12-15-2016

An Efficient Solution Methodology for Mixed-Integer Programming Problems Arising in Power Systems

Mikhail Bragin
University of Connecticut - Storrs, mikhail.bragin@uconn.edu

An Efficient Solution Methodology for Mixed-Integer Programming Problems Arising in Power Systems

Mikhail Bragin, PhD

University of Connecticut, 2016

For many important mixed-integer programming (MIP) problems, the goal is to obtain near-optimal solutions with quantifiable quality in a computationally efficient manner (within, e.g., 5, 10 or 20 minutes). A traditional method to solve such problems has been Lagrangian relaxation, but the method suffers from zigzagging of multipliers and slow convergence. When solving mixed-integer linear programming (MILP) problems, the recently adopted branch-and-cut may also suffer from slow convergence because when the convex hull of the problems has complicated facial structures, facet-defining cuts are typically difficult to obtain, and the method relies mostly on time-consuming branching operations. In this thesis, the novel Surrogate Lagrangian Relaxation method is developed and its convergence is proved to the optimal multipliers, without the knowledge of the optimal dual value and without fully optimizing the relaxed problem. Moreover, for practical implementations a stepsizing formula, that guarantees convergence without requiring the optimal dual value, has been constructively developed. The key idea is to select stepsizes in a way that distances between Lagrange multipliers at consecutive iterations decrease, and as a result, Lagrange multipliers converge to a unique limit. At the same time, stepsizes are kept sufficiently large so that the algorithm does not terminate prematurely. At convergence, the lower-bound property of the surrogate dual is guaranteed. To solve MIP problems, based on Surrogate Lagrangian Relaxation, stable and accelerated convergence is ensured by introducing quadratic penalty terms motivated by Augmented Lagrangian relaxation. Convergence of Lagrange

multipliers to their optimal values is significantly improved. When solving MILP problems, through the novel V-shape linearization, the relaxed problem is linearized to ensure that solutions consistent with those of the nonlinear relaxed problem can be obtained by branch-and-cut, thereby ensuring that convergence characteristics are very similar to those of Augmented Lagrangian relaxation. When solving block-structured MILP problems, the V-shaped relaxed problem allows the decomposition into much smaller MILP subproblems with exponentially reduced complexity as compared to the original problem thereby drastically reducing computational requirements. Moreover, analytical subproblem solutions can be obtained thereby making the reduction of computational requirements even more drastic. When solving large-scale unit commitment problems with combined cycle units as well as other problems for which facet defining cuts are difficult to obtain, it is demonstrated the new method is robust and much more efficient as compared to frequently-used branch-and-cut and surrogate Lagrangian relaxation, and represents a major step forward to solve difficult MIP and MILP problems.

An Efficient Solution Methodology for Mixed-Integer Programming Problems Arising in Power Systems

Mikhail Bragin

B.S., Voronezh State University, Voronezh, Russia, 2004

M.S., Voronezh State University, Voronezh, Russia, 2004

M.S. University of Nebraska-Lincoln, Lincoln, NE, 2006

M.S., University of Connecticut, Storrs, CT, 2014

A Dissertation

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Doctor of Philosophy

at the

University of Connecticut

2016

APPROVAL PAGE

Doctor of Philosophy Dissertation

An Efficient Solution Methodology for Mixed-Integer Programming Problems Arising in Power Systems

Presented by

Mikhail Bragin, B.S., M.S.

Major Advisor _____

Peter B. Luh

Associate Advisor _____

Peng Zhang

Associate Advisor _____

Lakshman Thakur

ACKNOWLEDGMENTS

First, I would like to express my deepest gratitude to my adviser, Prof. Peter. B. Luh for his continuous support of my doctoral study at the University of Connecticut. His guidance, encouragement, and immense knowledge helped me in all the time of research and paved the way for my successful dissertation.

I would also like to convey my sincere thanks to Professors Lakshman Thakur and Peng Zhang for joining my advisory committee and providing useful suggestions. I would also like to thank Dr. Joseph H. Yan of Southern California Edison for his support and insightful guidance.

I would like to thank all my labmates for the pleasant study and research experience. The discussions and exchanges of knowledge enriched my experience.

