sum_{n=1}^{N_4} u_{z,n}d_{z,n} \le \overline{COM}, z = 4 \tag{2.106}$$

$$\underline{IND} \le \sum_{n=1}^{N_5} u_{z,n}d_{z,n} \le \overline{IND}, z = 5 \tag{2.107}$$

$$d_{z,n} \in \{0,1\}, n \in \{N_z\}, \forall z \in \{1,2,3,4,5\} \tag{2.108}$$

This formulation is very similar to the formulation presented in the Lagrangian relaxation for the weighting method, the difference is only in the objective function, so we will not proceed to solve it as scope for the present work.

2.6.3. Dantzig-Wolfe Decomposition

The formulation can be decomposed by zones in a similar fashion as before where each one of the subproblems are identical to the ones presented in the weighting method, but the master problem has been modified to accommodate for the new complicating constraints. The formulation is as follows:

$$\max : PFA^{z,t}\, l_{z,t} \tag{2.109}$$

s.t.

$$\sum_{z=1}^{5}\sum_{n=1}^{N_z} a_{z,n}\Delta I_{z,n} d_{z,n}\, l_{z,t} \leq \overline{IMP} \tag{2.110}$$

$$\sum_{z=1}^{5}\sum_{n\in Sc} a_{z,n} d_{z,n}\, l_{z,t} \leq \overline{ENV} \tag{2.111}$$

$$\sum_{z=1}^{5}\sum_{n=1}^{N_z} p_{z,n} d_{z,n}\, l_{z,t} \geq \underline{PRO} \tag{2.112}$$

$$\underline{PFA} \leq \sum_{z=1}^{Z}\sum_{t=1}^{T_z} PFA^{z,t} d_{z,t}\, l_{z,t} \leq \overline{PFA} \tag{2.113}$$

$$\sum_{z=1}^{Z}\sum_{t=1}^{T_k} l_{z,t} = 1 \text{ for } k = 1,2,3,4,5 \tag{2.114}$$

$$l_{z,t} \in \{0,1\} \tag{2.115}$$

We tried to obtain a result comparable with a previous one, so we choose to use case 2 bounds and tried to find the solution obtained for weight set 150 as:

Weight	PFA	IMP	ENV	PRO	Obj. F.	Seconds
150	60.8501	1143.54	94.8272	762.086	919.901	6.028

Table 2.43 Result for weight 150 case 2

We first set maximum value of imperviousness change to 1144, a maximum area of environmentally sensitive area of 95 and a minimum profit of 762. Maximizing for PFA we expect to obtain 60.0067, we obtained 61.2067 with the following values:

Weight	PFA	IMP	ENV	PRO	Obj. F.	Seconds
150	60.8501	1143.54	94.8272	762.086	919.901	6.028
Bounds	61.2067	1143.63	94.8272	762.197		3.765

Table 2.44 Comparison between the result of weight 150 and the bounds using constraint method

Since the bounds seemed too loose, we tighten them to a maximum value of imperviousness change of 1143.55, a maximum area of environmentally sensitive area of 94.83 and a minimum profit of 762. Maximizing for PFA we to obtained 60.85 which is the value we were expecting. The computation time was 7.28 seconds.

2.7. ***Previous Work on Decomposition Heuristics***

Barnhart et al. (1996) presented formulations of integer programming problems involving a large number of variables with an example of the generalized assignment problem and crew assignment problem. They described how the pricing problem in Dantzig-Wolfe decomposition can be equivalently stated as a Lagrangian relaxation of the original integer programming problem.

Huisman et al. (2003) presented two different ways to combine Lagrangian relaxation with column generation. They applied the Dantzig-Wolfe decomposition technique to an integer problem, and solved the LP relaxation. Two approaches were followed. On the first one a Lagrangian relaxation was used to solve the sub-problems and on the second one, the Lagrangian relaxation was used to select the columns to be generated. They presented an example using lot sizing to show the applicability and to compare the computational efficiency of the two concepts.

These approaches are different from the approach taken in this work since we are using the Lagrangian relaxation as a technique to verify the optimality of the solution obtained by the branch and bound technique in the cases where more time is presumably needed and later proceeding to use Dantzig-Wolfe instead of solving the relaxation within Dantzig-Wolfe using Lagrangian relaxation.

2.8. Chapter Conclusions

This chapter presents a mixed integer programming model for land development using a weighted sum of objectives from different stakeholders. An algorithm involving a combination of the traditional branch and bound method, Lagrangian relaxation and Dantzig-Wolfe decomposition is presented and applied finding a large subset of Pareto optimal points in a shorter time compared to branch and bound alone. These techniques have been applied together in the past as presented in section 2.7 but none of those used the same sequence as presented in this work. That is, those heuristics solved one instance of the problem where the Lagrangian relaxation was used to solve the subproblems from the Dantzig-Wolfe decomposition or to generate new columns. In contrast, in this work the branch and bound, Lagrangian relaxation and Dantzig-Wolfe techniques are applied in series to a large number of instances (sets of weighting vectors) to solve the same problem.

The proposed algorithm to find an optimal solution follows three steps. First the problem is solved for all the weights using branch and bound with a time limit of one minute. Then, the Lagrangian relaxation is used as a bound to verify if the solution obtained in the cases where the branch and bound seems to need more time is equal to the bound in which case the solution found is optimal, if not then the Dantzig-Wolfe decomposition is used to verify optimality. The Lagrangian relaxation is used first since numerically it provided a bound faster than the Dantzig-Wolfe technique. Another conclusion from the numerical tests is that the Lagrangian relaxation technique is easier to implement than the Dantzig-Wolfe decomposition.

The 1000 weight vectors tried were solved within a reasonable time frame obtaining a relatively large set of Pareto optimal sets which would not have been possible by the use of branch and bound alone. The Lagrangian relaxation alone was not sufficient for one of the cases evaluated because it was not possible to obtain a feasible solution of the original problem by solving the relaxation. There was also one case where the Dantzig-Wolfe decomposition did not yield an integer solution so we needed to apply branch and bound to solve it. Although it was not the case in this dissertation, it is possible that neither Lagrangian relaxation nor Dantzig-Wolfe would find an optimal solution, so the only viable procedure would be to eliminate the time limit of the branch and bound search. This is true because in certain cases the Lagrangian relaxation is no stronger than the linear programming relaxation (Wolsey, 1998). Also, because the Lagrangian relaxation and the Dantzig-Wolfe decomposition are duals of each other (Geoffrion, 1974; Fisher, 1981) then their optimal solutions are the same. Therefore, it is possible that the optimal solution obtained by the algorithm after applying both Lagrangian relaxation and Dantzig-Wolfe would be no better than the linear programming relaxation (Fisher, 1981).

Chapter 3 Land Development Quadratic Mixed Integer Formulation

Gabriel, Faria and Moglen (2005) extended the work previously done by Moglen, Gabriel and Faria (2003) in several aspects. First Gabriel, Faria and Moglen (2005) allows for a set of parcels with “unassigned zoning” to be used. One of the decision variables of the model is to decide what type of development zone should be used for each unassigned parcels selected for development. Second, new constraints were added to handle preferences given to the parcels with zone category to be developed first, before considering any from the unassigned set. Third, the concept of compactness was treated in this work as the squared distance of a rectangle that enclosed all parcels previously developed and chosen for development by the model.

In this section we again extend that work by presenting a strategy to solve the problem using Lagrangian relaxation and Dantzig-Wolfe decomposition methods but do not present numerical results.

3.1. *Objective Functions*

This formulation considers four stakeholders as described below:

3.1.1. The Government Planner

The Government Planner is mostly concerned with the compact development of the land as to prevent the scattered patterns usually associated with sprawl (see page 112). This stakeholder seeks to minimize the size of a rectangle that surrounds all developed parcels. For computational reasons, the objective function minimized the square of the diagonal rather than the diagonal itself.

Figure 3.19 presents an example of the rectangle that encloses all previously developed parcels and the parcels proposed to be developed by the model.

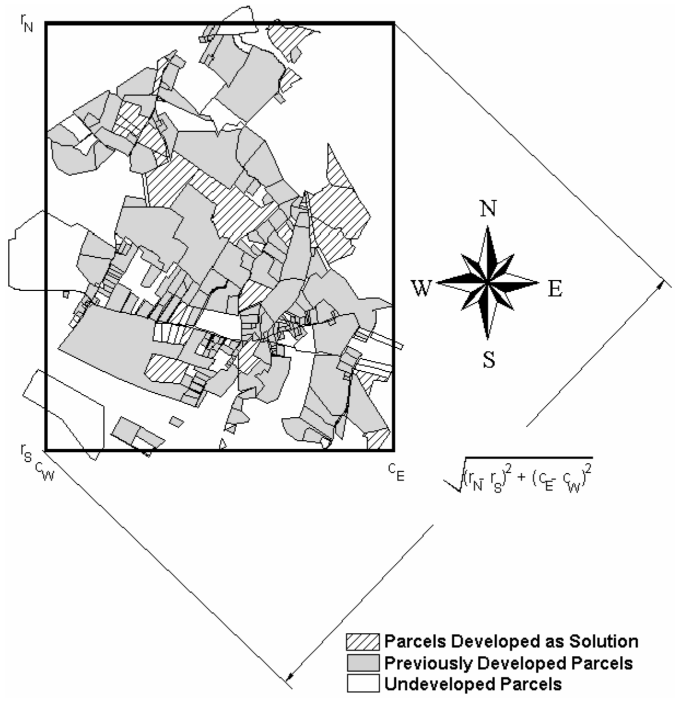

r_N
r_S
c_W
c_E
N
W
E
S
$\sqrt{(r_N - r_S)^2 + (c_E - c_W)^2}$
Parcels Developed as Solution
Previously Developed Parcels
Undeveloped Parcels

Figure 3.19 Depiction of the diagonal and the rectangle that encloses all developed parcels

To measure the rectangle it is first required to find the "largest northing" of the northernmost parcel, the "smallest northing" of the southernmost parcel, the "largest easting" of the easternmost parcel, and the "smallest easting" of the westernmost parcel. Let those coordinates be named r_N, r_S, c_E, and c_W respectively.

Then the length of the diagonal of the rectangle is given by:

$$dist = \sqrt{(r_N - r_S)^2 + (c_E - c_W)^2} \tag{3.1}$$

To simplify this equation but without loss of generality, the objective function was squared to obtain:

$$(r_N - r_S)^2 + (c_E - c_W)^2 \tag{3.2}$$

which represents the Planner's objective to be minimized.

3.1.2. *The Hydrologist*

The objective function used was the same as for the Hydrologist in Chapter 2, so the objective function can be written as:

$$\min \sum_{z=1}^{5} \sum_{n=1}^{N_z} a_{z,n} \Delta I_{z,n} d_{z,n} + \sum_{n \in S_{99}} a_n \Delta I_n d_n \tag{3.3}$$

3.1.3. *The Conservationist*

This stakeholder matches the one presented in Chapter 3, so the objective function can be written as:

$$\min \sum_{z=1}^{5} \sum_{n \in Sc} a_{z,n} d_{z,n} + \sum_{n \in (S_{99} \cap S_C)} a_n d_n \tag{3.4}$$

3.1.4. *The Land Developer*

This stakeholder matches the one presented in Chapter 3, so the objective function can be written as:

$$\max \sum_{z=1}^{5} \sum_{n=1}^{N_z} p_{z,n} d_{z,n} + \sum_{n \in S_{99}} p_n d_n \tag{3.5}$$

3.2. ***<u>Constraints</u>***

Similar to the formulation presented in Chapter 3, this formulation has constraints to accommodate the development to the expected growth of the population, commercial and industrial requirements.

$$\underline{RLD} \leq \sum_{n=1}^{N_1} u_{z,n} d_{z,n} + \sum_{n \in S_{99}} u_{z,n} RLD_{z,n} \leq \overline{RLD},\ z = 1 \tag{3.6}$$

$$\underline{RMD} \le \sum_{n=1}^{N_2} u_{z,n} d_{z,n} + \sum_{n \in S_{99}} u_{z,n} RMD_{z,n} \le \overline{RMD}, z = 2 \tag{3.7}$$

$$\underline{RHD} \le \sum_{n=1}^{N_3} u_{z,n} d_{z,n} + \sum_{n \in S_{99}} u_{z,n} RHD_{z,n} \le \overline{RHD}, z = 3 \tag{3.8}$$

$$\underline{COM} \le \sum_{n=1}^{N_4} u_{z,n} d_{z,n} + \sum_{n \in S_{99}} u_{z,n} COM_{z,n} \le \overline{COM}, z = 4 \tag{3.9}$$

$$\underline{IND} \le \sum_{n=1}^{N_5} u_{z,n} d_{z,n} + \sum_{n \in S_{99}} u_{z,n} IND_{z,n} \le \overline{IND}, z = 5 \tag{3.10}$$

$$d_{z,n} \in \{0,1\}, n \in \{N_z\}, \forall z \in \{1,2,3,4,5\} \tag{3.11}$$

$$RLD_{z,n}, RMD_{z,n}, RHD_{z\ n}, COM_{z,n}, IND_{z,n} \in \{0,1\} \forall z, n$$

where the new decision variables $RLD_{z,n}, RMD_{z,n}, RHD_{z\ n}, COM_{z,n}, IND_{z,n}$ have been included to associate a development type (residential low density, residential medium density, residential high density, commercial, or industrial respectively) to each of the parcels in the set of unassigned parcels labeled as set S_{99}.

The parcels in this set can be developed under only one type of zone:

$$RLD_{z,n} + RMD_{z,n} + RHD_{z\ n} + COM_{z,n} + IND_{z,n} = d_n, \forall z \in \{1,2,3,4,5\}, n \in S_{99} \tag{3.12}$$

$$d_n \in \{0,1\} \forall n \in S_{99} \tag{3.13}$$

Additional constraints are required for the computation of the corner coordinates of the outer rectangle for the planner's objective. These can be written as:

$$r_S - row_S(z,n) \le (1 - d_{z,n}) M \tag{3.14}$$

$$row_N(z,n) - r_N \le (1 - d_{z,n}) M \tag{3.15}$$

$$c_W - col_W(z,n) \le (1 - d_{z,n}) M \tag{3.16}$$

$$col_E(z,n) - c_E \le (1 - d_{z,n})M \tag{3.17}$$

where M is a suitably large positive constant.

Finally there are another two groups of constraints required to give priority to the parcels with zones assigned over those without it. Specifically, the first set of constraints stipulates that the parcels without assigned zone shouldn't be developed under a zone type z if there are enough parcels in that zone to cover the minimum requirements for growth. The constraints for the first group are written as:

$$\sum_{n\in S_{99}} RLD_n u_n \le My_{RLD}, \sum_{z=1}\sum_{n=1}^{N_z} u_{z,n} - \underline{RLD} \le M(1-y_{RLD}) \tag{3.18}$$

$$\sum_{n\in S_{99}} RMD_n u_n \le My_{RMD}, \sum_{z=2}\sum_{n=1}^{N_z} u_{z,n} - \underline{RMD} \le M(1-y_{RMD}) \tag{3.19}$$

$$\sum_{n\in S_{99}} RHD_n u_n \le My_{RHD}, \sum_{z=3}\sum_{n=1}^{N_z} u_{z,n} - \underline{RHD} \le M(1-y_{RHD}) \tag{3.20}$$

$$\sum_{n\in S_{99}} COM_n u_n \le My_{COM}, \sum_{z=4}\sum_{n=1}^{N_z} u_{z,n} - \underline{COM} \le M(1-y_{COM}) \tag{3.21}$$

$$\sum_{n\in S_{99}} IND_n u_n \le My_{IND}, \sum_{z=5}\sum_{n=1}^{N_z} u_{z,n} - \underline{IND} \le M(1-y_{IND}) \tag{3.22}$$

$$y_{RLD}, y_{RMD}, y_{RHD}, y_{COM}, y_{IND} \in \{0,1\} \tag{3.23}$$

Also, where the available number of parcels is not enough, all available parcels should be developed before assigning parcels from the unassigned set. The constraints for the second group can be mathematically written as:

$$\underline{RLD} - \sum_{z=1} \sum_{n=1}^{N_z} u_{z,n} \leq N\left(1 - w_{RLD}\right), \sum_{z=1} \sum_{n=1}^{N_z} u_{z,n} - \sum_{z=1} \sum_{n=1}^{N_z} u_{z,n} d_{z,n} \leq N w_{RLD} \tag{3.24}$$

$$\underline{RMD} - \sum_{z=2} \sum_{n=1}^{N_z} u_{z,n} \leq N\left(1 - w_{RMD}\right), \sum_{z=2} \sum_{n=1}^{N_z} u_{z,n} - \sum_{z=2} \sum_{n=1}^{N_z} u_{z,n} d_{z,n} \leq N w_{RMD} \tag{3.25}$$

$$\underline{RHD} - \sum_{z=3} \sum_{n=1}^{N_z} u_{z,n} \leq N\left(1 - w_{RHD}\right), \sum_{z=3} \sum_{n=1}^{N_z} u_{z,n} - \sum_{z=3} \sum_{n=1}^{N_z} u_{z,n} d_{z,n} \leq N w_{RHD} \tag{3.26}$$

$$\underline{COM} - \sum_{z=4} \sum_{n=1}^{N_z} u_{z,n} \leq N\left(1 - w_{COM}\right), \sum_{z=4} \sum_{n=1}^{N_z} u_{z,n} - \sum_{z=4} \sum_{n=1}^{N_z} u_{z,n} d_{z,n} \leq N w_{COM} \tag{3.27}$$

$$\underline{IND} - \sum_{z=5} \sum_{n=1}^{N_z} u_{z,n} \leq N\left(1 - w_{IND}\right), \sum_{z=5} \sum_{n=1}^{N_z} u_{z,n} - \sum_{z=5} \sum_{n=1}^{N_z} u_{z,n} d_{z,n} \leq N w_{IND} \tag{3.28}$$

$$w_{RLD}, w_{RMD}, w_{RHD}, w_{COM}, w_{IND} \in \{0,1\} \tag{3.29}$$

where N is a suitably large positive constant.

Since the region where the development is taking could be quite large, in order to better utilize the compactness measure presented in (3.2), the area can be broken down in sub areas or quadrants in such a way that the total area of the development is covered by those quadrants. For example consider the case presented in Gabriel, Faria and Moglen (2005) where the Montgomery County area under study is presented in Figure 3.20.

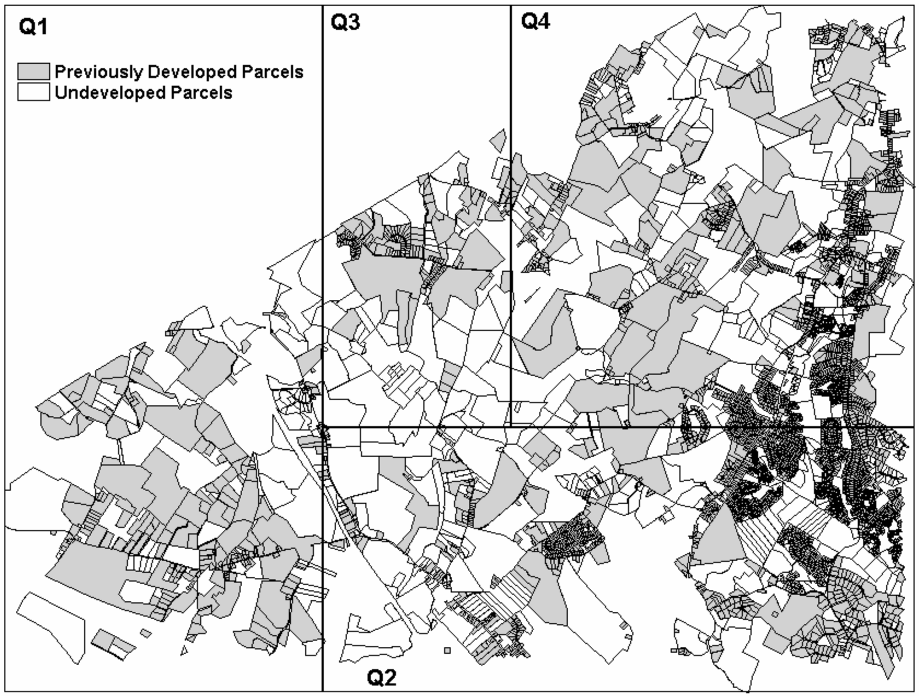


Figure 3.20 Division of Montgomery county study into four quadrants

The parcels that belong to each quadrant are shaded differently as in Figure 3.21.

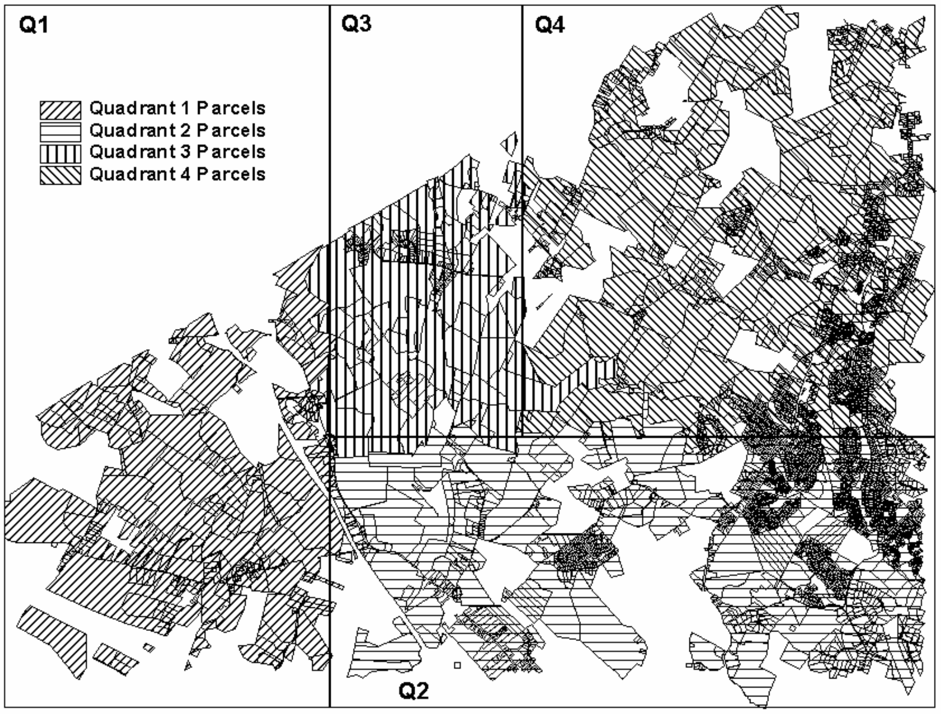


Figure 3.21 Parcels assigned to each quadrant

The objective function of the Planner can be slightly modified to minimize the sum of the four squared diagonals. More specifically, if there are Q quadrants, then the objective function of the Planner becomes:

$$\text{Min: } \sum_{q=1}^{Q}\left(\left(r_{q,N}-r_{q,S}\right)^2+\left(c_{q,E}-c_{q,W}\right)^2\right) \tag{3.30}$$

where $r_{q,N}$ is the northernmost coordinate, $r_{q,S}$ is the southernmost coordinate, $c_{q,E}$ is the easternmost coordinate and $c_{q,W}$ is the westernmost coordinate of the quadrant q.

3.2.1. Formulation Using the Weighting Method

Similar to the formulation presented in Chapter 2, we can use the weighting method to solve the problem as follows

$$\text{Min: } w_1\left(\sum_{q=1}^{Q}\left(\left(r_{q,N}-r_{q,S}\right)^2+\left(c_{q,E}-c_{q,W}\right)^2\right)\right)+\ w_2\left(\sum_{z=1}^{5}\sum_{n=1}^{N_z}a_{z,n}\Delta I_{z,n}d_{z,n}+\sum_{n\in S_{99}}a_n\Delta I_n d_n\right)+$$

$$w_3\left(\sum_{z=1}^{5}\sum_{n\in Sc}a_{z,n}d_{z,n}+\sum_{n\in\left(S_{99}\cap S_C\right)}a_n d_n\right)-\ w_4\left(\sum_{z=1}^{5}\sum_{n=1}^{N_z}p_{z,n}d_{z,n}+\sum_{n\in S_{99}}p_n d_n\right) \tag{3.31}$$

s.t.

(3.6) - (3.29)

3.2.2. Evaluation of the Nine Original Cases

Similarly to the study in Moglen, Gabriel and Faria (2003) this formulation was tested with nine cases as presented in Table 3.1.

Case		Planner (Compactness)	Hydrologist (Imperviousness Change)	Conservationist (Env. Sensitive Area)	Developer (Profit)	Relative Gap
1	Planner Alone	1	0	0	0	5e-005
2	Planner Pareto	1	0.001	0.001	0.001	5e-005
3	Hydrologist Alone	0	1	0	0	5e-005
4	Hydrologist Pareto	0.001	1	0.001	0.001	5e-005
5	Conservationist Alone	0	0	1	0	5e-005
6	Conservationist Pareto	0.001	0.001	1	0.001	5e-005
7	Developer Alone	0	0	0	1	5e-005
8	Developer Pareto	0.001	0.001	0.001	1	5e-004
9	All Perspectives	1	1	1	1	5e-005

Table 3.1 Weights assigned to each stakeholders' objective

In these nine cases each stakeholders' objective was optimized alone (setting the weight of the other stakeholders to zero), also giving a small positive weight to the others objectives (called Pareto) and with all stakeholders having the same weight (All Perspectives case).

Note that one of the cases required a larger relative gap measured as:

$$\frac{|\text{Best Solution - Best Bound}|}{\text{Best Bound}} \tag{3.32}$$

because the computation time required to reach the solution was not acceptable. Other cases took also a long time to solve i.e., the "Conservationist Alone" took a little over six hours to reach an optimal solution.

The tradeoff between stakeholders was presented in a value path graph as in Figure 3.22. The values have been normalized in the range 0-1 where 0 is best and 1 is worst. For example, case 7 (Developer Alone) does poorly in the compactness, imperviousness change and environmental measurements but provides a very high profit (value close to zero are desirable for all objectives).

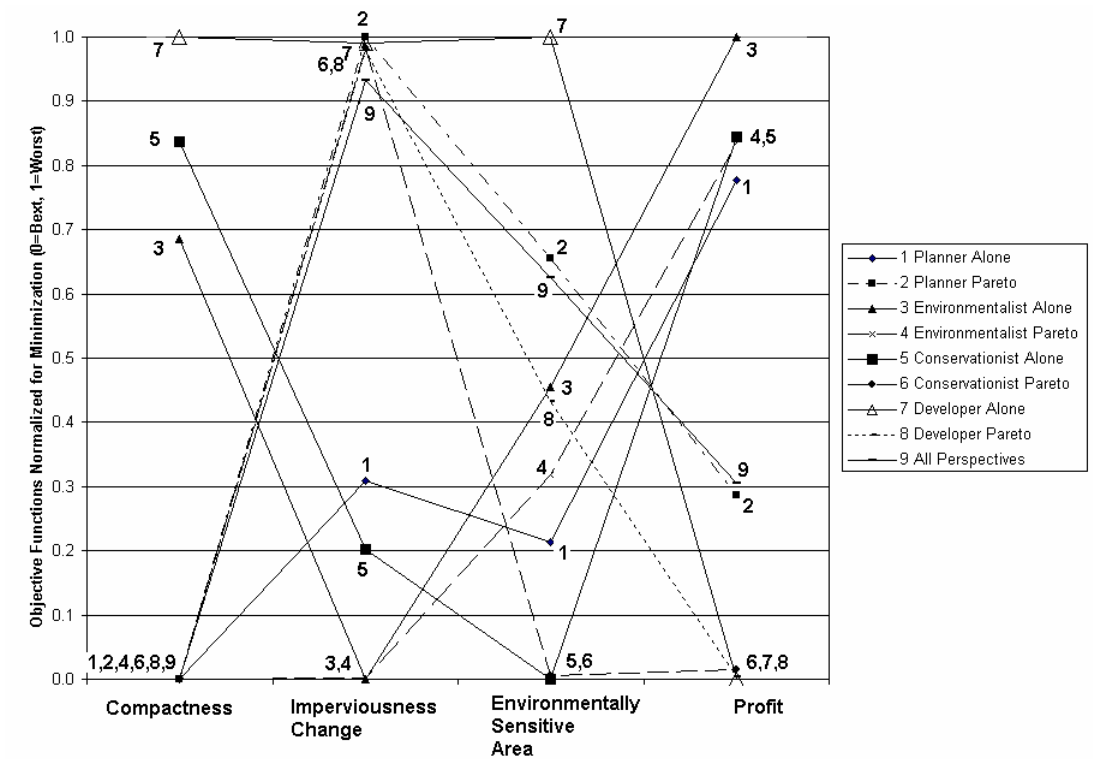


Figure 3.22 Value path representation of the nine cases evaluated

Figure 3.22 shows the normalized values of the four objective functions evaluated in the nine cases studied in a scale 0-1 where zero was the preferred solution and one the less desirable. Thus, *z* profit of zero means the highest profit and a imperviousness change of zero means the lowest imperviousness change. For example case 7 has the lowest compactness (parcels spread out more) among all cases, this case is also very high in the imperviousness and environmental measures while scores with a very high profit level (close to zero).

3.2.3. Problem's Structure

Once again, these long computational times are the motivation to implement relaxation and decomposition techniques similar to those already presented in the previous chapter. The implementation for this case should be straightforward similar to the previous chapter and thus are not presented in this dissertation.

Because the quadrants are divided in such way that the parcels belong to only one quadrant, then the structure of this formulation can be considered as Q independent formulations (one per quadrant) with common constraints that calculate the total development for each type of zone development (similar to the previous case) but with an additional set of complicating variables that appear along all zones (the unassigned parcels). Now, within each quadrant there are parcels from all type of zones including unassigned. Therefore, a combination of Benders decomposition and Dantzig-Wolfe decomposition would be required to solve this problem. We leave the formulation and related numerical implementations of this case for future work. We envision this structure as depicted in Figure 3.23.

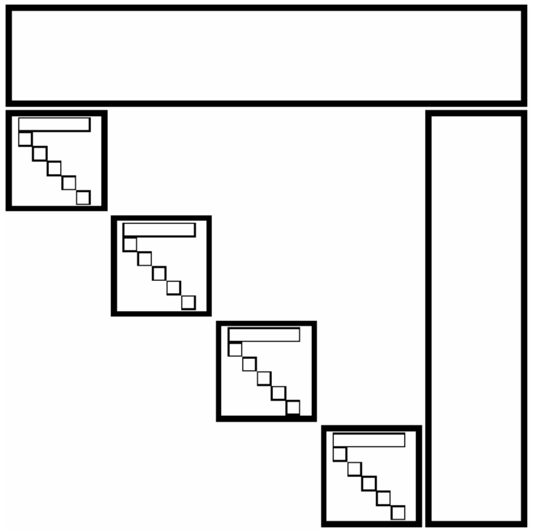


Figure 3.23 Decomposition structure of the quadratic model for land development

3.3. Nested Decomposition Strategies

Several researchers have applied sequentially Dantzig-Wolfe decomposition to solve large-scale problems raising a nested decomposition approach. Glassey (1973) applied this technique to a multi-stage linear programming problem (MLP). Ho (1977) applied the technique to a Manne's version of a linear programming problem of U.S. energy options with a staircase structure. Vanderbeck (2001) applied a nested decomposition to solve a cutting stock problem.

Similarly, nested Benders decomposition techniques has been used before in the context of multistage stochastic optimization (Birge, 1985; Gassman, 1990; Archibald and Buchanan, 1999; Watkins et al., 2000; Dempster and Thompson, 2005) where problems in the same time period can be solved independently with a decomposable structure exploited with a nested Benders algorithm.

Other applications arise in power systems where the complexity and size of the problem are addressed by nested decomposition methods (McCusker and Hobbs, 2003).

Thus far we have not been able to find any publications that combine Benders and Dantzig-Wolfe techniques together as proposed here to solve a problem of this structure.

3.4. Chapter Conclusions

The land development problem can incorporate compactness as an objective function. Using the rectangle approach the land development area can be divided into quadrants in order to prevent sprawl or used as a corridor to foster development in a specific direction set by the orientation of the coordinate system used. The rectangle-based measure has the advantage of convexity so any local solution is a global solution. However the parcels located inside the rectangle do not affect the measurement of compactness since they do not change the size of the rectangle. We propose in the next chapter a compactness measure that depends on all parcels selected for development.

This model has flexibility in the use of land since some of the parcels do not have a previously assigned zone type. However, the flexibility provided by allowing the model to decide the land use for each parcel increases the complexity of the model.

We envision Benders decomposition first to take care of the complicating variables for those parcels with unassigned zoning. Then we would have five subproblems, each being solvable by applying Dantzig-Wolfe decomposition.

Chapter 4 Embedded Minimum Spanning Tree Formulation

4.1. ***Selected Works About Minimum Spanning Trees***

The minimum spanning tree (MST) is one of the most widely studied problems in operations research (Graham and Hell, 1985) therefore there are numerous publications that analyze this problem from a variety of different perspectives. The following list is not meant to be exhaustive, just indicative of work previously done that reflect some resemblance with the concepts and algorithms later developed in this section.

Toussaint (1980) showed that the MST is a subset of the relative neighborhood graph (RNG) and presented two algorithms for obtaining the RNG of n points on the plane. This means that if one wants to construct a MST one could first construct a RNG and then use the edges as variables for the MST.

Vaidya (1984) studied the problem of finding a MST on a fully connected graph of n nodes in E^k with a bounded radius. He has developed a fast algorithm to find an approximation of the solution for a L_q distance metric where q = 2, 3, ... He used a search within a neighbor of each node for candidates to be included in the tree. This concept reinforces a basic property of the MST that is: each node will be connected to the closest node.

Lai and Sheng (1996) applied the concept of a closure defined as the set of all edges incident to a node with a specified length to select edges in an algorithm used to construct the Euclidean MST allowing a reduction on the size of the problem.

Zhou, G. and Gen (1999) considered simultaneously multicriteria in determining an MST, assigning multiple attributes on each edge creating a more realistic representation of the practical problem.

Yaman et al. (2001) modeled the robust spanning tree problem as a mixed integer programming formulation. In this formulation a single-commodity formulation and the dual of a multi-commodity formulation, both modeling the classic minimum spanning tree problem and both presented in Magnanti and Wolsey (1995), are joined together. Some rules are presented which allowed the author to reduce the size of the problem based on identify edges which will never be in the solution of a robust tree.

Montemanni and Gambardella (2002) presented a branch and bound algorithm for a robust version of the minimum spanning tree problem where edge costs are specified as intervals instead of fixed numbers. Based on the work of Yaman et al. (2001) a set of pre-processing rules are applied to reduce the dimension of the problem.

Graham et al. (2003) studied the capacitated minimum spanning tree (CMST) problem presenting a mixed integer programming formulation with a root node. They proposed an exact algorithm for solving the CMST problem using a heuristic since an exact procedure, which has to enumerate all feasible solutions, is exponential in the number of nodes is not applicable to very large size problems. They sorted the length of the edges and the algorithm chooses from the list starting with the smallest ones first, and applied a modification to the branch and bound search using an m-stage binary search tree.

4.2. The Minimum Spanning Tree in the Land Development Problem

According to Burchell et al. (2000), limiting the development of the land to areas close to those already developed is a control mechanism that could be applicable to reduce sprawl and the negative consequences associated with it. One can conclude from the literature review that the notion of compactness is often associated with the measure of density defined as dwelling units per unit of area. Sprawl and compactness are inversely related to each other. The more compact a development is, the less sprawl and vice versa.

We envision the parcels that are already developed as connected among them forming an existing infrastructure. New parcels will connect to this existing infrastructure by means of the minimum distance. With this idea in mind we have proposed the use of the minimum spanning tree (MST) as a compactness measure since it will promote the selection of parcels that are closer to existing developed parcels and therefore promoting compactness and preventing sprawl. This objective is considered in a multiobjective optimization problem in conjunction with other stakeholder objectives.

The foundation for the formulation is as follows: We propose the measurement and optimization of three objective functions. First the Planner's objective which will be to maximize the compactness measured as the resulting MST over all parcels chosen for development and the existing network of developed parcels. The existing network of developed parcels is presented as the MST which connects all already developed parcels, but this is only for purposes of representation and does not in any way implies that such a network is a tree. In fact, water supply networks, for example, are interconnected in densely settled areas by multiple cycles hence, the concept of the tree is not valid. The second objective considered is the Developer’s objective, which is maximizing the profits obtained from the development of the land. The third objective proposed is from the Hydrologist perspective the minimization of imperviousness. The constraints are to provide enough dwelling units to satisfy both the population growth in terms of residential units, and the economic growth in terms of acres for commercial and industrial use. Finally there is a set of constraints required to define the length of the minimum spanning tree.

Because a typical MST formulation involves an exponential number of variables constraints it is impractical if not impossible to solve with a large scale problem. Therefore a strategy is required to reduce the number of variables and constraints, such strategy was developed in this dissertation work. First, the fully connected concept was relaxed and the parcels are allowed to connect only to those parcels within vicinity. Second, not all constraints are imposed at once, rather just a subset of the full formulation is used and the relaxed problem is solved iteratively. On each iteration the disconnected elements are identified and new constraints to ensure connectivity of those components are added. Also, to speed up the process, the cycles within the graph are detected and new constraints are added to break the cycles. This double constraint generation approach has been tested here with networks of various sizes.

The algorithm developed is as presented in Figure 4.24. An additional step could be added to the algorithm in order to speed up the approach, where the potential cycles are identified ahead of time based on the parcels that have been selected for development during previous iterations.

In the next sections of this chapter, supporting arguments for the selection of the MST as a compactness measure are presented along with a discussion of the multiobjective formulation and implementation of the proposed algorithm supported by examples.

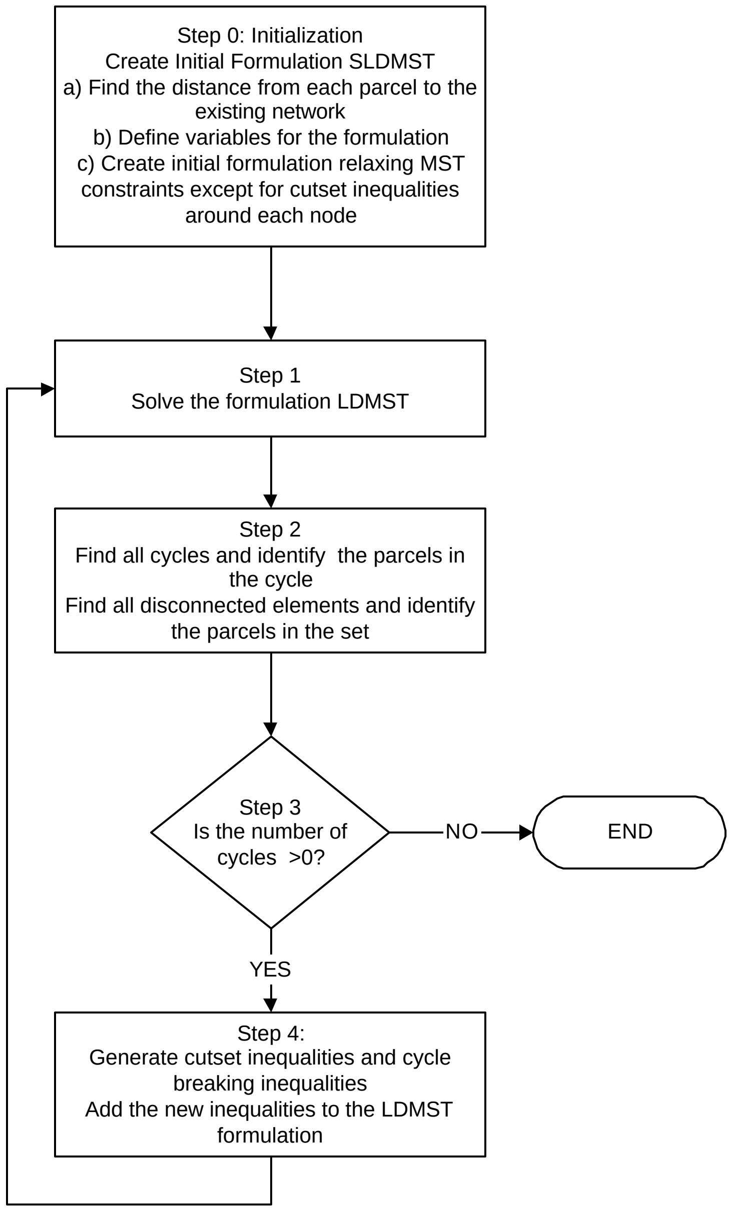

Step 0: Initialization
Create Initial Formulation SLDMST
a) Find the distance from each parcel to the existing network
b) Define variables for the formulation
c) Create initial formulation relaxing MST constraints except for cutset inequalities around each node
Step 1
Solve the formulation LDMST
Step 2
Find all cycles and identify the parcels in the cycle
Find all disconnected elements and identify the parcels in the set
Step 3
Is the number of cycles >0?
NO
END
YES
Step 4:
Generate cutset inequalities and cycle breaking inequalities
Add the new inequalities to the LDMST formulation

Figure 4.24 Algorithm to solve the land development formulation with embedded MST

4.3. The Minimum Spanning Tree as a Compactness Measure

Historically the roots of the minimum spanning tree (MST) could be traced as far back as the work of Kirchhoff (1824-1887) and other researchers of the last century (Ahuja et al. 1995). The discovery and formal presentation of the minimum spanning tree is attributed to Boruvka (1926a, 1926b) who considered the problem of an electric power company of Western Moravia seeking to interconnect cities to the existing power grid (Nesetril et al., 2000; Korte and Vygen, 2000). The problem was to distribute electricity, water, etc. from one point to another in the most efficient possible way. This is achieved by following a path of minimum distance or minimum cost which is found by the MST (Ahuja et al., 1995; Magnanti and Wolsey, 1995). Similar applications can be found in civil engineering when planning for new highways, one might first find the MST interconnecting the cities then try to fit a highway along the way since the distance to be covered would be minimum, or perhaps using a capacitated MST interconnect the major cities with highways and the smaller cities with routes (Magnanti and Wolsey, 1995).

By just reading some of the definitions of sprawl and the economic consequences one can conclude that the farther away a parcel is from a point of connection to the existing infrastructure the higher the cost of the development in terms of providing the required services to support the development.

The minimum spanning tree can be used to find the minimum length required to connect a group of points in the space, this concept can be applied to the connection of the parcels to the existing (or future) infrastructure. This problem is among the first combinatorial problems studied (Korte and Vygen, 2000).

4.3.1. Formulation of the Land Development Problem

4.3.1.1. Objectives

We are given a fully connected graph G of $V(G)$ nodes that represents the set of all parcels, and a set $V(H)$ of parcels previously developed such that

$V(H) \subseteq V(G)$. These two sets of nodes together with the interconnection between those previously developed parcels (edges $E(H)$) form the infrastructure network *H.* We propose the following objective functions for the stakeholders:

Planner' objective: maximize compactness

$$\text{Min: } \sum_{(i,j)\in E(G)} dist_{ij} e_{ij} \tag{4.1}$$

where $dist_{ij}$ is the length of the edge that joins parcels *i* and *j,* e_{ij} is the decision variable to include the edge *(i,j)* into the MST ($e_{ij} = 1$) or not ($e_{ij} = 0$). *E(G)* is the set of all edges in the graph.

Hydrologist' objective: minimization of imperviousness change

$$\text{Min: } \sum_{z=1}^{5} \sum_{n=1}^{N_z} a_{z,n} \Delta I_{z,n} d_{z,n} \tag{4.2}$$

Developer' objective: maximization of profits

$$\text{Max: } \sum_{z=1}^{5} \sum_{n=1}^{N_z} p_{z,n} d_{z,n} \tag{4.3}$$

were $a_{z,n}$ is the area of parcel *n* in zone *z.* $p_{z,n}$ is the profit from developing parcel *n* of zone *z,* d_{zn} is the decision variable to develop parcel *n* of zone *z. V(H)* is the set of available parcels, $\Delta I_{z,n}$ is the change in imperviousness when parcel *n* is developed of zone z. For this dissertation work as following the same zones as in Moglen, Gabriel and Faria (2003) and Gabriel, Faria and Moglen (2005), five zones were in the set of possible zones for development. Taking a weighted sum of the objectives, as previously done in Chapters 2 and 3, we can write the objective function as:

$$\text{Min: } w_1 \sum_{(i,j)\in E(G)} dist_{ij} e_{ij} + w_2 \sum_{z=1}^{5} \sum_{n=1}^{N_z} a_{z,n} \Delta I_{z,n} d_{z,n} - w_3 \sum_{z=1}^{5} \sum_{n=1}^{N_z} p_{z,n} d_{z,n} \qquad (4.4)$$

4.3.1.2. Constraints

The first set of constraints deal with the population, commercial and industrial growth of the region. The number of units developed for each zone should be bounded by the minimum and maximum required. These have been presented before in Chapter 2 as (2.10) - (2.14)

$$\underline{RLD} \le \sum_{n=1}^{N_1} u_{z,n} d_{z,n} \le \overline{RLD}, z = 1 \qquad (4.5)$$

$$\underline{RMD} \le \sum_{n=1}^{N_2} u_{z,n} d_{z,n} \le \overline{RMD}, z = 2 \qquad (4.6)$$

$$\underline{RHD} \le \sum_{n=1}^{N_3} u_{z,n} d_{z,n} \le \overline{RHD}, z = 3 \qquad (4.7)$$

$$\underline{COM} \le \sum_{n=1}^{N_4} u_{z,n} d_{z,n} \le \overline{COM}, z = 4 \qquad (4.8)$$

$$\underline{IND} \le \sum_{n=1}^{N_5} u_{z,n} d_{z,n} \le \overline{IND}, z = 5 \qquad (4.9)$$

The next sets of constraints are required to define the minimum spanning tree, Appendix 2 presents several formulations developed to define the MST. Among them we prefer to use the cutset formulation (A.210) - (A.213) over the packing formulation (A.206) - (A.209) because less inequalities are required and because they are easier to generate computationally.

Every parcel selected for development should be connected to the current infrastructure, this means it should be connected either to one of the previously developed parcels, or to the MST of parcels selected for development which in time is connected to the existing network. Since the MST of the previously developed parcels contains *n-1* nodes, for each newly developed parcel there will be one edge required to connect it to the existing MST. So the number of edges added to the MST has to be equal to the number of parcels selected for development.

$$\sum_{ij} e_{ij} = \sum_{z=1}^{5} \sum_{n=1}^{N_z} d_{z,n} \tag{4.10}$$

A node should have incident edges if and only if it is chosen for development. This double set of constraints is required given the fact that an edge is defined only if the parcels located at both ends of the edge are chosen for development.

$$\sum_{j} e_{ij} \le n \sum_{z} d_{z,n}, i = 1,2,...,n \tag{4.11}$$

$$\sum_{i} e_{ij} \le n \sum_{z} d_{z,n}, j = 1,2,...n \tag{4.12}$$

For example in Figure 4.25 edges (1,2), (2,3), (2,4) and (2,5) can only be defined if all the parcels in the set {1,2,3,4,5} are selected for development. If for example parcel 5 was not selected for development then the corresponding variable for the edge (2,5) $e_{2,5} = 0$.

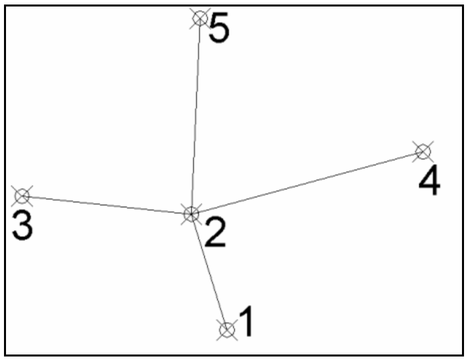


Figure 4.25 Example of four edges connecting five nodes

Both sets (4.11) and (4.12) are required since both nodes *i* and *j* are required to be developed for the edge to exist.

The cutset inequalities can be used to ensure the connectivity of all nodes in the tree. That is, any subset of nodes *S* should be connected at least with one edge to the rest of the nodes in the network, either previously developed or selected for development.

$$\sum_{\{(i,j)\in E(G): i\in S, j\in V(G)\setminus S\}} e_{ij} \geq (1-y_s) \forall S \subset V(G), S \neq \emptyset \tag{4.13}$$

$$\sum_{i\in S} d_i \leq n(1-y_s) \tag{4.14}$$

$$y_s \in \{0,1\} \tag{4.15}$$

To illustrate this set of inequalities consider Figure 4.26 and Figure 4.27.

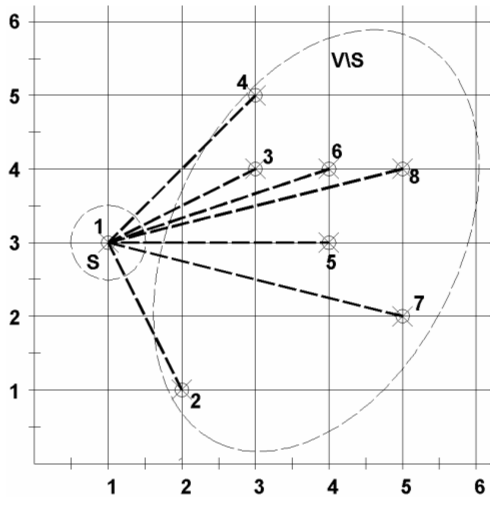


Figure 4.26 Set of one node connected to all other nodes

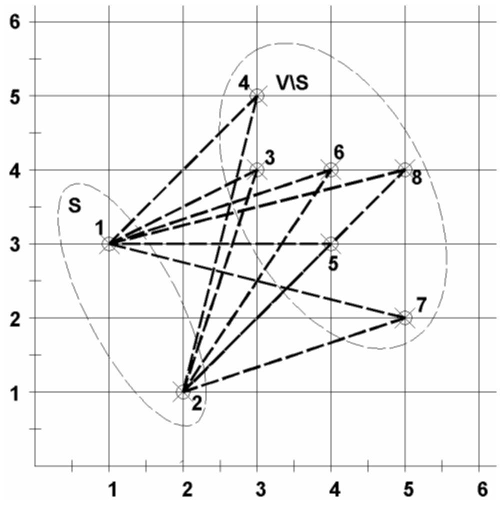


Figure 4.27 Set of two nodes connected to all other nodes

In Figure 4.26 if node 1 is selected for development, then that node should be connected to at least one of the other nodes (previously developed or selected for development parcels) in the graph. In Figure 4.27 if both nodes 1 and 2 are selected for development, then they should be also connected to the rest of the nodes in the graph.

Constraint (4.14) ensures that if there is at least one developed parcel in the set then the binary variable $y_S = 0$ which then forces $\sum_{\{(i,j)\in E(G): i\in S, j\in V(G)\setminus S\}} e_{ij} \geq 1$ by means of constraint (4.13). This ensures that the set *S* is connected to the rest of the graph by at least one edge.

Note that it is not required to have the packing constraints and the cutset constraints. The complete formulation would have redundant constraints if both are included at the same time. They are included here for reasons that will be come obvious when the proposed algorithm is presented.

Binary definition of the edge and development decision variables:

$$d_{iz} \in \{0,1\} \tag{4.16}$$

$$e_{ij} \in \{0,1\} \tag{4.17}$$

The implementation of formulation (4.4) - (4.17) as presented might be impossible to solve for a large network because the number of constraints involved in the definition of the MST is exponential.

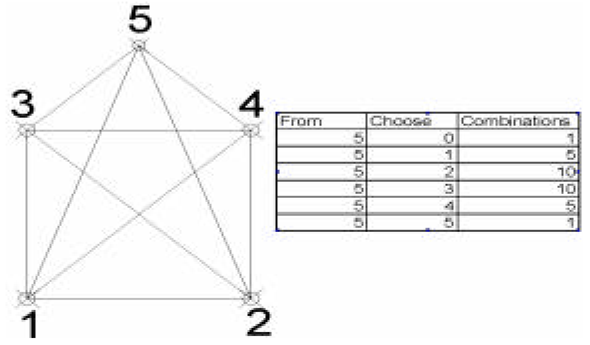
To understand the exponential nature of those constraints consider the following: we need to take groups of one node first, and connect them to the rest of the nodes, if there are *n=V(G)* nodes then the number of constraints is $\binom{n}{1} = \frac{n!}{(n-1)!}$, to take groups of two nodes and connect them with the rest of the graph we need $\binom{n}{2} = \frac{n!}{(n-2)!2!}$ constraints, and then we need to take groups of three nodes which are $\binom{n}{3} = \dfrac{n!}{(n-3)!3!}$ and so on. However, we do not need to add all the count together which would be $2^n - 2$ because taking combinations of one node is equivalent to take combinations of *(n-1)* nodes, combinations of two nodes is equivalent to taking combinations of (*n-2)* nodes and so on. Therefore, the number of constraints in this group would be

$$\frac{2^n - 2}{2} \text{ if } n \text{ is odd} \tag{4.18}$$

and

$$\frac{2^n - 2 - \binom{n}{n/2}}{2} + \binom{n}{n/2} \text{ if } n \text{ is even.} \tag{4.19}$$

To understand how we arrived to these formulas consider first the case presented in Figure 4.28.



From	Choose	Combinations
5	0	1
5	1	5
5	2	10
5	3	10
5	4	5
5	5	1

Figure 4.28 Graph of five nodes with a table of possible combinations

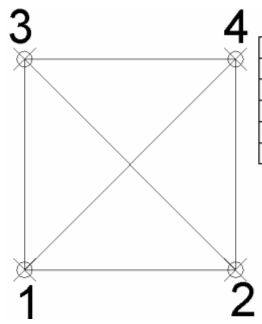
Consider for example a graph with five nodes such as presented in Figure 4.28, where the table lists all possible combinations from five nodes to choose. Note that the total number of possible combinations is given by:

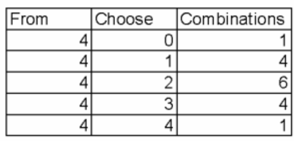
$$\sum_{i=0}^{n} \binom{n}{i} = 2^n \tag{4.20}$$

The first combination shown in the table of Figure 4.29 is the combination of five nodes taken by zero, this is identical to the combination of all nodes taken by all nodes. This combination is not explicitly included as a cutset inequality, rather is implicitly in constraint (4.10) which accounts the number of edges in the tree.

Note how the number of possible combinations is symmetrical to *n/2* meaning that the number of combinations of nodes taken by say a number of *c* nodes is identical to the number of possible combinations of the nodes taken by *n-c* when *c* is less than *n/2*. We only need one set of these constraints since they are redundant. Therefore, from all possible combinations $\left(2^n\right)$ we do not need the combinations taken by zero (or by n) so we can deduct those from the total number obtaining $\left(2^n - 2\right)$, because of the symmetry explained above, we only need half of those constraints leaving the final number of constraints required as $\left(\frac{2^n - 2}{2}\right)$. In our example we need to take combinations of five taken by one, and comb inations of five taken by two.

Now consider the case of a four node graph as the one presented in Figure 4.29.





From	Choose	Combinations
4	0	1
4	1	4
4	2	6
4	3	4
4	4	1

Figure 4.29 Graph of four nodes with a table of possible combinations

By the same argument as before, not all combinations of nodes are required to generate the constraints, the difference is that in this case (n/2) is integer so the total number of constraints can be computed as $\dfrac{2^n - 2 - \binom{n}{n/2}}{2}$ to account for the combinations from $1,2,\ldots,n/2-1$ constrains with symmetrical groups, to this we need to add the $\binom{n}{n/2}$ not yet accounted for.

But since we need an additional constraint to define the auxiliary variable y_s per cutset inequality, then the total number of constraints required is two times the number computed in (4.18) and (4.19), still these are an exponential number of constraints as presented in Table 4.1.

n	Constraints
3	6
4	20
5	30
6	82
7	126
8	324
9	510
10	1,274
11	2,046
12	5,018
13	8,190
14	19,814
15	32,766
16	78,404
17	131,070
18	310,762
19	524,286
20	1,233,330
21	2,097,150
22	4,899,734
23	8,388,606
24	19,481,370

Table 4.1 Number of constraints as a function of the number of nodes

Note that with 21 nodes the number of constraints exceeds one million. Figure 4.30 shows the exponential function for the number of constraints as a function of the number of nodes.

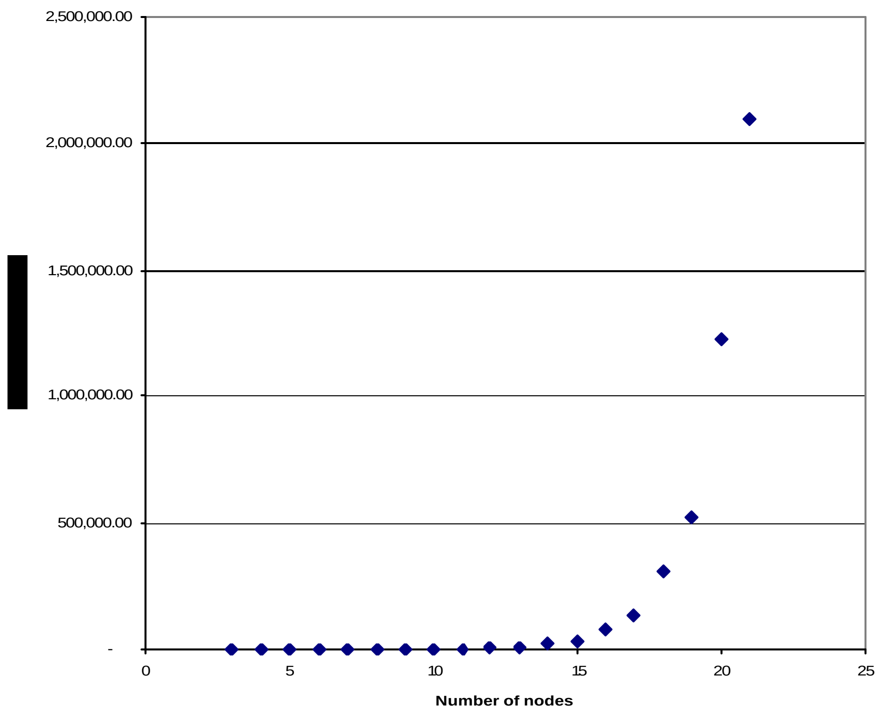


Figure 4.30 Number of constraints as a function of the number of nodes

The number of variables is also extremely large, since the nodes of the graph are parcels, and in theory if it can be a fully connected graph, then the number of edges is also exponential.

4.3.2. Solution Approach

We have developed an algorithm to solve this multiobjective land development problem with an embedded MST. The first step requires the solution of an initial formulation. This initial formulation is based on the previously presented formulation (4.4) - (4.17) with some modifications. In order to have any hope solving this problem given the extremely large number of variables and constraints we attacked the problem on two fronts as follows.

4.3.2.1. Reduction in the number of variables

The first and almost obvious reduction in the number of variables can be accomplished by eliminating the double counting of the edges. We identify each arc by e_{ij} where $i < j : \forall (i, j) \in E(G)$. Because we are only concerned with connecting the newly developed parcels to the existing infrastructure, we do not care about the direction of the edges. Hence, an edge $e_{ij} = e_{ji}$ rather than introducing these equalities in the formulation we define the edges in a lexicographic order. This allows us to effectively reduce the number of variables by half without losing any accuracy in the description of the problem.

The second reduction, although not that obvious is still easy to understand. Since each parcel represents a node in the graph, and the graph is considered fully connected, then there are $\binom{n}{2}$ edges to consider. This number is extremely large but not all those edges need to be used, just a subset of edges that connect each node to a group of geographically close nodes needs to be considered.

Theorem 1 In an optimal solution of the land development problem, two parcels *i* and *j* chosen for development will never be directly connected if the distance between them is larger than both the distance from *i* to the its closest previously developed parcel, and the distance from *j* to its closest previously developed parcel.

Proof Suppose that for an optimal solution, there is a parcel *i* selected for development which is connected in the MST to a parcel j. Now suppose that the distance from parcel *i* to its closest developed parcel (call it parcel 1) is $dist_{i1}$, and the distance from parcel *j* to its closest developed parcel (call it parcel 2) is $dist_{j2}$. Because the MST cannot have any cycles, it is impossible for the edges e_{i1}, e_{ij}, and e_{j2} to be selected simultaneously. Assume that the connection is made between the two nodes to the rest of the MST by means of edge e_{i1}, this can be done without loss of generality since it could be e_{j2} as well, see Figure 4.31 left.

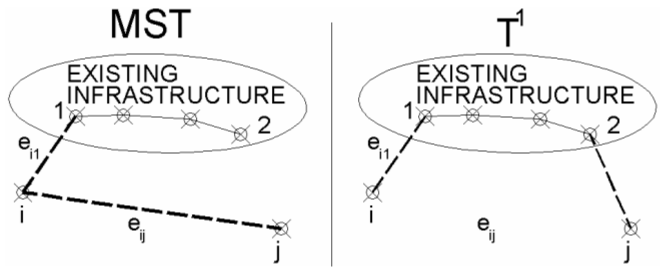


Figure 4.31 Nodes i and j connected to the existing infrastructure (left using e$_{ij}$, right using e$_{i1}$ and ej2)

If the distance from parcel *i* to parcel *j* $dist_{ij}$ satisfies, $dist_{ij} > dist_{i1}$ and $dist_{ij} > dist_{j2}$ then we could build another tree, namely T^1 for which we replace the edge e_{ij} by the edge e_{j2}. The total length of the tree $T^1 = MST - dist_{ij} + dist_{j2}$. Since the term $-dist_{ij} + dist_{j2} < 0$ because $dist_{ij} > dist_{i1}$, then we will have found a tree with a shorter dimension than MST so either MST is not a minimum tree, or $dist_{ij} \le dist_{i1}$ QED.

The consequence of Theorem 1 allows us to reduce the number of potential edges to consider by each parcel to only those that are within a circle of radius equal to the closest developed parcel (see Figure 4.32). The reduction in the number of edges and variables is substantial but cannot be calculated in general since it depends of the relative location of the developed nodes in reference to the available nodes.

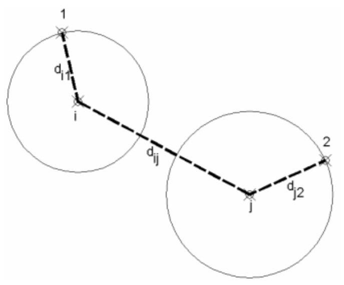


Figure 4.32 Two nodes *i,j* available for development connected to previously developed nodes 1 and 2

4.3.2.2. Reduction in the Number of Constraints

We also considered reducing the number of constraints. We designed an algorithm that iteratively moves from a series of forests to the optimal solution by adding cycle breaking constraints and cutset inequalities on each iteration.

The algorithm presented in Figure 4.33 begins with the formulation presented in (4.4) - (4.15) but slightly modified by relaxing the set (4.13) - (4.15) and adding cutset inequalities around each node. This decision was made based on the performance of the algorithm. It is clear that if a parcel is developed then it must be connected, so we save some iterations by including this set of constraints ahead of time. We then solve this initial formulation, and apply a heuristic based on a shortest path method to locate the cycles in the graph and then to locate the forest elements in the graph. Once we have identified these, we add only anti-cycling constraints (packing type of constraints) and connecting constraints (cutset type of inequalities) to avoid the cycles that we have found and connect the elements that are disconnected.

4.3.3. Solution Algorithm

4.3.3.1. Initial Formulation LDMST

$$\text{Min: } w_1 \sum_{(i,j)\in E(G)} dist_{ij} e_{ij} + w_2 \sum_{z=1}^{5}\sum_{n=1}^{N_z} a_{z,n}\Delta I_{z,n} d_{z,n} - w_3 \sum_{z=1}^{5}\sum_{n=1}^{N_z} p_{z,n} d_{z,n} \qquad (4.21)$$

s.t.

$$\underline{RLD} \le \sum_{n=1}^{N_1} u_{z,n} d_{z,n} \le \overline{RLD}, z = 1 \qquad (4.22)$$

$$\underline{RMD} \le \sum_{n=1}^{N_2} u_{z,n} d_{z,n} \le \overline{RMD}, z = 2 \qquad (4.23)$$

$$\underline{RHD} \le \sum_{n=1}^{N_3} u_{z,n} d_{z,n} \le \overline{RHD}, z = 3 \qquad (4.24)$$

$$\underline{COM} \le \sum_{n=1}^{N_4} u_{z,n} d_{z,n} \le \overline{COM}, z = 4 \qquad (4.25)$$

$$\underline{IND} \le \sum_{n=1}^{N_5} u_{z,n} d_{z,n} \le \overline{IND}, z = 5 \qquad (4.26)$$

$$\sum_{ij} e_{ij} = \sum_{z=1}^{5}\sum_{n=1}^{N_z} d_{z,n} \qquad (4.27)$$

$$\sum_{j} e_{ij} \le n \sum_{z} d_{z,n}, i = 1,2,...,n \qquad (4.28)$$

$$\sum_{i} e_{ij} \le n \sum_{z} d_{iz}, j = 1,2,...n \qquad (4.29)$$

$$\sum_{j\in V(G)} e_{ij} \ge d_{i,z}; \forall i \in V(H) \qquad (4.30)$$

$$d_{iz} \in \{0,1\} \qquad (4.31)$$

$$e_{ij} \in \{0,1\} \qquad (4.32)$$

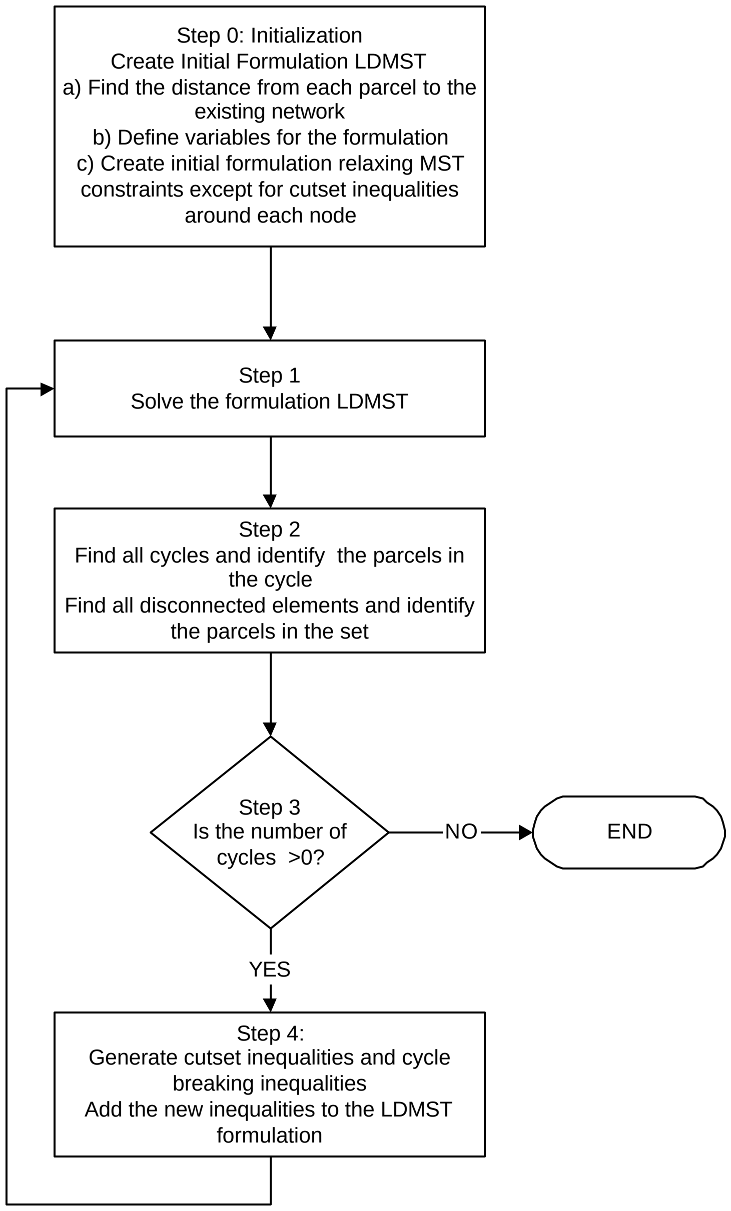

Step 0: Initialization
Create Initial Formulation LDMST
a) Find the distance from each parcel to the existing network
b) Define variables for the formulation
c) Create initial formulation relaxing MST constraints except for cutset inequalities around each node
Step 1
Solve the formulation LDMST
Step 2
Find all cycles and identify the parcels in the cycle
Find all disconnected elements and identify the parcels in the set
Step 3
Is the number of cycles >0?
NO
END
YES
Step 4:
Generate cutset inequalities and cycle breaking inequalities
Add the new inequalities to the LDMST formulation

Figure 4.33 Algorithm to solve the land development formulation with embedded MST

Constraints (4.30) prevents that a parcel selected for development remains disconnected, it will be connected to at least another parcel.

4.3.3.2. Step 0: Initialization

The initial step requires preparing the formulation LDMST. The edges to be included as decision variables need to be defined. The distance from each parcel available for development to the nearest developed parcel needs to be found, and then compared to the distance to the rest of the undeveloped parcels. If there is an undeveloped parcel closer than the closest developed parcel, then the edge that joins those two undeveloped parcels is included in the set of decision variables.

For example consider Figure 4.34 where two undeveloped parcels (A and B) and one developed parcel (1) are presented. First by evaluating node A, we note that the closest developed parcel is parcel 1, there is no other parcel in a radius of length $dist_{A1}$, so we define the variable e_{A1}. Then, by evaluating node B we find that the closest developed parcel is parcel 1, but there is one parcel (parcel A) in the radius of length $dist_{B1}$ so we define the decision variable e_{B1} and the decision variable e_{AB}.

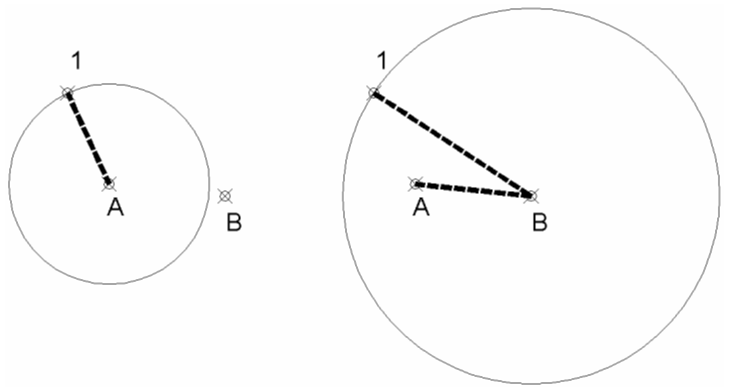


Figure 4.34 Two undeveloped parcels A, and B and one previously developed parcel 1

After all nodes available for development have been evaluated as described above, we solve the initial formulation LDMST and proceed to Step 1 in the algorithm.

4.3.3.3. Step 1: Solve the Formulation LDMST

Solve the formulation and record the solution which is the vector of decision variables for the parcels, and the vector of decision variables for the edges.

4.3.3.4. *Step 1: Find the Cycles and Identify the Disconnected Components*

Using an algorithm to find a shortest path to all nodes from a given node *s* take one of the parcels chosen for development and find all nodes connected to it, this will identify a component in the graph. Proceed until all developed nodes have been evaluated.

By evaluating each edge selected e_{ij} and finding the shortest path between nodes *i* and *j* (not including edge e_{ij}) the cycles can be identified. If there is a path between nodes I and j different than the edge e_{ij} then there is a cycle formed by the nodes in the path and the edge e_{ij}.

4.3.3.5. *Step 3: Optimality Check*

Is the number of cycles = 0. If so stop else go to next step.

4.3.3.6. *Step 3: Generate Inequalities to Break Cycles and Connect Disconnected Components*

Generate cutset inequalities of the form (4.13) - (4.15).

Take all trees found in step 1 of the algorithm and generate cutset inequalities of the form (4.30) to connect those isolated components into a tree.

In the example shown in Figure 4.35 a valid constraint would be

$$e_{1,3} + e_{2,3} + e_{3,4} + e_{3,5} \geq d_{3,z} \tag{4.33}$$

For all cycles found within each one of the components, including the tree of previously developed parcels, generate packing inequalities of the form:

$$\sum_{ij} e_{ij} \leq |S| - 1 \, \forall (i, j) \in S \subset E(G) \tag{4.34}$$

Any group of *S* nodes should not have any more than *S-1* edges. An example is presented in Figure 4.35

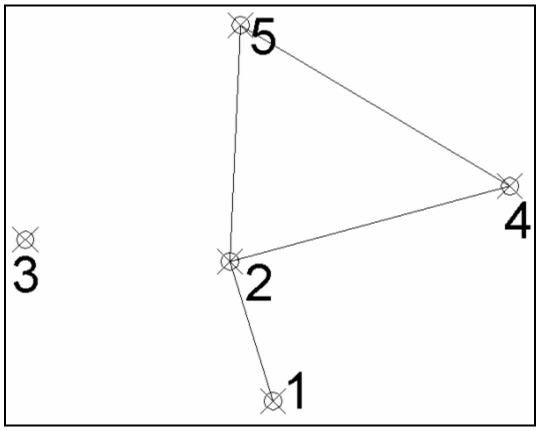


Figure 4.35 Example of a cycle and two disconnected sets

In Figure 4.35 there are three edges between the three nodes {2,4,5}. If one of the edges is removed then the cycle is removed. A valid constraint would be

$$e_{2,4} + e_{2,5} + e_{4,5} \leq d_{2,z} + d_{4,z} + d_{5,z} - 1 \tag{4.35}$$

The following example was solved step-by-step, to clarify the algorithm.

4.3.4. Example with 100 Nodes W=(1,1,1)

Suppose that there are 40 previously developed parcels connected in a MST[6] and 20 parcels available for development as presented in Figure 4.36.

[6] This is not a requirement and likely is not the case either, assumed only for clarity purposes.

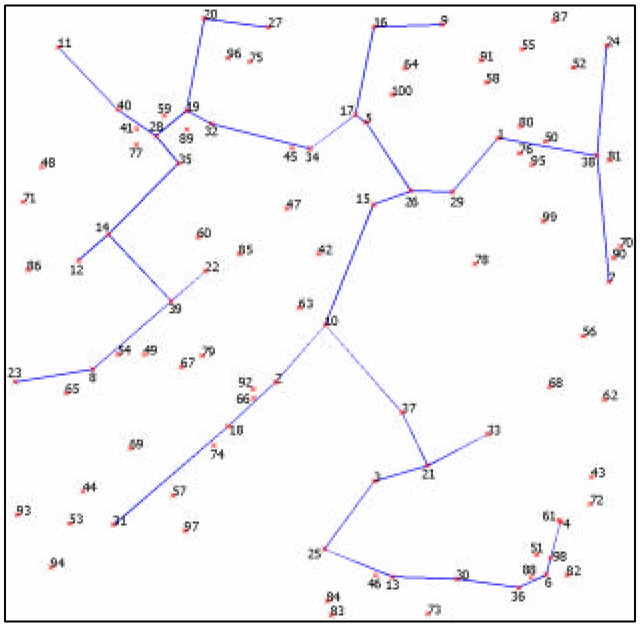


Figure 4.36 Initial set of 40 developed parcels connected with dark edges, and set of 60 parcels available for development (not connected)

4.3.4.1. Step 1: Solve the Initial Formulation

We solved the initial formulation (4.21) - (4.32) using a weight of one for each objective. Then we proceeded to graph the solution as in Figure 4.37.

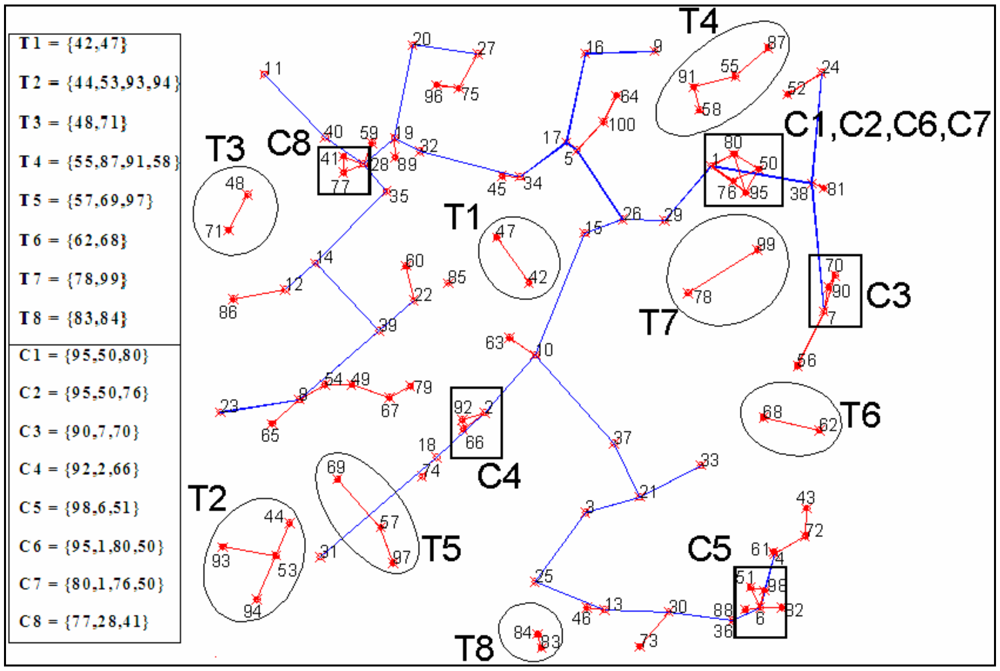


Figure 4.37 Solution of first iteration step 1, new edges shown lighter

4.3.4.2. Step 2: Find all Cycles and Disconnected Sets

Figure 4.37 shows that the solution of the initial step is a forest with eight disconnected elements identified with ellipses, and eight cycles identified with rectangles.

4.3.4.3. Step 3: Optimality Check

Since there are cycles and disconnected elements in the graph, the solution is neither optimal not feasible. We proceed with the next step.

4.3.4.4. Step 4: Generate Inequalities to Break Cycles and Connect Elements

We create inequalities to break the cycles found in the solution. The details of the cycles found are presented in Figure 4.38 and Figure 4.39.

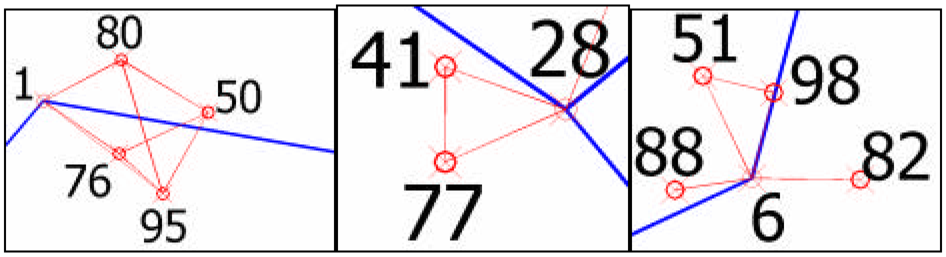


Figure 4.38 Three of the five cycle areas found with the first solution

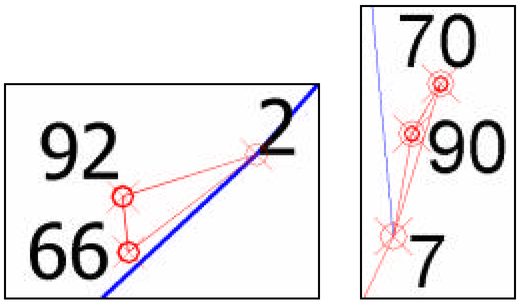


Figure 4.39 Two of the five cycle areas found in the initial solution

4.3.4.5. Step 4: Generate Cutset Inequalities to Connect the Trees in the Forest

We create inequalities to ensure the connectivity of the components that were not connected in the solution. There were at least eight trees not connected in the initial solution. For example, we observe in Figure 4.40 two of the disconnected components, one formed by the set {44, 53, 93, 94} and another by the set {57, 69, 97}.

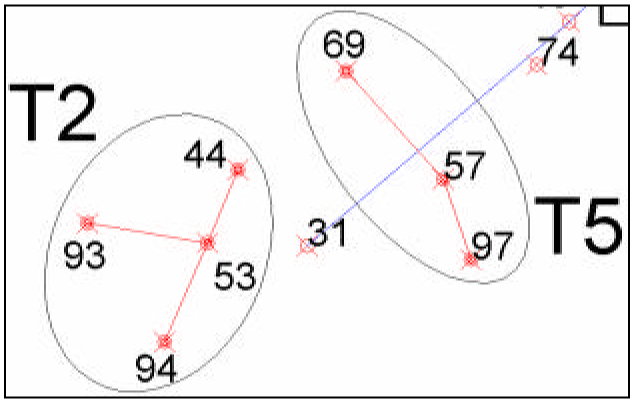


Figure 4.40 Two of the disconnected components found in the initial solution

4.3.4.6. Step 5: Solve the Augmented Formulation

After adding the cycle breaking constraints and the connectivity constraints to the previous formulation we proceed to solve the augmented formulation. Obtaining the solution presented in Figure 4.41.

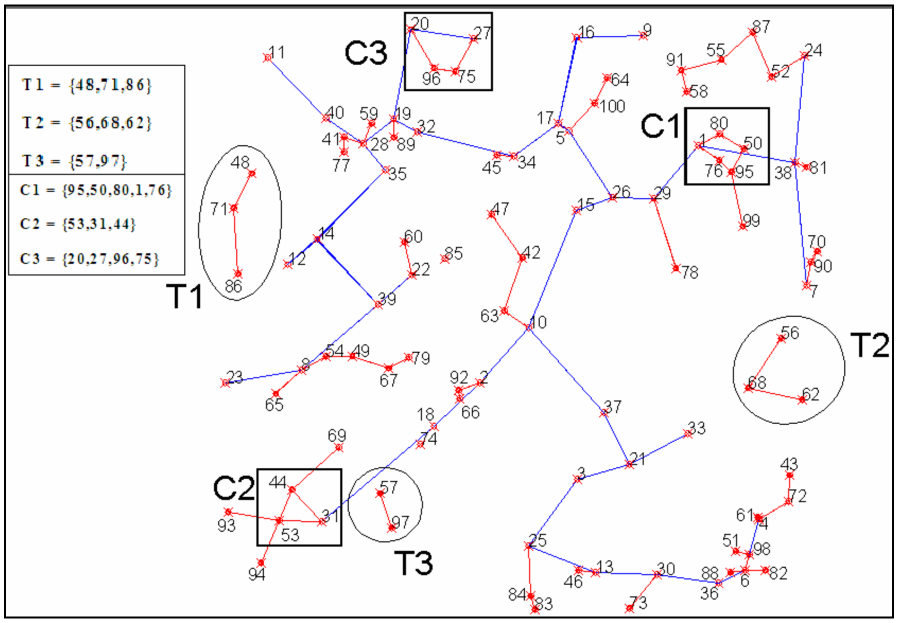


Figure 4.41 Second iteration, solution to the augmented formulation

Figure 4.41 shows that those components that were previously disconnected are now connected, for example the previously disconnected set {48, 71} is now connected to node 86. The cycles we had before are eliminated, but new cycles have been created, for example the edges around nodes {41, 77, 28} no longer form a cycle, but the edges around nodes {44, 31, 55} now form a cycle.

4.3.4.7. Step 2: Optimality Check

Since there are still some disconnected components the solution is neither optimal not feasible. We continue this process for two more iterations until we found an optimal solution as presented in Figure 4.42.

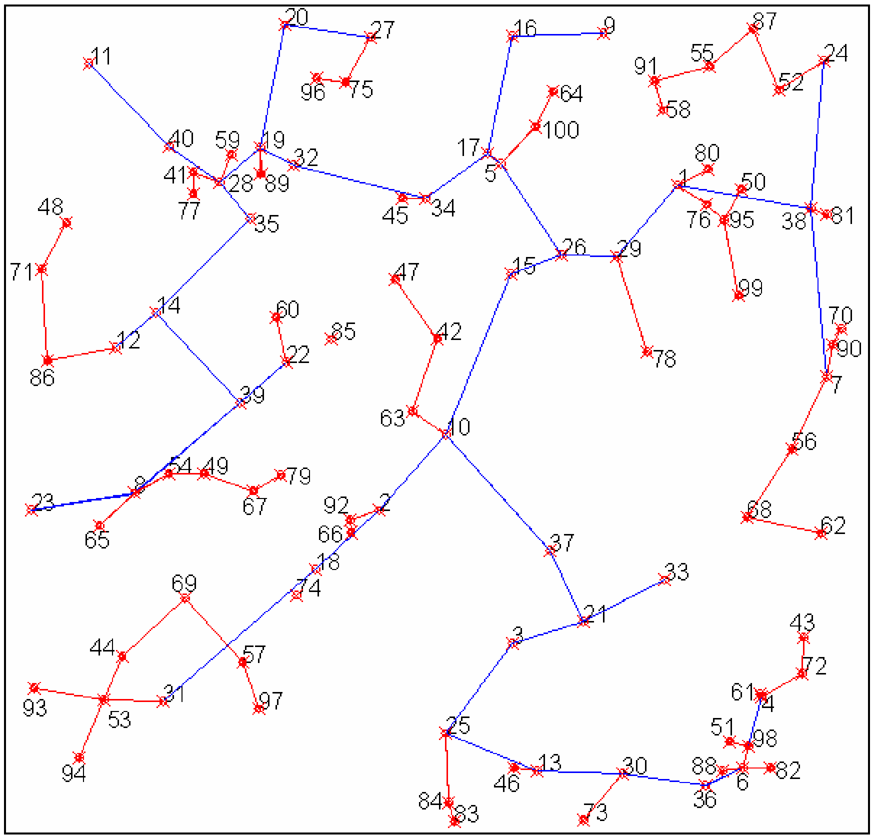


Figure 4.42 Optimal solution found

The solution found is a tree since it doesn't have cycles, has minimum distance (since the objective function is minimization) and complies with all other growth constraints therefore is optimal. This case was relatively easy to solve, in just four iterations we were able to fully create the constraints required to depict the problem. The data collected from each iteration is shown in Table 4.2.

	Add. Ineq.	Time sec	Variables	Constraints	MST	Profit	Imp Ch	Cycles	Disc. Elements
Iter 1	0	0.2	414	209	2,934.17	39,538.70	17,928.50	8	8
Iter 2	24	0.1	422	233	3,169.34	39,538.70	17,928.50	3	3
Iter 3	33	0.1	425	242	3,252.94	39,538.70	17,928.50	1	1
Iter 4	36	0.2	426	245	3,258.82	39,538.70	17,928.40	0	0

Table 4.2 Data collected per iteration

Table 4.2 provides information about the advance of the algorithm, we notice that the length of the MST is increasing as the algorithm moves forward, this is expected since at each iteration new constraints are added, therefore the feasible region is being reduced. The number of additional inequalities added decreases on each step from 24 added in the first iteration down to three added in the last one.