And last but not least, I would like to thank my lovely wife Yulia of 8 years, who has been always supporting all my research endeavors and understanding the gravity of this important step of my life.

Table of Contents

List of Figures

List of Tables

Publications Related to this Thesis

Journal Articles

Bragin, M. A., Luh, P. B., Yan, J. H., Yu, N., and Stern, G. A.: Convergence of the surrogate Lagrangian relaxation method, J. Optim. Theory Appl. 164(2.1), 173-201 (2015)

Bragin, M. A., Luh, P. B., Yan, J. H., and Stern, G. A.: An Efficient Approach for Solving Mixed-Integer Programming Problems under the Monotonic Condition. Journal of Control and Decision, January 2016, DOI: 10.1080/23307706.2015.1129916.

Yan, B., Fan, H., Luh, P. B., Moslehi, K., Feng, X., Yu, C.-N., Bragin, M. A. and Yu, Y., "Grid Integration of Wind Generation Considering Remote Wind Farms: Hybrid Markovian and Interval Unit Commitment," to appear in IEEE/CAA Journal of Automatica Sinica.

Bragin, M. A., Luh, P. B., Yan, J. H., and Stern, G. A.: A Solution Methodology for MIP and MILP Problems. Submitted to the Journal of Optimization Theory and Applications

Conference Proceedings

M. A. Bragin, X. Han, P. B. Luh, and J. H. Yan, "Payment Cost Minimization Using Lagrangian Relaxation and Modified Surrogate Optimization Approach," Proceedings of the IEEE PES 2011 GM, Detroit, Michigan, July 2011.

M. A. Bragin, P. B. Luh, J. H. Yan, N. Yu, X. Han and G. A. Stern, "An Efficient Surrogate Subgradient Method within Lagrangian Relaxation for the Payment Cost Minimization Problem," Proceedings of the IEEE PES 2012 GM, San Diego, California, July 2012

Han, X., Luh, P. B., Bragin, M. A., Yan, J. H., Yu, N., & Stern, G. A., "Solving payment cost co-

optimization problems," Proceedings of the IEEE PES 2012 GM

M. A. Bragin, P. B. Luh, and J. H. Yan, "An Efficient Surrogate Optimization Method for Solving Linear Mixed–Integer Problems with Cross–Coupling Constraints," Proceedings of the 10th WCICA, July 6-8, 2012, Beijing, China

M. A. Bragin, P. B. Luh, J. H. Yan, N. Yu and G. A. Stern, "Efficient Surrogate Optimization for Payment Cost Co-Optimization with Transmission Capacity Constraints," Proceedings of the IEEE PES 2013 GM, Vancouver, Canada, July 2013 ~ in Best Paper Session

M. A. Bragin, P. B. Luh, J. H. Yan and G. A. Stern., "Surrogate Lagrangian relaxation and branch-and-cut for unit commitment with combined cycle units," Proceedings of the IEEE PES 2014 GM, National Harbor, Maryland, July 2014 ~ in Best Paper Session

Yan, B., Luh, P. B., Bragin, M. A., Song, C., Dong, C., & Gan, Z. "Energy-efficient building clusters," *Proceedings of the IEEE 2014 IEEE CASE*

M. A. Bragin, P. B. Luh, J. H. Yan and G. A. Stern.: Novel exploitation of convex hull invariance for solving unit commitment by using surrogate Lagrangian relaxation and branch-and-cut. In: Proceedings of the IEEE PES 2015 GM, Denver, Colorado

M. A. Bragin, P. B. Luh, J. H. Yan and G. A. Stern, "An Efficient Approach for Unit Commitment and Economic Dispatch with Combined Cycle Units and AC Power Flow," IEEE Power and Energy Society 2016 General Meeting, Boston, MA, 2016.

M. A. Bragin, P. B. Luh, "Distributed and Asynchronous Unit Commitment and Economic Dispatch," submitted to 2017 IEEE Power and Energy Society General Meeting

B. Yan, P. B. Luh, E. Litvinov, T. Zheng, D. Schiro, M. A. Bragin, F. Zhao, J. Zhao, and I. Lelic "Unit Commitment with Generation-dependent Ramp Rates and Reserve Requirements," submitted to

2017 IEEE Power and Energy Society General Meeting

X. Sun, P.B. Luh, M. A. Bragin, Y. Chen, J. Wan, and F. Wang, "A Decomposition and Coordination Approach for Large-Scale Security Constrained Unit Commitment Problems with Combined Cycle Units," submitted to 2017 IEEE Power and Energy Society General Meeting

Chapter 1

Introduction

1.1. Motivations

For many important mixed-integer programming (MIP) problems, the goal is to obtain near-optimal solutions with quantifiable quality in a computationally efficient manner (within, e.g., 5, 10 or 20 minutes). An important subclass of MIP problems include "block-structured" [1] mixed-integer linear programming (MILP) problems, which can be viewed as multiple subsystems interconnected through system-wide coupling constraints. Examples include Generalized Assignment problems [2, 3], Location-Routing problems [4-7], and Unit Commitment and Economic Dispatch problems [8-14]. Linearity can be exploited by branch-and-cut [15-17], and separability can be exploited by Lagrangian relaxation [3, 18-22]. For example, after relaxing assignment constraints, Generalized Assignment Problems can be decomposed into machine subproblems [2, 3], and Unit Commitment and Economic Dispatch problems can be decomposed into unit subproblems after relaxing system demand constraints [8-14]. In the following, after presenting the unit commitment problems with combined cycle units, existing method and their difficulties will be presented.

Unit Commitment Problem with Combined Cycle Units. Combined cycle units are efficient because high temperature gas from combustion turbines is not released into the atmosphere but is used to generate steam in a heat recovery steam generator. Steam is then used to drive a steam turbine generator. However, transitions among generator states may be constrained. For example, steam turbines cannot generate electricity if there is not enough heat from combustion turbines. Such transitions among

commitment states of combustion and steam generators complicate unit commitment and economic dispatch problems and pose major computational challenges for existing methods. The associated unit commitment and economic dispatch problem is modeled as a mixed-integer linear programming problem, which is computationally intensive. CPU time required to obtain solutions with good quality by branch-and-cut and Lagrangian relaxation is frequently large [23-24]. Such difficulties arise because of fundamental difficulties as will be explained ahead.

Branch-and-Cut To solve MILP problems, branch-and-cut [15-17] attempts to obtain the "convex hull" by using "facet-defining" cuts. Then the optimal solution is obtained at one of the vertices of the convex hull. One way to obtain facets of the convex hull is by using "lift-and-project" cuts [25-32]. Another way to obtain facets of the convex hull is by using Gomory cuts [33-38]. Cuts are created after linearly combining constraints and then rotating and shifting resulting hyperplanes. When solving block-structured MILP problems, the method does not exploit "block structures," and constraints within one block are handled globally thereby affecting the solution process of the entire problem and leading to slow convergence. When the convex hull has complicated facial structures, facet-defining cuts are difficult to obtain, and the method relies mostly on time-consuming branching operations [39-40]. These difficulties have been vividly demonstrated for unit commitment problems with combined cycle units that arise in power systems [41-42].

Lagrangian Relaxation and Surrogate Lagrangian Relaxation The "block structure" can be exploited within Lagrangian relaxation, and complexity of resulting subproblems and associated convex hulls is drastically reduced. However, the method may require significant computational requirements and suffer from slow convergence because frequently used subgradient methods [8-14, 43-50] requires solving all subproblems to update Lagrange multipliers. These computational difficulties have been overcome by the surrogate subgradient method without solving all subproblems [51]. Resulting surrogate subgradient directions are smoother and form smaller acute angles toward the optimal multipliers as compared to subgradient directions, thereby alleviating zigzagging and reducing the number of iterations required for

convergence. However, convergence proof and practical implementations of the method critically require the knowledge of the optimal dual value, thereby leading to major convergence difficulties. These difficulties have been overcome by our recently-developed Surrogate Lagrangian Relaxation method [52]. The difficulty of the method is that levels of constraint violations may not reduce fast enough.