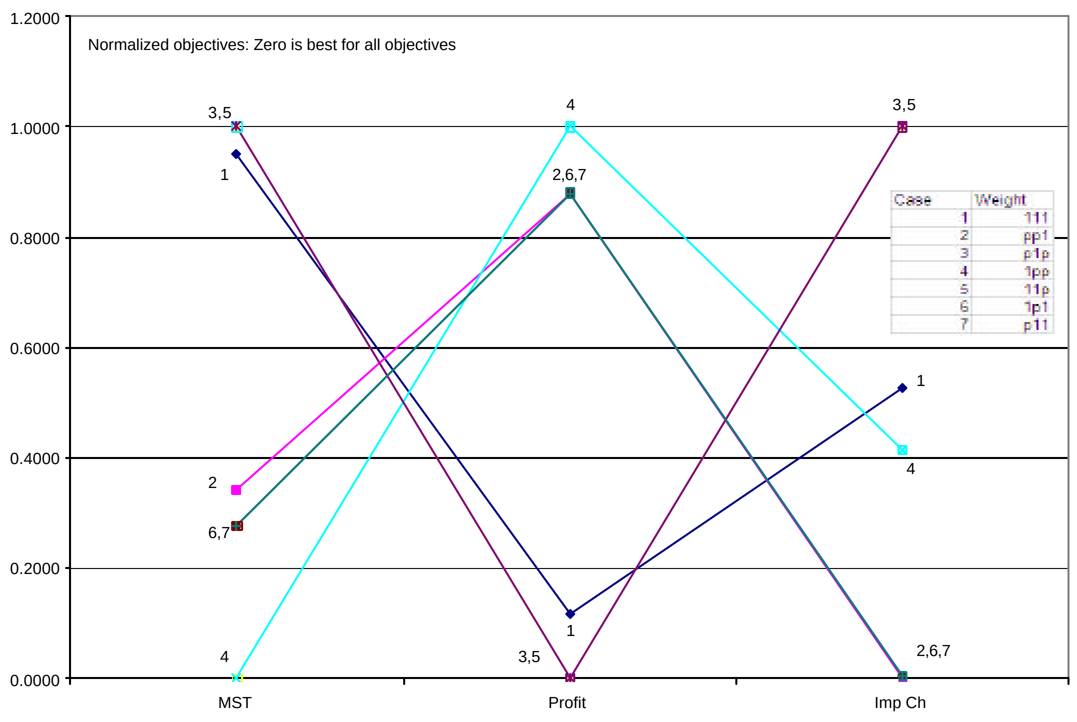
4.3.5. Example with 100 Nodes, 40 Previously Developed and 60 Available for Development Using Other Weight Combinations

Given the success with the small example, we decided to analyze the effects of changing the weights to the number of iterations, variables and constraints required to solve the problem. We are also interested in looking at the effect of such changes in the compactness of the solution. The cases evaluated and their solutions are presented in Table 4.3. The weight is represented by a three digit code with either a 1 or a *p*. The order of the digits represents the weight given to the compactness measurement, profit measurement and imperviousness change measurements respectively. The number 1 represents a weight of 1 to the objective, and the letter *p* stands for a small positive weight. For example case 2 with a weight code of *pp*1 means a small positive weight associated to the compactness and profit measure, and a weight of 1 to the imperviousness change.

Case	Weight	MST	Profit	Imp Ch	Iterations
1	111	3,258.82	39,538.70	17,928.50	4
2	pp1	1,971.21	15,446.00	8,086.86	3
3	p1p	3,358.87	43,233.20	26,777.50	4
4	1pp	1,254.32	11,600.20	15,845.50	5
5	11p	3,358.87	43,233.20	26,777.50	4
6	1p1	1,833.94	15,404.60	8,169.74	3
7	p11	1,833.94	15,404.60	8,169.74	3

Table 4.3 Cases evaluated with different weights

These cases were relatively easy to solve, within five iterations the optimal solution was found. The value of the objectives were normalized in a 0-1 scale where 0 is the preferred value and plotted together in Figure 4.43. As expected the solution with the most compact development strategy (Case 4) is also the one with the least profit, the rationale is that Case 4 with weight (1,p,p)



Case	Weight
1	111
2	pp1
3	p1p
4	1pp
5	11p
6	1p1
7	p11

Figure 4.43 Value path graph for the seven cases analyzed

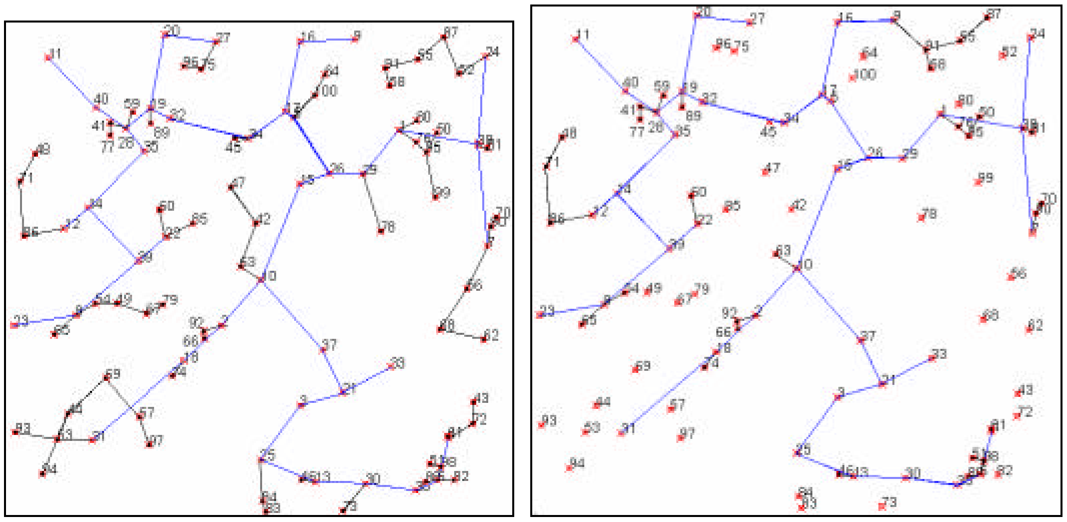


Figure 4.44 MST of Case 3 (Left) and Case 4 (right)

Figure 4.44 presents a comparison of the MST obtained by Cases 3 (left) and 4 (right). We notice that Case 3 has many more parcels developed than Case 4, this is because the profit is the objective with the highest weight. In contrast, Case 4 has fewer parcels and they tend to be located closer to the existing MST.

4.3.6. Example with parcels used in Gabriel, Faria and Moglen 2005

We started to solve the model using all parcels in Gabriel, Faria and Moglen (2005). There were a total of 1990 parcels with 1462 parcels previously developed that form an infrastructure as presented in Figure 4.45.

This task has proven to be an arduous, a solution was obtained after 237 iterations and a total of about 80 hours of computation time. We have noticed that at each iteration only a few constraints were added. This phenomenon was observed with other tests we ran, but became critical with this case.

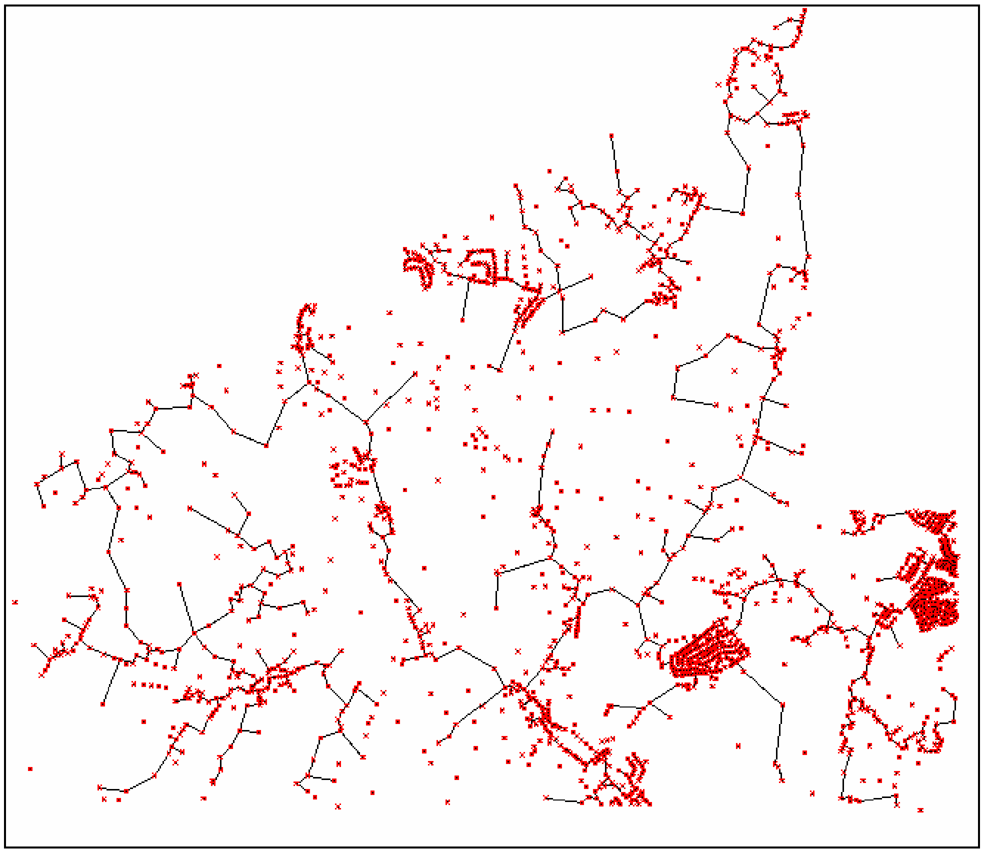


Figure 4.45 MST over all existing parcels

Figure 4.46 shows the number of constraints added per iteration. It is clear that at the beginning many cycles are found but as the algorithm progresses, the number of cycles actually created is reduced drastically. However the possible number of combinations for arcs to create cycles is still exponential. We had recorded all cycles and trees found per iteration with the hope of improve the current algorithm to predict the parcels that will tend to be tied together.

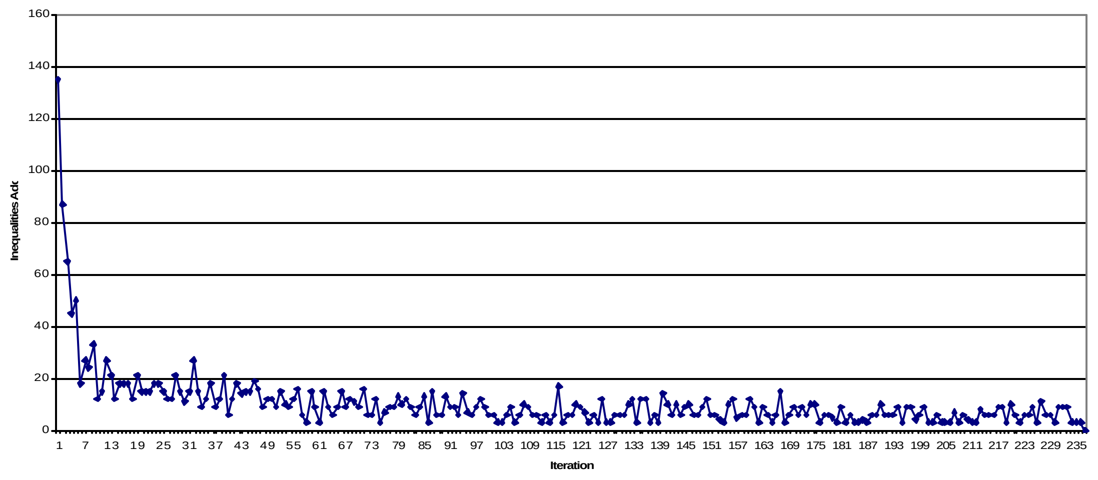


Figure 4.46 Number of constraints added per iteration using 1990 parcels

One of such ideas is to measure the distance from each parcel to all neighbor parcels (those closer than the closest previously developed parcel) then assign a likelihood of connection to each arc, given the average or maximum distance of arcs incident to a node. Having that information one can create an exponential number of constraints only for the group of arcs with very high likelihood of creating a cycle.

The difficulties obtained with this case were expected, the model has 4,711 variables and 4,089 constraints. Another interesting chart that could provide insight to the solution is presented in Figure 4.47.

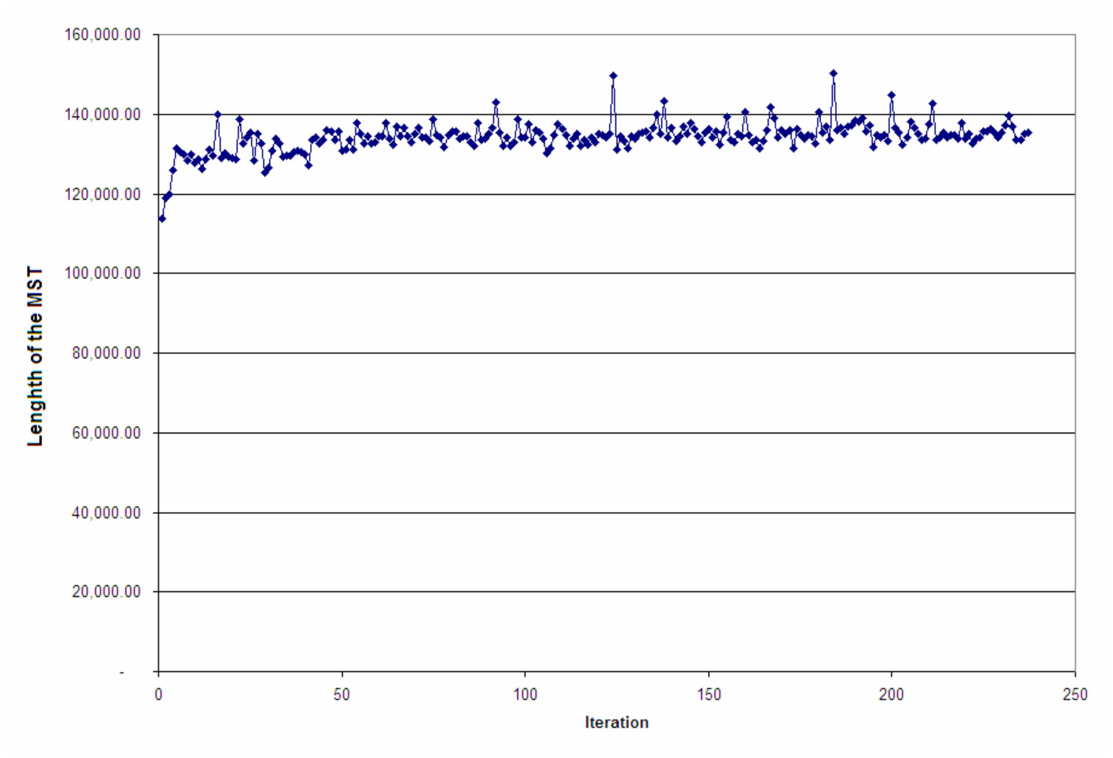


Figure 4.47 Length of the MST per iteration

This graph shows the how the MST length changes from one iteration to the next. Sometimes it increases and sometimes it decreases, the reason is due to the tradeoffs between other objectives in the problem. It seems like there is a tendency where the MST length increases slightly every iteration then suddenly drops, one might think that new parcels enter the solution as others are discarded while the algorithm goes through the steps.

Another idea is to identify the parcels selected for development at each iteration up to the point where the number of iterations added drops below a threshold (maybe 10) and then decide on the size of the radius to use.

We took the number of cycles identified over the 100 iterations and we discovered that some nodes appear more often than others, so we listed the nodes with high level of appearance. Table 4.4 contains the nodes with highest frequency.

Node ID	Frequency	Node ID	Frequency	Node ID	Frequency
1722	62	1538	17	1545	14
1721	61	1811	17	1808	13
1723	60	1540	16	332	12
1720	59	1547	16	1552	11
1725	50	1546	15	1687	11
1724	46	1756	15	1772	11
1542	20	1770	15	1543	10
1548	20	1771	15	1755	10
1541	18	213	14	1809	10

Table 4.4 List of nodes with 10 or more appearances in cycles up to iteration 150

Node 1722 is on top of the list, so this node tends to be selected for development. Due to the location of other nodes there is a tendency to form cycles. Figure 4.48 shows the neighborhood of node 1721. We note that there is a group of 6 nodes that are close together, they could form a fully connected network. A speedup strategy could be to include all cycle breaking constraints required for the 6 nodes network rather than iteratively add constraints as the cycles are found.

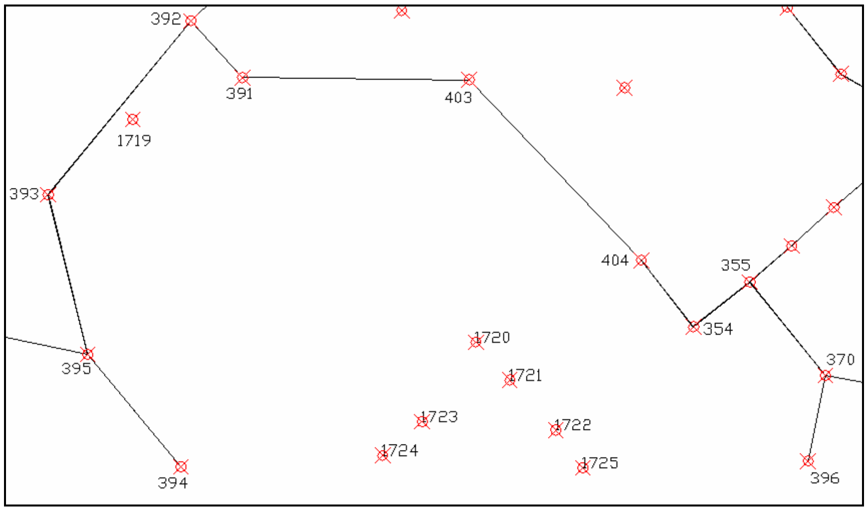


Figure 4.48 Neighborhood of node 1721

We note how there is a group of nodes close together all available for development. This group of nodes is delaying considerably the speed of the algorithm because of their proximity. To accelerate the procedure we could insert cycle breaking constraints for this group of six nodes.

A far more difficult situation comes in the neighborhood of node 1542 presented in Figure 4.49, there are 22 parcels available for development all grouped close between them and relatively far from the existing infrastructure. Figure 4.50 shows the arcs that could potentially connect the nodes in the neighbor, we created a circle around the area and counted about 220 arcs inside and about 28 nodes. It is clear that the number of arcs and nodes creates a level of complexity that slows down the algorithm. It would not be practical to include all nodes and arcs due to the exponential number of constraints, but perhaps not all of the nodes are required to be included because the attributes of the parcels.

Figure 4.51 shows all arcs used in the problem. We can see that there are areas with a large concentration of arcs, and therefore a large concentration of cycles.

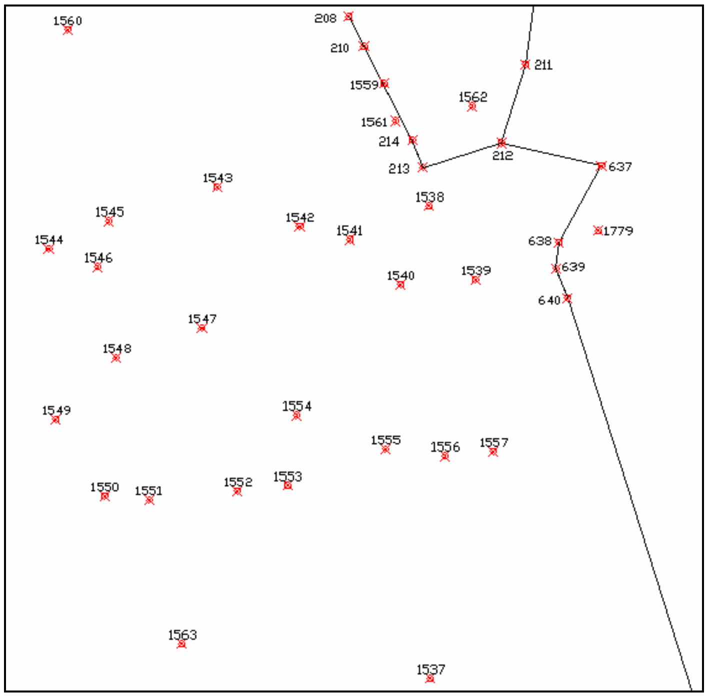


Figure 4.49 Neighborhood of node 1542

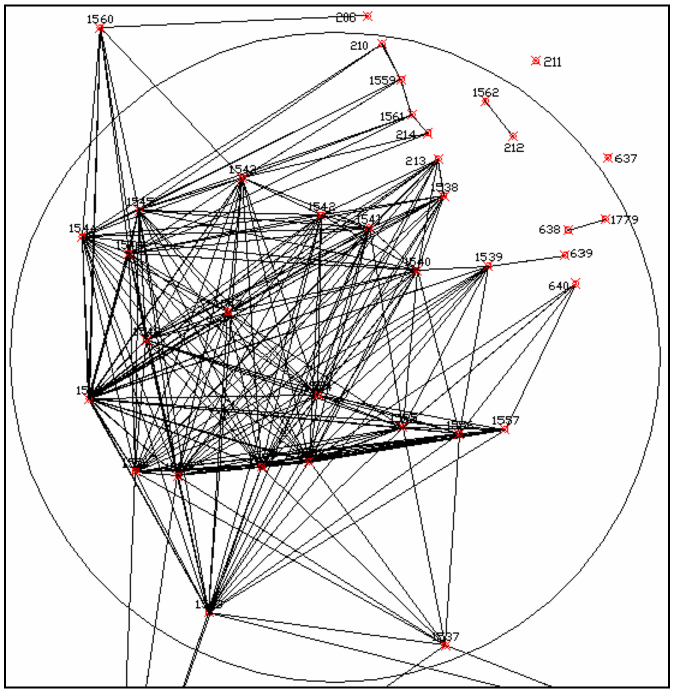


Figure 4.50 Potential arcs to connect nodes in the neighborhood of node 1542

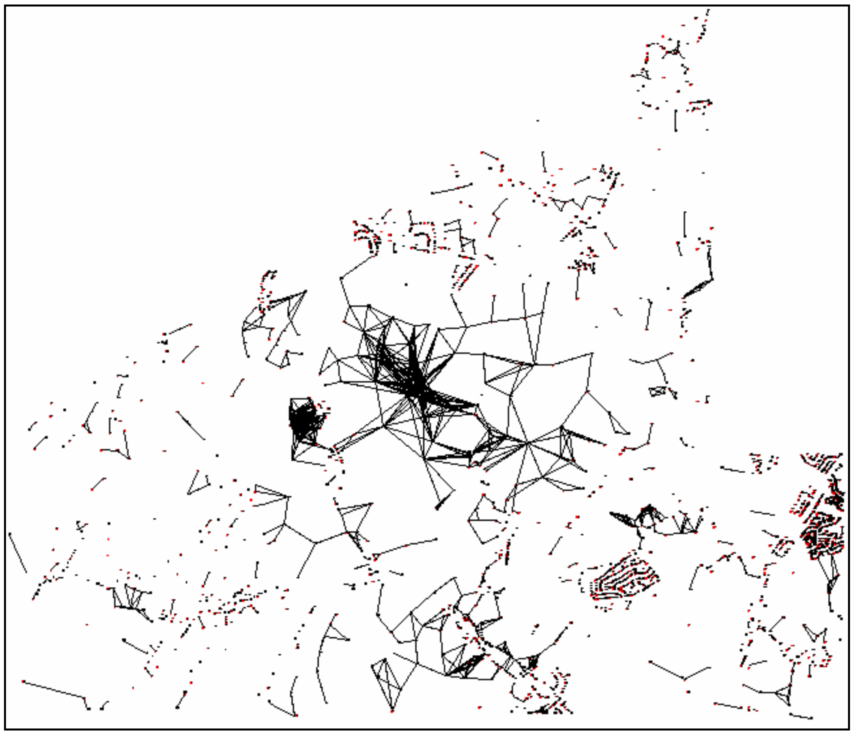


Figure 4.51 Network of potential arcs

Besides the neighbor of node 1542 there is another large area of concentration around node 1811 as shown in Figure 4.52. This creates an area of complexity due to the large number of edges eligible to enter the solution as shown in Figure 4.53.

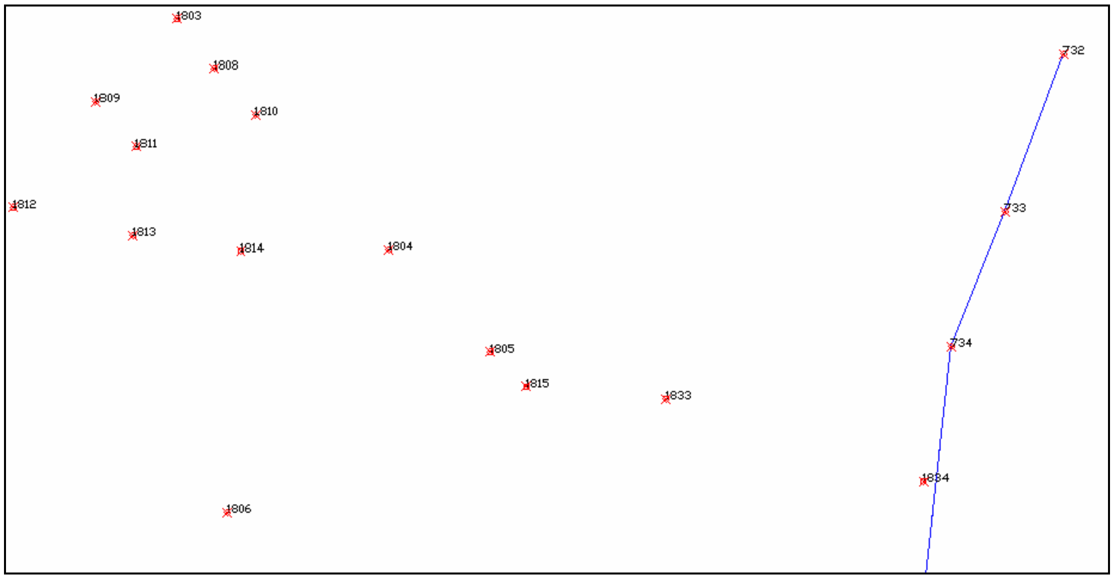


Figure 4.52 Nodes near node 1811

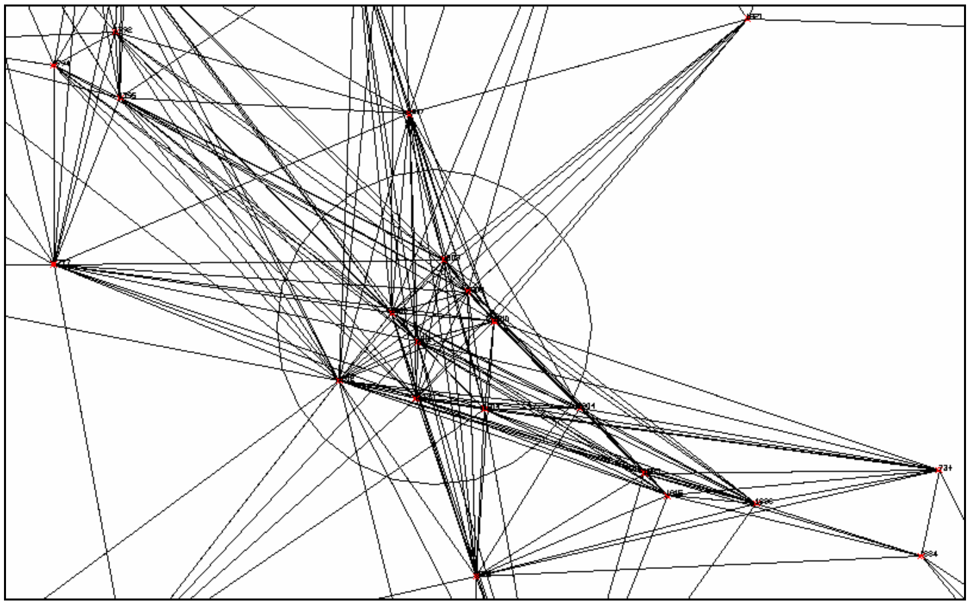


Figure 4.53 Potential arcs to connect nodes in the neighborhood of node 1811

Although the arcs in the neighborhood of node 1811 are quite large in length so perhaps they do not need to be all accounted for.

From the list of nodes in Table 4.4 there are 10 nodes in the neighborhood of node 1546 that appear frequently, for 10 nodes there are 637 constraints required to break the possible cycles 10 of which are already included in the initial formulation. This explains the slowness of the algorithm as it is currently implemented, at a pace of about 10 constraints per iteration it would take about 60 iterations to generate all constraints required for the neighborhood of node 1546. Given the hardware and software used it would take approximately 15 hours of computer time to complete 60 iterations. Figure 4.54 presents the solution obtained after the algorithm terminates.

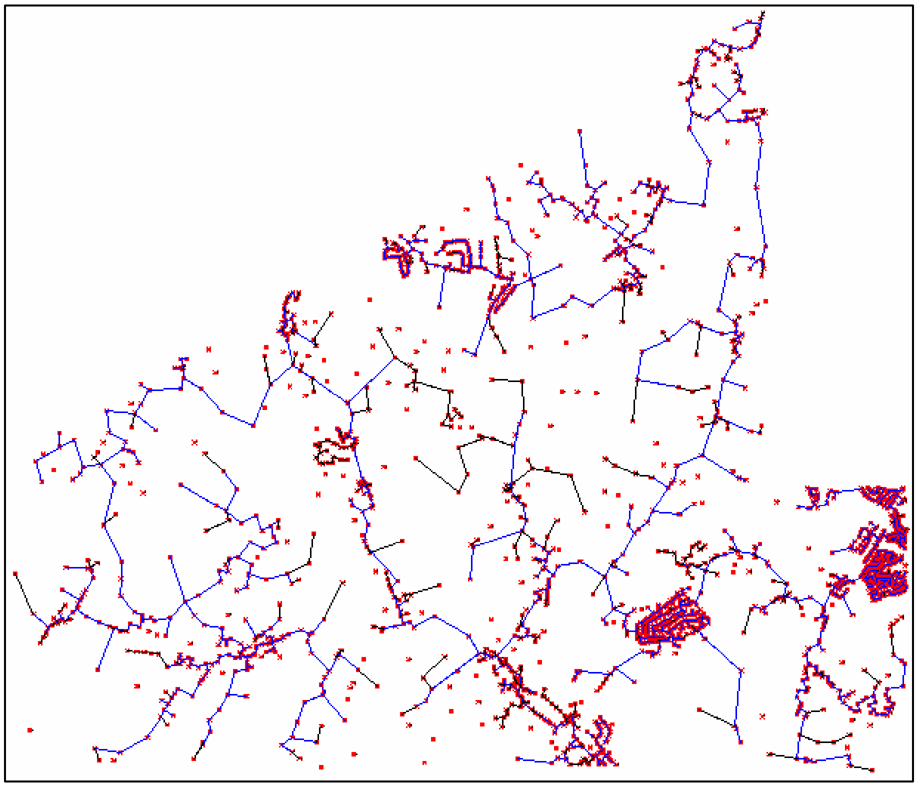


Figure 4.54 Solution obtained using set of all parcels.

Due to the scale it is difficult to differentiate the parcels selected for development from those that are not. The available parcels are indicated by a red dot not connected to the MST, all developed parcels are connected to the MST solution.

4.3.7. Comparison of Results Between Models

In this section we proceed to compare the results obtained with the model using the MST as a compactness measure, against the model using the squared diagonal as a compactness measurement presented in the work of Gabriel, Faria and Moglen (2005). Although the sets of parcels for the two models are different, and the objectives in the formulations are different we compare the results in terms of the stakeholders' objectives and analyze the compactness of the results for both models.

In Gabriel, Faria and Moglen the compactness measure used was the squared value of the diagonal surrounding all developed parcels. When this measurement is compared to the MST we can immediately notice that the squared diagonal measure is determined by the four extreme parcels that define the rectangle. Any development inside the rectangle is not going to affect that measure of the compactness. In contrast, when the MST is used, all parcels contribute to the compactness measure so the final result should be that the parcels tend to be developed close to one another. Figure 4.55 presents the solution of both models in one picture taken in quadrant 3. The shaded parcels are those selected by the model when the square diameter is minimized, while the thick gray lines represent the arcs required to connect the parcels selected for development to the existing infrastructure. The inner rectangle represents the smallest rectangle that can be drawn around the developed parcels while the outer rectangle represents the area of the quadrant. We choose quadrant 3 because that was the quadrant with the most potential of savings from the point of view of compactness, this quadrant had the most difference between the inner rectangle and the outer rectangle.

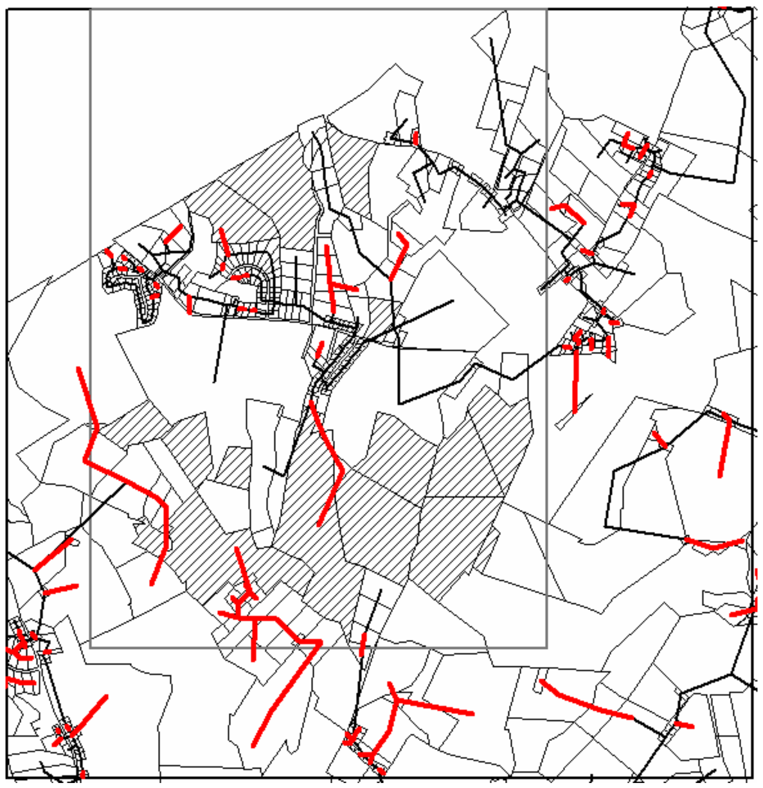


Figure 4.55 Overlapping of Solutions in Quadrant 3

It is clear by looking at the figure that the smallest rectangle method contained the development of the parcels within a reduced area much more effectively than the MST method. The MST method chooses smaller parcels, located closer together but overall dispersed over the region, while the smallest rectangle method instead chooses larger parcels within a compact area.

Figure 4.56 presents the solutions in quadrant 1. Again the parcels selected by the MST model fall outside the inner rectangle. However the difference between the two solutions is not as dramatic as in quadrant 3. As in the quadrant 3 case a larger number of smaller parcels were selected; this makes sense from the point of view of the MST measure since smaller parcels have smaller arcs required to connect them to the existing parcels, so it is natural for the model to favor small parcels closer to the existing infrastructure as opposed to large parcels separated.

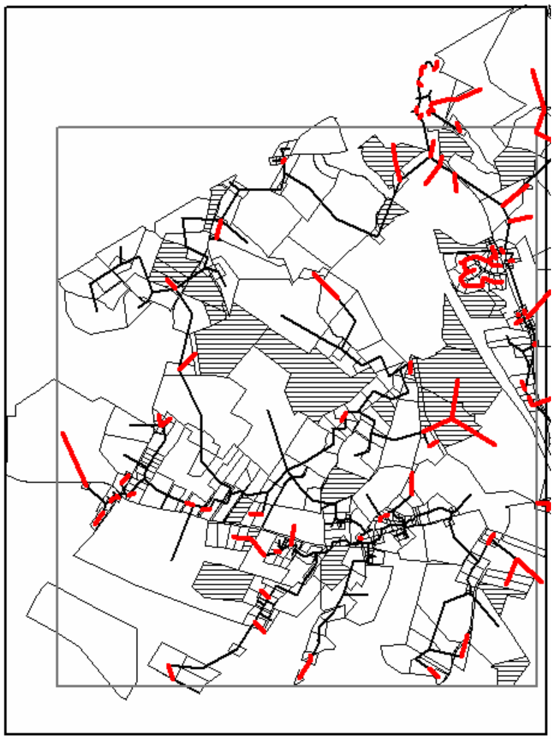


Figure 4.56 Overlapping of Solutions in Quadrant 1

Although these results are not strictly speaking comparable since they were obtained with different objectives, the conclusions can be generalized because it is expected that the MST will select smaller parcels located as close as possible to all developed parcels. Such parcels are probably distinct from the set of geographically proximate parcels that would be chosen by the squared diagonal measure.

4.3.8. Improvements to the Algorithm

Based on the result of the various test ran, we observed that the total number of constraints was not exponential. For example in the 100 node case with weight W=(1,1,1), the algorithm found a solution using 245 constraints. This is an indication that the constraints required was far from the number of constraints used by including all possible cuts. Therefore we could add all constraints required to break cycles on small group of nodes and save some iterations.

4.3.9. Future Work with MST

Although we have some ideas about other uses of the MST in this context, we have not pursued them due to different reasons. Some of those ideas and a brief explanation of the work we had done follows:

4.3.9.1. Minimization of the Maximum Diameter

In lieu of using the MST as the compactness measure, use instead the minimization of the maximum diameter of a tree that connects all parcels to be developed to the existing network. This might prove somewhat better since considering the two MST presented below in Figure 4.57

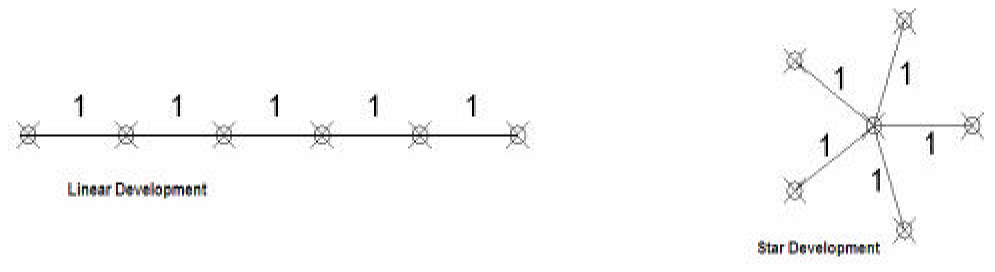


Figure 4.57 MST of five units on a linear fashion (left) and star fashion (right)

Most people would consider the star development arrangement shown on the right of Figure 4.57 to be more compact than the development on the left. The diameter of a tree is defined as the longest path between any pair of nodes in the tree. The tree show on the left has a diameter of five which is greater than the diameter of two found on the right figure.

The difficulty of such a measure is that there is no easy way to optimize this objective function since is a minimization of a maximization function of the form:

$$\text{minimize } D = \max\{P_{ij}\} \forall (i, j) \in E(G) \tag{4.36}$$

To solve a problem with (4.36) as one of the objective functions we would recommend using genetic algorithms.

4.3.10. Connectivity to Hubs

A point can be made that the new developments will not be connected to a neighbor parcel but rather to a hub located in some geographical point. The model as formulated can be easily changed to accomplish this, we would only need to change the input information of the existing MST to be the MST of the existing hubs.

4.3.11. Connectivity to Large Populated Cities

It is possible to use the concept of compactness as development around cities with large population density, in this case we propose to use as objective function the maximization of a normalized weighted sum of the distances from the available parcels to neighboring highly dense cities. The normalization formula could be one such as the following

$$ND_{ij} = \frac{\underset{ij}{Max}\{Dist_{ij}\} - Dist_{ij}}{Max\{Dist_{ij}\} - Min\{Dist_{ij}\}} \tag{4.37}$$

And then the objective function would be similar to:

$$\text{Maximize } \sum_i \sum_j Pop_j ND_{ij} d_i \tag{4.38}$$

Equation (4.37) would provide a normalized weight from each parcel i to each highly populated city j in such way that the closest parcels to the node j would have higher score (would be preferred as to other parcels further away). The term Pop_j represents the population of the city j. Objective function (4.38) would then try to prefer the development of parcels that are closer to highly populated cities.

This concept would require the use of Census data and relatively minor changes to the formulation presented in this work.

4.3.12. Use of Planar Graphs

Based on the work of Williams (2001) who developed an integer programming model to find a MST in a planar graph, we decided to test the concept with the MST setting, but found that some parcels cannot be selected for development unless another neighbor parcel is developed as well. This limitation is based on the characteristics of planar graphs which does not have any edges crossing. For example consider Figure 4.58 where for example node 7 can connect only to nodes 4, 5, 6, 8, 9, or 10. Otherwise the edge would cross another existing edge. Therefore, the model as currently envisioned cannot be implemented.

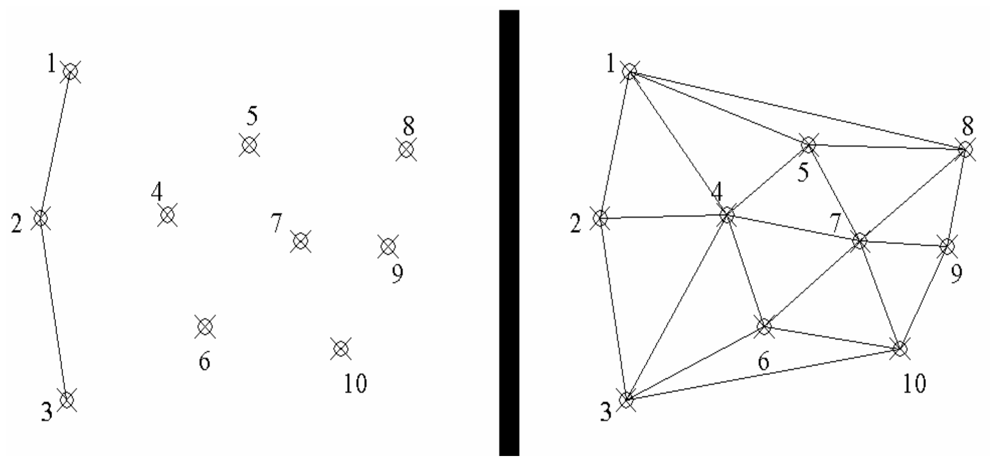


Figure 4.58 Existing MST connects nodes 1,2, and 3 other disconnected nodes (left) form a planar graph (right)

The desirables properties of the planar graph is that the formulation required to find a MST has the unimodularity property which ensures that the solution is integer. This property might be of interest if implemented into the formulation because it could be possible to apply decomposition techniques where the some of the sub-problems might have the total unimodularity property.

4.3.13. Cluster Analysis

Cluster analysis techniques are concerned with the grouping of items that present closely related characteristics. Gower and Ross (1969), Magnanti and Wolsey (1995) and Zahn (1971) are among others some of the researchers who have applied the MST concept to identify and analyze clusters. These concepts could be applied to the land development problem to decide upon the type of development to take place in the set of unassigned parcels. One possible objective function would be to generate one MST per zone using the existing developed parcels, and then minimize the MST resulting from connecting the parcels to those *z* trees instead of the connection to one existing MST as presented in this work.

Also, the clustering principle can be used by the Department of Planning office of a county or state to determine the zone types most convenient for parcels.

4.4. <u>Chapter Conclusions</u>

An innovative measure for compactness in land development is presented in this chapter along with a methodology to solve it. The minimum spanning tree has a long history in operations research. It has been studied since 1926 and many formulations have been created to solve it (see A 3.5). The work of this dissertation presents a novel approach to integrate the minimum spanning tree into a multiobjective optimization problem for land development accounting for the perspective of several stakeholders.

The problem of finding the MST typically involves a large formulation since the number of variables and constraints grow exponential to the number of nodes in the network, some researchers had already developed mechanisms to reduce the size of the problem, we presented a different approach since we consider the existence of a previous infrastructure in our analysis. By using a reduced (relaxed) formulation, which is solved and augmented by including additional inequalities that were violated by the solution of the previous iteration, a new solution is found and checked again continuing a procedure that stops when the solution is a tree (no cycles found and all parcels selected for development are connected).