Method of Multipliers ("Augmented Lagrangian Relaxation") To accelerate the reduction of violation levels of relaxed constraints, the method of multipliers (referred to as "Augmented Lagrangian relaxation" (ALR)) [53-57] was proposed in the late 1960s by Hestenes [53] and Powell [54] whereby constraint violations are penalized by introducing quadratic penalty terms. Under assumptions of convexity and smoothness of the objective function and constraints, convergence has been established [55]. The ALR method has been used for MILP [57] problems and it has been shown duality gap approaches zero. However, the associated computational effort may be large because of the combinatorial nature of the problems. While Augmented Lagrangian relaxation has been one of the fastest methods, because of the introduction of quadratic penalty terms, the relaxed problem is nonlinear and non-separable.

1.2. Major Contributions

To efficiently solve the unit commitment problem with combined cycle units as well as other complicated mixed-integer programming problems to obtain near-optimal solutions with quantifiable quality and within strict time limits, a novel solution methodology is developed. First, based on Lagrangian relaxation whereby through decomposition complexity of subproblems reduces drastically, to guarantee convergence to the optimal multipliers, the novel surrogate Lagrangian relaxation method is developed whereby convergence is proved without the knowledge of the optimal dual value and without fully optimizing the relaxed problem. Moreover, for practical implementations, a stepsizing formula that guarantees convergence without requiring the optimal dual value, has been constructively developed. The key idea is to select stepsizes in a way that distances between Lagrange multipliers at consecutive iterations decrease, and as a result, Lagrange multipliers converge to a unique limit. At the same time, stepsizes are kept sufficiently large so that the algorithm does not terminate prematurely. At

convergence, the lower-bound property of the surrogate dual is guaranteed. To significantly improve convergence, the surrogate Augmented Lagrangian relaxation methodology is developed based on Surrogate Lagrangian Relaxation with major improvements on convergence through the introduction of quadratic penalty terms as motivated by the fast convergence of Augmented Lagrangian relaxation. When specializing to solving MILP problems by using the surrogate Augmented Lagrangian relaxation method, existing linearization methods and their difficulties are presented. To preserve fast convergence characteristics of surrogate Augmented Lagrangian relaxation while exploiting linear solvers, the augmented relaxed problem is linearized through the novel V-shape linearization scheme that preserves positions of minima. When further specializing to solving block-structured MILP problems, the linearized relaxed problem is decomposed into smaller subproblems with exponentially reduced complexity as compared to the original problem thereby drastically reducing computational requirements. Moreover, analytical subproblem solutions are obtained thereby making the reduction of computational requirements even more drastic. It is demonstrated numerically that convergence of the new method is more stable as compared to standard subgradient methods. When solving large-scale MILP problems whereby convex hulls are difficult to obtain such as generalized assignment problems and unit commitment problems with combined cycle units, it is demonstrated the new method is robust and much more efficient as compared to frequently-used branch-and-cut and our recent surrogate Lagrangian relaxation, and represents a major step forward to solve difficult MILP problems.

To address the increasing uncertainties associated penetration levels of intermittent renewable such as wind and solar, an alternative distributed and asynchronous unit commitment end economic dispatch will be considered presented. Within the framework, unit subproblems are solved locally in an asynchronous manner, and price-based coordination is performed by an ISO. Since prices are updated based on unit solutions obtained using prices of different vintages, convergence may not be guaranteed. To overcome this difficulty, surrogate Lagrangian relaxation is distributed, and conditions on price convergence are innovatively established through a Lyapunov energy function of the distance from current prices to the optimum. To consider the effects of asynchronous unit solutions, an upper bound of the energy function

is shown to approach zero. Numerical testing on the unit commitment end economic dispatch problem with 1000 units without transmission capacity constraints indicates that the method is robust and fast. To further explore the scalability of asynchronous coordination, a simplified unit commitment end economic dispatch problem with 10,000 units each with generation capacity constraints only indicate that the method converges, is robust and more efficient as compared to alternate direction method of multipliers.

1.3. Organization of this Thesis

The rest of this thesis is organized as follows. Chapter 2 introduces the novel Surrogate Lagrangian relaxation method and its convergence proof. Chapter 3 introduces the novel surrogate Augmented Lagrangian relaxation and its combination with branch-and-cut. Chapter 4 discusses a distributed and asynchronous unit commitment problem and extensions of surrogate Lagrangian relaxation to handle the asynchronous nature of the problem.