An optimal solution has been found for small problems, for large ones we had to accept suboptimal solutions to the problem due to the time requirement to solve the iterations. Some techniques can be implemented to expedite the current procedure by looking ahead and include potential cycles in earlier iterations. There will be a tradeoff between the number of constraints actually required and the number of iterations performed. This work can be extended (and simplified) by assuming large hubs instead of individual previously developed parcels that would define the existing infrastructure. This simplification should improve computation time and reduce the number of variables required.

From the point of view of containing the development within certain boundaries the smallest rectangle model presented in Chapter 3 resulted more effective than the MST model presented in this section, however from the cost of infrastructure point of view the solution in the MST model should be less since the distances to the existing infrastructure are smaller.

Appendix 1 Numerical Results for Models

A 1.1. Solution of Difficult Weights

A 1.1.1. Weight 964- Original Bounds

This weight took the longest computation time, over 12 hours before the program stopped.

A 1.1.2. Lagrangian Relaxation

Applying the Lagrangian relaxation method described earlier to this case we obtained the following result:

Using u=(0,0) The solution is

PFA	IMP	ENV	PRO	z(u)	z
78.39008	2044.813	21.7481	1086.466	-1016.27	-1016.27

Table A.1 Lagrangian relaxation result weight 964 original bounds

This solution is feasible to the original problem, so it is optimal.

A 1.1.3. Dantzig-Wolfe Decomposition

After two iterations the Dantzig-Wolfe decomposition method provided the following solution,

Objective function: -1016.27

Note that this is the same solution reported by the solver as the “Best Solution” before the procedure halted, and it is consistent with the solution obtained by the Lagrangian relaxation.

A 1.1.4. Weight 662- Original Bounds

This weight took the second longest computation time, a little over 7 hours before the program stopped.

A 1.1.4.1. Lagrangian Relaxation:

Applying the Lagrangian relaxation method described earlier to this case we obtained the following result:

Using u=(0,0) The solution is

PFA	IMP	ENV	PRO	z(u)	z
79.01148	2059.857	21.7481	1085.822	-387.976	-1023.88

Table A.2 Lagrangian relaxation result weight 662 original bounds

This solution is feasible to the original problem, so it is optimal.

A 1.1.4.2. Dantzig-Wolfe Decomposition

Iteration 1: Objective function: - 978.237

Iteration 2: Objective function: - 1024.14

After two iterations the Dantzig-Wolfe decomposition method provided the optimal solution,

Note that this is the same solution reported by the solver as the "Best Solution" before the procedure halted, and it is better than the solution obtained by the Lagrangian relaxation.

A 1.1.5. Weight 178- Original Bounds

This weight took the longest computation time from those where the optimal solution was actually found, took a little over 2 hours before the optimal solution was found.

A 1.1.5.1. Lagrangian Relaxation:

Applying the Lagrangian relaxation method described earlier to this case we obtained the following result:

Using u=(0,0) the solution is

PFA	IMP	ENV	PRO	z(u)	z
77.65128	2030.104	21.0726	1086.897	-640.727	-640.727

Table A.3 Lagrangian relaxation result weight 178 original bounds

This solution is feasible to the original problem, so it is optimal.

A 1.1.5.2. Dantzig-Wolfe Decomposition

Iteration 1: Objective function: - 607.534

Iteration 2: Objective function: - 640.727

After two iterations the Dantzig-Wolfe decomposition method provided the optimal solution,

Note that this is the same solution reported by the solver as the optimal solution, and it is consistent with the solution obtained by the Lagrangian relaxation.

A 1.1.6. Weight 921- Original Bounds

This weight took a little under 2 hours before the optimal solution was found.

A 1.1.6.1. Lagrangian Relaxation

Applying the Lagrangian relaxation method described earlier to this case we obtained the following result:

Using u=(0,0) the solution is

PFA	IMP	ENV	PRO	z(u)	z
79.01148	2059.857	21.7481	1085.822	-387.976	-387.977

Table A.4 Lagrangian relaxation result weight 921 original bounds

This solution is feasible to the original problem, so it is optimal.

A 1.1.7. Weight 389 – Tightened Bounds Case 1

A 1.1.7.1. Lagrangian Relaxation

We started with weight $u = (0,0)$ and obtained a feasible solution (a solution within the lower and upper PFA bounds) so we have found the optimal solution.

IMP	ENV	PFA	PRO	z(u)	z	u1	u2	Feasible?
2045.17	21.07	78.27	1086.25	412.21	412.21	0.00	0.00	TRUE

Table A.5 Results of Lagrangian relaxation case 1 weight 389

We note that the solution obtained is between the lower and the upper bound reported by the branch and bound procedure.

A 1.1.7.2. Dantzig-Wolfe Decomposition

The procedure goes through three iterations and ends with the following result:

Objective function: 412.208, $l_{1,6} = 1$, $l_{2,6} = 1$, $l_{3,4} = 1$, $l_{4,7} = 0.984371$, $l_{5,6} = 1, p_1 = 0$, $p_2 = 0$, $m_1 = 45.2669$, $m_2 = 177.754$, $m_3 = 80.8921$, $m_4 = 76.1641$, $m_5 = 32.1313$

Since there are no fractional lambdas the problem is solved. The solution coincides with the Lagrangian relaxation.

A 1.1.8. Weight 176 – Tightened Bounds Case 1

A 1.1.8.1. Lagrangian Relaxation:

We started with weight m =(0,0) and obtained an infeasible solution. Since the PFA was under the lower bound we increased the value of m_1 until the lower bound was exceeded. Since all solutions will be feasible to the constrain limiting the development under the upper bound, we kept $m_2 = 0$ and varied only m_1. We started with step size of 0.1 until we got a feasible solution, then we applied a decreasing factor $r = 0.9$ obtaining the following results:

Iteration	IMP	ENV	PFA	PRO	z(u)	z	u1	u2	Feasible?
1	2004.05	1.9578	59.71	1029.29	75.86	75.86	0	0.00	FALSE
2	2004.05	1.9578	59.71	1029.29	75.83	75.86	0.1	0.00	FALSE
3	2004.05	1.9578	59.71	1029.29	75.80	75.86	0.2	0.00	FALSE
4	2004.05	1.9578	59.71	1029.29	75.77	75.86	0.3	0.00	FALSE
5	2044.35	20.6290	78.38	1085.12	76.51	69.16	0.4	0.00	TRUE
6	2044.35	20.6290	78.38	1085.12	75.77	69.16	0.36	0.00	TRUE
7	2004.05	1.9578	59.71	1029.29	75.76	75.86	0.324	0.00	FALSE

Table A.6 Results of Lagrangian relaxation case 1 weight 176

From the table we note that the distance between the upper bound found 75.77 (Iteration 6) and the best feasible solution found 69.16 (iteration 6) is quite large. The gap in this case is 8.73%.

When we used a bisection approach with a ten iteration limit, starting with 0.4 we obtained the following results.

Iteration	IMP	ENV	PFA	PRO	z(u)	z	u1	u2	Feasible?
1	2004.05	1.96	59.71	1029.29	75.86	75.86	0	0.00	FALSE
2	2004.05	1.96	59.71	1029.29	75.83	75.86	0.1	0.00	FALSE
3	2004.05	1.96	59.71	1029.29	75.80	75.86	0.2	0.00	FALSE
4	2004.05	1.96	59.71	1029.29	75.77	75.86	0.3	0.00	FALSE
5	2044.35	20.63	78.38	1085.12	76.51	69.16	0.4	0	TRUE
6	2004.05	1.96	59.71	1029.29	75.76	75.86	0.35	0	FALSE
7	2044.36	20.63	78.38	1085.12	76.05	69.16	0.375	0	TRUE
8	2044.35	20.63	78.38	1085.12	75.82	69.16	0.3625	0	TRUE
9	2004.05	1.96	59.71	1029.29	75.75	75.86	0.35625	0	FALSE
10	2044.35	20.63	78.38	1085.12	75.76	69.16	0.359375	0	TRUE

Table A.7 Results of the Lagrangian relaxation case 1 using Bisection

The bisection method performed slightly better, the best upper bound found was 75.76 and the best lower bound 69.15 for a relative gap of 8.72%. Since there was no real improvement using bisection as compared to the step procedure we kept using the step procedure to find the solution to the Lagrangian relaxation.

A 1.1.8.2. Dantzig-Wolfe Decomposition

The procedure goes through two iterations and ends with the following result:

Objective function: 75.754, $l_{1,6} = 1$, $l_{2,7} = 1$, $l_{3,4} = 1$, $l_{4,6} = 0.984371$, $l_{4,7} = 0.0156294$, $l_{5,6} = 1, p_1 = \text{-}0.358858$, $p_2 = 0$, $m_1 = 9.33512$, $m_2 = 35.5167$, $m_3 = 29.6239$, $m_4 = 14.4388$, $m_5 = 8.37099$

Since there are values of the lambdas fractional the problem is not completely solved. To obtain all lambdas binary we should use a branch and bound approach to eliminate the fractional values.

We first divide the problem into two sets, one with $l_{4,6} = 1$ and the other with $l_{4,6} = 0$ obtaining the following results:

$l_{4,6} = 1$	$l_{4,6} = 0$
Objective function: 2.27505	Objective function: 69.1584
$l_{1,6} = 1$	$l_{1,6} = 1$
$l_{2,5} = 0.547937$	$l_{2,7} = 1$
$l_{2,7} = 0.452063$	$l_{3,4} = 1$
$l_{3,4} = 1$	$l_{4,7} = 1$
$l_{4,6} = 1$	$l_{5,6} = 1$
$l_{5,5} = 1$	$p_1 = 0$
$p_1 = -488.224$	$p_2 = 0$
$p_2 = 0$	$m_1 = 7.76549$
$m_1 = 2143.23$	$m_2 = 31.3968$
$m_2 = 5636.4$	$m_3 = 20.317$
$m_3 = 12682.2$	$m_4 = 3.11374$
$m_4 = 15717$	$m_5 = 6.56531$
$m_5 = 2532.07$	

Table A.8 Branching for Dantzig-Wolfe decomposition case 1 weight 176

Since one of the solutions ($l_{4,6} = 0$) produces all binary values for the l vector then that branch is pruned by optimality, since the other branch ($l_{4,6} = 1$) has a maximum value of 2.27 it is pruned by bound. The best solution found is the bound 69.1584 which is the same bound obtained with the Lagrangian relaxation.

A 1.1.9. Weight 178 – Tightened Bounds Case 1

A 1.1.9.1. Branch and Bound

The branch and bound procedure obtained an optimal solution in a little over 2 hours as shown in Table A.9.

PFA	IMP	ENV	PRO	Obj. F.	Time	WID	Hrs
77.6513	2030.14	21.0726	1086.9	-640.727	7465.24	178	2.073678

Table A.9 Branch and bound result for case 1 weight 178

A 1.1.9.2. Lagrangian Relaxation

In just one iteration the Lagrangian relaxation provided the optimal solution.

PFA	IMP	ENV	PRO	z(u)	z	u1	u2	Feasible?
77.65	2030.10	21.07	1086.90	-640.73	-640.73	0.00	0.00	TRUE

Table A.10 Lagrangian relaxation result for case 1 weight 178

A 1.1.9.3. Dantzig-Wolfe

The procedure took three iterations before it arrived to the following solution:

Objective function: 640.727, $l_{1,6}=1$, $l_{2,6}=1$, $l_{3,4}=1$, $l_{4,6}=1$, $l_{5,6}=1$, $p_1=0$, $p_2=0$, $m_1=75.0204$, $m_2=289.553$, $m_3=121.21$, $m_4=104.211$, $m_5=50.7325$.

Since there are no fractional lambdas the problem is solved. The solution coincides with the Lagrangian relaxation.

A 1.1.10. Weight 459 – Tightened Bounds Case 1

A 1.1.10.1. Lagrangian Relaxation

The Lagrangian relaxation obtained a feasible solution on the first try (W=0,0) as follows:

PFA	IMP	ENV	PRO	z(u)	z
76.84	2015.06	21.07	1087.54	-883.56	-883.57

Table A.11 Lagrangian relaxation solution case 1 weight 459

A 1.1.10.2. Dantzig-Wolfe

After three iterations the procedure provided the following solution:

Objective function: 883.565, $l_{1,7} = 1$, $l_{2,6} = 1$, $l_{3,4} = 1$, $l_{4,6} = 1$, $l_{5,6} = 1$, $p_1 = 0$, $p_2 = 0$, $m_1 = 109.38$, $m_2 = 401.677$, $m_3 = 158.621$, $m_4 = 145.146$, $m_5 = 68.7407$.

Since there are no fractional lambdas the problem is solved. The solution coincides with the Lagrangian relaxation.

A 1.1.11. Weight 176- Tightened Bounds Case 2

A 1.1.11.1. Lagrangian Relaxation

We tried different values of *u* without finding any feasible solution. The result of the relaxation was switching between two infeasible solutions as

Iteration	IMP	ENV	PFA	PRO	z(u)	z	u1	u2	Feasible?
1	2004.05	1.96	59.71	1029.29	75.86	75.86	0	0.00	FALSE
2	1796.07	1.77	23.48	995.35	105.30	58.77	0	1.00	FALSE
3	2004.05	1.96	59.71	1029.29	85.86	75.86	1	1.00	FALSE
4	2044.35	20.63	78.38	1085.12	88.35	69.16	1.5	1.00	FALSE
5	2004.05	1.96	59.71	1029.29	86.79	75.86	1.35	1.10	FALSE
6	2044.35	20.63	78.38	1085.12	88.16	69.16	1.485	0.99	FALSE
7	2004.05	1.96	59.71	1029.29	86.68	75.86	1.3365	1.089	FALSE
8	2044.35	20.63	78.38	1085.12	87.97	69.16	1.47015	0.9801	FALSE
9	2004.05	1.96	59.71	1029.29	86.57	75.86	1.323135	1.07811	FALSE
10	2044.35	20.63	78.38	1085.12	87.78	69.16	1.455449	0.970299	FALSE

Table A.12 Lagrangian relaxation results weight 176 case 2

A 1.1.11.2. Dantzig-Wolfe

After four iterations we obtained the following values:

Objective function: 75.754, $l_{1,6}=1$, $l_{2,7}=1$, $l_{3,4}=1$, $l_{4,6}=0.984371$, $l_{4,7}=0.0156294$, $l_{5,6}=1$, $p_1=-0.358858$, $p_2=0$, $m_1=9.33512$, $m_2=35.5167$, $m_3=29.6239$, $m_4=14.4388$, $m_5=8.37099$.

A 1.1.12. Weight 643- Tightened Bounds Case 2

A 1.1.12.1. Lagrangian Relaxation

Once again, the Lagrangian relaxation was not able to find a feasible solution within 10 iterations. The solution kept switching between two non feasible solutions to the original problem.

Iteration	IMP	ENV	PFA	PRO	z(u)	z	u1	u2	Feasible?
1	2044.35	20.63	78.38	1085.12	112.33	112.33	0	0.00	FALSE
2	1832.65	1.77	24.33	1001.63	128.11	82.44	0	1.00	FALSE
3	1891.41	1.77	38.15	1033.77	121.53	95.05	0.1	0.90	FALSE
4	1988.59	1.77	58.41	1029.37	118.84	109.77	0.2	0.81	FALSE
5	2004.05	1.96	59.71	1029.29	117.95	110.53	0.3	0.73	FALSE
6	2004.05	1.96	59.71	1029.29	117.17	110.53	0.4	0.6561	FALSE
7	2044.36	20.63	78.38	1085.12	116.57	112.33	0.5	0.59049	FALSE
8	2004.05	1.96	59.71	1029.29	117.08	110.53	0.45	0.649539	FALSE
9	2044.36	20.63	78.38	1085.12	116.53	112.33	0.495	0.584585	FALSE
10	2004.05	1.96	59.71	1029.29	117.02	110.53	0.4455	0.643044	FALSE

Table A.13 Lagrangian relaxation results weight 643 case 2

A 1.1.12.2. Dantzig-Wolfe

After four iterations the procedure stopped at the following solution

Objective function: 111.52, $l_{1,6} = 1$, $l_{2,6} = 1$, $l_{3,4} = 1$, $l_{4,6} = 0.551214$, $l_{4,8} = 0.448786$, $l_{5,6} = 1$, $p_1 = 0$, $p_2 = 0.0962767$, $m_1 = 10.8096$, $m_2 = 41.5207$, $m_3 = 28.8211$, $m_4 = 14.6866$, $m_5 = 8.94299$.

A 1.1.13. Weight 724- Tightened Bounds Case 2

A 1.1.13.1. Lagrangian Relaxation

The Lagrangian relaxation approach failed to find a feasible solution within 10 iterations. The solution keeps switching between two infeasible solutions to the original problem.

Iteration	IMP	ENV	PFA	PRO	z(u)	z	u1	u2	Feasible?
1	1311.46	1.77	47.79	763.22	-36.74	-36.74	0	0.00	FALSE
2	1328.45	19.30	76.99	751.93	-34.48	-51.47	1	0.00	FALSE
3	1297.55	19.30	75.70	751.40	-36.70	-50.26	0.9	0.10	FALSE
4	1389.34	1.96	58.28	754.68	-38.52	-39.48	0.81	0.20	FALSE
5	1282.46	19.30	74.89	752.04	-37.28	-49.66	0.891	0.18	FALSE
6	1297.55	19.30	75.70	751.40	-35.79	-50.26	0.9801	0.162	FALSE
7	1282.46	19.30	74.89	752.04	-37.40	-49.66	0.88209	0.1782	FALSE
8	1389.34	1.96	58.28	754.68	-38.54	-39.48	0.793881	0.19602	FALSE
9	1282.46	19.30	74.89	752.04	-37.52	-49.66	0.873269	0.176418	FALSE
10	1389.34	1.96	58.28	754.68	-38.55	-39.48	0.785942	0.19406	FALSE

Table A.14 Lagrangian relaxation results weight 724 case 2

A 1.1.13.2. Dantzig-Wolfe Decomposition

After four iterations we obtained the following values:

Objective function: -40.5298, $l_{1,7} = 1$, $l_{2,7} = 1$, $l_{3,1} = 1$, $l_{4,7} = 0.103538$, $l_{4,8} = 0.896462$, $l_{5,7} = 1$, $p_1 =$ -0.613294, $p_2 = 0$, $m_1 =$-9.19256, $m_2 =$-1.55426, $m_3 =$ 8.15193, $m_4 =$-1.59139, $m_5 =$0.454139.

Since the values of lambda are fractional we need to apply branch and bound to find binary solutions. We obtained the following results:

$l_{4,8} = 1$	$l_{4,8} = 0$
Objective function: -338.042	Objective function: -40.5298
$l_{1,5} = 0.894116$	$l_{1,7} = 1$
$l_{1,7} = 0.105884$	$l_{2,7} = 1$
$l_{2,5} = 1$	$l_{3,1} = 1$
$l_{3,1} = 1$	$l_{4,7} = 0.103538$
$l_{4,8} = 1$	$l_{4,9} = 0.896462$
$l_{5,5} = 1$	$l_{5,7} = 1$
$p_1 = -260.398$	$p_1 = -0.613294$
$p_2 = 0$	$p_2 = 0$
$m_1 = 917.467$	$m_1 = -9.19256$
$m_2 = 2944.48$	$m_2 = -1.55426$
$m_3 = 6745.56$	$m_3 = 8.15193$
$m_4 = 8358.37$	$m_4 = -1.59139$
$m_5 = 1331.94$	$m_5 = 0.454139$

Table A.15 Results for Dantzig-Wolfe weight 724 case 2 first branch

Since both solution have fractional values we need to branch again on each one obtaining the following results.

$l_{4,8} = 1$	$l_{4,8} = 1$
$l_{1,5} = 1$	$l_{1,5} = 0$
Objective function: -349.08 $l_{1,5} = 1$	Infeasible

$l_{2,5} = 1$ $l_{3,1} = 1$ $l_{4,8} = 1$ $l_{5,5} = 0.54616$ $l_{5,7} = 0.45384$ $p_1 = -131.198$, $p_2 = 0$ $m_1 = 456.608$ $m_2 = 1436.8$ $m_3 = 3394.82$ $m_4 = 4199.34$ $m_5 = 657.523$	

Table A.16 Results for Dantzig-Wolfe weight 724 case 2 second branch

$l_{4,8} = 0$	$l_{4,8} = 0$
$l_{1,5} = 1$	$l_{1,5} = 0$
Objective function: -250.155 $l_{1,5} = 1$ $l_{2,7} = 1$ $l_{3,1} = 1$ $l_{4,7} = 0.0549682$ $l_{4,9} = 0.945032$ $l_{5,7} = 1$ $p_1 = -0.613294$ $p_2 = 0$ $m_1 = 0$ $m_2 = -1.55426$ $m_3 = 8.15193$ $m_4 = -1.59139$ $m_5 = 0.454139$	Objective function: -40.5298 $l_{1,7} = 1$ $l_{2,7} = 1$ $l_{3,1} = 1$ $l_{4,7} = 0.103538$ $l_{4,9} = 0.896462$ $l_{5,7} = 1$ $p_1 = -0.613294$ $p_2 = 0$ $m_1 = -9.19256$ $m_2 = -1.55426$ $m_3 = 8.15193$ $m_4 = -1.59139$ $m_5 = 0.454139$

Table A.17 Results for Dantzig-Wolfe weight 724 case 2 third branch

The optimal solution found is:

Objective function: -47.0496, $l_{1,6} = 1$, $l_{2,6} = 1$, $l_{3,1} = 1$, $l_{4,7} = 1$, $l_{5,1} = 1$, $p_1 = 0$, $p_2 = 0$, $m_1 = -11.3802$, $m_2 = -8.21401$, $m_3 = 12.6749$, $m_4 = -9.37907$, $m_5 = -1.55416$.

A 1.2. List of Weights Used to Find New Cases

Weight ID	IMP w1	ENV w2	PFA w3	PRO w4
1	0.871526	0.485054	0.147507	0.024457
2	0.929101	0.597805	0.56447	0.299908
3	0.172352	0.14218	0.050867	0.795779
4	0.643518	0.749261	0.568631	0.974561
5	0.357277	0.095527	0.39646	0.681209
6	0.40432	0.948522	0.850309	0.940149
7	0.618936	0.21963	0.684365	0.734583
8	0.649157	0.858379	0.72361	0.677511
9	0.224105	0.848818	0.036482	0.310137
10	0.889452	0.820314	0.677809	0.073912
11	0.347651	0.919449	0.83145	0.792619
12	0.591866	0.665722	0.195238	0.276977
13	0.202299	0.999514	0.719099	0.310625
14	0.543946	0.2944	0.028189	0.275954
15	0.331171	0.634428	0.177728	0.024844
16	0.079885	0.388387	0.539113	0.27257
17	0.502698	0.815553	0.139456	0.078516
18	0.520485	0.312005	0.50296	0.29955
19	0.201249	0.967271	0.965474	0.130321
20	0.332656	0.76364	0.828601	0.475267
……..	……..	……..	……..	……..
983	0.468824	0.542138	0.138855	0.145115
984	0.598395	0.329389	0.890222	0.038672
985	0.259766	0.671314	0.334407	0.485953
986	0.064853	0.791861	0.000946	0.620814
987	0.64908	0.540634	0.939127	0.868454
988	0.566762	0.045235	0.521804	0.80763
989	0.8458	0.612932	0.45743	0.18619
990	0.28172	0.668197	0.373642	0.149056
991	0.368481	0.509523	0.0197	0.723988
992	0.965993	0.978895	0.088428	0.437695
993	0.20147	0.845853	0.363693	0.036743
994	0.461205	0.172675	0.102678	0.742378
995	0.110919	0.792794	0.989794	0.078222
996	0.82453	0.007803	0.08482	0.245826
997	0.907274	0.899252	0.181329	0.74313
998	0.530632	0.490737	0.564156	0.198193
999	0.666129	0.51261	0.922694	0.096678
1000	0.699326	0.807198	0.914195	0.941198

Table A.18 Extract of the list of weights used to find new cases

A 1.3. Formulation to Find u in the Lagrangian Relaxation Example

Formulation

```
min eta
s.t.
eta - 5.5u >=  0
eta - 4.5u >=  2
eta - 3.5u >=  4
eta - 2.5u >=  6
eta - 1.5u >=  8
eta - 4.5u >=  3
eta - 3.5u >=  5
eta - 2.5u >=  7
eta - 1.5u >=  9
eta - 0.5u >= 11
eta - 3.5u >=  6
eta - 2.5u >=  8
eta - 1.5u >= 10
eta - 0.5u >= 12
eta + 0.5u >= 14
eta - 2.5u >=  9
eta - 1.5u >= 11
eta - 0.5u >= 13
eta + 0.5u >= 15
eta + 1.5u >= 17
eta - 1.5u >= 12
eta - 0.5u >= 14
eta + 0.5u >= 16
eta + 1.5u >= 18
eta + 2.5u >= 20
end
```

Solution

```
        OBJECTIVE FUNCTION VALUE

        1)      15.00000

  VARIABLE        VALUE          REDUCED COST
       ETA        15.000000           0.000000
         U         2.000000           0.000000


       ROW   SLACK OR SURPLUS     DUAL PRICES
        2)         4.000000           0.000000
        3)         4.000000           0.000000
        4)         4.000000           0.000000
        5)         4.000000           0.000000
        6)         4.000000           0.000000
        7)         3.000000           0.000000
        8)         3.000000           0.000000
        9)         3.000000           0.000000
       10)         3.000000           0.000000
       11)         3.000000           0.000000
       12)         2.000000           0.000000
       13)         2.000000           0.000000
       14)         2.000000           0.000000
       15)         2.000000           0.000000
       16)         2.000000           0.000000
       17)         1.000000           0.000000
       18)         1.000000           0.000000
       19)         1.000000           0.000000
       20)         1.000000           0.000000
       21)         1.000000           0.000000
       22)         0.000000           0.000000
       23)         0.000000          -0.500000
       24)         0.000000          -0.500000
       25)         0.000000           0.000000
       26)         0.000000           0.000000
```

A 1.4. Extract of the Solutions for the 1000 Weights Using Original Bounds

Weight ID	PFA	IMP	ENV	PRO	Obj. F.	Time in seconds
1	52.746	1083.69	174.035	764.393	1002.41	2.013
2	54.5145	1129.87	93.8476	763.799	846.025	2.634
3	55.7931	1643.67	131.847	1068.54	-551.124	0.26
4	51.5783	1405.85	26.0652	954.101	-34.9364	0.1
5	51.5783	1327.7	322.146	998.349	-195.401	0.13
6	63.8971	1519.98	31.8475	1011.34	-360.375	0.201
7	51.5783	1123.86	179.818	816.735	99.8317	0.12
8	54.1578	1201.43	25.0856	795.793	223.102	0.12
9	51.5783	1389.28	19.3033	931.728	36.8838	0.12
10	54.5145	1175	25.0856	761.867	972.42	0.641
11	63.8971	1519.98	31.8475	1011.34	-297.028	0.19
12	54.5145	1175	25.0856	761.867	490.478	1.142
13	63.8971	1429.13	19.3033	949.162	-32.3763	0.09
14	54.1578	1084.63	173.056	765.685	428.104	0.601
15	54.5145	1175	25.0856	761.867	376.424	0.851
16	67.1099	1571.57	26.0652	1017.96	-177.977	0.241
17	54.5145	1175	25.0856	761.867	543.706	1.642
18	54.5145	1129.87	93.8476	763.799	361.147	2.944
19	67.2085	1211.95	19.3033	756.826	99.057	0.18
20	61.7628	1411.24	19.3033	941.056	-14.2309	0.121
……	……..	……..	……..	……..	……..	……..
981	54.5145	1084.72	173.056	765.795	1008.32	0.531
982	54.5145	1185.89	25.0856	777.479	414.509	0.41
983	54.5145	1175	25.0856	761.867	446.338	1.012
984	54.5145	1084.72	173.056	765.795	627.947	0.43
985	53.6588	1421.91	19.3033	954.62	-99.5222	0.091
986	67.6249	1928.06	21.0726	1091.11	-535.711	0.831
987	61.7628	1361.88	26.0652	914.105	46.1994	0.12
988	51.5783	1249.7	350.915	952.687	-72.1771	0.12
989	54.5145	1175	25.0856	761.867	842.4	2.143
990	55.4357	1189.42	19.3033	761.076	213.827	1.132
991	51.5783	1462.96	31.8475	990.497	-162.823	0.09
992	54.1578	1174.91	25.0856	761.756	821.3	2.043
993	65.0742	1204.98	19.3033	761.141	207.462	0.26
994	51.5783	1292.23	267.45	969.425	-82.8082	0.141
995	71.5404	1241.22	19.3033	754.919	23.1166	0.23
996	60.8135	1052.32	343.533	764.06	677.364	1.031
997	54.5145	1185.89	25.0856	777.479	510.835	0.411
998	54.5145	1175	25.0856	761.867	454.051	1.212
999	54.5145	1175	25.0856	761.867	671.602	1.232
1000	61.7628	1361.88	26.0652	914.105	56.6207	0.13

Table A.19 Extract of the solutions for the 1000 Weights Using Original Bounds

A 1.5. Extract of the Pareto Optimal Solutions Using Original Bounds

Weight ID	IMP	ENV	PFA	PRO	Obj. F.	Time
3	1643.67	131.85	55.79	1068.54	-551.12	0.2600
9	1389.28	19.30	51.58	931.73	36.88	0.1200
20	1411.24	19.30	61.76	941.06	-14.23	0.1210
26	1432.27	26.07	51.58	968.75	-207.04	0.0900
38	1223.56	19.30	53.66	805.08	68.60	0.1200
44	1846.58	26.07	65.32	1084.88	-659.68	0.5900
45	1969.72	14.99	49.06	1086.49	-347.50	0.9320
58	1790.90	26.07	65.32	1074.53	-652.80	0.2700
60	1910.07	21.07	67.62	1088.33	-427.55	0.4010
62	1409.96	179.82	55.79	1011.09	-192.92	0.1300
63	1809.65	26.07	65.32	1078.19	-677.88	0.2710
67	1113.27	174.04	54.16	802.65	433.48	0.1100
69	1536.68	26.07	56.71	1014.30	-519.64	0.1400
70	1203.55	26.07	54.16	798.72	255.80	0.3010
74	1514.28	131.85	63.90	1032.71	-335.62	0.2800
75	1140.77	93.85	54.51	779.41	373.66	0.2200
79	1105.10	322.15	51.58	834.08	148.37	0.1100
80	2000.01	21.07	75.62	1088.18	-667.56	2.2530
…..	…..	…..	…..	…..	…..	…..
965	1201.43	25.09	54.16	795.79	391.49	0.4610
967	1190.88	25.09	64.15	762.39	133.80	0.4110
971	1224.19	19.30	51.58	805.96	125.44	0.1200
972	1123.86	179.82	51.58	816.74	168.35	0.1200
973	1464.00	26.07	63.90	975.65	-195.06	0.2000
974	1420.82	19.30	51.58	954.41	12.95	0.1100
976	1469.92	398.99	53.71	1063.75	-582.11	0.1700
977	1425.57	19.30	64.40	945.50	-6.63	0.1310
978	1069.85	212.22	54.51	766.69	877.30	0.8310
979	1054.16	267.89	52.75	766.25	750.90	1.3720
980	1410.46	277.85	63.90	1023.53	-335.67	0.1100
984	1084.72	173.06	54.51	765.80	627.95	0.4300
985	1421.91	19.30	53.66	954.62	-99.52	0.0910
986	1928.06	21.07	67.62	1091.11	-535.71	0.8310
987	1361.88	26.07	61.76	914.11	46.20	0.1200
988	1249.70	350.92	51.58	952.69	-72.18	0.1200
990	1189.42	19.30	55.44	761.08	213.83	1.1320
991	1462.96	31.85	51.58	990.50	-162.82	0.0900
992	1174.91	25.09	54.16	761.76	821.30	2.0430
993	1204.98	19.30	65.07	761.14	207.46	0.2600
994	1292.23	267.45	51.58	969.43	-82.81	0.1410
995	1241.22	19.30	71.54	754.92	23.12	0.2300
996	1052.32	343.53	60.81	764.06	677.36	1.0310
997	1185.89	25.09	54.51	777.48	510.84	0.4110
999	1175.00	25.09	54.51	761.87	671.60	1.2320

Table A.20 Extract of the Pareto Optimal Solutions for the 1000 Weights Using Original Bounds

A 1.6. Value Path Graph for Pareto Optimal Points

The values from Table A.20 were normalized in the scale 0-1 with 1 being the most desirable solution (the one that either maximizes or minimizes the objective) and 0 the less desirable. Those points were plotted and joined with lines and presented in Figure A.59. This information is of very little help since the number of solutions is large.

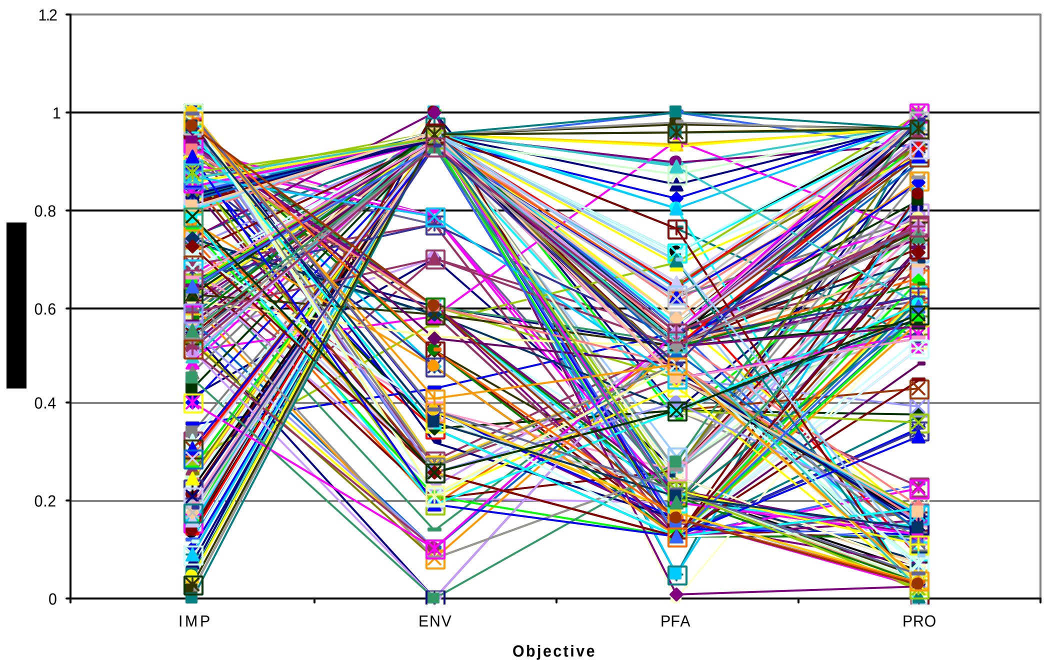


Figure A.59 Value Path Graph for Pareto Optimal Points

A 1.7. Extract of the Solutions Using Tighter Bounds Case 1

Weight ID	IMP	ENV	PFA	PRO	Obj. F.	Time
1	1,098.39	174.04	60.85	764.08	1,014.03	0.7610
2	1,145.43	93.85	64.15	763.87	855.02	1.1720
3	1,661.05	131.85	63.90	1,071.24	-550.69	0.6010
4	1,394.78	26.07	61.76	938.36	-32.51	0.1000
5	1,365.09	311.76	61.76	1,007.89	-193.57	0.1500
6	1,519.98	31.85	63.90	1,011.34	-360.38	0.2700
7	1,143.85	179.82	60.18	823.93	101.04	0.1510
8	1,224.03	25.09	62.26	806.00	224.99	0.1500
9	1,411.24	19.30	61.76	941.06	38.54	0.1500
10	1,190.56	25.09	64.15	761.93	979.72	0.6910
11	1,519.98	31.85	63.90	1,011.34	-297.03	0.2400
12	1,189.69	25.09	62.62	761.56	497.68	1.0620
13	1,429.13	19.30	63.90	949.16	-32.38	0.1100
14	1,099.32	173.06	62.26	765.38	435.96	0.9710
15	1,190.31	25.09	63.84	761.60	379.84	0.6910
16	1,571.57	26.07	67.11	1,017.96	-177.98	0.3810
17	1,189.69	25.09	62.62	761.56	549.99	1.7830
18	1,145.43	93.85	64.15	763.87	364.38	1.1910
19	1,211.95	19.30	67.21	756.83	99.06	0.1910
20	1,411.24	19.30	61.76	941.06	-14.23	0.1300
…..	…..	…..	…..	…..	…..	…..
981	1,084.75	201.83	62.62	765.84	1,013.70	1.1720
982	1,208.49	25.09	62.62	787.69	423.48	0.2200
983	1,189.69	25.09	62.62	761.56	452.15	0.6710
984	1,100.28	173.06	64.15	765.86	628.68	0.3110
985	1,439.28	19.30	61.76	957.32	-99.03	0.2400
986	1,928.06	21.07	67.62	1,091.11	-535.71	1.5620
987	1,361.88	26.07	61.76	914.11	46.20	0.1400
988	1,269.69	350.92	60.18	959.88	-71.14	0.1410
989	1,190.31	25.09	63.84	761.60	851.14	1.0210
990	1,204.98	19.30	65.07	761.14	214.60	0.7710
991	1,503.04	31.85	61.76	1,003.88	-157.94	0.2200
992	1,189.60	25.09	62.26	761.45	834.92	1.4020
993	1,204.98	19.30	65.07	761.14	207.46	0.2910
994	1,298.70	228.29	60.18	957.21	-78.40	0.1200
995	1,241.22	19.30	71.54	754.92	23.12	0.2200
996	1,052.32	343.53	60.81	764.06	677.36	0.3910
997	1,208.49	25.09	62.62	787.69	522.28	0.3710
998	1,190.56	25.09	64.15	761.93	456.86	0.9210
999	1,190.56	25.09	64.15	761.93	673.07	0.5310
1000	1,361.88	26.07	61.76	914.11	56.62	0.1300

Table A.21 Extract of the solutions for the 1000 Weights Tighter Bounds Case 1

A 1.8. Extract of the Pareto Optimal Set Using Tighter Bounds Case 1

Weight ID	IMP	ENV	PFA	PRO	Obj. F.	Time
1	1,098.39	174.04	60.85	764.08	1,014.03	0.7610
3	1,661.05	131.85	63.90	1,071.24	-550.69	0.6010
9	1,411.24	19.30	61.76	941.06	38.54	0.1500
14	1,099.32	173.06	62.26	765.38	435.96	0.9710
26	1,472.35	26.07	61.76	982.12	-203.07	0.2210
44	1,846.58	26.07	65.32	1,084.88	-659.68	0.6310
58	1,790.90	26.07	65.32	1,074.53	-652.80	0.2700
60	1,910.07	21.07	67.62	1,088.33	-427.55	0.3910
63	1,809.65	26.07	65.32	1,078.19	-677.88	0.2800
64	1,144.57	93.85	62.62	763.49	563.53	1.5320
67	1,135.87	174.04	62.26	812.86	443.56	0.1300
70	1,226.15	26.07	62.26	808.93	258.01	0.3000
74	1,514.28	131.85	63.90	1,032.71	-335.62	0.2810
75	1,145.75	93.85	64.15	764.32	380.42	1.8220
80	2,000.01	21.07	75.62	1,088.18	-667.56	2.3830
82	1,203.87	19.30	62.93	760.44	268.43	1.7830
83	1,395.91	26.07	61.76	939.07	-97.50	0.1000
86	1,243.55	19.30	62.26	812.27	24.95	0.1310
88	1,439.54	19.30	61.76	957.44	-167.71	0.2800
90	1,175.88	93.85	62.62	804.41	456.95	0.1500
……	……	……	……	……	……	……
976	1,484.60	388.60	61.82	1,062.60	-578.87	0.3700
977	1,425.57	19.30	64.40	945.50	-6.63	0.1300
979	1,054.16	316.36	60.85	766.25	757.47	0.5910
980	1,410.46	277.85	63.90	1,023.53	-335.67	0.1100
981	1,084.75	201.83	62.62	765.84	1,013.70	1.1720
983	1,189.69	25.09	62.62	761.56	452.15	0.6710
984	1,100.28	173.06	64.15	765.86	628.68	0.3110
985	1,439.28	19.30	61.76	957.32	-99.03	0.2400
986	1,928.06	21.07	67.62	1,091.11	-535.71	1.5620
987	1,361.88	26.07	61.76	914.11	46.20	0.1400
988	1,269.69	350.92	60.18	959.88	-71.14	0.1410
989	1,190.31	25.09	63.84	761.60	851.14	1.0210
991	1,503.04	31.85	61.76	1,003.88	-157.94	0.2200
992	1,189.60	25.09	62.26	761.45	834.92	1.4020
993	1,204.98	19.30	65.07	761.14	207.46	0.2910
994	1,298.70	228.29	60.18	957.21	-78.40	0.1200
995	1,241.22	19.30	71.54	754.92	23.12	0.2200
996	1,052.32	343.53	60.81	764.06	677.36	0.3910
997	1,208.49	25.09	62.62	787.69	522.28	0.3710
999	1,190.56	25.09	64.15	761.93	673.07	0.5310

Table A.22 Extract of the Pareto optimal solutions for the 1000 weights using tighter bounds case 1

A 1.9. Extract of the Solutions Using Tighter Bounds Case 2

Weight ID	IMP	ENV	PFA	PRO	Obj . F.	Time in seconds
1	1098.39	174.035	60.8501	764.083	1014.03	0.761
2	1145.43	93.8476	64.153	763.865	855.021	1.152
3	1661.05	131.847	63.8971	1071.24	-550.688	0.511
4	1394.78	26.0652	61.7628	938.358	-32.5097	0.1
5	1365.09	311.756	61.7628	1007.89	-193.571	0.14
6	1519.98	31.8475	63.8971	1011.34	-360.375	0.21
7	1143.85	179.818	60.1814	823.929	101.036	0.13
8	1224.03	25.0856	62.2619	806.003	224.992	0.121
9	1411.24	19.3033	61.7628	941.056	38.5416	0.12
10	1190.56	25.0856	64.153	761.932	979.722	0.641
11	1519.98	31.8475	63.8971	1011.34	-297.028	0.2
12	1189.69	25.0856	62.6185	761.557	497.68	1.072
13	1429.13	19.3033	63.8971	949.162	-32.3763	0.09
14	1099.32	173.056	62.2619	765.375	435.955	0.991
15	1190.31	25.0856	63.839	761.598	379.844	0.631
16	1571.57	26.0652	67.1099	1017.96	-177.977	0.25
17	1189.69	25.0856	62.6185	761.557	549.987	1.423
18	1145.43	93.8476	64.153	763.865	364.379	1.151
19	1211.95	19.3033	67.2085	756.826	99.057	0.18
20	1411.24	19.3033	61.7628	941.056	-14.2309	0.121
…..	…..	…..	…..	…..	…..	…..
982	1208.49	25.0856	62.6185	787.688	423.476	0.22
983	1189.69	25.0856	62.6185	761.557	452.148	0.701
984	1100.28	173.056	64.153	765.861	628.675	0.31
985	1439.28	19.3033	61.7628	957.316	-99.0294	0.231
986	1928.06	21.0726	67.6249	1091.11	-535.711	1.562
987	1361.88	26.0652	61.7628	914.105	46.1994	0.13
988	1269.69	350.915	60.1814	959.881	-71.1434	0.13
989	1190.31	25.0856	63.839	761.598	851.135	0.952
990	1204.98	19.3033	65.0742	761.141	214.6	0.791
991	1503.04	31.8475	61.7628	1003.88	-157.941	0.2
992	1189.6	25.0856	62.2619	761.446	834.915	1.322
993	1204.98	19.3033	65.0742	761.141	207.462	0.25
994	1298.7	228.291	60.1814	957.209	-78.4017	0.121
995	1227.31	19.3033	69.5	756.641	23.4584	0.11
996	1052.32	343.533	60.8135	764.06	677.364	0.38
997	1208.49	25.0856	62.6185	787.688	522.284	0.371
998	1190.56	25.0856	64.153	761.932	456.857	0.931
999	1190.56	25.0856	64.153	761.932	673.068	0.541
1000	1361.88	26.0652	61.7628	914.105	56.6207	0.13

Table A.23 Extract of the solutions for 1000 weights tighter bounds case 2

A 1.10. Extract of the Pareto Optimal Set Tighter Bounds Case 2

Weight ID	IMP	ENV	PFA	PRO	Obj. F.	Time in seconds
1	1098.39	174.035	60.8501	764.083	1014.03	0.761
3	1661.05	131.847	63.8971	1071.24	-550.688	0.511
9	1411.24	19.3033	61.7628	941.056	38.5416	0.12
14	1099.32	173.056	62.2619	765.375	435.955	0.991
26	1472.35	26.0652	61.7628	982.124	-203.07	0.21
44	1846.58	26.0652	65.3173	1084.88	-659.684	0.601
58	1790.9	26.0652	65.3173	1074.53	-652.798	0.26
60	1910.07	21.0726	67.6249	1088.33	-427.548	0.391
63	1809.65	26.0652	65.3173	1078.19	-677.875	0.28
64	1144.57	93.8476	62.6185	763.489	563.529	1.562
67	1135.87	174.035	62.2619	812.86	443.557	0.12
70	1226.15	26.0652	62.2619	808.932	258.007	0.32
74	1514.28	131.847	63.8971	1032.71	-335.621	0.291
75	1145.75	93.8476	64.153	764.324	380.416	1.772
82	1203.87	19.3033	62.9305	760.435	268.426	1.882
83	1395.91	26.0652	61.7628	939.067	-97.5043	0.101
86	1243.55	19.3033	62.2619	812.274	24.9496	0.13
88	1439.54	19.3033	61.7628	957.444	-167.71	0.241
90	1175.88	93.8476	62.6185	804.405	456.949	0.15
92	1087.24	322.146	60.1814	809.193	101.413	0.12
…..	…..	…..	…..	…..	…..	…..
976	1484.6	388.603	61.8167	1062.6	-578.867	0.391
977	1425.57	19.3033	64.3961	945.503	-6.62552	0.13
979	1054.16	316.363	60.8501	766.254	757.47	0.581
980	1410.46	277.852	63.8971	1023.53	-335.669	0.11
981	1084.75	201.825	62.6185	765.837	1013.7	1.092
983	1189.69	25.0856	62.6185	761.557	452.148	0.701
984	1100.28	173.056	64.153	765.861	628.675	0.31
985	1439.28	19.3033	61.7628	957.316	-99.0294	0.231
986	1928.06	21.0726	67.6249	1091.11	-535.711	1.562
987	1361.88	26.0652	61.7628	914.105	46.1994	0.13
988	1269.69	350.915	60.1814	959.881	-71.1434	0.13
989	1190.31	25.0856	63.839	761.598	851.135	0.952
991	1503.04	31.8475	61.7628	1003.88	-157.941	0.2
992	1189.6	25.0856	62.2619	761.446	834.915	1.322
993	1204.98	19.3033	65.0742	761.141	207.462	0.25
994	1298.7	228.291	60.1814	957.209	-78.4017	0.121
995	1227.31	19.3033	69.5	756.641	23.4584	0.11
996	1052.32	343.533	60.8135	764.06	677.364	0.38
997	1208.49	25.0856	62.6185	787.688	522.284	0.371
999	1190.56	25.0856	64.153	761.932	673.068	0.541

Table A.24 Extract of the Pareto optimal solutions for the 1000 weights using tighter bounds case 2

A 1.11. Data Collected Iteration by Iteration Solving Embedded Minimum Spanning Tree Large Scale Parcel Set

	Add								Disconn.
	Ineq	Time sec	Var	Constr	MST	Profit	Imp Ch	Cycles	Elements
Iter 1	0	309.00	3,915.00	1,650.00	113,849.00	1,346,630.00	947.37	25	42
Iter 2	135	305.50	3,957.00	1,783.00	118,787.00	1,346,560.00	947.93	18	28
Iter 3	222	304.00	3,985.00	1,870.00	119,756.00	1,346,660.00	947.67	13	20
Iter 4	287	306.00	4,005.00	1,933.00	126,061.00	1,346,740.00	947.54	9	15
Iter 5	332	306.40	4,020.00	1,978.00	131,487.00	1,346,580.00	948.37	7	16
Iter 6	382	306.80	4,036.00	2,028.00	130,418.00	1,346,800.00	948.00	4	6
Iter 7	400	310.30	4,042.00	2,046.00	129,960.00	1,346,710.00	948.10	5	8
Iter 8	427	304.50	4,050.00	2,071.00	128,485.00	1,346,680.00	948.08	4	8
Iter 9	451	308.60	4,058.00	2,095.00	129,909.00	1,346,690.00	948.26	9	11
Iter 10	484	304.70	4,069.00	2,128.00	127,644.00	1,346,720.00	947.43	2	4
Iter 11	496	303.00	4,073.00	2,140.00	128,609.00	1,346,750.00	947.67	3	5
Iter 12	511	303.80	4,078.00	2,155.00	126,328.00	1,346,590.00	947.27	6	9
Iter 13	538	303.30	4,087.00	2,182.00	128,606.00	1,346,740.00	947.54	6	7
Iter 14	559	303.00	4,094.00	2,203.00	131,061.00	1,346,760.00	948.31	2	4
Iter 15	571	302.40	4,098.00	2,215.00	129,695.00	1,346,620.00	947.99	4	6
Iter 16	589	303.30	4,104.00	2,233.00	139,790.00	1,346,760.00	948.11	3	6
Iter 17	607	303.50	4,110.00	2,251.00	128,868.00	1,346,870.00	947.88	4	6
Iter 18	625	305.10	4,116.00	2,269.00	130,130.00	1,346,660.00	948.06	4	4
Iter 19	637	307.30	4,120.00	2,281.00	129,356.00	1,346,750.00	947.89	6	7
Iter 20	658	307.00	4,127.00	2,302.00	129,080.00	1,346,740.00	947.87	3	5
Iter 21	673	308.60	4,132.00	2,317.00	128,817.00	1,346,830.00	947.70	2	5
Iter 22	688	310.60	4,137.00	2,332.00	138,630.00	1,346,750.00	949.01	4	5
Iter 23	703	310.10	4,142.00	2,347.00	132,722.00	1,346,640.00	947.66	4	6
Iter 24	721	310.10	4,148.00	2,365.00	134,031.00	1,346,690.00	947.82	3	6
Iter 25	739	308.60	4,154.00	2,383.00	135,298.00	1,346,460.00	947.72	4	5
Iter 26	754	303.90	4,159.00	2,398.00	128,358.00	1,346,660.00	947.10	3	4
Iter 27	766	307.20	4,163.00	2,410.00	135,187.00	1,346,620.00	948.20	3	4
Iter 28	778	308.90	4,167.00	2,422.00	132,734.00	1,345,210.00	948.33	6	7
Iter 29	799	308.00	4,174.00	2,443.00	125,396.00	1,344,000.00	947.50	4	5
Iter 30	814	309.30	4,179.00	2,458.00	126,459.00	1,345,520.00	947.43	2	4
Iter 31	825	307.10	4,183.00	2,469.00	130,720.00	1,345,720.00	947.92	4	5
Iter 32	840	302.50	4,188.00	2,484.00	133,765.00	1,343,110.00	948.21	8	9
Iter 33	867	306.30	4,197.00	2,511.00	132,754.00	1,345,410.00	947.39	5	5
Iter 34	882	302.80	4,202.00	2,526.00	129,198.00	1,345,570.00	947.55	1	3
Iter 35	891	304.80	4,205.00	2,535.00	129,591.00	1,345,730.00	946.40	3	4
Iter 36	903	303.60	4,209.00	2,547.00	129,678.00	1,344,130.00	947.56	5	6
Iter 37	921	304.30	4,215.00	2,565.00	130,573.00	1,345,430.00	946.86	2	3
Iter 38	930	302.50	4,218.00	2,574.00	130,841.00	1,344,160.00	946.32	3	4
Iter 39	942	302.70	4,222.00	2,586.00	130,404.00	1,344,930.00	947.14	6	7
Iter 40	963	307.30	4,229.00	2,607.00	129,789.00	1,344,310.00	947.30	1	2
Iter 41	969	306.30	4,231.00	2,613.00	127,198.00	1,345,130.00	946.91	3	4
Iter 42	981	308.50	4,235.00	2,625.00	133,460.00	1,345,980.00	947.85	5	5
Iter 43	999	308.60	4,241.00	2,643.00	134,124.00	1,346,260.00	947.84	2	4
Iter 44	1013	309.80	4,245.00	2,657.00	132,681.00	1,346,780.00	948.13	3	5

Iter 45	1028	307.50	4,250.00	2,672.00	133,606.00	1,346,630.00	947.64	3	5
Iter 46	1043	305.20	4,255.00	2,687.00	136,018.00	1,346,390.00	947.65	4	6
Iter 47	1062	308.60	4,261.00	2,706.00	135,586.00	1,346,830.00	949.03	4	5
Iter 48	1078	313.30	4,266.00	2,722.00	133,654.00	1,346,710.00	947.83	2	3
Iter 49	1087	307.80	4,269.00	2,731.00	135,546.00	1,346,440.00	947.61	3	4
Iter 50	1099	308.20	4,273.00	2,743.00	130,784.00	1,346,450.00	947.79	3	4
Iter 51	1111	308.90	4,277.00	2,755.00	131,177.00	1,346,650.00	947.50	2	3
Iter 52	1120	306.10	4,280.00	2,764.00	133,560.00	1,346,480.00	947.60	4	5
Iter 53	1135	305.60	4,285.00	2,779.00	131,223.00	1,346,650.00	947.86	1	3
Iter 54	1145	307.10	4,288.00	2,789.00	137,718.00	1,346,650.00	947.51	1	3
Iter 55	1154	306.00	4,291.00	2,798.00	135,111.00	1,346,740.00	948.02	2	4
Iter 56	1166	307.70	4,295.00	2,810.00	132,482.00	1,346,490.00	947.27	3	5
Iter 57	1182	307.10	4,300.00	2,826.00	134,469.00	1,346,750.00	948.19	2	2
Iter 58	1188	306.60	4,302.00	2,832.00	132,740.00	1,346,710.00	947.97	1	1
Iter 59	1191	302.90	4,303.00	2,835.00	132,855.00	1,346,690.00	947.83	4	5
Iter 60	1206	302.60	4,308.00	2,850.00	134,582.00	1,346,680.00	948.61	3	3
Iter 61	1215	302.70	4,311.00	2,859.00	134,481.00	1,346,540.00	947.77	1	1
Iter 62	1218	302.90	4,312.00	2,862.00	137,775.00	1,346,230.00	948.48	4	5
Iter 63	1233	302.10	4,317.00	2,877.00	133,990.00	1,346,310.00	947.69	2	3
Iter 64	1242	303.20	4,320.00	2,886.00	132,377.00	1,346,110.00	946.87	1	2
Iter 65	1248	302.20	4,322.00	2,892.00	137,026.00	1,346,400.00	947.54	1	3
Iter 66	1257	306.70	4,325.00	2,901.00	134,548.00	1,346,440.00	947.08	4	5
Iter 67	1272	306.90	4,330.00	2,916.00	136,720.00	1,346,190.00	947.69	3	3
Iter 68	1281	305.10	4,333.00	2,925.00	134,585.00	1,346,410.00	947.72	2	4
Iter 69	1293	305.70	4,337.00	2,937.00	132,819.00	1,346,560.00	947.82	2	3
Iter 70	1304	302.80	4,340.00	2,947.00	135,062.00	1,346,460.00	947.71	1	3
Iter 71	1313	303.60	4,343.00	2,956.00	136,500.00	1,346,270.00	948.03	2	5
Iter 72	1329	304.30	4,348.00	2,972.00	134,011.00	1,346,270.00	946.99	2	2
Iter 73	1335	306.90	4,350.00	2,978.00	133,997.00	1,346,270.00	947.50	1	2
Iter 74	1341	305.30	4,352.00	2,984.00	133,277.00	1,346,530.00	947.67	2	4
Iter 75	1353	302.70	4,356.00	2,996.00	138,779.00	1,346,820.00	948.33	1	1
Iter 76	1356	302.40	4,357.00	2,999.00	134,737.00	1,346,550.00	947.38	1	2
Iter 77	1363	302.20	4,359.00	3,006.00	134,194.00	1,346,550.00	947.56	2	3
Iter 78	1372	302.90	4,362.00	3,015.00	131,819.00	1,346,710.00	947.35	3	3
Iter 79	1381	308.10	4,365.00	3,024.00	134,896.00	1,346,190.00	947.05	2	4
Iter 80	1394	309.50	4,369.00	3,037.00	135,769.00	1,346,220.00	947.88	2	3
Iter 81	1404	307.90	4,372.00	3,047.00	135,713.00	1,346,640.00	948.03	2	4
Iter 82	1416	308.60	4,376.00	3,059.00	133,775.00	1,345,960.00	947.38	2	3
Iter 83	1425	307.70	4,379.00	3,068.00	134,515.00	1,346,480.00	948.34	1	2
Iter 84	1431	305.40	4,381.00	3,074.00	134,402.00	1,346,670.00	947.66	2	3
Iter 85	1440	302.70	4,384.00	3,083.00	132,995.00	1,346,340.00	947.87	2	4
Iter 86	1453	308.00	4,388.00	3,096.00	132,107.00	1,346,480.00	947.44	1	1
Iter 87	1456	305.60	4,389.00	3,099.00	137,724.00	1,346,420.00	947.80	4	4
Iter 88	1471	307.30	4,393.00	3,112.00	133,662.00	1,346,530.00	948.29	1	2
Iter 89	1477	306.30	4,395.00	3,118.00	133,950.00	1,346,490.00	947.56	1	2
Iter 90	1483	306.30	4,397.00	3,124.00	135,176.00	1,346,380.00	947.45	2	4
Iter 91	1496	304.80	4,401.00	3,137.00	136,523.00	1,346,430.00	947.68	2	3
Iter 92	1505	307.90	4,404.00	3,146.00	143,053.00	1,346,580.00	948.66	3	3
Iter 93	1514	303.70	4,407.00	3,155.00	135,230.00	1,346,400.00	947.93	1	2
Iter 94	1520	307.40	4,409.00	3,161.00	132,161.00	1,346,620.00	947.57	6	4

Iter 95	1534	306.50	4,413.00	3,173.00	134,105.00	1,346,250.00	948.25	1	2
Iter 96	1541	305.50	4,415.00	3,180.00	131,873.00	1,346,800.00	947.76	1	2
Iter 97	1547	307.40	4,417.00	3,186.00	132,956.00	1,346,260.00	947.71	2	3
Iter 98	1556	303.50	4,420.00	3,195.00	138,586.00	1,346,230.00	948.26	3	4
Iter 99	1568	306.10	4,424.00	3,207.00	134,256.00	1,346,010.00	947.59	2	3
Iter 100	1577	304.40	4,427.00	3,216.00	134,211.00	1,346,680.00	947.25	2	2
Iter 101	1583	307.00	4,429.00	3,222.00	137,568.00	1,346,510.00	947.60	1	2
Iter 102	1589	308.30	4,431.00	3,228.00	132,992.00	1,346,660.00	947.66	1	1
Iter 103	1592	305.90	4,432.00	3,231.00	136,012.00	1,346,390.00	947.14	1	1
Iter 104	1595	304.90	4,433.00	3,234.00	135,406.00	1,346,420.00	947.54	1	2
Iter 105	1601	302.60	4,435.00	3,240.00	133,899.00	1,346,100.00	947.02	3	3
Iter 106	1610	304.80	4,438.00	3,249.00	130,286.00	1,345,650.00	947.23	1	1
Iter 107	1613	305.60	4,439.00	3,252.00	131,394.00	1,346,380.00	948.02	1	2
Iter 108	1619	303.10	4,441.00	3,258.00	134,892.00	1,346,500.00	947.37	1	3
Iter 109	1629	305.60	4,444.00	3,268.00	137,391.00	1,346,610.00	947.42	2	3
Iter 110	1638	305.40	4,447.00	3,277.00	136,315.00	1,346,460.00	948.83	1	2
Iter 111	1644	306.30	4,449.00	3,283.00	134,635.00	1,346,250.00	948.28	1	2
Iter 112	1650	308.40	4,451.00	3,289.00	131,899.00	1,346,420.00	947.59	1	1
Iter 113	1653	306.70	4,452.00	3,292.00	133,986.00	1,346,300.00	946.62	1	2
Iter 114	1659	307.20	4,454.00	3,298.00	134,986.00	1,346,580.00	947.54	1	1
Iter 115	1662	306.00	4,455.00	3,301.00	132,103.00	1,346,820.00	947.54	2	2
Iter 116	1668	304.00	4,457.00	3,307.00	133,452.00	1,346,230.00	947.35	3	5
Iter 117	1685	305.70	4,462.00	3,324.00	132,356.00	1,346,250.00	947.56	1	1
Iter 118	1688	307.90	4,463.00	3,327.00	134,001.00	1,346,200.00	947.29	2	2
Iter 119	1694	305.60	4,465.00	3,333.00	132,891.00	1,346,450.00	947.45	1	2
Iter 120	1700	303.90	4,467.00	3,339.00	134,961.00	1,346,460.00	947.77	2	3
Iter 121	1710	307.00	4,470.00	3,349.00	134,725.00	1,346,120.00	948.76	1	3
Iter 122	1719	305.30	4,473.00	3,358.00	134,227.00	1,346,460.00	947.85	1	2
Iter 123	1726	302.50	4,475.00	3,365.00	135,027.00	1,346,630.00	948.14	1	1
Iter 124	1729	302.30	4,476.00	3,368.00	149,630.00	1,346,640.00	948.37	2	2
Iter 125	1735	302.90	4,478.00	3,374.00	131,084.00	1,346,810.00	947.50	1	1
Iter 126	1738	302.80	4,479.00	3,377.00	134,560.00	1,346,290.00	947.75	3	4
Iter 127	1750	303.00	4,483.00	3,389.00	133,317.00	1,346,780.00	948.21	1	1
Iter 128	1753	304.60	4,484.00	3,392.00	131,555.00	1,346,650.00	948.02	1	1
Iter 129	1756	303.40	4,485.00	3,395.00	134,442.00	1,346,370.00	947.96	2	2
Iter 130	1762	303.50	4,487.00	3,401.00	133,893.00	1,346,260.00	947.94	2	2
Iter 131	1768	307.70	4,489.00	3,407.00	135,138.00	1,346,410.00	947.58	2	2
Iter 132	1774	307.00	4,491.00	3,413.00	135,496.00	1,346,550.00	948.32	1	3
Iter 133	1784	303.80	4,494.00	3,423.00	135,620.00	1,346,310.00	947.76	4	4
Iter 134	1796	306.30	4,498.00	3,435.00	134,191.00	1,346,640.00	947.84	1	1
Iter 135	1799	304.30	4,499.00	3,438.00	136,443.00	1,346,120.00	947.17	3	4
Iter 136	1811	308.20	4,503.00	3,450.00	139,942.00	1,346,480.00	947.58	3	4
Iter 137	1823	306.70	4,507.00	3,462.00	135,141.00	1,346,020.00	947.12	1	1
Iter 138	1826	306.70	4,508.00	3,465.00	143,183.00	1,346,550.00	948.71	2	2
Iter 139	1832	305.30	4,510.00	3,471.00	134,028.00	1,346,330.00	947.41	1	1
Iter 140	1835	307.00	4,511.00	3,474.00	136,525.00	1,346,710.00	947.75	2	4
Iter 141	1849	304.90	4,515.00	3,488.00	133,287.00	1,346,680.00	947.71	1	3
Iter 142	1859	307.10	4,518.00	3,498.00	134,593.00	1,346,030.00	947.29	2	2
Iter 143	1865	305.40	4,520.00	3,504.00	136,804.00	1,346,560.00	948.42	1	3
Iter 144	1875	305.00	4,523.00	3,514.00	135,068.00	1,346,610.00	947.26	1	2

Iter 145	1881	303.80	4,525.00	3,520.00	137,718.00	1,346,450.00	947.62	1	3
Iter 146	1890	306.20	4,528.00	3,529.00	136,174.00	1,346,240.00	947.63	1	2
Iter 147	1900	305.90	4,530.00	3,539.00	134,451.00	1,346,620.00	947.52	1	2
Iter 148	1906	304.10	4,532.00	3,545.00	132,991.00	1,346,510.00	947.28	1	2
Iter 149	1912	305.80	4,534.00	3,551.00	135,442.00	1,346,370.00	947.09	2	3
Iter 150	1921	305.50	4,537.00	3,560.00	136,198.00	1,346,490.00	948.05	4	4

Table A.25 Data collected iteration by iteration from solving using all parcels in Moglen, Gabriel, and Faria (2003)

Appendix 2 Optimization Methods

This section presents the theoretical framework required for the development of the formulations and solution methods presented in this work. The topics are:

- Single Objective Optimization
- Multiobjective Optimization and Solution Methods including
  - The Weighted Sum Method
  - The Constraint Method
  - The Weighted Metric Method
  - Goal Programming
  - Multiobjective Simplex
- Methods to Solve Integer Programming Problems, including
  - Branch and Bound
  - Lagrangian Relaxation
  - Dantzig-Wolfe Decomposition Method
  - Dantzig-Wolfe Algorithm for Integer Programming
  - Benders Decomposition (Not used but included for completeness)
- Duality Gap

Optimization models are mathematical representation of problems with an objective function that is either maximized or minimized (Nash and Sofer, 1996), programming problems are concerned with finding an efficient allocation of resources to either maximize or minimize an objective (Gass, 1985). Linear programming problems are optimization problems where the functions used are linear.

The optimization problems can be distinguished as either single objective or multiobjective, constrained or unconstrained, linear or nonlinear, stochastic or deterministic, among others distinctions.

The objective functions are mathematical expressions that measure values of interest to the decision makers. For example profits, cost, risk, loss, efficiency, etc. When only one of these objectives is considered at a time, then the problem is considered a single objective optimization problem, when more than one objective is considered simultaneously then it is called a multiobjective optimization problem.

If the variables in the optimization problem can take any value in the domain of the objective function then the problem is called unconstrained. On the other hand, there are additional restrictions that limit the values that these decision variables can take, then the problem is said to be constrained. The set of restrictions that limit the values of the decision variables are called constraints. Depending on the type of functions used to define the objectives and the constraints, the problem can be either linear or nonlinear. Linear problems are those whose objective function(s) and constraints are expressed as linear combinations of the decision variables and nonlinear problems are those whose either objective function(s) or at least one constraint is not a linear function of the decision variables.

Deterministic models are those whose coefficients or functions are known with certainty. By contrast, stochastic models consider some of the problem's data to be uncertain (Birge and Louveaux, 1997).

The distinction between these types of problems is important because in general the solution approach used to solve problems in one category does not necessarily carry over in other categories. The work presented in this dissertation does not consider unconstrained or stochastic optimization problems, and besides a case with a quadratic objective function, nonlinear problems in general are not considered either.

This chapter is intended to present the basic concepts from the theory of linear and integer programming optimization since those concepts are required for the algorithms presented in later chapters.

A 2.1. Single Objective Optimization

Linear programming was conceived in 1947 by George Dantzig. Although Fourier (1823), de la Vallee Poussin (1911), and Kantorovich (1939) produced work that suggest their authors were aware of the potential of linear programming. Prior to Dantzig the efforts in programming resources were mathematically studied but they lacked the concept of the objective function (Dantzig, 1982). The simplex method invented by Dantzig is a basic tool to solve practical problems of large complex systems (Dantzig, 1982).

The following equations are taken from Nash and Sofer (1996), to describe the solution method invented by Dantzig using vector-matrix notation.

The general linear programming problem can be written in standard form as:

Min: $c^T x$ (A.1)

subject to $Ax = b$ (A.2)

$x \geq 0$ (A.3)

with $b \geq 0$.

Here $c \in \mathbb{R}^n$ is a column vector of objective function coefficients, $b \in \mathbb{R}^m$ is a column vector of the "right hand side", and $A \in \mathbb{R}^{mxn}$ is the constraint matrix of coefficients, and $x \in \mathbb{R}^n$ is a column vector of decision variables (Nash and Sofer, 1996).

Associated with any linear programming there is another linear programming problem called the "dual problem" in which the roles of the variables and constraints are reversed (Nash and Sofer, 1996). The original linear programming problem is then called by association the primal. The dual problem to (A.1) - (A.3) can be written as:

Max: $b^T y$ (A.4)

subject to: $A^T y \geq c$ (A.5)

There are strong relationships between the primal and the dual problems. Three of which are presented below without proof. The interested reader is referred to Nash and Sofer (1996), Gass (1985), Winston (2004) for the proof of these theorems.

Theorem 2 Weak Duality

Let x be a feasible solution to the (primal) linear programming problem (A.1) - (A.3), and let y be as feasible solution to the (dual) linear programming problem (A.4) - (A.5). Then

$c^T x \geq b^T y$ *(A.6)*

Theorem 3 Strong Duality

Consider a pair of primal and dual linear programming problems. If one of the problems has an optimal solution then so does the other one, and the value of both objective functions is the same.

Let x^* be an optimal solution of the primal problem (A.1) - (A.3), then y^* is an optimal solution to the dual and:

$$c^T x^* = b^T y^* \tag{A.7}$$

Theorem 4 Complementary Slackness

Consider a primal problem written in standard form and the corresponding dual linear program. If x^* is optimal for the primal and y^* is optimal for the dual, then:

$$x^{*T}(c - A^T y) = 0 \tag{A.8}$$

If x is a feasible solution to the primal, and y is a feasible solution of the dual such that

$$x^T (c - A^T y) = 0 \tag{A.9}$$

Then x and y are optimal solutions to their respective problems (Nash and Sofer, 1996).

A 2.2. Multiobjective Optimization and Solution Methods

This section explains some of the different techniques that traditionally have been implemented in other multiobjective optimization settings. This section also briefly describes how those techniques can be used or combined as the foundation to what would be the algorithm to solve the land development planning problem (LDPP). This section is not intended to serve as an exhaustive list of techniques but rather to brief the reader on methods that are typically available to solve this type of problem.

Figure A.60 has been adapted from ReVelle and McGarity (1997) and shows the relative position of multiobjective optimization within the decision support methods in operations research.

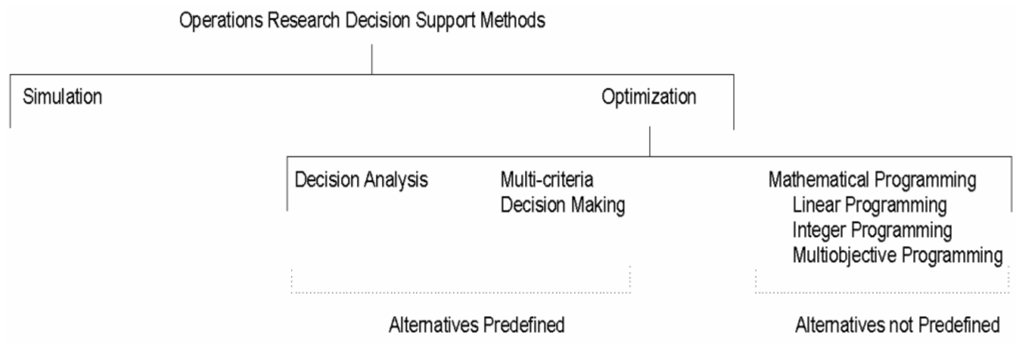

Operations Research Decision Support Methods
Simulation
Optimization
Decision Analysis
Multi-criteria Decision Making
Mathematical Programming
Linear Programming
Integer Programming
Multiobjective Programming
Alternatives Predefined
Alternatives not Predefined

Figure A.60 Hierarchy of decision support methods

Multiobjective methods are mathematical tools to solve problems with conflicting objectives. These problems arise in a wide variety of settings since there is typically more than one interest, objective or goal to pursue. A typical example is in manufacturing where there is a constant tradeoff between cost and quality. Typically better quality implies higher cost but then, the lower the cost the greater the profits. In a grocery store the relationship between customer satisfaction and the number of cashiers available is also conflicting since the greater the number of cashiers available, the higher the cost but also the higher the level of customer satisfaction. Every time there are two or more objectives to optimize in a problem there is an opportunity to implement multiobjective optimization techniques. The most interesting problems, and also the most challenging ones are those where the objectives are in conflict with one another, meaning the increase in one is obtained as a loss to another. The problem then becomes to balance these objectives to reach solutions that are considered satisfactory.

Cohon (1978) divides multiobjective decision making into preference and not preference-based. Figure A.61 has been adapted from his book.

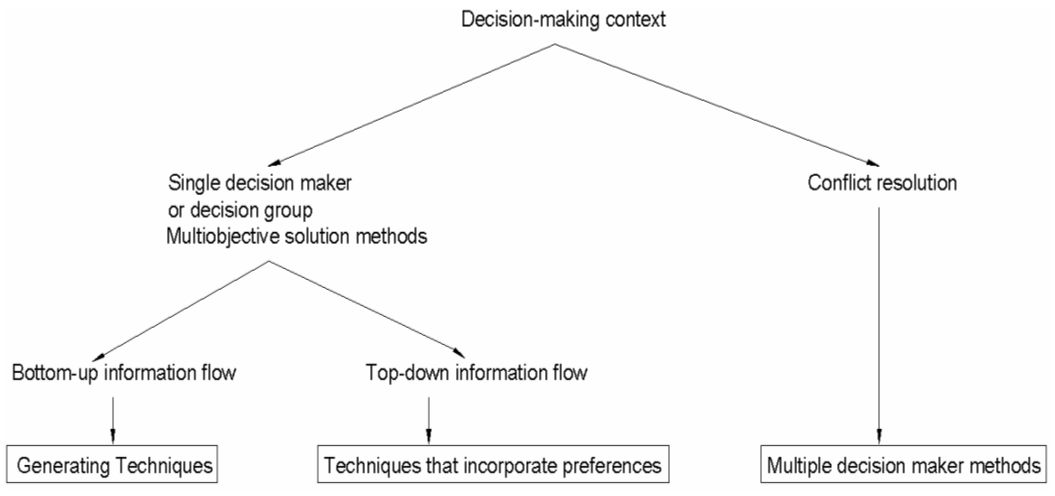

Decision-making context
Single decision maker
or decision group
Multiobjective solution methods
Conflict resolution
Bottom-up information flow
Top-down information flow
Generating Techniques
Techniques that incorporate preferences
Multiple decision maker methods

Figure A.61 Multiobjective solution methods classification by Cohon (1978)

What follows is a discussion of solution methods as presented in the literature. This is not an exhaustive discussion of all possible methods but a review of the most relevant ones. Consider the general multiobjective optimization problem typically formulated as follows:

$$\left.\begin{array}{ll}\text{Minimize: } f_m(x), m=1,2,\ldots,M & \\ \text{Subject to:} & g_j(x)\geq 0,\ j=1,2,\ldots,J \\ & h_k(x)=0, k=1,2,\ldots,K \\ & x_i^L \leq x_i \leq x_i^U, i=1,\ldots,n\end{array}\right\} \qquad (A.10)$$

There are *M* objectives subject to *J* inequalities, *K* equalities and the solution vector *x* has *n* components of which each one x_i, is bounded below and above by x_i^L and x_i^U, respectively. This general form can be used recognizing that a minimization problem can be posed as a maximization problem by multiplying the objective function by -1 so:

$$\min : f_u(x) \leftrightarrow \max : -f_u(x) \qquad (A.11)$$

And also by recognizing that any less than or equal to inequality can be converted into a greater than or equal to inequality by multiplying the constraint by -1

$$g_v(x) \geq 0 \leftrightarrow -g_v(x) \leq 0 \qquad \text{(A.12)}$$

A 2.2.1. The Weighted Sum Method

Known as the weighting method (Cohon, 1978), this method is based on combining all the M objectives into one objective by assigning each of the objective functions a weight w_m, and then solving a single objective optimization problem. The general multiobjective formulation becomes:

$$\left.\begin{array}{ll} \text{Minimize:} & \sum_{m=1}^{M} w_m f_m(x) \\ \text{Subject to:} & g_j(x) \geq 0,\ j = 1,2,\ldots,J \\ & h_k(x) = 0, k = 1,2,\ldots,K \\ & x_i^L \leq x_i \leq x_i^U, i = 1,\ldots,n \end{array}\right\} \qquad \text{(A.13)}$$

To obtain the Pareto optimal set[7] with this method, first all the weights w_m must be strictly positive (Miettinen, 1999; Deb, 2004; Steuer, 2004) since values of zero on the weights might produce weakly Pareto points, and negative weights would produce an opposite effect to the one desired (maximization instead of minimization or vice-versa). Also, the feasible region needs to be convex or there is a risk of missing Pareto optimal points around the non convex area[8] (Deb, 2004). In the case of the related Land Development Planning Problem, since the problem is a mixed

[7] Set of efficient solutions for which the improvement of one objective is only obtained by the diminishing of another objective.

[8] Specifically for integer programming problems where the feasible region is non convex.

integer programming problem, there is a risk of missing some Pareto optimal points as presented in ReVelle and McGarity (1997) due to a “duality gap”.

An analysis of the Pareto optimal set provides insight into the different tradeoffs to the stakeholders and the decision makers. Each one of the stakeholders can identify how their objective changes when other players gain or lo se weight in the evaluation process. This provides a useful negotiation tool to the individuals involved. The problem with this method is that there is little consensus on what the weights should be, moreover to determine the Pareto set a potentially large number of runs is required each one with a different weighting vector. For the land development problem, the identification of the complete Pareto optimal set is not a goal since at this point no decision on alternative developments are required. We know that this method can provide some of the Pareto optimal points, and we are interested in finding methods to solve each one of the optimization problems within reasonable times.

A 2.2.2. The Constraint Method

This method optimizes one objective while the other *M*-1 objectives are constrained to be not worse than a certain value.

The general formulation becomes:

$$\left.\begin{array}{ll} \text{Minimize: } f_u(x) & \\ \text{Subject to:} & f_m(x) \le e_m, m = 1,2,\ldots,M \setminus m = u \\ & g_j(x) \ge 0, j = 1,2,\ldots,J \\ & h_k(x) = 0, k = 1,2,\ldots,K \\ & x_i^L \le x_i \le x_i^U, i = 1,\ldots,n \end{array}\right\} \tag{A.14}$$

Where e_m is a limiting value for the *m* objective.

One of the striking problems with this method is feasibility. It is quite possible that the values e_m selected to constrain the objective functions, render the problem infeasible. Therefore, some caution needs to be taken in the selection of the e_m's values. These values need to be selected such that feasible solutions to the single objective problem exists (Cohon, 1978).

Note that the value of the e_m's are parameters in the optimization model and not decision variables. However, for a point to be in the Pareto optimal set it is required that all the constraints of the objectives should be binding at an optimal solution.

"If this is not the case and if there are alternative optima to the constrained problem, then some of these alternative optimal solutions might be inferior" (Cohon, 1978).

Cohon (1978) presented an algorithm that could be used to overcome the infeasibility problem by solving *M* single objective optimization problems obtaining the optimal solution to each objective. Then this information is used as a bound to preserve feasibility.

A 2.2.3. Weighted Metric Method

An ideal solution z^* would be one that simultaneously optimizes all the objectives. Such a solution although desirable is in most settings a point that lies outside of the feasible region. One fact worth noting is that the ideal solution z^* is in most of the cases infeasible since it is usually located outside of the feasible region. Consider Figure A.62 where z^* is the minimum value obtainable for objective functions 1 and 2 by themselves.

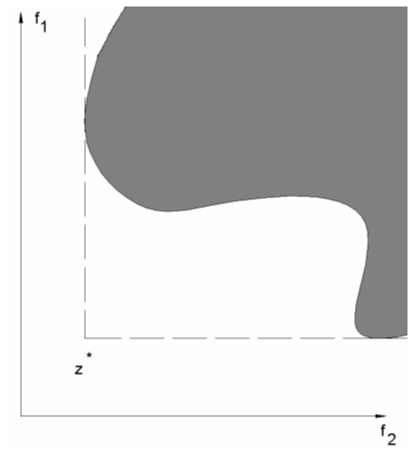


Figure A.62 Ideal solution to the two objective case with feasible region in gray

This next method is based on finding a solution with minimal distance to this ideal solution z^*.

The general formulation becomes:

$$\left.\begin{array}{ll} \text{Minimize:} & \left(\sum_{m=1}^{M} w_m \left| f_m(x) - z_m^* \right|^p \right)^{1/p} \\ \text{Subject to:} & g_j(x) \geq 0,\ j = 1,2,\ldots,J \\ & h_k(x) = 0,\ k = 1,2,\ldots,K \\ & x_i^L \leq x_i \leq x_i^U \end{array}\right\} \qquad \text{(A.15)}$$

As presented in Deb (2001), when $p = 1$ the problem is equivalent to the weighting method proposed before, when $p = 2$ the problem is to find the minimum Euclidean distance between the ideal solution and the solution provided by x. When a large p is used, the problem is to minimize the largest deviation to the ideal solution. This special case is also known as the weighted Tchebycheff problem, which can be written as:

$$\left.\begin{array}{ll} \text{Minimize:} & \max_{m=1}^{M}\left(w_m \left| f_m(x) - z_m^* \right|\right) \\ \text{Subject to:} & g_j(x) \geq 0,\ j = 1,2,\ldots,J \\ & h_k(x) = 0, k = 1,2,\ldots,K \\ & x_i^L \leq x_i \leq x_i^U \end{array}\right\} \qquad \text{(A.16)}$$

A 2.2.4. Goal Programming

To overcome the problem of infeasibility of the original problem, a goal programming technique can be used. Originally introduced by Charnes and Cooper (1961), this method provides a tool that guarantees feasibility to an augmented problem while seeking a solution as close as possible to the best values for each objective. Goal programming is based on the utilization of deviational variables. A weighted sum of the deviations becomes the objective function of the extended problem, and the objective functions of the original problem are now included as constraints that are functions of the original variables and the deviational variables.

Consider again the general formulation written in a slight different form:

$$\left\{\begin{array}{ll} \text{Minimize} : & f_m(x), m = 1,2,\ldots,M \\ Subject\ to : & x \in S \end{array}\right\} \qquad \text{(A.17)}$$

$$S = \begin{cases} g_j(x) \geq 0,\ j = 1,2,\ldots,J \\ h_k(x) = 0, k = 1,2,\ldots,K \\ x_i^L \leq x_i \leq x_i^U \end{cases} \qquad \text{(A.18)}$$

There are *M* objectives and it is desirable to achieve them as closely as possible to each one of their *M* goals. The actual value of the goals are set by the stakeholders, and their relative importance is key to the solution approach. One approach known as the weighted goal programming assign weights to the deviations from the goals, while another approach known as the preemptive goal programming assumes that the goals are listed in order of importance, and each goal is infinitely more important than the next goal.

A general expression for the weighted goal programming problem can be written as:

$$\left.\begin{array}{l} \text{Minimize: } \sum_{m=1}^{M}(w_m^+ d_m^+ + w_m^- d_m^-) \\ \text{s.t.} \\ e_m = f_m(x) + d_m^- - d_m^+, m = 1,2,\ldots,M \\ d_m^+, d_m^- \geq 0 \\ x \in S \end{array}\right\} \qquad \text{(A.19)}$$

where d_m^+ and d_m^- are deviational variables from the goal e_m, w_m^+ and w_m^- are non negative weights on the positive and negative deviations from each objective m.

This formulation was adapted from Cohon (1978), other formulations of goal programming techniques are presented in Nijkamp (1979), Winston (1994), Gass (1985) and Deb (2001) among others.

If the original problem is feasible then the set S is nonempty. The ideal case is that in which there is a point for which the objective function is exactly equal to the goal so that $f_m(x) = e_m, m = 1,2,\ldots,M$ but this might not be the case. By adding the deviational variables d_m^+ and d_m^- an augmented version of the original problem is guaranteed to be feasible.

Consider the constraint for each objective function

$$e_m = f_m(x) + d_m^- - d_m^+ \qquad \text{(A.20)}$$

In this constraint when the objective function is equal to the goal, the deviational variables will both be equal to zero, when the deviational variable d_m^- is positive the objective function $f_m(x)$ falls short from the goal e_m, since the objective function cannot be under e_m and over e_m at the same time, the deviational variable d_m^+ equals to zero. Likewise, when the deviational variable d_m^+ is positive, then d_m^- equals zero, this holds because both deviational variables are minimized in the objective function.

It is possible to eliminate one of the deviation variables if it is known with certainty the relation between the objective function to the goal. If the objective function $f_m(x)$ is always greater or equal to the goal then only d_m^+ is required. If the objective function $f_m(x)$ is always lower or equal to the goal then only d_m^- is required.

This method provides a mechanism for optimization of goals that might prove very useful in the solution of the LDPP. One possible setting is to optimize the objective function of each stakeholder alone and then optimize them all together by minimizing their deviation from the optimal solution. This type of approach might be considered by decision makers because it provides a solution that the stakeholders could accept since it represents a minimum deviation from their goals. It is easier to accept that their optimum value has been decreased while everyone else's has decreased as well to the minimum possible extent. The drawback is that the method will find potentially only one point. The one for which the deviations are minimized, to find or approximate the Pareto set, or an approximation of it we need another method.

Possibly the preemptive goal programming method could be used also by considering all possible orders of priorities for the stakeholders. Although, this may become computationally challenging due to the number of combinations and the number of subproblems that need to be solved.

Other variations of this method such as the min-max Goal Programming (Deb, 2001; Winston, 2004) consider minimizing the maximum distance among all goals to the ideal solution.

A 2.2.5. Multiobjective Simplex

The simplex method is a well known technique to solve single objective linear programming problems. It is based on a two-step procedure: first find an extreme point of the feasible region and then move to an adjacent extreme point with a better objective function until an optimal solution is found. The procedure to move from one extreme point to the next is based on elementary operations of the matrix of coefficients A and the values of the right hand side vector b. Typically some tableaus are used to ease the computation of the method. The multiobjective simplex method is an extension of the original simplex method in which additional rows are used to evaluate the multiobjective aspects of the problem.

The simplex method for single objective optimizations is extensively presented in the literature (Dantzig, 1963; Gass, 1985; Nash and Sofer, 1996; Cohon, 1978; Steuer, 2004; Winston 2004).

The following explanation was adapted from Nash and Sofer (1996). Consider the original linear programming problem (A.1) - (A.3).

Equation (A.2) can be re-written as:

$$\begin{pmatrix} B & N \end{pmatrix}\begin{pmatrix} x_B \\ x_N \end{pmatrix} = b \qquad \text{(A.21)}$$

where B and N are partitions of the coefficient matrix A, x_B is the set of basic variables and x_N is the set of non-basic variables.

Now the objective function (A.21) can be re-written as:

$$\text{Min: } c_B^T x_B + c_N^T x_N \qquad \text{(A.22)}$$

where c_B is the vector of coefficients of the basic variables and c_N is the vector of coefficients of the non-basic variables. The constraints (A.21) written as:

$$Bx_B + Nx_n = b \qquad \text{(A.23)}$$

$$x \geq 0 \qquad \text{(A.24)}$$

can be solved for x_B as:

$$x_B = B^{-1}b - B^{-1}Nx_N \qquad \text{(A.25)}$$

assuming that B^{-1} exists.