References

1. Giovanni, F. and Gentile, G.: Zero-lifting for integer block structured problems. Journal of combinatorial optimization 7(2) 161-167 (2003)
2. Posta, M., Ferland, J. A., and Michelon, P.: An exact method with variable fixing for solving the generalized assignment problem. Comput. Optim. Appl. 52(3), 629–644 (2012)
3. Fisher, M. L.: The Lagrangian relaxation method for solving integer programming problems. Management Sci. 27(1), 1-18 (1981)
4. Nagy, G., and Salhi, S.: Location-routing: Issues, models and methods. Eur. J. Oper. Res. 177(2), 649-672 (2007)
5. Prins, C., Prodhon, C., Ruiz, A., Soriano, P., and Wolfler Calvo, R.: Solving the capacitated location-routing problem by a cooperative Lagrangean relaxation-granular tabu search heuristic. Transp. Sci. 41(4), 470-483 (2007)

6. Belenguer, J.-M., Benavent, E., Prins, C., Prodhon, C., and Wolfler-Calvo, R.: A branch-and-cut algorithm for the capacitated location routing problem. Comp. Oper. Res. 38(6), 931–941 (2010)

7. Kohl, N. and Madsen, O. B. G.: An Optimization Algorithm for the Vehicle Routing Problem with Time Windows Based on Lagrangian Relaxation. Oper. Res. 45, 395–406 (1997)

8. Guan, X., Luh, P. B., Yan, H. and Amalfi, J. A.: An optimization-based method for unit commitment. Int. J. Elec. Power Energy Syst. 14(1), 9–17 (1992)

9. Guan, X., Luh, P. B., Yan, H., and Rogan, P. M.: Optimization-based scheduling of hydrothermal power systems with pumped-storage units. IEEE Trans. Power Syst. 9(2), 1023-1031 (1994)

10. Zhuang, F. and Galiana, F.D.: Towards a More Rigorous and Practical Unit Commitment by Lagrangian Relaxation. IEEE Trans. Power Syst., 3, 763–773 (1988)

11. Zhai, Q. and Guan, X.: Unit commitment with identical units: Successive subproblem solving method based on Lagrangian relaxation. IEEE Trans. Power Syst., 17(4), 1250–1257 (2002)

12. Weerakorn, O., and Petcharaks, N.: Unit commitment by enhanced adaptive Lagrangian relaxation. IEEE Trans. Power Syst., 19(1), 620-528 (2004)

13. Jimenez, N., and Conejo, A.: Short-Term Hydrothermal Coordination by Lagrangian Relaxation: Solution of the Dual Problem. IEEE Trans. Power Syst., 14, 89–95 (1999)

14. Virmani, S., Adrian, E. C., Imhof, K. and Mukherjee, S.: Implementation of a Lagrangian relaxation based unit commitment problem. IEEE Trans. Power Syst., 4(4), 1373-1380 (1989)

15. Padberg, M., and Rinaldi, G.: A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. SIAM Review 33(1), 60-100 (1991)

16. Hoffman, K. L., and Padberg, M.: Solving airline crew scheduling problems by branch-and-cut. Mgmt. Sci. 39(6), 657-682 (1993)

17. Balas, E., Ceria, S., and Cornuéjols, G.: Mixed 0-1 Programming by lift-and-project in a branch-and-cut framework. Mgmt. Sci. 42(9), 1229-1246 (1996)

18. Everett, H.: Generalized Lagrange multiplier method for solving problems of optimum allocation of resources. Oper. Res. 11(3), 399-417 (1963)

19. Held, M., and Karp, R. M.: The traveling salesman problem and minimum spanning trees. Oper. Res. 18(6), 1138-1162 (1970)

20. Held, M., and Karp, R. M.: The traveling salesman problem and minimum spanning trees: Part II. Math. Program. 1(1), 6-25 (1971)

21. Fisher, M. L.: Optimal solution of scheduling problems using Lagrange multipliers: Part I. Oper. Res. 21(5), 1114-1127 (1973)