When (A.25) is substituted into (A.22) we obtain the following:

$$\text{Min: } c_B^T\left(B^{-1}b - B^{-1}Nx_N\right) + c_N^T x_N \qquad \text{(A.26)}$$

which is equivalent to:

$$\text{Min: } c_B^T B^{-1}b - c_B^T B^{-1}Nx_N + c_N^T x_N \qquad \text{(A.27)}$$

or:

$$\text{Min: } c_B^T B^{-1}b + \left(c_N^T - c_B^T B^{-1}N\right)x_N \qquad \text{(A.28)}$$

In (A.28), the term $\left(c_N^T - c_B^T B^{-1} N\right)$ is known as the vector of reduced costs of the non-basic variables x_N . This term is nonnegative when the optimal solution to the general linear programming problem is found.

If we define

$$y = \left(c_B^T B^{-1}\right)^T \qquad \text{(A.29)}$$

then the objective function can be written as:

$$\text{Min: } z = y^T b + \left(c_N^T - y^T N\right) x_N \qquad \text{(A.30)}$$

the vector y is known as the vector of simplex multipliers or dual variables.

At any extreme point, the basic variables and the value of the objective function can be determined by setting the non-basic variables x_N to zero obtaining from (A.25):

$$x_B = B^{-1} b \qquad \text{(A.31)}$$

and from (A.30)

$$z = y^T b \qquad \text{(A.32)}$$

A point x is a basic solution if it satisfies the equality constraints (A.23) and if the columns of the constraint matrix corresponding to the nonzero components of x are linearly independent (Nash and Sofer, 1996). A basic feasible solution is a basic solution that also satisfies the nonnegativity constraint $x \geq 0$. The simplex method starts by finding a basic feasible solution. There are several methods to obtain this initial point (Gass, 1985; Steuer, 2004). If the procedure is unable to do so then the problem is infeasible. Assuming this initial basic feasible solution (extreme point) is found, the simplex method evaluates the point for optimality. If the optimality test fails, an adjacent basic feasible solution (extreme point) is evaluated until an optimal solution is found. The data of the variables is arranged in tableaus whose last rows are the reduced cost of each variable. The interested reader is referred to the literature for a detailed step by step description.

Cohon (1978), Philip (1972), and Ecker and Kouada (1975) presented mathematical properties for noninferior solutions and developed algorithms for identifying noninferior solutions. Holl (1973), Evans and Steuer (1973), and Zeleny (1974) have all presented specific simplex-based methods for the generation of noninferior solutions.

The algorithm by Zeleny (1974) extends the original simplex algorithm to accommodate multiple objectives instead of a single one. These additional objectives are added at the bottom of the original tableau. Zeleny's theorem for multiobjective simp lex method states that if when the reduced costs are evaluated, there is a solution for which all reduced cost are non negative and at least one strictly negative, then the current solution is inferior. Based on this and other related theorems Zeleny designed an algorithm that ends when all noninferior basic feasible solutions have been found.

A 2.3. Methods to Solve General Mixed Integer Programming Problems

This work in this dissertation uses integer programming techniques in all of the three models analyzed. Be cause of this, besides presenting the theory of multiobjective optimization, the author has considered equally important to present some of the theory behind integer programming problems and the methods available to solve them.

The “divide and conquer” approach is typically the preferred solution method to solve mixed integer programming problems (MIP), these are the core of the current commercial solvers (Eiselt and Sandblom, 2000). This approach divides the original problem into subproblems that are easier to solve. The downside of course is the need to solve multiple subproblems instead of only one problem. The effectiveness of the method depends on how much easier it is to solve the subproblems as compared to the original problem. The next three sections are dedicated to these types of solution methods.

A 2.3.1. Branch and Bound

Branch and bound is a simple yet powerful technique widely used by the optimization community to solve mixed integer programming problems. The core of the procedure is to first relax the integer requirements and solve a linear programming relaxation. Then, select those variables that do not have integer solutions and create new problems in which the variables are bounded by the closest integer values obtained.

To clarify the procedure, consider the general formulation of a programming problem as presented in Wolsey (1998):

$$z = \max\{c^T x : x \in S\} \tag{A.33}$$

where S is the feasible region.

Suppose that the feasible region S can be divided into k smaller sets such that

$$S = S_1 \cup S_2 \cup \ldots \cup S_k \tag{A.34}$$

Let $z^k = \max\{c^T x : x \in S_k\}$ for $k = 1,2,\ldots,K$

then $z = \max_k z^k$ (A.35)

One way to represent the different possible sets is by using an enumeration tree. This is a graphical representation with a node on the top representing the feasible region S with arcs connecting nodes that represent subsets of S.

Now consider the general formulation of a MIP as follows:

Max: z = $c^T x + d^T y$ (A.36)

s.t.

$$\begin{aligned} & x \in S \\ & y \in T \\ & x \in \mathbb{R}^m, y \in \mathbb{Z}^n \end{aligned} \tag{A.37}$$

where $\mathbb{Z}^n = \{Z = (z_1, z_2, ..., z_n)^T\} / z_i$ is integer $\forall i$.

Suppose that we relax the integer requirements so we solve instead the linear programming relaxation:

Max: z = $c^T x + d^T y$ (A.38)

s.t.

$$\begin{aligned} & x \in S \\ & y \in T \\ & x \in \mathbb{R}^m, y \in \mathbb{R}^n \end{aligned} \tag{A.39}$$

Obtaining as a solution at least one $y_j \notin \mathbb{Z}$. Then we can create two problems as follows:

Subproblem 1	and	Subproblem 2
max z$_1$ = $c^T x + d^T y$		max z$_2$ = $c^T x + d^T y$
s.t.		s.t.
$x \in S$		$x \in S$
$y \in T$		$y \in T$
$y_j \le \lfloor y_j^* \rfloor$		$y_j \ge \lceil y_j^* \rceil$
$x \in \mathbb{R}^m, y \in \mathbb{R}^n$		$x \in \mathbb{R}^m, y \in \mathbb{R}^n$

Where $\lfloor \bullet \rfloor$ is the floor of $\bullet$ defined as the greatest integer $\leq \bullet$, and $\lceil \bullet \rceil$ called ceiling of $\bullet$ is defined as the smallest integer $\geq \bullet$. Is not difficult to see that z = Max:$\{z_1, z_2\}$

The procedure of separating the initial problem into subproblems is called branching. We draw a tree structure with an initial node that represents the LP relaxation of the MIP and two branches coming out of this root node, one where we solve subproblem 1 and another where we solve subproblem 2 as presented in Figure A.63.

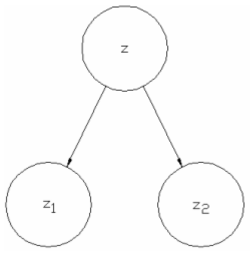


Figure A.63 Tree representation with two branches

The case in which only one variable has a fractional result is not common, rather, several of the y_j variables could be fractional so the tree grows quite large very fast. Fortunately, there are smart strategies that take in account the result of the linear programming relaxation to fix upper and lower bounds on each branch. This information is used to prevent the exploration of those branches that will not have the optimal solution. This procedure is commonly called “pruning the tree” and it is based on the following proposition (Wolsey, 1998).

Let $S = S_1 \cup S_2 \cup \ldots \cup S_k$ be a partition of S into smaller sets, and let $z^k = \max\{c^T x : x \in S_k\}$ for $k = 1,2,\ldots,K$, $\overline{z}^k$ be an upper bound on z^k and $\underline{z}^k$ be a lower bound on z^k. Then since (A.38) is a maximization problem,

$\overline{z} = \max_k \overline{z}^k$ is an upper bound on z (A.40)

and

$\underline{z} = \max_k \underline{z}^k$ is a lower bound on z (A.41)

$\overline{z}$ will tend to be reduced as the procedure advances since the feasible region gets smallest at each step, so the objective function tends to worsen (gets smaller in a maximization problem).

A tree can be pruned by optimality, or by infeasibility. A branch is pruned by optimality if the solution of the relaxation yields an integer solution, then no more subproblems are to be found within the node.

$z^t = \{\max c^T x : x \in S_t\}$ has been solved (A.42)

A branch of the tree is pruned by infeasibility if the LP relaxation solved at the node is infeasible.

$S_t = \varnothing$ (A.43)

A branch is pruned by bound if the optimal solution of the LP relaxation at the node is outside the best bound found. In the case of maximization if the solution falls below the lower bound found, or in a minimization problem the solution falls above the best upper bound found.

$\overline{z}^t \leq \underline{z}$ (A.44)

In a maximization problem to obtain the upper bound we solve the linear programming relaxation, to obtain a lower bound we need to find a feasible solution either by searching down the branches of the tree, or by using an algorithm.

Example: Consider the following problem:

(IP) $z^{IP} = \max : 3x_1 + 2x_2$ (A.45)

s.t.

$x_1 + x_2 \leq 5.5$ (A.46)

$x_1 \leq 4.5$ (A.47)

$x_2 \leq 3.5$ (A.48)

$x_1, x_2 \in \mathbb{Z}^1_+$ (A.49)

We solve the linear programming relaxation:

(LP) $z^{LP} = \max : 3x_1 + 2x_2$ (A.50)

s.t.

$x_1 + x_2 \leq 5.5$ (A.51)

$x_1 \leq 4.5$ (A.52)

$x_2 \leq 3.5$ (A.53)

$x_1, x_2 \in \mathbb{R}^1_+$ (A.54)

Whose optimal solution is:

$z^{LP} = 15.5$, $x_1 = 4.5$, $x_2 = 1$

Since x_1 is fractional we break down the problem into two subproblems, one with $x_1 \geq 5$ and the other with $x_1 \leq 4$ as presented in Figure A.64.

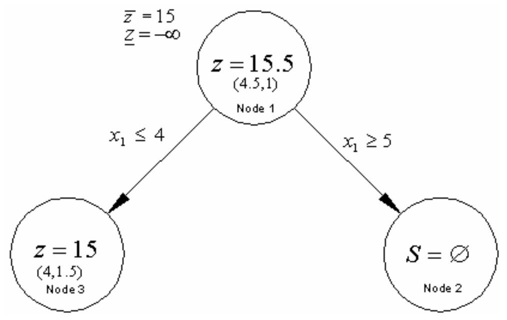


Figure A.64 First branch

The first problem evaluated in node 2 is infeasible so the tree gets pruned by infeasibility, and the second problem evaluated in node 3 has an optimal solution of

$$z^{LP} = 15, x_1 = 4, x_2 = 1.5$$

We update the value of the upper bound from 15.5 to 15. Since now x_1 is fractional we need to branch on this variable creating two new subproblems one with $x_2 \ge 2$ and the other with $x_2 \le 1$ as presented in Figure A.65.

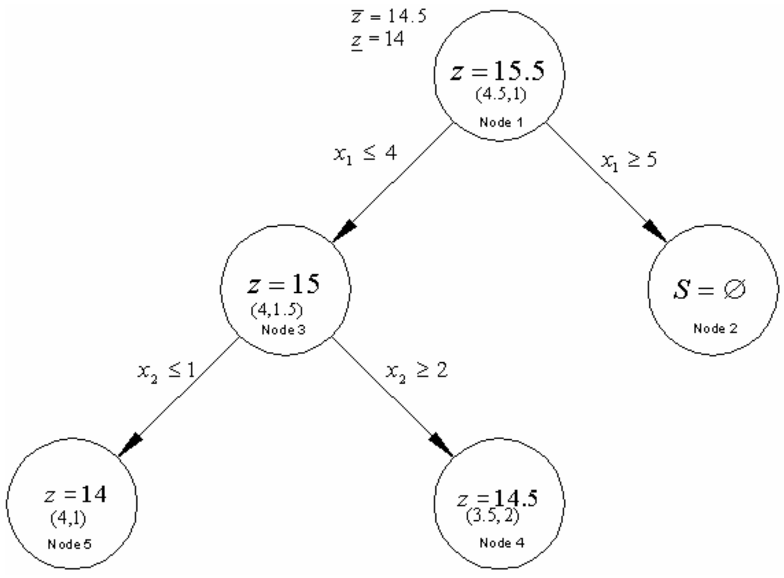


Figure A.65 Second branch

Since the optimal solution of node 5 is integer, we do not need to evaluate that node any more, thus it is pruned by optimality. Also, the lower bound of the problem gets updated since we have found the first feasible solution. Node 4 has an optimal solution of

$$z^{LP} = 14.5, x_1 = 3.5, x_2 = 2$$

Since x_1 is fractional we break down the problem into two subproblems, one with $x_1 \geq 4$ and the other with $x_1 \leq 3$ as presented in Figure A.66.

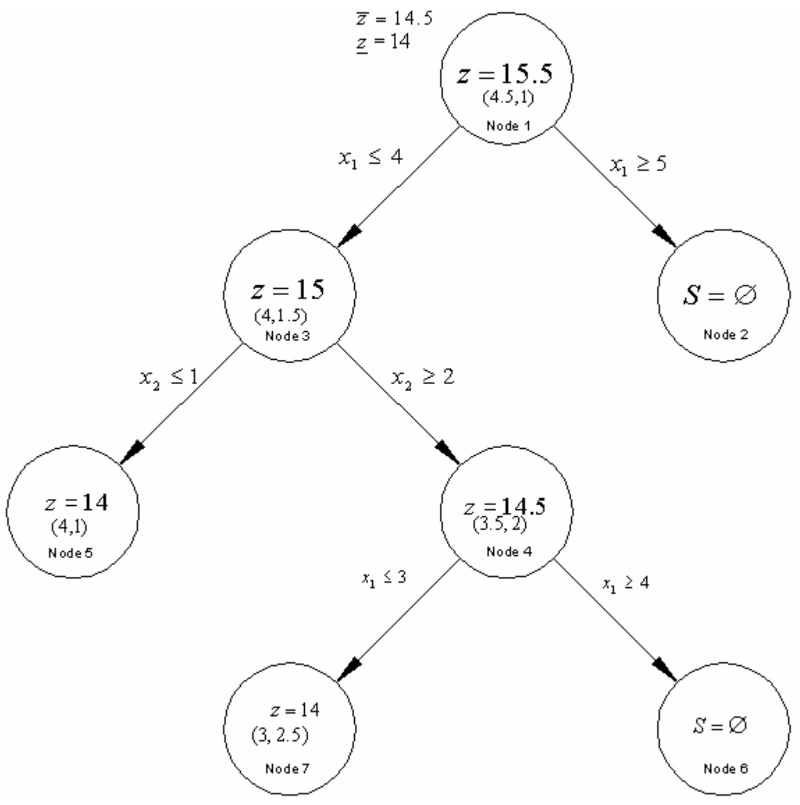


Figure A.66 Third branch

Now we find that node 6 is infeasible so it gets pruned by infeasibility and since node 7 has the same value of the bound but a fractional number then we conclude that node 5 represents an optimal solution so $z^{IP} = 14, x_1 = 4, x_2 = 1$.

A 2.3.2. Lagrangian Relaxation

The Lagrangian relaxation technique is based on simplifying the MIP problem by eliminating the constraints that makes it difficult to solve. The constraints are not completely eliminated, rather they are moved to the objective function multiplied by a factor. This factor is called a Lagrange multiplier.

The rationale for this procedure comes from realizing that by eliminating the constraints from the feasible region the resulting feasible region is an expansion of the original one. By moving the nonnegative slack of these constraints to the objective function penalized by a certain nonnegative factor, the objective function of the relaxation is always larger, or in the best case equal, to the original problem for any feasible point in the original problem. This method does not always provide an optimal solution, but at least provides a bound on the MIP problem (Wolsey, 1998).

We can express the IP problem as:

$$z(x) = \max: c^T x \tag{A.55}$$

s.t.

$$\begin{aligned} Dx &\le d \\ x &\in X \end{aligned} \tag{A.56}$$

where D is a matrix of "difficult or complicating" constraints coefficients, d is the right hand side of those constraints, and X is a set. The constraint set $Dx \le d$ is a complicating set of "k" constraints, in the sense that if they were eliminated then the problem would be computationally simpler to solve. A relaxation of the original is:

$$z(u) = \max: c^T x + u^T (d - Dx) \tag{A.57}$$

s.t.

$$x \in X \tag{A.58}$$

Note that the feasible region is larger since a set of inequalities have been eliminated. For any point such that $(d - Dx) \geq 0$, the objective function can only be larger because $u \geq 0$. Therefore the optimal solution *z*(x)* is bounded as,

$$z^*(u) \geq z^*(x) \forall x : (d - Dx) \geq 0, x \in X \tag{A.59}$$

Since $z^*(u)$ is an upper bound on the value of $z^*(x)$, the problem now become to find the smallest multipliers over all infinite possible values of u. To find these multipliers we need to solve the Lagrangian dual problem:

$$w_{LD} = \min\{z(u) : u \geq 0\} \tag{A.60}$$

As presented in Wolsey (1998) if $u \geq 0$,

i) $x(u)$ is an optimal solution of (A.57), and

ii) $Dx(u) \leq d$, and

iii) $(Dx(u))_i = d_i$ whenever $u_i > 0$ (complementarity),

then $x(u)$ is optimal in (A.55) - (A.56)

Example: Consider again the problem presented in (A.45) - (A.49). We could relax constraint (A.51) and apply the Lagrangian relaxation procedure as follows:

(IP) $z(u) = \max : 3x_1 + 2x_2 + (5.5 - x_1 - x_2)u$ (A.61)

s.t.

$$x_1 \leq 4.5 \tag{A.62}$$

$$x_2 \leq 3.5 \tag{A.63}$$

$$x_1, x_2 \in \mathbb{Z}^1_+ \tag{A.64}$$

We need to solve the Lagrangian dual problem

$$wLD = \min_{u} \{\max : 3x_1 + 2x_2 + (5.5 - x_1 - x_2)u\} \quad \text{(A.65)}$$

s.t.

$$x_1 \le 4.5 \quad \text{(A.66)}$$

$$x_2 \le 3.5 \quad \text{(A.67)}$$

$$x_1, x_2 \in \mathbb{Z}^1_+, u \in \mathbb{R}^1_+ \quad \text{(A.68)}$$

Before solving the Lagrangian dual we need to present some information about the solution of the Lagrangian dual problem that can be used to solve it.

The following explanation comes directly from Wolsey (1998). Suppose for simplicity that the feasible region of the relaxed problem contains a large but finite number of points $X = \{x^1, ..., x^T\}$

Now

$$w_{LD} = \min_{u \ge 0} z(u) \quad \text{(A.69)}$$

$$w_{LD} = \min_{u \ge 0} \left\{ \max_{x \in X} \left[c^T x + u^T (d - Dx) \right] \right\} \quad \text{(A.70)}$$

$$w_{LD} = \min_{u \ge 0} \left\{ \max_{t=1,..,T} \left[c^T x^t + u^T (d - Dx^t) \right] \right\} \quad \text{(A.71)}$$

$$w_{LD} = \min h \quad \text{(A.72)}$$

s.t.

$$h \ge c^T x^t + u^T (d - Dx^t) \text{ for all } t = 1,2,...,T \quad \text{(A.73)}$$

$$u \in \mathbb{R}^m_+, h \in \mathbb{R}^1 \quad \text{(A.74)}$$

The new variable h represents an upper bound on $z(u)$. Problem (A.72) - (A.74) is a linear problem whose dual is:

$$w_{LD} = \max \sum_{t=1}^{T} m_t (c^T x^t) \tag{A.75}$$

s.t.

$$\sum_{t=1}^{T} m_t \left(Dx^t - d \right) \le 0 \tag{A.76}$$

$$\sum_{t=1}^{T} m_t = 1 \tag{A.77}$$

$$m_t \in \mathbb{R}_+^T \tag{A.78}$$

Setting $x = \sum_{t=1}^{T} m_t x^t$, with $\sum_{t=1}^{T} m_t = 1$ and $m_t \in \mathbb{R}_+^T$ we get

$$w_{LD} = \max c^T x^t \tag{A.79}$$

s.t.

$$Dx^t \le d \tag{A.80}$$

$$x \in conv(X) \tag{A.81}$$

This can be generalized to the case when X is the feasible region of any integer programming problem:

$$X = \left\{ x \in \mathbb{Z}_+^n : Ax \le b \right\} \tag{A.82}$$

Then

$$w_{LD} = \max \left\{ c^T x^t : Dx \le b, x \in conv(X) \right\} \tag{A.83}$$

The result of (A.83) provides the strength of the relaxation which in some cases is not any stronger than the simple linear programming relaxation (Wolsey, 1998).

Now we can proceed to solve the Lagrangian dual using a subgradient algorithm as described in Wolsey (1998).

Iteration 1, Step 1: Initialization

set $u = u^0$

Iteration k, Step 2: Solve the Lagrangian Subproblem

set $u = u^k$

Solve the Lagrangian problem IP(u^k) with optimal solution $x(u^k)$.

$u^{k+1} = \max\left\{u^k - m_k\left(d - Dx(u^k)\right), 0\right\}$

$k \leftarrow k+1$

The algorithm at each iteration takes a step from the present point u^k in a direction opposite to a subgradient $d - Dx(u^k)$. The difficulty is in defining the step lengths $\{m_k\}_{k=1}^{\infty}$ (Wolsey, 1998).

Wolsey (1998) presents a theorem to aid in the selection of the steps.

If $\sum_k m_k \to \infty$, and $m_k \to 0$ as $k \to \infty$ then $z(u_k) \to w_{LD}$ (A.84)

where w_{LD} is an optimal of LD

If $m_k = m_0 r^k$ for some parameter $r < 1$ then if m_0 and r are sufficiently large

$z(u^k) \to w_{LD}$ (A.85)

If $\overline{w} \geq w_{LD}$ and $m_k = \dfrac{e_k\left[z\left(u^k\right) - \overline{w}\right]}{\left\|d - Dx\left(u^k\right)\right\|^2}$ with $0 < e_k < 2$ then

$z\left(u^k\right) \to \overline{w}$ (A.86)

Or the algorithm finds u^k with $w \geq z\left(u^k\right) \geq w_{LD}$ for some finite *k*.

Iteration k, Step 3: Convergence Check

Unless a value of u^k is obtained for which is known that $Z_D(u^k)$ equals the cost of a known feasible solution, there is no way to prove that the subgradient method has reached optimality (Fisher 1981). Therefore, we need to recourse to classic stopping criteria, one of which is setting a limit in the number of iterations (Fisher 1981), another could be by setting a tolerance that measures the improvement between one iteration and the next such as $\left|z(u^k) - z(u^{k-1})\right| \leq tol$ or one could also let the procedure run for certain predetermined maximum time .

Going back to our previous example, se set $u^0 = 5$ and $r = 0.9$ as per (A.85) we expect to obtain the optimal value of w_{LD} .

Iteration 1, Step 1: Initialization

set $u = 4$

Iteration 1, Step 2: Solve the Lagrangian Subproblem

Solve the Lagrangian problem LR(u^k) with optimal solution $x(u^k)$.

LR(u^1) $z(u) = \max{:}3x_1 + 2x_2 + (5.5 - x_1 - x_2)4$ (A.87)

s.t.

$x_1 \leq 4.5$ (A.88)

$x_2 \leq 3.5$ (A.89)

$x_1, x_2 \in \mathbb{Z}_+^1$ (A.90)

we obtain $z(u) = 22, x_1 = 0, x_2 = 0$

$m_k = m_0 r^k \leftrightarrow m_1 = 0.9$

$k = 1$

Iteration 2

set $u = 4*0.9 = 3.6$

LR(u^2) $z(u) = \max: 3x_1 + 2x_2 + (5.5 - x_1 - x_2)3.6$ (A.91)

s.t.

$x_1 \leq 4.5$ (A.92)

$x_2 \leq 3.5$ (A.93)

$x_1, x_2 \in \mathbb{Z}_+^1$ (A.94)

we obtain $z(u) = 19.8, x_1 = 0, x_2 = 0$

We continue in the same fashion obtaining the following results:

Iteration	u	z	x1	x2
1	4.0000	22.0000	0	0
2	3.6000	19.8000	0	0
3	3.2400	17.8200	0	0
4	2.9160	16.3740	4	0
5	2.6244	15.9366	4	0
6	2.3620	15.5429	4	0
7	2.1258	15.1886	4	0
8	1.9132	15.1302	4	3
9	1.7219	15.4172	4	3
10	1.5497	15.6755	4	3
11	1.3947	15.9079	4	3
12	1.2552	16.1171	4	3

Table A.26 Values of the optimal solution for different iterations

When there values are plotted we obtain

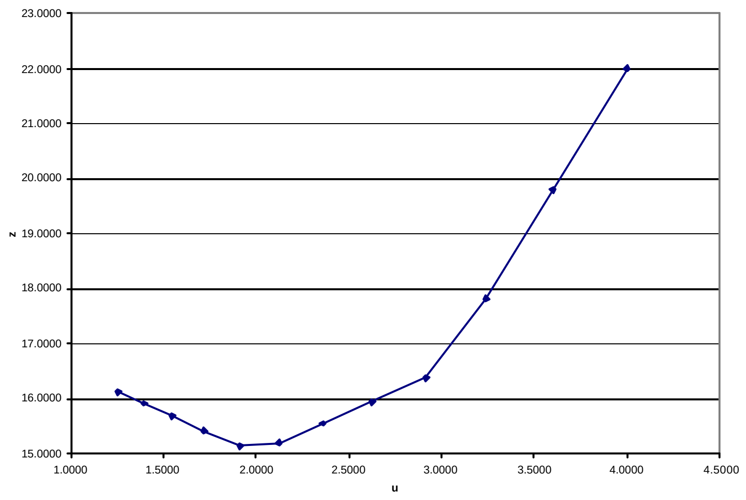


Figure A.67 Values of the objective function for different values of u

The optimal solution found is $z(u) = 15.1302$ with u=1.9132, $x_1 = 4$, $x_2 = 3$ which results in $z(x) = 18$. This value of z^{LR} is the closest to the optimal value of z (= 14).

However, there is a better solution, because we know explicitly the feasible region of (A.66) - (A.68), and it is not too large (contains only 20 points) we can explicitly write the problem in the form (A.72) - (A.74) obtaining the optimal solution $h = 15, u = 2$ (See appendix for formulation and solution using LINDO). This value is better than the one obtained earlier. This realization has created another area of research on the Lagrangian relaxation method based on column generation techniques similar to those explained under the Dantzig-Wolfe decomposition technique, and multiplier adjustment methods among others (Fisher, 1981).

Although we found a close bound on the value of z, the values for x_1 and x_2 are not close to the optimal values of x. But this close bound on the objective function can be employed to efficiently prune the branch and bound algorithm (Fisher, 1981; Wolsey 1996).

The strength of the Lagrangian dual as presented in (A.83) is evident when considering the feasible region of the original problem and the relaxed problem as presented in Figure A.68.

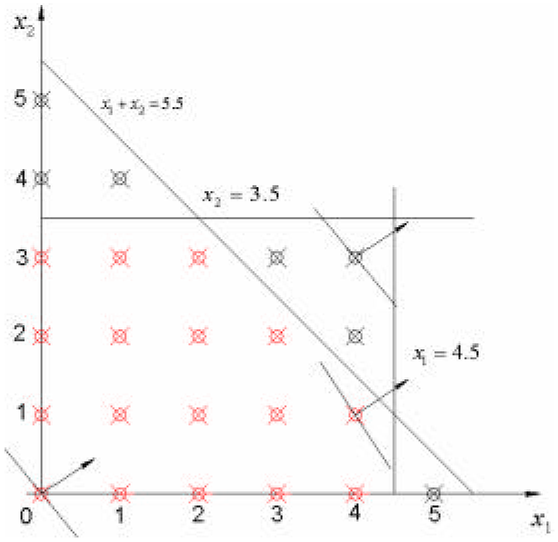


Figure A.68 Feasible region

The original problem contains 17 points in the feasible region while the relaxation contains 20. The point (4,3) is optimal for any u that results in a positive coefficient of x_1 and x_2. The point (4,0) is optimal for any u that results in a positive coefficient of x_1 and negative for x_2. The point (0,0) will be optimal for any u such that both coefficients are negative. These ranges can be found by writing the objective function as

$$\text{LR}(u)\ \ z(u) = \max\text{:}(3-u)x_1 + (2-u)x_2 + 5.5u \qquad \text{(A.95)}$$

For $0 \le u < 3$ the coefficient of x_1 is positive and for $0 \le u < 2$ the coefficient of x_2 is positive.

It is also possible to find the best value of *u* by applying the bisection method or the regula falsi method when we find two values of *u* such that one makes the original problem feasible and the other one infeasible.

The reasoning is that when we consider the Lagrangian problem:

$$z(u) = \max\left\{c^T x + u^T (d - Dx) : x \in X\right\} \tag{A.96}$$

as

$$u \to 0 \;\; z^*(u) \to \max\left\{c^T x : x \in X\right\} \tag{A.97}$$

Since the feasible region of the original problem is contained in the feasible region of this relaxation, if

$$\left\{x : Dx \le D, x \in X\right\} \subseteq \left\{x \in X\right\} \tag{A.98}$$

Then

$$z^*(u) \ge z^*(x) \tag{A.99}$$

This result was presented earlier as (A.59).

If the original problem if feasible, then a large value of *u* would produce a feasible solution since it would maximize the term $\left(d - Dx\right)$, which would be feasible on the original problem for any value of x such that $\left(d - Dx\right) \ge 0$.

One could then use the regula falsi method to find the minimum value of *u* that maximizes *z(u)*.

For example, consider the values

$u^{(1)} = 2.1258$ with $z(u)^{(1)} = 15.1886$ where $\left(d - Dx^{(1)}\right) \ge 0$ and

$u^{(2)} = 1.9132$ with $z(u)^{(2)} = 15.1302$ where $\left(d - Dx^{(2)}\right) \le 0$.

$$u^{(3)} = \frac{(d - Dx^{(1)})u^{(2)} - (d - Dx^{(2)})u^{(1)}}{(d - Dx^{(1)}) - (d - Dx^{(2)})}$$

in our example

$$u^{(3)} = 2.0194758 \rightarrow z(u)^3 = 15.0292137, x_1 = 4, x_2 = 0$$

So the bound obtained with regula falsi was in this case closer than the one obtained using the subgradient method. Similarly a bisection approach could have been used as follows:

$$u^{(3)} = \frac{u^{(2)} + u^{(1)}}{2}$$

in our example

$$u^{(3)} = 2.0194758$$

so the result would have been the same.

A point could be made that under different values of u_0 and r the subgradient method could have provided a better bound. Nevertheless, the application of successive iterations of the regula falsi method would match the bound found by the subgradient. The problem is to decide when to use one method or the other.

A 2.3.3. Dantzig-Wolfe Decomposition Method

The following method takes advantage of certain "decomposable" structure present in certain types of formulations. This decomposable structure permits the feasible region to be broken into sets $S = S_1 \cup S_2 \cup \ldots \cup S_k$. These sets result from breaking the original formulation into k independent subproblems. However, some problems also include a set of m constraints that prevent the straightforward decomposition of the original problem, these constraints are called complicating constraints (Nash and Sofer 1996; Conejo et al., 2003), central constraints (Winston, 2004) or joint constraints (Wolsey, 1998).

A formulation has a decomposable structure with complicating constraints, if one can arrange the constraints in the following general fashion:

There is a group of *m* complicating constraints that involve any of the variables, ant there are *k* groups of constraints that involve only $k_1, k_2, ..., k_n$ variables.

$$Max: c^{1T}x^1 + c^{2T}x^2 + ... + c^{KT}x^K \tag{A.100}$$

s.t.

$$\begin{aligned}
&A^1x^1 + A^2x^2 + ... + A^Kx^K = b \\
&D^1x^1 \qquad\qquad\qquad \le d_1 \\
&\qquad + D^2x^2 \qquad\quad \le d_2 \\
&... \\
&\qquad\qquad\qquad D^Kx^K \le d_K \\
&x^1 \in \mathbb{R}^{k_1}, x^2 \in \mathbb{R}^{k_2}, ..., x^K \in \mathbb{R}^{k_n}, b \in \mathbb{R}^m, D^j \in \mathbb{R}^{k_i x q_i}, d_j \in \mathbb{R}^{q_i}
\end{aligned} \tag{A.101}$$

Looking at the structure of the problem one can notice that the first set of constraints includes all the variables. If this set of constraints were not included, then the problem could be broken into *k* separate independent problems with $k_1, k_2, ..., k_n$ variables and $q_1, q_2, ..., q_k$ constraints, respectively. This problem is said to present complicating constraints because a constraint or set of constraints prevents a straightforward decomposition of the problem.

In other words, a formulation has a decomposable structure with complicating constraints, if one can divide the constraints and the variables in sets such that:

Constraints in set 1 only involve only variables in set 1

Constraints in set 2 only involve only variables in set 2

….

Constraints in set *k* only involve only variables in set *k*

There is a set of constraints *k+1* that involves any variable (Winston, 2004).

This set *k+1* is referred to as the central constraints or complicating constraints.

The Dantzig-Wolfe technique has two different but similar algorithms to solve problems with decomposable structure. One is for linear programming problems (Dantzig, 1969; Bradley, Hax and Magnanti, 1977; Nash and Sofer 1996; Conejo et al., 2003; Winston 2004), and another one for mixed integer programming problems (Wolsey, 1998; Vanderbeck, 1998).

4.4.1. Dantzig –Wolfe Algorithm for Linear Programming Problems

In the book edited by Aronofsky (1969) there is an explanation of the decomposition principle by Dantzig which is presented as follows:

This method decomposes the original linear programming problem into:

a) subprograms corresponding to its almost independent parts, and

b) a master program which ties together the subprograms.

The price paid for this decomposition is that the master program and the subprograms may have to be solved several times. The algorithm is based on the results of the following theorem (Winston, 2004; Wolsey, 1998)

Theorem 5 All feasible points can be expressed as a combination of the convex hull

Suppose the feasible region for an LP is bounded and the extreme points (or basic feasible solutions) of the LP's feasible region are: $P_1, P_2, \ldots, P_k$*. Then any point x* *in the LP feasible region may be written as a linear combination of* $P_1, P_2, ..., P_k$ *. In other words there exist weights* $m_1, m_2, ..., m_k$ *satisfying:*

$$x = m_1 P_1 + m_2 P_2 + ... + m_k P_k \qquad (A.102)$$

Moreover, the weights $m_1, m_2, ..., m_k$ *may be chosen such that*

$$m_1 + m_2 + ... + m_k = 1 \text{ and } m_1, m_2, ..., m_k \geq 0 \qquad (A.103)$$

Any linear combination of vectors for which the weights satisfy (A.103) is called a convex combination (Korte and Vygen, 1999; Winston, 2004). And the set of points $P_1, P_2, ..., P_k$ is called the convex hull of the feasible region *S* denoted as *conv(S)* (Korte and Vygen, 1999; Wolsey, 1998). This theorem states that any point in a bounded feasible region can be written as a convex combination of the extreme points of the feasible region.

A set X is convex if $l x + (1 - l) y \in X \ \forall x, y \in X$ and $l \in [0,1]$, so the set X is convex if and only if all convex combinations of points in *X* are again in *X*. The convex hull of a set *X* is the smallest convex set containing *X* (Korte and Vygen, 1999).

Consider the linear programming problem that has a decomposable structure with complicating constraints. If the complicating constraints are ignored we are relaxing the linear programming problem. Therefore in general the feasible region is expanded. The feasible region of the relaxed problem is divided into sets, each one having extreme points. Because the feasible region of the original problem is a subset of the feasible region of the relaxed problem, then the extreme points of the feasible region of the original problem would be feasible points of some of these subsets.

Therefore, by the theorem mentioned above, then they can be expressed as convex combinations of these extreme points. To reinforce this concept consider an analogous situation presented in Figure A.69 where the original feasible region (hatched) has extreme points x_1, x_2, x_3, x_4.This feasible region is a subset of the feasible region of the relaxed problem which can be decomposed in two sets (white and shaded). The extreme points of the original problem are feasible points to the sets so they can be expressed as a convex combination of the extreme points of the subsets.

For example x_1 can be expressed as $x_1 = m_1^1 x_1^1 + m_2^1 x_2^1 + m_3^1 x_3^1 + m_4^1 x_4^1$

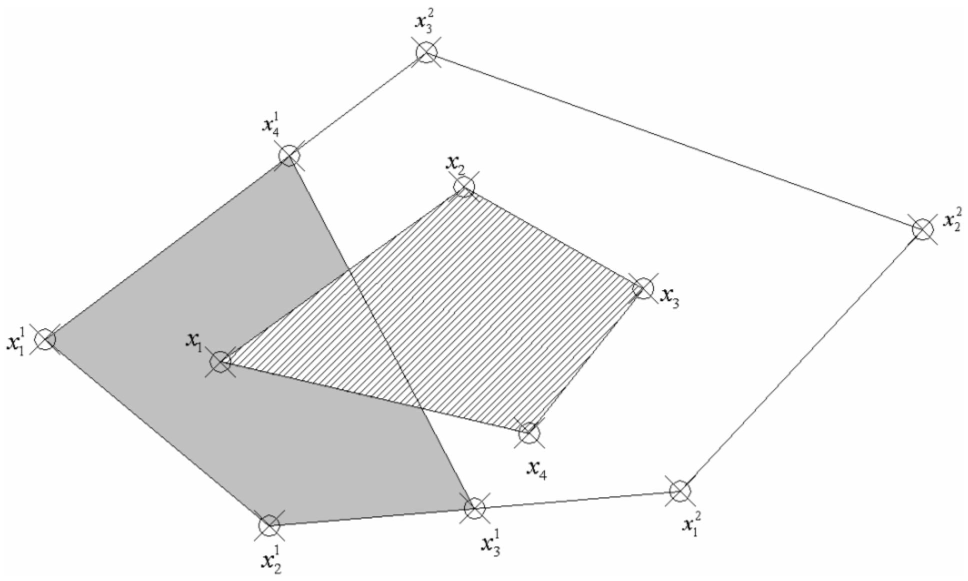


Figure A.69 Example of feasible region of original problem (shown hatched) as a subset of the feasible region of the relaxed problem with two subsets (shaded gray and unshaded)

The Dantzig-Wolfe decomposition method seeks to find the extreme points of the feasible region by first decomposing or dividing the feasible region into sets. These sets result from taking the original formulation and ignoring the complicating constraints. By solving the linear programming subproblems multiple times, each time with a different objective function, different extreme points of each set are found. Finally, feasible points for the original problem are found by using convex combinations of these extreme points. Because the number of extreme points of each set can be extremely large, the algorithm first finds few extreme points and then solves a linear programming problem to determine which of the variables not currently considered (non-basic variables) can be included in order to improve the current solution. This selection process is known as column generation.

The following algorithm was adapted and generalized from Conejo et al. (2003) and Winston (2004).

Iteration 1, Step 0: Initialization

Initialize the iteration counter $v = 0$.

Obtain p_v solutions to the k subproblems $i = 1,2...k$ by solving p_v times:

$$Max: c^{iT} x^i \tag{A.104}$$

s.t.

$$\begin{aligned} &D^i x^i \le d_i \\ &D^i \in \mathbb{R}^{q_i x k_i} x^i \in \mathbb{R}^{k_i}, d_i \in \mathbb{R}^{q_i} \end{aligned} \tag{A.105}$$

The coefficients of the variables in (A.104) are arbitrary (nonnegative) coefficients required to obtain the initial p_v solutions.

Iteration 1, Step 1: Solve the Restricted Master Problem

Increase the iteration counter $v = v + 1$

Express the LP's objective function and complicating constraints in terms of the solutions obtained before, and multipliers m_i^j. Add the convexity and sign constraints to obtain the restricted master problem:

Max: Objective function in terms of the m_i^j (A.106)

s.t.

Complicating constraints in terms of the m_i^j (A.107)

$m_1 + m_2 + ... + m_k = 1$ (A.108)

$m_1, m_2, ..., m_k \geq 0$ (A.109)

To obtain the solution $u_1^{(v)}, u_2^{(v)}, ..., u_p^{(v)}$ and the dual variables $l_1^{(v)}, l_2^{(v)}, ..., l_p^{(v)}$ and $s^{(v)}$.

Iteration 1, Step 2: Solve the k Subproblems

$$\underset{x}{Max}: v = \sum_{j=1}^{k_i} \left(c_{ij} - l_i^{(v)} a_{ij} \right) x_{ij}$$

s.t.

$$\sum_{j=1}^{k_i} a_{ij} x_{ij} \leq d_i$$

$$lb_{ij} \leq x_{ij} \leq ub_{ij}; \, j = 1..k_n$$

With the solution to the relaxed problem, evaluate the objective function of the original problem obtaining:

$$z^{(p+v)} = \sum_{i=1}^{k} \sum_{j=1}^{k_i} c_{ij} x_{ij}^{(p+v)}$$

And the value of the complicating constraints obtaining:

$$r_i^{(p+v)} = \sum_{i=1}^{k} \sum_{j=1}^{k_i} a_{ij} x_{ij}^{(p+v)} = b$$

Iteration 1, Step 3: Convergence checking

If $v \geq s$ then the optimal solution of the original problem has been achieved and the algorithm stops. If $v < s$ then the current solution can be used to improve the solution of the master problem so go to step 2.