22. Geoffrion, A. M.: Lagrangean relaxation for integer programming, Springer Berlin Heidelberg 82-114 (1974)

23. Özyurt, Z., and Aksen, D.: Solving the multi-depot location-routing problem with Lagrangian relaxation. Extending the horizons: Advances in computing, optimization, and decision technologies. Springer US, 125-144 (2007)

24. Avella, P., Boccia, M., and Vasilyev, I.: A computational study of exact knapsack separation for the generalized assignment problem. Comput. Optim. Appl. 45(3), 543–555 (2010)

25. Balas, E., Ceria, S., and Cornuejols, G.: A Lift-and-Project Cutting Plane Algorithm for Mixed 0-1 Programs Math. Program. 58, 295-324 (1993)

26. Balas, E.: Disjunctive Programming. Annals of Discrete Mathematics, 5, 3-51 (1979)

27. Balas, E.: Recent Advances in Lift-and-Project, Technical Report, GSIA, Carnegie Mellon University (1997)

28. Balas, E. and Jeroslow, R.G.: Strengthening Cuts for Mixed Integer Programs. Eur. J. Oper. Res 4, 224–234 (1980)

29. Ceria, S. and Pataki, G.: Solving Integer and Disjunctive Programs by Lift-and-Project. In R.E. Bixby, E.A. Boyd, and R.Z. Rios-Mercado (eds.), IPCO VI, Lecture Notes in Computer Science, 1412. Springer, 271–283 (1998).

30. Balas, E.: A modified lift-and-project procedure. Math. Program. 79, 19–32 (1997)

31. Balas, E., and Perregaard, M.: Lift-and-project for mixed 0-1 programming: recent progress. Discrete Applied Mathematics, 123, 129–154 (2002)

32. Giandomenico, M., Rossi, F., and Smriglio. S.: Strong lift-and-project cutting planes for the stable set problem. Math. Program. Ser. A, 141(1–2), 165–192 (2013)

33. Ceria, S., Cornuéjols, G., and Dawande, M.: Combining and strengthening Gomory cuts. E. Balas and J. Clausen, eds. Proc. 4th IPCO Conference, Copenhagen, Denmark. Springer-Verlag, 438-451, (1995)

34. Caprara, A., and Fischetti, M.: {0,1/2}-Chvátal-Gomory cuts. Math. Program. 74(3), 221-235 (1996)

35. Eisenbrand, F.: Gomory-Chvátal cutting planes and the elementary closure of polyhedra. Doctoral Dissertation, Universität des Saarlandes, http://www2.math.uni-paderborn.de/fileadmin/Mathematik/AG-Eisenbrand/publications/diss.pdf (2000)

36. Aliev, I., and Letchford, A. N.: Iterated Chvátal-Gomory cuts and the geometry of numbers. arXiv preprint arXiv:1306.6031 (2013)

37. Gomory, R. E.: An algorithm for the mixed integer problem (No. RAND-P-1885). RAND Corp, Santa Monica, CA (1960)

38. Gomory, R. E.: Solving linear programming problems in integers. Combinatorial Analysis, 10, 211-215 (1960)

39. Land, A. H. and Doig, A. G.: An automatic method of solving discrete programming problems. Econometrica 28(3), 497-520 (1960)

40. Mitchell, J. E.: Branch-and-cut. Wiley encyclopedia of operations research and management science. John Wiley & Sons, Inc (2010)

41. Bragin, M. A., Luh, P. B., Yan, J. H., and Stern, G. A.: Surrogate Lagrangian relaxation and branch-and-cut for unit commitment with combined cycle units. In: Proceedings of the IEEE Power and Energy Society, General Meeting, National Harbor, Maryland (2014)

42. Bragin, M. A., Luh, P. B., Yan, J. H., and Stern, G. A.: Novel exploitation of convex hull invariance for solving unit commitment by using surrogate Lagrangian relaxation and branch-and-cut. In: Proceedings of the IEEE PES GM, Denver, Colorado (2015)

43. Ermoliev, Y. M.: Methods for solving nonlinear extremal problems. Cybernetics 2(4), 1–14 (1966)

44. Polyak, B. T.: A general method of solving extremum problems. Sov. Math. 8, 593–597 (1967)

45. Shor, N. Z.: On the rate of convergence of the generalized gradient method. Cybernetics 4(3), 79-80 (1968)

46. Shor, N. Z.: Generalized gradient methods for non-smooth functions and their applications to mathematical programming problems. Econ. Math. Methods 12(2), 337–356 (1976) (in Russian)