A 2.3.4. Dantzig –Wolfe Algorithm for Integer Programming Problems

Consider a slight variation of the problem presented in (A.100) - (A.101). The general formulation for a single objective integer programming problem that could be decomposed into *k* subproblems can be written as:

$$\text{(IP)}\ z = Max : c^{1T}x^1 + c^{2T}x^2 + \ldots + c^{KT}x^K \qquad \text{(A.110)}$$

s.t.

$$\begin{aligned}
&A^1x^1 + A^2x^2 + \ldots + A^Kx^K = b \\
&D^1x^1 \qquad\qquad\qquad \leq d_1 \\
&\qquad + D^2x^2 \qquad\quad \leq d_2 \\
&\ldots \\
&\qquad\qquad\qquad D^Kx^K \leq d_K \\
&x^1 \in \mathbb{Z}^{k_1}, x^2 \in \mathbb{Z}^{k_2}, \ldots, x^K \in \mathbb{Z}^{k_n}, b \in \mathbb{R}^m, D^j \in \mathbb{R}^{k_j x q_j}, d_j \in \mathbb{R}^{q_j}
\end{aligned} \qquad \text{(A.111)}$$

or more succinctly as (Wolsey, 1998):

$$\begin{aligned}
&Minimize : \sum_{k=1}^{K} c^{kT}x^k \\
&Subject\ \text{to:} \\
&\sum_{k=1}^{K} A^kx^k = b \\
&x^k \in X^k \text{ for } k = 1 \ldots K
\end{aligned} \qquad \text{(A.112)}$$

where

$$X^k = \{x^k \in \mathbb{Z}_+^{n_k} : D^kx^k \leq d_k\} \text{ for } k = 1..K \qquad \text{(A.113)}$$

The goal is to decompose the problem by blocks into *k* subproblems. Assuming the feasible region of each subproblem is non empty, we can restate the problem formulation as a function of the (possibly quite large) feasible points

$\left\{x^{k,t}\right\}_{t=1}^{T_k}$ contained in X^k so we have (Wolsey, 1998):

$$X^k = \{x^k \in \mathbb{R}^{n_k} : x^k = \sum_{t=1}^{T_k} \left(l_{k,t}\right)\left(x^{k,t}\right), \sum_{t=1}^{T_k} l_{k,t} = 1, l_{k,t} \in \{0,1\} \text{ for t = 1..T}_k\} \quad \text{(A.114)}$$

Now substituting for x^k leads to the equivalent IP Master problem:

$$\text{(IPM) } z = \max \sum_{k=1}^{K} \sum_{t=1}^{T_k} (c^k x^{k,t})\, l_{kt} \quad \text{(A.115)}$$

s.t.

$$\sum_{k=1}^{K} \sum_{t=1}^{T_k} (A^k x^{k,t})\, l_{k,t} = b \quad \text{(A.116)}$$

$$\sum_{t=1}^{T_k} l_{k,t} = 1 \text{ for } k = 1, 2, \ldots, K \quad \text{(A.117)}$$

$$l_{k,t} \in \{0,1\} \text{ for } t = 1, 2, .., T_k \text{ and } k = 1, 2, \ldots, K \quad \text{(A.118)}$$

To illustrate these formulations consider the example used before in (A.45) - (A.49) with an additional variable x_3 used as slack variable to convert the complicating constraint into equality form:

$$\text{(IP) } z = \max : 3x_1 + 2x_2 \quad \text{(A.119)}$$

s.t.

$$x_1 + x_2 + x_3 = 5.5 \quad \text{(A.120)}$$

$$\left.\begin{array}{l} x_1 \in X^1 \\ x_2 \in X^2 \\ x_3 \in X^3 \end{array}\right\} \quad \text{(A.121)}$$

where

$$X^1 = \left\{ x_1 \in \mathbb{Z}_+^1 : x_1 \le 4.5 \right\} \tag{A.122}$$

$$X^2 = \left\{ x_2 \in \mathbb{Z}_+^1 : x_2 \le 3.5 \right\} \tag{A.123}$$

$$X^3 = \left\{ x_3 \in \mathbb{R}_+^1 : x_3 \ge 0 \right\} \tag{A.124}$$

The graphical representation of this problem is presented in Figure A.70

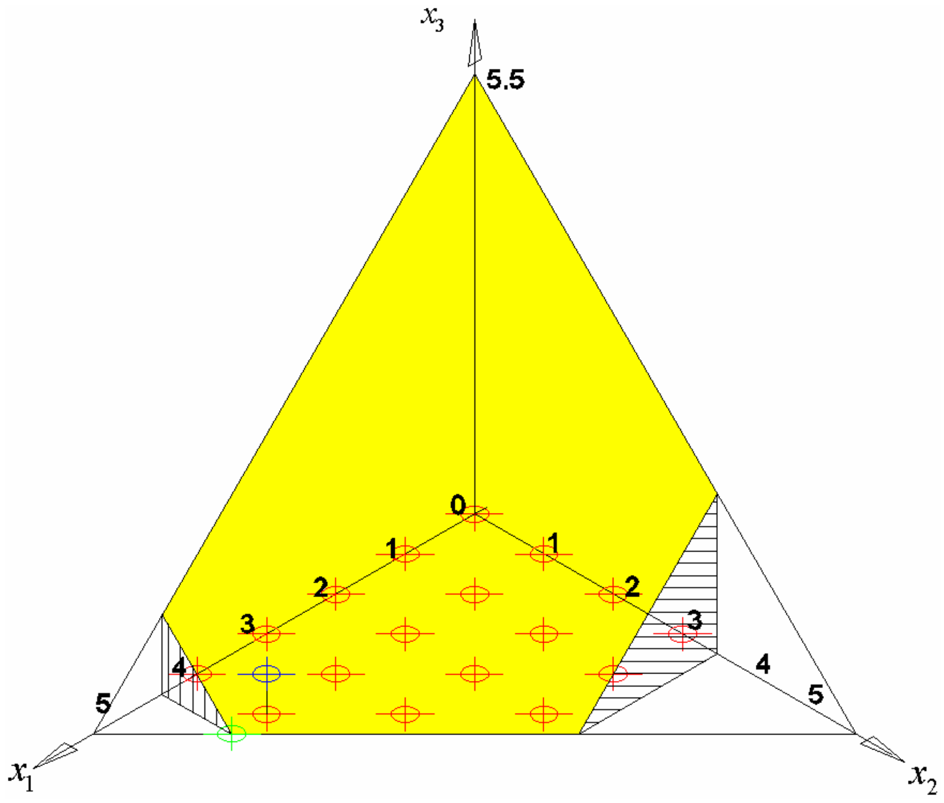


Figure A.70 Feasible region of example

The feasible region are the vertical lines that start on each of the integer points located in the plane x_1x_2 and end on the shaded plane. Note that these points correspond to integer values of x_1 and x_2 and the shaded plane is formed by the intersection of the plane $x_1 + x_2 + x_3 = 5.5$ with the planes $x_1 = 4.5$ (shown with vertical hatch) and $x_2 = 3.5$ (shown with horizontal hatch). The optimal solution for this problem is: $z^{IP} = 14, x_1 = 4, x_2 = 1, x_3 = 0.5$

In this case there is one complicating constraint and three constraints that can be decomposed into three subproblems. The sets X^1 and X^2 both contain a relatively small finite set of points $X^1 = \{0,1,2,3,4\}$ and $X^2 = \{0,1,2,3\}$ while the set X^3 has an infinite number of points. Variable x_3 can be regarded as a slack variable to force the inequality $x_1 + x_2 \leq 5.5$ as equality. Therefore, Figure A.70 can be presented in two dimensions as in Figure A.71.

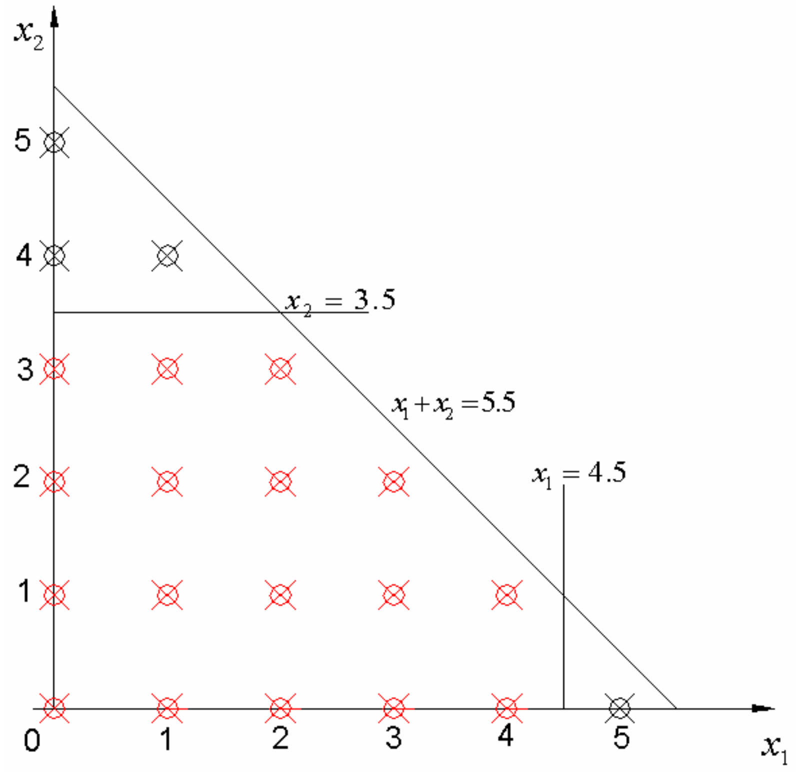


Figure A.71 Two dimensional representation of the example's feasible region

We have that:

$$X^1 = \left\{ x_1 \in \mathbb{R}^1_+ : x_1 = \sum_{t=1}^{4} (l_{1,t})(x^{1,t}), \sum_{t=1}^{4} (l_{1,t}) = 1, l_{1,t} \in \{0,1\} \right\} \tag{A.125}$$

$$X^2 = \left\{ x_2 \in \mathbb{R}_+^1 : x_2 = \sum_{t=1}^{5} \left(l_{2,t} \right) \left(x^{2,t} \right), \sum_{t=1}^{5} \left(l_{2,t} \right) = 1, \; l_{2,t} \in \{0,1\} \right\} \tag{A.126}$$

$$X^3 = \left\{ x_3 \in \mathbb{R}_+^1 : x_3 \geq 0 \right\} \tag{A.127}$$

or written explicitly as:

$$\text{(IPM)} \quad \begin{array}{l} z = \max\!:(3)(0)\,l_{1,1} + (3)(1)\,l_{1,2} + (3)(2)\,l_{1,3} + (3)(3)\,l_{1,4} + (3)(4)\,l_{1,5} \\ \qquad (2)(0)\,l_{2,1} + (2)(1)\,l_{2,2} + (2)(2)\,l_{2,3} + (2)(3)\,l_{2,4} \end{array} \tag{A.128}$$

s.t.

$$\begin{array}{l} 0\,l_{1,1} + 1\,l_{1,2} + 2\,l_{1,3} + 3\,l_{1,4} + 4\,l_{1,5} \\ 0\,l_{2,1} + 1\,l_{2,2} + 2\,l_{2,3} + 3\,l_{2,4} + x_3 = 5.5 \end{array} \tag{A.129}$$

$$l_{1,1} + l_{1,2} + l_{1,3} + l_{1,4} = 1 \tag{A.130}$$

$$l_{2,1} + l_{2,2} + l_{2,3} + l_{2,4} + l_{2,5} = 1 \tag{A.131}$$

$$x_3 \geq 0 \tag{A.132}$$

$$l_{1,t} \in \{0,1\} \text{ for } t = 1,2,3,4 \tag{A.133}$$

$$l_{2,t} \in \{0,1\} \text{ for } t = 1,2,3,4,5 \tag{A.134}$$

The procedure to solve these problems is based on a linear programming relaxation of the integer programming problem known as the "Master linear problem" as follows.

$$\text{(LPM)}\; z = \max \sum_{k=1}^{K} \sum_{t=1}^{T_k} (c_k x^{k,t})\, l_{kt} \tag{A.135}$$

s.t.

$$\sum_{k=1}^{K} \sum_{t=1}^{T_k} (A_k x^{k,t})\, l_{k,t} = b \tag{A.136}$$

$\sum_{t=1}^{T_k} l_{k,t} = 1$ for $k = 1, 2, \ldots, K$ (A.137)

$l_{k,t} \geq 0$ for $t = 1, 2, .., T_k$ and $k = 1, 2, \ldots, K$ (A.138)

The only difference in the formulation is the relaxation of the binary constraint for the $l_{k,t}$ factors. This formulation has a column $\begin{pmatrix} c^k x \\ A^k x \\ e_k \end{pmatrix}$ for each $x \in X^k$.

The objective is to solve this relaxed problem using the simplex method but since there is a very large number of columns, then the variable to enter the basis is selected by solving an optimization problem for each of the *k* subproblems rather than by finding the reduced cost of each possible variable.

The set of variables $\{p_i\}_{i=1}^{m}$ will be used as the dual variables associated with the joint constraints (A.136) and the set of variables $\{m_k\}_{k=1}^{K}$ as dual variables associated with the convexity constraints (A.137).

The algorithm has five steps as follows:

Iteration 1, Step 0: Initialization

Find a set of feasible solutions, at least one for each subproblem.

Iteration1, Step 1: Solve the restricted linear programming master problem

(RLPM) $\tilde{z} = \max \tilde{c}^T \tilde{l}$ (A.139)

s.t.

$\tilde{A}\tilde{l} = \tilde{b}$ (A.140)

$\tilde{l} \geq 0$ (A.141)

$\tilde{A}$ is generated using the available set of columns (feasible solutions) and it is a subset of the matrix A composed of all the feasible points in the feasible region. The solution to this problem provides a primal optimal solution $\tilde{l}^*$ and a dual optimal solution (p, m).

Because the restricted linear programming problem has been created with a subset of feasible points of the master problem then it follows that any feasible solution to the restricted problem is feasible to the master problem.

Iteration 1, Step 2: Optimality Check.

We need to check whether the set of variables (p, m) is dual feasible for the master problem. But rather than evaluate all possible points, we solve the following optimization problem:

$$V_k = \max\{\left(c^{kT} - p\, A^k\right)^T x - m_k : x \in X^k\} \qquad \text{(A.142)}$$

Iteration 1, Step 3: Stopping Criterion

If $V_k = \max\{\left(c^{kT} - p\, A^k\right)^T x - m_k : x \in X^k\} = 0$ for *k = 1, 2 ,… K* then the solution (p, m) is dual feasible for the master problem so

$$z^{LPM} \le \sum_{i=1}^{m} p_i b_i + \sum_{k=1}^{K} m_k \qquad \text{(A.143)}$$

When the value of the objective function z^{LPM} gets to be equal to the value of its upper bound then the solution $\tilde{l}$ is optimal for (A.135) - (A.138).

An alternative criterion is to check if the complicating constraint is met, if so then the solution at hand is optimal.

Iteration 1, Step 4: Generating a New Column if $V_k = \max\{\left(c_k - p\, A^k\right)^T x - m_k : x \in X^k\} > 0$ for some k implies that the k^{th} column would improve the value of the objective function if it enters the basis. Now the problem should be re-optimized using this k^{th} column entering the basis.

A 2.3.5. Example

Following this algorithm for the small example (A.119) - (A.123) we have:

A 2.3.5.1. Iteration 1, Step 1: Initialization

Consider the initial feasible point $x = (1,1)$

We solve the Restricted Linear Programming Master problem

(RLPM) $z^{LPM} = \max{:}(3)(1)\, l_{1,1} + (2)(1)\, l_{2,1}$ (A.144)

s.t.

$1 l_{1,1} + 1 l_{2,1} + x_3 = 5.5$ (A.145)

$x_3 \geq 0$ (A.146)

$l_{1,1} = 1$ (A.147)

$l_{2,1} = 1$ (A.148)

Solving this problem gives the following optimal solution:

$z^{LPM} = 5,\ l_{1,1} = 1,\ l_{2,1} = 1,\ x_3 = 3.5,\ p = 0,\ m_1 = 3,\ m_2 = 2$

A 2.3.5.2. Iteration 1, Step 2: Optimality Check.

We need to solve the following problems:

$V_k = \max\{\left(c^{kT} - p\, A^k\right)^T x - m_k : x \in X^k\}$ (A.149) written explicitly:

$$V_1 = \max\{(3-(0)(1))\,x_1 - 3 : x_1 \le 4.5\} \tag{A.150}$$

$$V_2 = \max\{(2-(0)(1))\,x_2 - 2 : x_1 \le 3.5\} \tag{A.151}$$

whose solutions are:

$$V_1 = 10.5, x_1 = 4.5 \tag{A.152}$$

$$V_2 = 7, x_2 = 3.5 \tag{A.153}$$

Since both are positive the current value of $z^{LPM} = 5$ can be improved by generating a new column. We then arbitrarily introduce $x_1 = 4.5$ and solve the restricted linear programming master problem again as follows:

(RLPM) $z^{LPM} = \max{:}(3)(1)\,l_{1,1} + (3)(4.5)\,l_{1,2} + (2)(1)\,l_{2,1}$ (A.154)

s.t.

$$1l_{1,1} + 4.5\,l_{1,2} + 1l_{2,1} + x_3 = 5.5 \tag{A.155}$$

$$l_{1,1} + l_{1,2} = 1 \tag{A.156}$$

$$l_{2,1} = 1 \tag{A.157}$$

Solving this problem gives the following optimal solution:

$z^{LPM} = 15.5$, $l_{1,1} = 0$, $l_{1,2} = 1$, $l_{2,1} = 1$, $x_3 = 0$, $p = 0$, $m_1 = 13.5$, $m_2 = 2$

A 2.3.5.3. Iteration 1, Step 2: Optimality Check.

We need to solve the following problems:

$$V_k = \max\{\left(c^{kT} - p\,A^k\right)^T x - m_k : x \in X^k\} \tag{A.158}$$

written explicitly:

$$V_1 = \max\{\left(3-(0)(1)\right)x_1 - 13.5 : x_1 \leq 4.5\} \tag{A.159}$$

$$V_2 = \max\{\left(2-(0)(1)\right)x_2 - 2 : x_2 \leq 3.5\} \tag{A.160}$$

whose solutions are:

$$V_1 = 0, x_1 = 4.5 \tag{A.161}$$

$$V_2 = 7, x_2 = 3.5 \tag{A.162}$$

Since V_2 is positive the current value of $z^{LPM} = 15.5$ can be improved by entering $x_2 = 3.5$. Then we solve again the restricted linear programming master problem:

(RLPM) $$z^{LPM} = \max{:}(3)(1)l_{1,1} + (3)(4.5)l_{1,2} + (2)(1)l_{2,1} + (2)(3.5)\,l_{2,2} \tag{A.163}$$

s.t.

$$1l_{1,1} + 4.5l_{1,2} + 1l_{2,1} + 3.5l_{2,2} + x_3 = 5.5 \tag{A.164}$$

$$x_3 \geq 0 \tag{A.165}$$

$$l_{1,1} + l_{1,2} = 1 \tag{A.166}$$

$$l_{2,1} + l_{2,2} = 1 \tag{A.167}$$

Solving this problem gives the following optimal solution:

$z^{LPM} = 15.5$, $l_{1,1} = 0$, $l_{1,2} = 1$, $l_{2,1} = 1$, $l_{2,2} = 0$, $x_3 = 0$, $p = 2$, $m_1 = 4.5$, $m_2 = 0$

A 2.3.5.4. Step 2: Optimality Check.

We need to solve the following problems:

$$V_k = \max\{\left(c^{kT} - p\,A^k\right)^T x - m_k : x \in X^k\} \tag{A.168}$$

written explicitly:

$$V_1 = \max\{\left(3-(2)(1)\right)x_1 - 4.5 : x_1 \le 4.5\} \tag{A.169}$$

$$V_2 = \max\{\left(2-(2)(1)\right)x_2 - 0 : x_2 \le 3.5\} \tag{A.170}$$

whose solutions are:

$$V_1 = 0, x_1 = 4.5 \tag{A.171}$$

$$V_2 = 0, x_2 = 3.5 \tag{A.172}$$

Since both are zero we conclude that there is no other point that would improve the objective function. Therefore, the solution (4.5,1) is optimal and the algorithm stops. Note that this solution is not the optimal solution to the original problem. This leads us to look into the strength of the formulation.

Wolsey (1998) has the following proposition:

$$z^{LPM} = \max\left\{\sum_{k=1}^{K} c^{kT}x^k : \sum_{k=1}^{K} A^k x^k = b, x^k \in conv(x^k)\text{for } k = 1,2,\dots,K\right\} \tag{A.173}$$

A 2.3.6. Benders Decomposition

Similar to the Dantzig-Wolfe decomposition method, there are problems that can be decomposed in blocks but slightly different than before.

Consider a problem that has the following structure:

$$Min: c^1x^1 + c^2x^2 + \dots + c^Kx^K \tag{A.174}$$

s.t.

$$\begin{array}{l} D^1x^1 \qquad\qquad + A_1^K x^K \le d_1 \\ \quad + D^2x^2 \qquad + A_2^K x^K \le d_2 \\ \dots \\ \qquad + D^{K-1}x^{K-1} + A_M^K x^K \le d_M \\ x^1 \in \mathbb{R}^{k_1}, x^2 \in \mathbb{R}^{k_2}, \dots, x^K \in \mathbb{R}^{k_n} \end{array} \tag{A.175}$$

In this case, there is a group of variables x^K that prevent the decomposition of the problem in *K-1* blocks. These variables are known as “complicating variables” (Conejo et al., 2003).

In the previous case we had a set of constraints that involve variables from any block (complicating constraints), and we had another group of constraints with a set of variables that appear only in those constraints. In this case we have a set of variables that appear only in a group of constraints and then we have another set of “complicating variables” that appear on all constraints.

An algorithm to solve this kind of problems is known as Benders decomposition. We will not explain the details of the method here since it will not be used to solve any or the problems presented in this dissertation work. The section is included for briefing the reader on the existence of such algorithm.

A 2.4. Duality Gap

Previously we briefly mentioned the “duality gap” problem as a downside of the weighting method to find all the Pareto optimal points. Here we will expand on this issue since it is considered of utmost importance in finding the Pareto optimal set for this problem.

Because the variables to decide if a parcel gets chosen to be developed or not are binary variables the feasible region is non-convex. Therefore, the convex combinations of solutions to the problem are not necessarily feasible. Moreover, the contour of the feasible region could lead to missing Pareto optimal points as the weights of the objectives are changed. To illustrate this point consider our previous single-objective problem with a second objective function (A.177):

$$z^1 = \max: 3x_1 + 2x_2 \tag{A.176}$$

$$z^2 = \max: -6x_1 - 5x_2 \tag{A.177}$$

s.t.

$$x_1 + x_2 \leq 5.5 \tag{A.178}$$

where

$$x_1 \leq 4.5 \tag{A.179}$$

$$x_2 \leq 3.5 \tag{A.180}$$

This problem has a graphical representation already shown in Figure A.71. The interesting aspect is that from the 17 feasible points in the feasible region, 10 of them are Pareto optimal. The data presented in Table A.27 show the value of the variables for all feasible points and the evaluation of the two objective functions. Values in gray represent dominated points.

x_1	x_2	**max** z_1	**max** z_2
4	1	13	-29
4	0	12	-24
3	2	11	-28
3	1	10	-23
3	0	9	-18
2	3	9	-27
2	2	8	-22
2	1	7	-17
2	0	6	-12
1	3	6	-21
1	2	5	-16
1	1	4	-11
1	0	3	-6
0	3	3	-15
0	2	2	-10
0	1	1	-5
0	0	0	0

Table A.27 Feasible points with objective function values

If the weighting method is applied to solve this problem, one would combine both objectives and the formulation would be:

$$z = \max\text{: } w_1(3x_1 + 2x_2) + w_2(-6x_1 - 5x_2) \tag{A.181}$$

s.t.

$$x_1 + x_2 \leq 5.5 \tag{A.182}$$

$$x_1 \leq 4.5 \tag{A.183}$$

$$x_2 \leq 3.5 \tag{A.184}$$

$$x_1, x_2 \in \mathbb{Z}_+^1 \tag{A.185}$$

As we vary the weights for w_1 and w_2 we can obtain some of the Pareto optimal points. Table A.28 presents 21 different values of w_1 and w_2 that produced only three of the ten Pareto optimal points.

Point	w_1	w_2	z	x_1	x_2
0	1.00	0.00	14.00	4	1
1	0.95	0.05	11.85	4	1
2	0.90	0.10	9.70	4	1
3	0.85	0.15	7.55	4	1
4	0.80	0.20	5.40	4	1
5	0.75	0.25	3.25	4	1
6	0.70	0.30	1.20	4	0
7	0.65	0.35	0.00	0	0
8	0.60	0.40	0.00	0	0
9	0.55	0.45	0.00	0	0
10	0.50	0.50	0.00	0	0
11	0.45	0.55	0.00	0	0
12	0.40	0.60	0.00	0	0
13	0.35	0.65	0.00	0	0
14	0.30	0.70	0.00	0	0
15	0.25	0.75	0.00	0	0
16	0.20	0.80	0.00	0	0
17	0.15	0.85	0.00	0	0
18	0.10	0.90	0.00	0	0
19	0.05	0.95	0.00	0	0
20	0.00	1.00	0.00	0	0

Table A.28 Pareto optimal points obtained by changing weights

The gradient of a linear expression is formed by the coefficients of the variables, there is one gradient for the objective function formed as a vector with the coefficients of the variables (Steuer, 2004). Since the coefficients are a function of *u*, there are different gradients for different values of *u*. The feasible region, and some of the gradients of these weighted function are presented in Figure A.72.

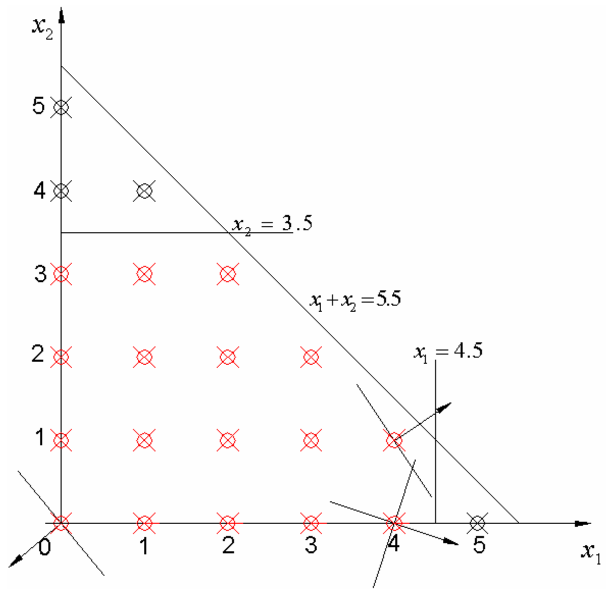


Figure A.72 Feasible region, and gradients of the weighted objective function

The gradients are important because they indicate the direction of greatest increase of the linear function (Steuer, 2004).

Notice how point (3, 1) with objective functions (11,-23) wasn't found as result from the weighted method given the step between weights, since there is no combination of weights that can produced this point.

However, if we use the constraint method instead, we would obtain all the Pareto optimal points in the feasible region.

Consider the problem (A.176) - (A.180) written as:

$$z^1 = \max : 3x_1 + 2x_2 \qquad \text{(A.186)}$$

s.t.

$$-6x_1 - 5x_2 \geq \underline{z}^2 \qquad \text{(A.187)}$$

$$x_1 + x_2 \leq 5.5 \qquad \text{(A.188)}$$

$$x_1 \leq 4.5 \qquad \text{(A.189)}$$

$$x_2 \leq 3.5 \qquad \text{(A.190)}$$

$$x_1, x_2 \in \mathbb{Z}_+^1 \qquad \text{(A.191)}$$

Where $\underline{z}^2$ is a lower bound on the value of the second objective function. As we solve this problem for different values of $\underline{z}^2$, we obtain different optimal points.

	$\underline{z}^2$	z^1	z^2	x
1	-30	14	-29	(4,1)
2	-28	13	-28	(3,2)
3	-27	12	-27	(2,3)
4	-26	12	-24	(4,0)
5	-23	11	-23	(3,1)
6	-22	10	-22	(2,2)
7	-21	9	-21	(1,3)
8	-20	9	-18	(3,0)
9	-17	8	-17	(2,1)
10	-16	7	-16	(1,2)
11	-15	6	-15	(0,3)
12	-14	6	-12	(2,0)
13	-11	5	-11	(1,1)
14	-10	4	-10	(0,2)
15	-9	3	-6	(1,0)
16	-5	2	-5	(0,1)
17	-4	0	0	(0,0)

Table A.29 Solutions to the constraint method for different values of $\underline{z}^2$

There are some aspects worth noting in Table A.29. First, we used integer values of $\underline{z}^2$ because we noted that the function $-6x_1 - 5x_2$ would produce integer values for all integer combinations of x_1 and x_2. Second, there seems to be values missing for example -8, -7, -6 but upon a closer inspection we observe that for

$\underline{z}^2 = -9$, the optimal solution provides a value of $z^2 = -6$ so any value of $\underline{z}^2$ between -9 and -6 will provide the same solution as $\underline{z}^2 = -9$. Therefore the next value to test is $z^2 = -5$. Also, not all the points obtained are Pareto optimal, for example, when comparing solutions 11 and 12 we note that the values for z^1 are the same while z^2 is better for solution 12 as compared to 11. This means that solution 12 dominates solution 11 and therefore solution 11 is not Pareto optimal. Lastly, there are no positive values of $\underline{z}^2$ because the function $-6x_1 - 5x_2$ is no positive for all values of x_1 and x_2.

The conclusion of this example is that the constrained method is a better method to search for the Pareto optimal set as compared to the weighting method but still one need to check if any of the points obtained are dominated by others. This still does not guarantee that all the obtained points are Pareto optimal, since we could miss some points by an inappropriate selection of $\underline{z}^2$.

Cohon (1978) warns about the possibility of obtaining inferior solutions using the constraint method saying that

> "all the constraints on objectives should be binding at the optimal solution to the constrained problem. If this is not the case and if there are alternative optima to the constrained problem, then some of these optimal solutions may be inferior alternatives for the original multiobjective problem."

Since we are dealing with integer variables, we have the additional complication that the objective constraints need not be binding in order for the solution to be optimal. Consider for example the case where we do not have an integer lower bound $\underline{z}^2$. Then since the left hand side is integer, and the right hand side is fractional there must be a slack on the constraint but the solution may still be Pareto optimal. Consider for example $\underline{z}^2 = -10.5$ when solving (A.186) - (A.191) we obtained $x = (0,2)$, $z^1 = 4$, $z^2 = -10$. This is a Pareto optimal point regardless of the constraint for z^2 having slack.

Appendix 3 Introduction to Graph Theory

This chapter is an introduction to the graph theory concepts used to prepare the algorithm presented in Chapter 4. The topics to be covered are:

- Definitions Related to Graph Theory
- The Shortest Path Problem
    - Graph Scanning Algorithm
    - Bellman's Principle of Optimality
- Solutions to the Shortest Path Problem
    - Algorithmic Approach
    - Mixed Integer Programming Formulations
- The Minimum Spanning Tree Problem
- Solutions to the Minimum Spanning Tree
    - Algorithms
    - Mixed Integer Programming Formulations

Many of the algorithms explained in the following sections are applied in an algorithm to solve the Land Development Planning Problem with embedded minimum spanning tree presented in Chapter 6.

A 3.1. Definitions Related to Graph Theory

This section presents a brief introduction of the graph theory concepts that are required to completely follow the discussion of minimum spanning trees and their relationship with compactness and infrastructure as discussed in Chapter 4. It also provides the basis for the mixed integer programming formulation presented later.

An undirected graph *G* is a collection of nodes *V(G)* and edges *E(G)* that connects the nodes by pairs. A directed graph *D* or digraph is similar to the undirected graph with the difference that the edges are directed so the order of the nodes that define the edge is important. When an edge *e* that joins two nodes *v* and *w*, we say that *v* and *w* are adjacent; *v* and *w* are neighbors. If *v* is an endpoint of an edge *e* then *v* is incident to *e*. (Korte and Vygen, 2000).

A path is a sequence of edges that connects two nodes. A graph is connected if there is a path from each node to every other node in the graph. A non-connected graph is made up of connected pieces called components, each component consists of vertices that are all reachable from one another (Gross and Yellen, 1999). An edge *e* is a bridge of *G* if the graph *G-e* has more connected components than *G*. A graph has a cycle (circuit) if there are at least two different paths between two nodes. An undirected graph without a cycle is called a forest. A connected forest is a tree. A spanning tree of a graph *G* is a tree that contains all the nodes of *G*. (Korte and Vygen, 2000).

For undirected graphs G and $X \subset V(G)$ we define a cut:

$$d(X) = E(X, V(G) \setminus X) \tag{A.192}$$

For directed graphs D and $X \subset V(D)$ we define the cuts out of the set and into the set by Magnanti and Wolsey (1995):

$$d^{+}(X) = e_{ij} \in E(X, V(D) \setminus X) : i \in X,\ j \in V(D) \setminus X \tag{A.193}$$

and

$$d^{-}(X) = e_{ij} \in E(V(D) \setminus X, X) : i \in V(D) \setminus X,\ j \in X \tag{A.194}$$

respectively.

In words this is the set of all incident arcs to a set of nodes X. An example of these different types of cuts is presented in Figure A.73.

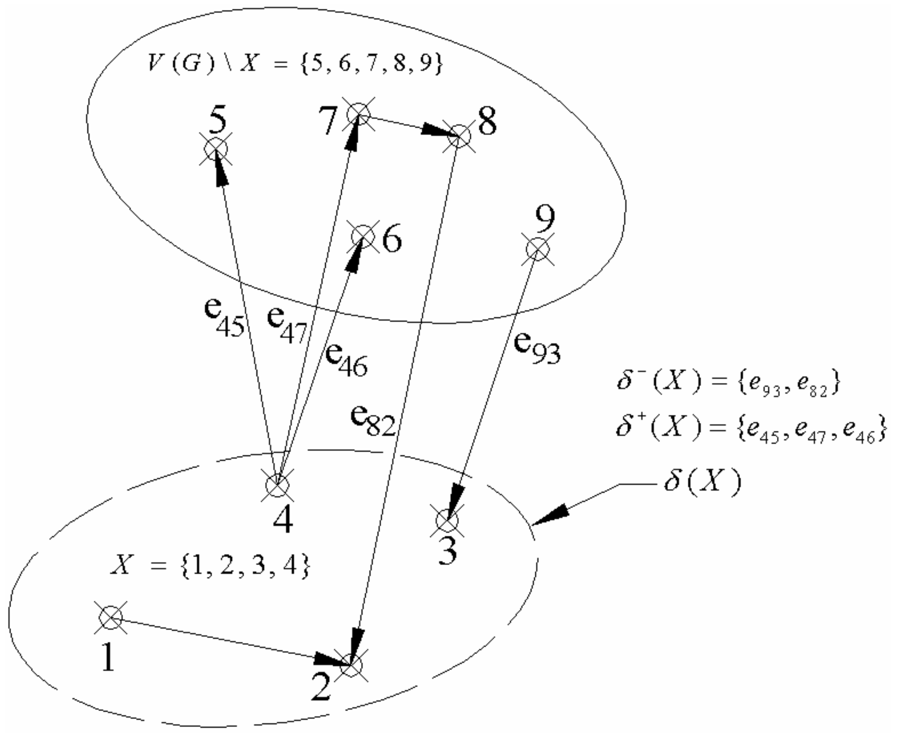

$V(G) \setminus X = \{5,6,7,8,9\}$
5
7
8
6
9
e_{45}
e_{47}
e_{46}
e_{93}
e_{82}
$\delta^-(X) = \{e_{93}, e_{82}\}$
$\delta^+(X) = \{e_{45}, e_{47}, e_{46}\}$
$\delta(X)$
4
3
$X = \{1,2,3,4\}$
1
2

Figure A.73 Cutset around set of nodes X={1,2,3,4}

Theorem 6 Equivalent statements of for a MST

Let G be an undirected graph on n vertices. Then the following statements are equivalent:

(a) G is a tree (i.e. is connected and has no circuits)

(b) G has n-1 edges and no circuits

(c) G has n-1 edges and is connected

(d) G is a minimal connected graph (i.e., every edge is a bridge)

(e) G is a minimal graph with $d(X) \neq \emptyset$ *for all* $\emptyset \neq X \subset V(G)$

(f) G is a maximal cycle free graph (i.e. the addition of any other edge would create a cycle)

(g) G contains a unique path between any pair of vertices.

For a proof of this theorem the reader is referred to Korte and Vygen (2000).

A 3.2. The Shortest Path Problem

The shortest path problem is one of the best known combinatorial optimization problems (Korte and Vygen, 2000), also it is among the simplest network flow problems (Ahuja et al., 1995). It consists of finding the shortest path $P_{[u,v]}$ between two nodes *u* and *v* of a graph *G*, or determines that none exists (in the case where the nodes are not connected). The shortest path is that for which the sum of the edge weights is a minimum. This problem is difficult to solve if the weights are arbitrary, in particular negative weights adds an extra complication because some paths can end up with negative values (Korte and Vygen, 2000). For the purposes of this dissertation, unless noted, all weights are to be considered nonnegative which greatly reduces the time to achieve a solution. Given a graph G with weights $c: E(G) \rightarrow R$,c is called conservative if there is no cycle with negative total weight.

Connectivity is very important when searching for a shortest path, there are algorithms that find if there is a connection between two nodes of a graph. In particular there is a general algorithm that can find the path from a node *s* to all other nodes that are reachable from *s*. We call this algorithm the Graph Scanning Algorithm (Korte and Vygen, 2000).

A 3.2.1. Graph Scanning Algorithm[9]

This algorithm is relevant to the dissertation work, because in the algorithm presenter in Chapter 6, one of the steps is to find all disconnected components in the graph. The graph scanning algorithm was implemented to find such components.

Given a graph G (directed or undirected) and some vertex $s \in V(G)$, this algorithm finds the set R of vertices reachable from s, and the set $T \subseteq E(G)$ such that (R,T) is a tree rooted at s.

Steps

(1) Set $R=\{s\}$, $Q=\{s\}$ and $T=\emptyset$

(2) If $Q=\emptyset$ then stop, else choose a $v \in Q$

(3) Choose a $w \in V(G) \setminus R$ with $e=(v,w) \in E(G)$ if there is no such w then set $Q = Q \setminus \{v\}$ and go to (2)

(4) Set $R = R \cup \{w\}$, $Q = Q \cup \{w\}$, $T = T \cup \{e\}$ and go to (2)

Step 1 is an initialization step, R is the set of nodes that can be reached from s, Q is a queue for the nodes to be evaluated, and T is the list of edges that connect s with the other nodes in G. Step 2 is a termination check, if the queue is empty then the procedure stops, otherwise a node v is chosen from the set of nodes stored in the queue Q. Step 3 finds a node w that is connected to v, if none can be found then v is removed from the queue. Step 4 adds the node w to the set of nodes that can be reached from s, updates the queue by adding node w and updates the tree by adding the edge e.

[9] Unless referenced otherwise this section and its subsections were extracted from Korte and Vygen, (2000).

The implementation of this algorithm requires information about the graph G, one way to provide that information is by using the incidence matrix. The incidence matrix of an undirected graph G is given by $A = \left(a_{v,e}\right)_{v \in V(G), e \in E(G)}$ where:

$$a_{v,e} = \begin{cases} 1 \text{ if } v \in E(G) \\ 0 \text{ if } v \notin E(G) \end{cases} \qquad \text{(A.195)}$$

For example consider the network presented in Figure A.74.

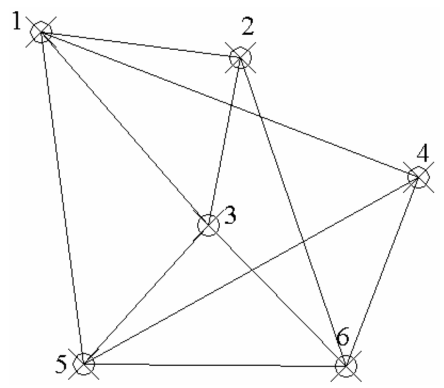


Figure A.74 Small undirected graph example

The incidence matrix corresponding to this network is:

		Edges										
		1,2	1,3	1,4	1,5	2,3	2,6	3,5	3,6	4,5	4,6	5,6
Nodes	1	1	1	1	1							
	2	1				1	1					
	3		1			1		1	1			
	4			1						1	1	
	5				1			1		1		1
	6						1		1		1	1

Table A.30 Incidence matrix of small undirected graph presented in Figure A.74

For a directed graph, the elements of the matrix are given by

$$a_{v,e} = \begin{cases} -1 \text{ if } v = i \\ 1 \text{ if } v = j \\ 0 \text{ if } v \neq \{i, j\} \end{cases} \qquad \text{(A.196)}$$

where *e=(i,j)* is the edge joining nodes *i* and *j*. For example consider the network presented in Figure A.75

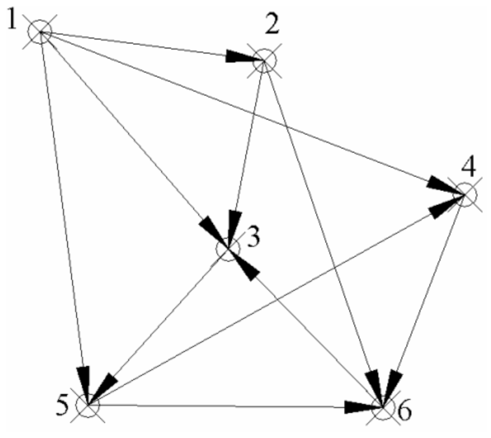


Figure A.75 Small directed graph example

The incidence matrix corresponding to this network is:

		Edges										
		1,2	1,3	1,4	1,5	2,3	2,6	3,5	4,6	5,4	5,6	6,3
Nodes	1	-1	-1	-1	-1							
	2	1				-1	-1					
	3		1			1		-1				1
	4			1					-1	1		
	5				1			1		-1	-1	
	6						1		1		1	-1

Table A.31 Incidence matrix of small example presented in Figure A.75

Using such matrix to represent a graph is not the most efficient representation because each column contains only two non zero entries. A better way to store the information is by using the adjacency matrix $A = \left(a_{v,w}\right)_{v,w \in V(G)}$ where:

$$a_{v,w} = \begin{cases} 1 \text{ if } (v,w) \in E(G) \\ \quad 0 \text{ otherwise} \end{cases} \qquad \text{(A.197)}$$

An adjacency matrix for the network presented in Figure A.75 is presented in Table A.32.

	1	2	3	4	5	6
1		1	1	1	1	
2			1			1
3					1	
4						1
5				1		1
6			1			

Table A.32 Adjacency matrix for the network presented in Figure A.75

Yet, there is still a better way to store the information of the graph, just by storing the edges incidence to each vertex in a so called adjacency list. There could be one ordered list of all edges sorted by vertex, or a list of edges per vertex. For example, consider the small network presented in Figure A.75, a list of all edges sorted by vertex is presented in Table A.33.

Edge
1,2
1,3
1,4
1,5
2,3
2,6
3,5
4,6
5,4
5,6
6,3

Table A.33 Adjacency list of edges sorted by vertex for Figure A.75

A list of edges by vertex are presented in Table A.34.

Node 1	Node 2	Node 3	Node 4	Node 5	Node 6
1,2	2,3	3,5	4,6	5,4	6,3
1,3	2,6			5,6	
1,4					
1,5					

Table A.34 Adjacency list of edges per node for network in Figure A.75

There are two typical methods to implement this algorithm, depth first search (DFS) and breadth first search (BFS), they depend on the choice to select the vertex in step (3) they are as follows:

A 3.2.2. Depth First Search

In this implementation we choose the node $v \in Q$ that was last to enter Q. The implementation follows a Last In – First Out (LIFO stacking) strategy.

A 3.2.3. Breadth First Search

In this implementation we choose the node $v \in Q$ that was first to enter Q. The implementation follows a First In – First Out (FIFO stacking) strategy. With a small modification to the general algorithm, the BFS contains the shortest path from s to any vertex reachable from s given that the weight of each edge is one.

Modification:

In step (1) include the statement: $l(s) = 0$, in step (4) include the statement $l(w)=l(v)+1$. We then have $l(v) = dist_{(R,T)}(s,v)$ for all $v \in R$ at any stage of the algorithm. Where $l(v)$ represents the distance in number of edges from the root to the node v.

A 3.2.4. Bellman's Principle of Optimality

Some of the solutions to the shortest path problem are based on the so called Bellman's principle of optimality.

Theorem 7 Bellman's Principle of Optimality

Given s and w as two different vertices of a digraph G with nonnegative weights, if the edge $e=(v,w)$ *is the final edge of some shortest path* $P_{[s,w]}$ *from s to w then the path* $P_{[s,v]}=P_{[s,w]}-e$ *is the shortest path from the node s to the node* v. The interested reader is referred to the proof as presented in (Korte and Vygen, 2000).

This result also holds for undirected graphs with nonnegative weights and for acyclic digraphs with arbitrary weights.

A 3.3. Solutions to the Shortest Path Problem

There are many approaches at least to find a shortest path in a graph. For example, one set of approaches is based on the implementation of algorithms, and the other one is by the application of network optimization concepts recurring to solve a mixed integer programming formulation. Both approaches will be presented here.

A 3.3.1. Algorithmic Approach

Because some of the algorithms to solve the shortest path are based on Bellman's principle of optimality, those algorithms also find the shortest path between a node and all other nodes in the graph (see BFS presented before). This holds because at the beginning it is unknown which nodes belong to the shortest path $P_{[s,v]}$ then it is easy to compute all shortest path to each node t until we find v, the information can be efficiently stored by saving only the final edge of each path (Korte and Vygen, 2000).

A 3.3.2. Modification of the BFS Algorithm

One could replace the edge e in the BFS algorithm by a path of length $c(e) = l(v, w), e \in E(G)$ which could introduce an exponential number of edges.

A 3.3.3. Dijkstra's Algorithm

Given a graph G with associated weights $c : E(G) \to R_+$ and some vertex $s \in V(G)$, this algorithm finds the shortest paths from s to all $v \in V(G)$ and their lengths.