47. Brannlund, U., Lindberg, P. O., Niu, A. and Nilsson, J. E.: Railway timetabling using Lagrangian relaxation. Transportation Sci., 32. 358-369 (1998)

48. Fisher, M. L.: An applications oriented guide to Lagrangian relaxation. Interfaces 15(2), 10-21 (1985)

49. Sherali, H. D. and Choi, G.: Recovery of Primal Solutions When Using Subgradient Optimization Methods to Solve Lagrangian Duals of Linear Programs. Operations Research Letters, 19, 105-113 (1996)

50. Mulvey, J. M. and Crowder, H. P.: Cluster analysis: an application of Lagrangian relaxation. Management Sci. 25, 329-340 (1979)

51. Zhao, X., Luh, P. B, and Wang, J.: Surrogate gradient algorithm for Lagrangian relaxation. J. Optim. Theory Appl. 100(3), 699–712 (1999)

52. Bragin, M. A., Luh, P. B., Yan, J. H., Yu, N., and Stern, G. A.: Convergence of the surrogate Lagrangian relaxation method, J. Optim. Theory Appl. 164(1), 173-201 (2015)

53. Hestenes, M. R.: Multiplier and gradient methods. J. Optim. Theory Appl. 4(5), 303-320 (1969)

54. Powell, M. J. D.: A method for nonlinear constraints in minimization problems. In: Optimization, (R. Fletcher, ed.), Academic Press (1969)

55. Bertsekas, D. P.: Nonlinear Programming, 3rd Edition, Athena Scientific, Belmont, MA (2016)

56. Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundations and Trends in Machine Learning. 3(1), 1–122 (2010)

57. Feizollahi, M. J., Ahmed, S, and Sun, A.: Exact augmented Lagrangian duality for mixed integer linear programming. Mathematical Programming, 1-23, 2016

Chapter 2

Convergence of the Surrogate Lagrangian Relaxation Method

Studies have shown that the surrogate subgradient method, to optimize non-smooth dual functions within the Lagrangian relaxation framework, can lead to significant computational improvements as compared to the subgradient method. The key idea is to obtain surrogate subgradient directions, that form acute angles toward the optimal multipliers without fully minimizing the relaxed problem. The major difficulty of the method is its convergence, since the convergence proof and the practical implementation require the knowledge of the optimal dual value. Adaptive estimations of the optimal dual value may lead to divergence and the loss of the lower bound property for surrogate dual values. The main contribution of this section is on the development of the surrogate Lagrangian relaxation method and its convergence proof to the optimal multipliers, without the knowledge of the optimal dual value and without fully optimizing the relaxed problem. Moreover, for practical implementations a stepsizing formula, that guarantees convergence without requiring the optimal dual value, has been constructively developed. The key idea is to select stepsizes in a way that distances between Lagrange multipliers at consecutive iterations decrease, and as a result, Lagrange multipliers converge to a unique limit. At the same time, stepsizes are kept sufficiently large so that the algorithm does not terminate prematurely. At convergence, the lower-bound property of the surrogate dual is guaranteed. Testing results demonstrate that non-smooth dual functions can be efficiently optimized, and the new method leads to faster convergence as compared to other methods available for optimizing non-smooth dual functions, namely, the simple subgradient method, the subgradient-level method and the incremental subgradient method.

2.1. Introduction

When solving complicated mixed-integer optimization problems, the effort needed to obtain an optimal solution increases dramatically as the problem size increases. Therefore, the goal for practical applications is often to obtain a near-optimal solution with quantifiable quality in a computationally efficient manner. Lagrangian relaxation has successfully achieved this goal by exploiting separability of a problem. In the method, the relaxed problem is fully optimized, and the dual function is obtained. Dual functions are always concave, and the feasible set of dual solutions is always convex regardless of the characteristics of the original problem such as convexity. To optimize non-smooth concave dual functions, Lagrange multipliers are adjusted based on appropriately defined stepsizes and by using subgradient directions [1-13] or surrogate subgradient directions [14-20]. At convergence of multipliers, heuristics are typically used to obtain feasible solutions.