Steps

(1) Set $R = \varnothing, l(s) = 0, l(v) = \infty \; \forall v \in V(G) \setminus \{s\}$

(2) Find a vertex $v \in V(G) \setminus R : l(v) = \min_{w \in V(G) \setminus R} l(w)$

(3) Set $R = R \cup \{v\}$

(4) For all $w \in V(G) \setminus R : (v, w) \in E(G)$ do:

a. If $l(w) > l(v) + c((v, w))$ then $l(w) = l(v) + c((v, w)), p(w) = v$

(5) If $R \neq V(G)$ then go to (2)

Step 1 is an initialization step where R is a node checklist used to terminate the algorithm. The $l(v)$'s are the list of distances from the node s to any node v. Step 2 finds the closest node in the graph G that is not already into the checklist R to any node in the list R. Step 3 adds the node found in the checklist. Step 4 looks at all nodes in the graph G that are not in the checklist R such that there exists an edge (v,w) and updates the length of the shortest path from the node s to the node w if a shortest path has been found. Step 5 checks for termination if the checklist includes all nodes of the graph.

Theorem 8 Correctness of Dijkstra's algorithm

Dijkstra's algorithm finds a shortest path between two nodes, and its running time is $O(n^2)$, *where* $n = |V(G)|$. For a proof of this theorem the reader is referred to Korte and Vygen (2000).

A 3.3.4. Moore-Bellman-Ford Algorithm

Given a digraph G with conservative weights $c : E(G) \to R$ and some vertex $s \in V(G)$, this algorithm finds all the shortest paths from s to all $v \in V(G)$ and their lengths.

Steps

```
(1) Set $l(s) = 0, l(v) = \infty \ \forall v \in V(G) \setminus \{s\}$
(2) For $i$=1 to $n$-$1$ do:
    a. For each edge $(v, w) \in E(G)$ do
        i. If $l(w) > l(v) + c((v, w))$ then $l(w) = l(v) + c((v, w)), p(w) = v$
```

Theorem 9 Correctness of Moore – Bellman – Ford algorithm

The Moore-Bellman-Ford algorithm finds a shortest paths from s to all other reachable nodes in the network, and its running time is $O(nm)$ *where* $n = |V(G)|, m = |E(G)|$. For a proof of this theorem the reader is referred to Korte and Vygen (2000).

A 3.3.5. Floyd-Warshall Algorithm

Given a digraph G with conservative weights $c: E(G) \to R$, and nodes $V(G) = \{1,...,n\}$ this algorithm finds the shortest path between all pairs of nodes s and v where $s, v \in V(G)$ and their lengths.

Steps

```
(1) Set $l_{i,j} = c((i,j)) \forall (i,j) \in E(G)$
        $l_{i,j} = \infty \forall (i,j) \in \left(V(G) \times V(G)\right) \setminus E(G) : i \neq j$
        $l_{ii} = 0 \forall i$
        $p_{i,j} = i \forall i, j \in V(G)$
(2) For $j$=1 to $n$ do
    a. For i = 1 to n do: If $i \neq j$ then:
        i. For k = 1 to n do: If $k \neq j$ then
            1. If $l_{i,k} > l_{i,j} + l_{j,k}$ then set $l_{i,k} = l_{i,k} + l_{j,k}, p_{i,k} = p_{jk}$
```

Theorem 10 Correctness of the Floyd-Warshall algorithm

The Floyd-Warshall algorithm finds all shortest paths between all pairs of nodes v and s and their lengths with a running time of $O(n^3)$. For a proof of this theorem the reader is referred to Korte and Vygen (2000).

A 3.3.6. Mixed Integer Programming Formulations

A 3.3.6.1. Minimum Cost Flow Problem

Some of these formulations have their origin in a problem called the minimum cost flow problem. This problem consists in finding the minimum cost of shipment for a commodity along a network , satisfying the demand constraints at each node.

The decision variables are flows along the arcs so the variable x_{ij} represents the flow from node i to node j. The minimum cost flow problem can be formulated as (Ahuja et al. 1999):

Minimize $\sum_{(i,j)\in E(G)} c_{ij}x_{ij}$ (A.198)

s.t.

$$\sum_{\{j:(i,j)\in E(G)\}} x_{ij} - \sum_{\{j:(j,i)\in E(G)\}} x_{ji} = b(i), \forall i \in N \quad \text{(A.199)}$$

$$l_{ij} \le x_{ij} \le u_{ij}, \forall (i,j) \in E(G) \quad \text{(A.200)}$$

The constraints in (A.199) balance the mass flow at each node *i*, while the constraints (A.200) prevent to exceed flow the capacity of each arc (*i*,*j*).

Note how if when using the above formulation we set:

$$b(s)=1, b(t)=-1, b(w)=0 : w\in V(G)\setminus\{s,t\} \quad \text{(A.201)}$$

$$l_{ij}=0, u_{ij}\ge 1 \forall (i,j)\in A \quad \text{(A.202)}$$

then the solution to the minimum cost flow provides the shortest path between *s* and *t* by sending one unit from *s* to *t*.

To find the shortest path from a node *s* to all other nodes in the network, the formulation can be changed to set:

$$b(s)=(n-1), b(w)=-1 : w\in V(G)\setminus\{s\} \quad \text{(A.203)}$$

$$l_{ij}=0, u_{ij}=(n-1)\forall (i,j)\in E(G) \quad \text{(A.204)}$$

The value *n-1* for the arc capacity wouldn't set any unnecessary restrictions on the flow. By sending one unit of product from node s to all other nodes at minimum cost we are finding the shortest paths.

A 3.4. The Minimum Spanning Tree Problem

By definition a minimum spanning tree (MST) is a spanning tree of minimum weight[10]. In other words, a MST is a tree that connects all the nodes of G and the sum of the distance of all edges is minimum (Ahuja et al., 1995; Papadimitriou and Steiglitz, 1998; Jungnickel, 1999; and Korte and Vygen, 2000).

A theorem from Cayley (1889) proves that the number of spanning trees in a graph with n nodes is given by

$$n^{n-2} \tag{A.205}$$

It would be an extremely laborious task to identify each one of these trees, measure the total length and then find those with minimum weight. Fortunately as with some other combinatorial problems there are other procedures to reach the solution without resorting to an exhaustive search.

A 3.5. Solutions to the Minimum Spanning Tree

We provide two different approaches to solve the MST, the first one is based on the algorithms developed by Boruvka, Prim, and Kruskal among others. The second approach is using mixed integer programming to create formulations that can be used to find the solution to the MST.

[10] The minimum weight is a general term which associates a weight or a cost to each edge of the tree, for our purposes this tree can also be said to be of minimum length.

A 3.6. Algorithms

A 3.6.1. Prim (1930)

Given a connected, undirected graph G with weights $c: E(G) \to R$, this algorithm finds a spanning tree T of minimum weight.

Steps

```
(1) Choose $v \in V(G)$. Set $T = (\{v\}, \emptyset)$
(2) While $V(T) \neq V(G)$ do
    a. Choose an edge $e \in d_G(V(T))$ of minimum cost. Set $T = T + e$
```

Theorem 11 Correctness of Prim's algorithm

Prim's algorithm finds a MST and its running time is $O(n^2)$. For a proof of this theorem the reader is referred to Korte and Vygen (2000).

A 3.6.2. Kruskal (1956)

Given a connected undirected graph G with weights $c: E(G) \to R$, this algorithm finds a spanning tree T of minimum weight.

Steps

```
(3) Sort the edges such that $c(e_1) \leq c(e_2) \leq ... \leq c(e_m)$
(4) Set $T = (V(G), \emptyset)$
(5) For I = 1 to m do:
    a. if $T + e_i$ contains no circuit then set $T = T + e_i$
```

Theorem 12 Correctness of Kruskal's algorithm

Kruskal's Algorithm finds a MST and can be implemented to run in $O(mn)$ running time, can also be implemented to run in $O(m \log n)$. For a proof of this theorem the reader is referred to Korte and Vygen (2000).

A 3.6.3. Mixed Integer Programming Formulations

There has been some work in finding the solution to the minimum spanning tree using mixed integer programming by various researchers, the following is a collection of some of the research done in this area. We do not intend to be exhaustive but we consider the list to cover the most relevant formulations that we found to apply into the land development problem.

A 3.6.4. Polyhedral Description or Packing Formulation

Edmonds (1970) presented a formulation to find the minimum spanning tree called the polyhedral description of the MST. Then Magnanti and Wolsey (1995) presented this formulation with a slightly different notation as the packing formulation. We present the notation of Magnanti and Wolsey (1995). Given a connected undirected graph G with $n = |V(G)|$ nodes, then a MST can be found by solving:

$$\text{Min: } \sum_{e \in E(G)} w_e x_e \tag{A.206}$$

s.t.

$$\sum_{e \in E(G)} x_e = n - 1 \tag{A.207}$$

$$\sum_{e \in E(G)} x_e \le |S| - 1 \forall S \subset V(G) \tag{A.208}$$

$$x_e \in \{0,1\} \forall e \in E(G) \tag{A.209}$$

where x_e is a binary decision variable set to 1 if the edge *e* belongs to the minimum spanning tree and 0 otherwise. Note that by Theorem 6 the number of edges in the minimum spanning tree is *n-1* which is enforced by the cardinality constraints (A.207). Also, there should not be cycles in the MST, this is enforced by the constraints known as packing constraints (A.208). Suppose that there is a cycle, then the number of edges between the set of nodes connected by the cycle would be at least $|S|$. Finally, (A.209) prevents fractional values for the edges which would not correspond to a MST.

Theorem 13 A polyhedral description of the MST provides an integer solution

The polytope (A.207) - (A.209) has an integer solution and its vertices are exactly the incidence vectors of spanning trees of G. The interested reader is referred to Korte and Vygen (2000) for a proof.

Consider for example Figure A.76, the set S={1,3,5} has $|S| = 3$ and a cycle, therefore is on violation of an inequality of the form (A.208).

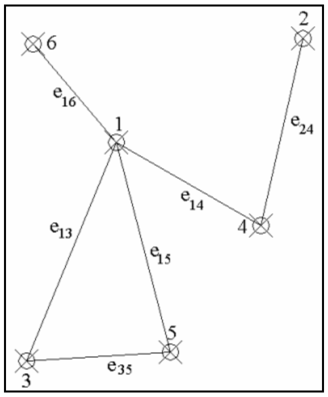


Figure A.76 The cycle between nodes 1, 3 and 5 has three edges

The drawback of this formulation is the exponential number of constraints required in the set (A.208).

A 3.6.5. Cutset Formulation

Another formulation of the MST presented by Magnanti and Wolsey (1995) is as follows:

Min: $\sum_{e \in E(G)} w_e x_e$ (A.210)

subject to

$\sum_{e \in E(G)} x_e = n - 1$ (A.211)

$\sum_{e \in d(S)} x_e \geq 1, \forall S \subset V(G), S \neq \emptyset$ (A.212)

$x_e \in \{0,1\}$ (A.213)

Similarly to the packing formulation, this formulation requires the number of edges to be equal to *n-1* and that all nodes are connected. Constraints (A.212) are included to enforce connectivity. They take a cut around a set *S* of nodes and force that there will be at least one edge from the set *S* to the set of nodes *V\S*. This, together with (A.211) forces the tree description. For example consider Figure A.77 where a cut includes only one node.

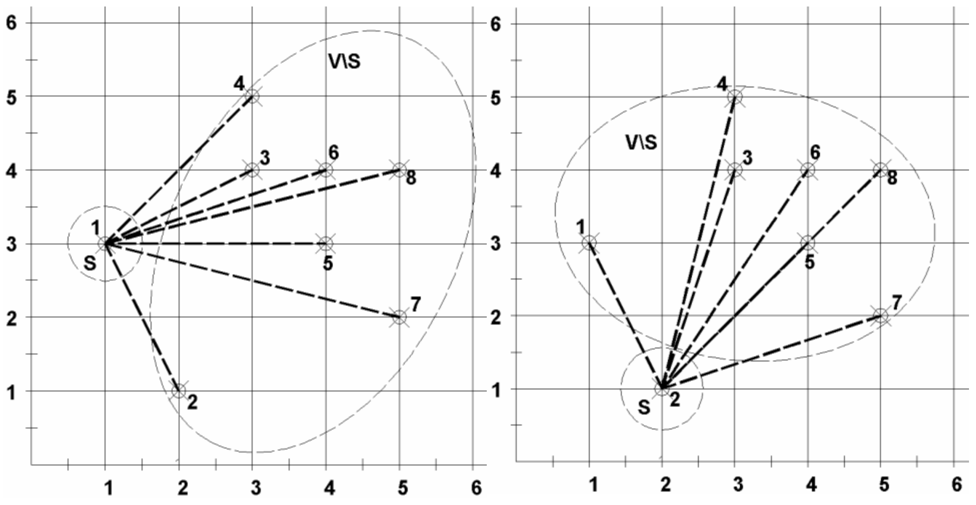


Figure A.77 Cut around node 1 (Left) and around node 2 (Right)

Now in Figure A.78 the cut includes two nodes.

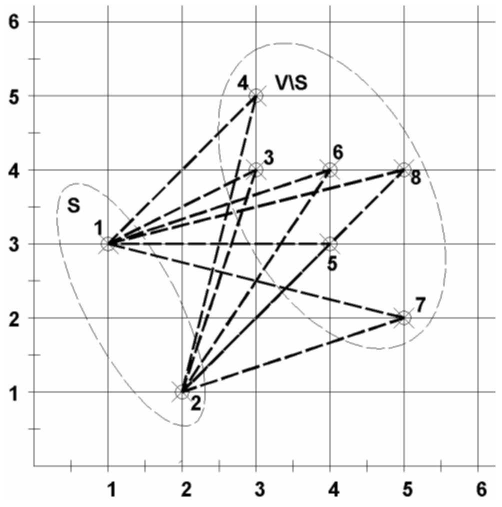


Figure A.78 Cut around nodes 1 and 2

When all possible sets are considered, together with the fact that there must be exactly *n-1* edges, then it is impossible to have a cycle. Because there is no disconnected set, there are *n* nodes and *n-1* edges, then no cycles are possible because otherwise either one of the nodes would be disconnected, or there would be more than *n-1* edges. This follows from Theorem 6.

A 3.6.6. Modified Cutset Formulation

The above formulation can be tightened so that the underlying polyhedron equals the convex hull of incidence vectors of the spanning trees (Magnanti and Wolsey, 1995).The modification involves replacing each undirected edge from the graph by two opposite directed edges converting the undirected graph *G* in a digraph *D*. This formulation requires a root node *r* from which the flow is sent into the network.

The resulting formulation is as follows:

Min: $\sum_{e\in E(G)} w_e x_e$ (A.214)

subject to

$$\sum_{e\in E(D)} y_e = n-1 \tag{A.215}$$

$$\sum_{e\in d^+(S)} y_e \le |S|-1 \forall S \subset V(D), S \neq \varnothing \tag{A.216}$$

$$\sum_{e\in d^-(v)} y_e = 1 \forall v \in V(D) \setminus \{r\} \tag{A.217}$$

$$y_e \ge 0 \forall e \in E(G) \tag{A.218}$$

$$x_e = y_{ij} + y_{ji} \forall e \in E(G) \tag{A.219}$$

The interested reader is referred to Magnanti and Wolsey (1995) for the correctness of this formulation.

A 3.6.7. Multi-Cut Formulation

The above formulation finds a MST, but it has the inconvenience that if the binary restriction (A.217) is relaxed , the relaxation does not define the convex set of incidence vectors of spanning trees (Magnanti and Wolsey, 1995). With this in mind, an alternative formulation is:

Min: $\sum_{e \in E} w_e x_e$ (A.220)

subject to

$$\sum_{e \in E(G)} x_e = n - 1 \tag{A.221}$$

$$\sum_{\{e \in d(C_0, C_1, \ldots, C_k)\}} x_e \geq k \forall C_0, C_1, \ldots, C_k \subset V, S \neq \emptyset \tag{A.222}$$

$$x_e \in \{0,1\} \tag{A.223}$$

This formulation is a more general case of the previous one (where *k=1*). Here a set of cuts $C_0, C_1, \ldots, C_k$ are connected to the rest of the tree by at least *k* edges. For a proof of correctness of this formulation the interested reader is referred to Magnanti and Wolsey (1995). An example with three cuts is presented in Figure A.79.

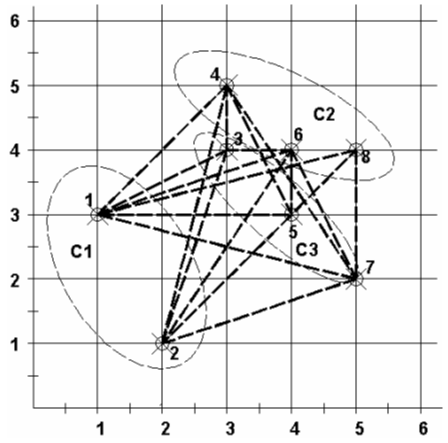


Figure A.79 Example of three cuts

A 3.6.8. Single Commodity Flow Formulation

Another approach to find the MST is by a modification of the general network design problem. This formulation considers that there is a flow of *n-1* products sent through the network from one of the nodes (node 1 for simplicity). Each one of the other *n-1* nodes has a demand of one item. The decision variable x_e is used to decide if an edge *e* will have flow ($x_e = 1$) or not ($x_e = 0$). The formulation assumes that the graph is undirected but the flows are directed, so there is a sign consideration if the flow comes into the node or if the flow is leaving the node. The formulation can be written as:

Min: $$\sum_{e \in E(G)} w_e x_e \tag{A.224}$$

subject to

$$\sum_{e \in d^+(1)} f_e - \sum_{e \in d^-(1)} f_e = n-1 \tag{A.225}$$

$$\sum_{e \in d^-(v)} f_e - \sum_{e \in d^+(v)} f_e = 1, \forall 1 \neq v \in V(G) \tag{A.226}$$

$$f_{ij} \leq (n-1)x_e, \forall e \in E(G) \tag{A.227}$$

$$f_{ji} \leq (n-1)x_e, \forall e \in E(G) \tag{A.228}$$

$$\sum_{e \in E(G)} x_e = n-1 \tag{A.229}$$

$$f_e \geq 0 \tag{A.230}$$

$$x_e \in \{0,1\}, \forall e \in E(G) \tag{A.231}$$

Here equations (A.225) and (A.226) are flow balance equations around the nodes, inequalities (A.227) and (A.228) set the flow to zero through a non selected edge ($x_e = 0$). An interesting aspect of this formulation is that there is not a cost associated to the objective function, any feasible solution is a MST.

Figure A.80 explains graphically an example of the formulation.

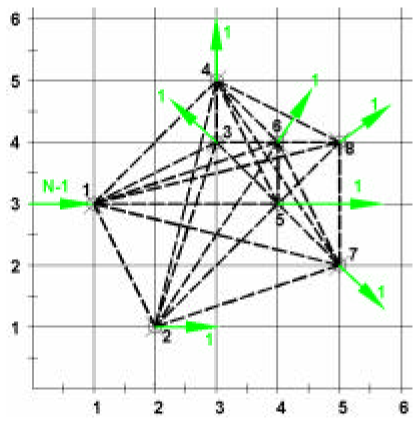


Figure A.80 Network with n-1 commodities leaving to each node from node 1

Node 1 is the root node, *(N-1)* items are sent into the node represented by the arrow pointing towards the root node. All other nodes have a demand of one item represented by the arrow leaving each node in the graph. The items travel from one node to another by the edges connecting the nodes.

A 3.6.9. Directed Multi-commodity Flow Model

Ahuja et al., (1995) and Magnanti and Wolsey (1995) both suggested yet another modification to the general formulation for network design to find the MST. Such formulation would be as follows:

$$\text{Min: } \sum_{e \in E(G)} d_e y_e \tag{A.232}$$

s.t.

$$\sum_{e \in d^-(r)} f_e^k - \sum_{e \in d^+(r)} f_e^k = -1, \forall k \neq r \tag{A.233}$$

$$\sum_{e \in d^-(v)} f_e^k - \sum_{e \in d^+(v)} f_e^k = 0, \forall v \neq r, v \neq k \text{ and all } k \tag{A.234}$$

$$\sum_{e \in d^-(k)} f_e^k - \sum_{e \in d^+(k)} f_e^k = 1, \forall k \neq r \in V(G) \tag{A.235}$$

$f_e^k \leq y_e, \forall e \in E(G)$ (A.236)

$\sum_{e \in E(G)} y_e = n-1$ (A.237)

$x_e = y_{ij} + y_{ji}$ (A.238)

$f_e^k \geq 0, \forall e \in E(G)$ (A.239)

$y_e \in \{0,1\} \quad \forall e \in E(G)$ (A.240)

Each commodity *k=1,2,…K* has an origin node that for purposes of simplicity can be node 1 in the network, and a destination node *D(k)*, and a flow requirement of one unit at each node; f_e^k is the fraction of commodity *k* that flows over the edge *e*. The upper bound on the capacity of the arcs is defined by y_{ij}. Constraints (A.236) allows flow to cross an edge only if the edge is selected in *T*. The interested reader is referred to Ahuja et al. (1995) and Magnanti and Wolsey (1995) for proof of the correctness of the formulation.

A 3.6.10. Extended Multi-commodity Flow Model

A slight change of the previous formulation where the y_{ij} variables are eliminated and replaced by a flow constraint. The resulting formulation is as follows:

Min: $\sum_{e \in E(G)} d_e y_e$ (A.241)

s.t.

$\sum_{e \in d^-(r)} f_e^k - \sum_{e \in d^+(r)} f_e^k = -1, \forall k \neq r$ (A.242)

$\sum_{e \in d^-(v)} f_e^k - \sum_{e \in d^+(v)} f_e^k = 0, \forall v \neq r, v \neq k$ and all k (A.243)

$$\sum_{e \in d^-(k)} f_e^k - \sum_{e \in d^+(k)} f_e^k = 1, \forall k \neq r \in V(G) \tag{A.244}$$

$$f_e^k \leq y_e, \forall e \in E(G) \tag{A.245}$$

$$\sum_{e \in E(G)} x_e = n - 1 \tag{A.246}$$

$$x_e \geq f_{ij}^k + f_{ji}^k \tag{A.247}$$

$$f_e^k \geq 0, \forall e \in E(G) \tag{A.248}$$

$$y_e \in \{0,1\} \quad \forall e \in E(G) \tag{A.249}$$

Bibliography

[1] Aerts, J., Van Herwijnen, M., Janssen, R. and Stewart, T. Evaluating Spatial Design Techniques for Solving Land-use Allocation Problems. Journal of Environmental Planning and Management, January 2005, vol. 48, no. 1, pp. 121-142(22)

[2] Agarwal, C., Green, G.L., Grove, J. M., Evans, T., and Schweik, C. Review and Assessment of Land Use Change Models; Dynamics of Space, Time, and Human Choice. 4th International Conference on Integrating GIS and Environmental Modeling (GIS/EM4): Problems, Prospects and Research Needs. Banff, Alberta, Canada, September 2 - 8, 2000

[3] Ahuja, R.K., Magnanti, T. L., Orlin, J. B. and Reddy, M. R. Applications of Network Optimization. Handbooks of Operations Research and Management Science. Volume 7: Network Models, Edited by M. O. Ball, T. L. Magnanti, C. L. Monma, and G. L. Nemhauser, Elsevier, North-Holland, Amsterdam. 1995.

[4] Alonso, W. Location and Land Use. Cambridge, Mass. Harvard University Press. 1964

[5] Archibald, T. W.; Buchanan, C. S Nested Benders decomposition and dynamic programming for reservoir optimization. Journal of the Operational Research Society, 50 (5).1999

[6] Aronofsky, J.S. Editor. Progress in Operations Research. Relationship Between Operations Research and the Computer. Volume III. John Wiley & Sons Inc. 1969.

[7] Badiru, A.B. and Pulat, P.S. Comprehensive Project Management, Integrating Optimization Models, Management Principles, and Computers. Prentice Hall, 1995.

[8] Balling, R., Taber, J.T., Brown, M. R. and Day, K. Multiobjective Urban Planning Using Genetic Algorithm. Journal of Urban Planning and Development. June 1999

[9] Bammi, D. and Bammi, D. Land use planning: an optimizing model. OMEGA-The International Journal of Management Science 3 (5), 583-593. 1975.

[10] Bammi, D. and Bammi, D. Development of Comprehensive Land use Plan by Means of a Multiple Objective Mathematical Programming Model. Interfaces 9 (2) 50-63. 1979

[11] Barnhart, C., Johnson, E., Nemhauser, G. L., Savelsbergh, M., Vance, P. Branch and Price Column Generation for Solving Huge Integer Programs. School of Industrial and Systems Engineering Georgia Institute of Technology Atlanta, GA. 1996

[12] Bazaraa, M.S., Jarvis, J.J. and Sherali, H.D. Linear Programming and Network Flows. Third Edition John Wiley & Sons, Inc. 2005

[13] Beinat, E. and Nijkamp, P. Multicriteria Analysis for Land-Use Management. Kluwer Academic Publishers, Dordrecht. 1998.

[14] Benabdallah, S. and Wright, J.R. Multiple Subregion Allocation Models. *Journal of Urban Planning and Development* 118 (1), 24-40. 1992.

[15] Birge, J.R. and Louveaux. Introduction to Stochastic Programming. Springer-Verlag New York Inc. 1997

[16] Birge, J.R. Decomposition and partitioning methods for multistage stochastic programs. Operations Research 33. 1985

[17] Black, T. The economics of sprawl. Urban Land 55(3): 6-52. 1996

[18] Boruvka., O. O Jistém Problému Minimálním. Acta Societ. Scient. Natur. Moravicae 3, p.p. 37-58. 1926a.

[19] Boruvka., O. Príspevek k resení otázky economické stavby. Elektrovodních sítí. Elektrotechnicky Obzor 15. 1926b

[20] Bradley, S.A., Hax, A. and Magnanti, T. Applied Mathematical Programming. Reading Mass. Adison-Wesley, 1977.

[21] Briassoulis, H. Analysis of Land Use Change: Theoretical and Modeling Approaches. The Web Book of Regional Science. Regional Research Institute West Virginia University. http://www.rri.wvu.edu/WebBook/Briassoulis/contents.htm

[22] Bryant, C. and Johnson, J. Agriculture in the City's Countryside, Belhaven, London, 1992.

[23] Burchell, R. W. and Listokin, D. Fiscal Studies for the Governor's Commission on Growth in the Chesapeake Bay region. Baltimore, Maryland Office of Planning. 1991

[24] Burchell, R. W., Lowenstein, G. Dolphin, W. R. Galley, C. Costs of Sprawl—2000. Research Sponsored by the Federal Transit Administration in Cooperation with the Transit Development Corporation. National Academy Press. Washington , D.C. 2002

[25] Burchell, R. W., Lowenstein, G., Dolphin, W. R., Galley, C. C., Downs A., Seskin, S., and Moore T. The Benefits of Sprawl. Chapter 12A in The Costs of Sprawl – Revisited, Washington, D.C.: Transportation Research Board and National Research Council, 351-391. 2000.

[26] Burchell, R. W., Shad, N.A., Listokin D., Phillips H., Downs, A., Siskin, S., Davis, J. S., Moore, T., Helton, T., Gall, M., and ECONort-west. Cost of sprawl revisited Washington D.C. National Academy Press 1998

[27] Chambers, L. Practical Handbook of Genetic Algorithms. Applications Volume I. CRC Press, Inc. 1995.

[28] Charnes, A. and Cooper, W. Management Models and Industrial Applications of Linear Programming. Vol. I. Wiley, New York. 1961.

[29] Christaller, Walter. Die Zentralen Orte in Süddeutschland . Jena: Gustav Fischer, 1933. (Translated (in part), by Charlisle W. Baskin, as Central Places in Southern Germany. Prentice Hall. 1966.

[30] Cohon, J. L. Multiobjective Programming and Planning. Academic Press 1978.

[31] Colorado Smart Growth State Web Site http://www.state.co.us/smartgrowth/

[32] Conejo, A.J., Castillo, E., Mínguez, R., and García-Bertrand, R. Decomposition Techniques in Mathematical Programming. Engineering and Science Applications. Work in progress. 2003.

[33] Conte, C. R. The Boys of Sprawl, Governing, May, 28-33. 2000

[34] Costanza, R. and Wainger, L. Modeling complex ecological economic systems. Bioscience; Vol. 43 Issue 8. 1993

[35] Costanza, R. Ecological Economics and Sustainability. Ecological Applications, Vol. 6, No.4. 1996

[36] Cotta, C., Aldana, J. F., Nebro, A. J. and Troya, J.M.. Hybridizing Genetic Algorithms with Branch and Bound Techniques for the Resolution of the TSP. In Pearson, D.W., Steele, N.C. and Alberec, R.F. (eds.) Artificial Neural Nets and Genetic Algorithms. Proceedings of the International Conference on Artificial Neural Nets and Genetic Algorithms pp. 277-280. Ales, France. 1995

[37] Dantzig, G. B., and Wolfe, P. Decomposition Principles for Linear Programming. Operations Research, 8:101-111, 1960.

[38] Dantzig, George B. Linear Programming and Extensions. Princeton University Press. Princeton, NJ. 1963

[39] Dantzig, George B. Reminiscences About the Origins of Linear Programming. Operation Research Letters Volume 1, Number 2. April 1982

[40] Deb, K. Multiobjective Optimization using Evolutionary Algorithms. John Wiley & Sons, Ltd. 2001.

[41] Dempster, M.A.H. and Thompson, R.T., Parallelization and Aggregation of Nested Benders Decomposition. http://ssrn.com/abstract=37765

[42] Desaulniers, G., Desrosiers, J., Solomon, M. (Eds.). Column Generation. Springer. 2005

[43] Dijkstra, E. W. A Note on Two Problems in Connection with Graphs. Numer. Math., 1, 269 – 271. 1959

[44] Downs, A. How America's Cities are Growing: The Big Picture. Brookings Review 16(4):8-12. 1999

[45] Downs, A. Some Realities about Sprawl and Urban Decline. Housing Policy Debate 10(4):955-74. 1998 http://www.anthonydowns.com/

[46] Edmonds, J. Submodular Functions, Matroids and Certain Polyhedra. In Combinatorial Structures and Their Applications; Proceedings of the Calgary International Conference on Combinatorial Structures and Their Applications 1969 (R. Guy, H Hanani, N. Sauer, J. Schonheim, eds.) Gordon and Breach, New York 1970

[47] Eiselt, H.A. and Sandblom, C. L. with contributions by Spielberg, K., Richards, E., Smith, B.T., Laporte, G. and Boffey B. T. Integer Programming and Network Models. Springer-Verlag Berlin. 2000

[48] EPA Mid-Atlantic Integrated Assessment: Urban Sprawl. U.S. Environmental Protection Agency. http://www.epa.gov/emfjulte/tpmcmaia/html/sprawl.html 2005

[49] Ewing, R. Characteristics, Causes, and Effects of Sprawl: A Literature Review. Environmental and Urban Issues 21, 2: 1-15. 1994

[50] Ewing, Reid. Is Los Angeles-Style Sprawl Desirable? Journal of the American Planning Association 63, 1: 107-126. 1997

[51] Farley, J. and Costanza, R. Envisioning shared goals for humanity: a detailed, shared vision of a sustainable and desirable USA in 2100. Ecological Economics 43. 2002

[52] Fisher, M. L. The Lagrangian Relaxation Method for Solving Integer Programming Problems. Management Science, 27:1-18, 1981.

[53] Fonseca, C. M. and Fleming, P. J. Multiobjective Optimization and Multiple Constraint Handling with Evolutionary Algorithms – Part I: A Unified Formulation. IEEE Transactions on systems, man and cybernetics – Part A: Systems and Humans Vol. 28, No 1 January 1998.

[54] French, A. P., Robinson, A.C. and Wilson, J. M. Using a Hybrid Genetic-Algorithm/Branch and Bound Approach to Solve Feasibility and Optimization Integer Programming Problems. Journal of Heuristics 7:551-546 (2001) Kluwer Academic Publishers. The Netherlands.

[55] Gabriel, S.A., Faria, J.A., and Moglen, G.E. A Multiobjective Optimization Approach to Smart Growth in Land Development. Socio-Economic Planning Sciences (in press) 2005.

[56] Galster, G., Hanson, R., Ratcliffe, M., Wolman, H., Coleman, S. and Freihage, J. Wrestling Sprawl to the Ground: Defining and Measuring an Elusive Concept. Housing Policy Debate 12, 4, pp. 681-718. 2001

[57] Garfinkel, R.S. and Nemhauser, G.L. Optimal Political Districting by Implicit Enumeration Techniques. Management Science No. 8 April 1970.

[58] Gass, S. I. Linear Programming Methods and Applications. Fifth Edition. McGraw Hill, Inc. 1985.

[59] Gassman, H.I. MSLiP: A Computer Code for the Multistage Stochastic Linear Programming Problem. Math. Prog 47, 407±423. 1990

[60] Geoffrion, A. Lagrangian Relaxation for Integer Programming. Mathematical Programming Study, 2:82 – 114, 1974.

[61] Gilbert, K. C., Holmes, D.D. and Rosenthal, R.E. A Multiobjective Discrete Optimization Model for Land Allocation. Management Science Vol. 31 No 12 December 1985. http://www.epa.gov/smartgrowth/

[62] Gillham, O. The Limitless City A Primer on the Urban Sprawl Debate. Island Press. 2002

[63] Glaeser, E. L., and Kahn M. E. Sprawl and Urban Growth. Harvard Institute of Economic Research Discussion Paper, Number 2004. 2003

[64] Glassey, C.R. Nested Decomposition and Multi-Stage Linear Programs. Management Science Vol. 20. No 3, November, 1973.

[65] Gordon, P. and Richardson, H. W. Critiquing Sprawl's Critics. Policy Analysis, January 24, 1 – 18. 2000

[66] Gower, J.C. and Ross, G.J.S. Minimum Spanning Trees and Single Linkage Cluster Analysis. Appl. Stat. 18, 54-64. 1969

[67] Gross, J., and Yellen, J. Graph Theory and its Applications. CRC Press, 1999.

[68] H. Yaman, O.E. Karassan, and M.C. Pinar. The robust spanning tree problem with interval data. Operations Research Letters, 29:31 - 40, 2001.

[69] Han, J., McMahon, G., and Sugden, S. A Node-oriented Branch and Bound Algorithm for the Capacitated Minimum Spanning Tree Problem. Proceedings of the 28th Annual IEEE International Conference on Local Computer Networks (LCN'03). 2003

[70] Heenan, D. A. An America Odyssey: The Corporate Migration to Rural America. PacNet Newsletter #9, March 5, 1999. http://www.csis.org/pacfor/pac0999.html

[71] Heim, C. E. Leapfrogging, Urban Sprawl, and Growth Management: Phoenix, 1950-2000. The American Journal of Economics and Sociology. 2001

[72] Ho, J.K. Nested Decomposition of a Dynamic Energy Model. Management Science Vol. 23. No 9. May, 1977

[73] Hoover and Giarratani. An Introduction to Regional Economic. Alfred A. Knopf, Inc.1984

[74] Huisman, D., Jans, R., Peeters, M., Wagelmans, A.P.M. Combining Column Generation and Lagrangian Relaxation. Erasmus Research Institute of Management (ERIM), RSM Erasmus University. ERS-2003-092-LIS. 2003

[75] Jungnickel, D. Graphs, Networks and Algorithms. Springer-Verlag Berlin Heidelberg. 1999.

[76] Knaap, G. J., Song, Y. and Nedovic-Budic, Z. Measuring Patterns of Urban Development: New Intelligence for the War on Sprawl. Working paper available at: http://www.smartgrowth.umd.edu/research/researchpapers-landuseandenvironment.htm. 2003

[77] Korte, B. and Vygen, J. Combinatorial Optimization Theory and Algorithms. Springer-Verlag Berlin Heidelberg. 2000.

[78] Kruskal, J. B. On the Shortest Spanning Subtree of a Graph and The Traveling Salesman Problem. Proc AMS 7, 48-50. 1956.

[79] Lai, T.H. and Sheng, M.J. Constructing Euclidean Minimum Spanning Trees and All Nearest Neighbors on Reconfigurable Meshes. IEEECS No D95205. 1996

[80] Lewis, R.K. Urban Planning: It's Time for a Foreign Concept to Hit Home in the U.S. The Washington Post On Line Saturday July 28, 2001. Page H03

[81] Lockwood C. Sprawl. Hemispheres, September 1999

[82] Magnanti, T.L., and Wolsey, L.A. Optimal Trees. Volume 7: Network Models, Edited by M. O. Ball, T. L. Magnanti, C. L. Monma, and G. L. Nemhauser, Elsevier, North-Holland, Amsterdam. 1995.

[83] Malpezzi, S. Estimates of the Measurement and Determinants of Urban Sprawl in U.S. Metropolitan Areas. University of Wisconsin Madison. Center for Urban Land Economics Research. 1999.

[84] McCusker, S.A., and B.F. Hobbs. A Nested Benders Decomposition Approach to Locating Distributed Generation in a Multiarea Power System. Networks & Spatial Economics. vol.3. pp. 197 - 223. 2003

[85] McCusker, S.A., and B.F. Hobbs. A Nested Benders Decomposition Approach to Locating Distributed Generation in a Multiarea Power System. Networks & Spatial Economics. vol.3. 2 pp. 197 - 223. 2003

[86] Milkova, E. Graph Theory and Information Technology. University of Education, Faculty of Management and Information Technology Department of Theoretical Informatics Hradec Kralové.

[87] Moglen, G.E., Gabriel, S.A. and Faria, J.A.. A Framework for Quantitative Smart Growth in Land Development. Journal of American Water Resources Association (JAWRA). 2002.

[88] Montemanni, R. and Gambardella, L.M. A Branch and Bound Algorithm for the Robust Spanning Tree Problem with Interval Data. TECHNICAL REPORT IDSIA-10-02. 2002

[89] Moskowitz, H. S. and Lindbloom, C. G. The New Illustrated Book of Development Definitions. New Brunswick, N.J. Rutgers University Center for Urban Policy Research.

[90] Myrdal, G. Economic Theory and Under-Developed Regions. London: Gerald Duckworth. 1957

[91] Nagar, A., Heragu, S.S. and Haddock, J. A Combined Branch and Bound and Genetic Algorithm Based for a Flowshop Scheduling Algorithm. Annals of Operations Research 63, 397 – 414.1996.

[92] Nash, S.G and Sofer, A. Linear and Nonlinear Programming. The McGraw-Hill Companies, Inc. 1996.

[93] Nesetril, J., Milkova, E., and Nesetrilova H. Otakar Boruvka On Minimum Spanning Tree Problem (translation of the 1926 papers, comments, and history). 2000

[94] Nijkamp, P. Multidimensional Spatial Data and Decision Analysis. John Wiley & Sons Ltd. 1979.

[95] Ontario Smart Growth State Web Site http://www.smartgrowth.gov.on.ca/

[96] Orfield, M. A Regional Agenda for Community and Stability. Metropolitics. A Regional Agenda for Community and Stability (Revised Edition). Brookings Institution Press 1997

[97] Papadimitriou, C.H. and Steiglitz, K. Combinatorial Optimization, Algorithms and Complexity. Dover publications Inc. 1988.

[98] Popenoe, D. Urban Sprawl: Some Neglected Sociology. Sociology & Social Research 31(2):181–88. 1979

[99] Pred, Allan R. The Spatial Dynamics of U. S. Urban-Industrial Growth, 1800-1914. Cambridge, M.I.T. Press. 1966

[100] Prim, C. R. Shortest Connection Networks and Some Generalizations. Bell Systems Tech. J. 36, p.p. 1389 – 1401. 1957

[101] Pullar, D. Using an Allocation Model in Multiple Criteria Evaluation. Journal of Geographic Information and Decision Analysis, Vol 3. No 2. pp. 9-17, 1999.

[102] Reeves, C. Hybrid Genetic Algorithms for Bin-Packing and Related Problems. Annals of Operations Research 63, 371-296. 1996

[103] ReVelle, C. and McGarity, A.E. Design and Operation of Civil and Environmental Engineering Systems. John Wiley & Sons, Inc. 1997.

[104] Romanos, M. Residential Spatial Structure. Lexington Books regional science monograph series. Lexington, Mass.: Lexington Books. 1976

[105] Siegel, F. The Sunny Side of Sprawl. The New Democrat, March/April, 21-22. 1999

[106] Sierra Club. Sprawl: The dark side of the American Dream. http://www.sierraclub.org/sprawl/report98/

[107] Sonka, M., Hlavac, V., and Boyle R. Image Processing: Analysis and Machine Vision (2nd ed). Thomson Learning Vocational. 1998

[108] Thünen, J. H. von, 1783-1850 Isolated state; an English edition of Der Isolierte Staat. Translated by Carla M. Wartenberg. Edited with by Peter Hall, Oxford, New York. Pergamon Press. 1966.

[109] Thünen, J. H. von. Der isolirte Staat in Beziehung auf Landwirthschaft und Nationalökonomie. G.W. Leopold's Universitätsbuchhandlung. 1842

[110] Torrens, P. and Alberti, M. Measuring sprawl. London, England: Center for Advanced Spatial Analysis, University College London, Unpublished paper # 27. 2000

[111] Toussaint, G.T. The Relative Neighborhood Graph of a Finite Planar Set. Pattern Recognition, Vol. 12 pp. 261-268. 1980

[112] U.S. Environmental Protection Agency. Smart Growth Web Site. http://www.epa.gov/smartgrowth/

[113] United States General Accounting Office. GAO Report to Congressional Requesters. Community Development Extent of Federal Influence on "Urban Sprawl" Is Unclear.1999

[114] Vaidya, P. M. A Fast Approximation Algorithm for Minimum Spanning Trees in k-Dimensional Space. IEEE 1984

[115] Vanderbeck, F. On Dantzig-Wolfe Decomposition in Integer Programming and Ways to Perform Branching in a Branch and Price Algorithm. Mathématiques Appliquées Bordeaux, Université Bordeaux. Cedex, France. 1998

[116] Vatalis, K. and Manoliadis, O. A Two Level Multicriteria DSS for Landfill Site Selection Using GIS: Case Study in Western Macedonia, Greece. Journal of Geographic Information and Decision Analysis Vol. 6, No 1, pp. 49 – 56. 2002.

[117] Watkins Jr., D.W., McKinneya, D. C., Lasdonb, L. S., Nielsenc, S.S., and Martind Q. W. A scenario-based stochastic programming model for water supplies from the highland lakes. Intl. Trans. in Op. Res. 7. 2000

[118] Wheeler, J.O. and P.O. Muller. Economic Geography. New York: John Wiley and Sons, pp. 133-137. 1981

[119] Williams, Justin C. A Linear-Size Zero-One Programming Model for the Minimum Spanning Tree Problem in Planar Graphs. Networks, Vol. 39. 2001

[120] Winston, W.L. Operations Research Applications and Algorithms. Fourth Edition. Brooks/Col. 2004.

[121] Wolman, H., Galster, G., Hanson, R., Ratcliffe, M., Furdell, K., Sarzynski, A. The Fundamental Challenge in Measuring Sprawl: Which Land Should Be Considered? The Professional Geographer. Vol. 57, no. 1. 2005

[122] Wolsey, L.A. Integer Programming. John Wiley & Sons, Inc. 1998.

[123] Wright, J., ReVelle, C. and Cohon, J. A Multiobjective Integer Programming Model for the Land Acquisition Problem. Regional Science and Urban Economics 13 (1983) 31-53. North Holland 1983.

[124] Wu, J. and Plantinga A.J. The Influence of Public Open Space on Urban Spatial Structure. Journal of Environmental Economics & Management, Vol. 46. 2003

[125] Yu, P.L. and Zeleny, M. Linear Multi-Parametric Programming by Multicriteria Simplex Method. management Science, Vol. 23 No 2. 1976

[126] Zahn, C.T. Graph Theoretical Methods for Detecting and Describing Gestalt Clusters. IEEE Trans. Comput. C20, 68-86. 1971

[127] Zeleny, M. Linear Multiobjective Programming. Lecture notes in Economics and Mathematical Systems. No 95 Berling Springer-Verlag. 1994

[128] Zhou, G. and Gen M. Genetic Algorithm Approach on Multi-Criteria Minimum Spanning Tree Problem. European Journal of Operational Research 114 p.p. 141-152. 1999