The subgradient method has been extensively studied starting with the pioneering works of Ermoliev [1], Polyak [2-3] and Shor [4-5]. The general convergence has been established in [1] and [2]. To ensure convergence with a geometric rate, convergence was proved by requiring the optimal dual value [3]. In practical implementations, adaptive rules to adjust estimates of the optimal dual value were first developed in [3], and such rules have been improved in [7–9, 11, 21] to guarantee convergence to the optimum. Difficulties associated with the unavailability of the optimal dual value have been overcome owing to the fact that subgradients form acute angles with directions toward the optimal multipliers and owing to the convexity of the dual function. Therefore, for properly chosen stepsizes, multipliers move closer to the optimal multipliers. For example, in the subgradient-level method [7], stepsizes are set by using estimates of the optimal dual value based on the highest dual value obtained so far, and such estimates are further adjusted when significant oscillations of multipliers are detected. However, subgradient methods require the relaxed problem to be fully optimized, which can be difficult when the relaxed problem is non-separable or NP-hard. Moreover, convergence can be slow because multipliers often zigzag across the ridges of the dual function, and the zigzagging is especially noticeable when

ridges are sharp. While incremental subgradient methods [8] reduce computational requirements by optimizing one subproblem at a time and converge without requiring the optimal dual value, these methods require separability of the problem and cannot be used to solve non-separable problems, or when subproblems are NP-hard.

The surrogate subgradient method, developed within the Lagrangian relaxation framework, is a variation of the subgradient method that seeks to reduce computational requirements and to obtain surrogate subgradient directions that form acute angles with directions toward to the optimal multipliers [15–20]. Without requiring the relaxed problem to be fully optimized, surrogate subgradient directions do not change drastically, thereby alleviating the zigzagging of multipliers and reducing the number of iterations required for convergence. The major difficulty of the method is its convergence, since the convergence proof and the practical implementation require the knowledge of the optimal dual value, which is unavailable in practice. In the method, since the relaxed problem is not fully optimized, surrogate dual values are no longer on but are above the dual surface. As a result, surrogate subgradient directions may not form acute angles with directions toward the optimal multipliers, and divergence may occur. In addition, the lower bound property of surrogate dual values may be lost. While such difficulties can sometimes be overcome by occasionally obtaining subgradients during the convergence process, the computational effort can still be prohibitive when the relaxed problem is non-separable or NP-hard.

In this Section, surrogate Lagrangian relaxation with novel conditions on stepsizes is developed, and convergence of the method is proved without requiring the optimal dual value and without fully optimizing the relaxed problem in Section 2.2. The idea is to select stepsizes in a way that distances between Lagrange multipliers at consecutive iterations decrease, and as a result, multipliers converge to a unique limit. At the same time, stepsizes are kept sufficiently large so that the algorithm does not terminate prematurely. At convergence, a surrogate dual value provides a lower bound to the primal cost. Moreover, a particular stepsizing formula that satisfies the set of conditions has been obtained. Convergence of the interleaved method [14], in which one subproblem is solved at a time to update

multipliers, has also been proved. Under additional assumptions used in subgradient methods [8], the convergence rate of the new method is linear.

Section 2.3 presents testing results for a small nonlinear integer optimization problem, large generalized assignment problems and quadratic assignment problems. For the small problem, the new method is compared with the subgradient method to demonstrate that the zigzagging is alleviated, and calculations of surrogate subgradient directions require significantly lower computational effort. The new method is then compared with the methods available for non-smooth optimization such as the simple subgradient method, the subgradient-level method and the incremental subgradient method when solving generalized assignment problems with separable dual problems, and quadratic assignment problems with non-separable dual problems.

2.2. Convergence of the Surrogate Lagrangian Relaxation Method

This section is on the development of the novel surrogate Lagrangian relaxation method and its convergence proof without requiring the optimal dual value. In Subsection 2.1, a generic mixed-integer problem formulation and the Lagrangian relaxation framework are presented. To maximize non-smooth dual functions, subgradient directions and stepsizes requiring the optimal dual value are frequently used [22]. To find multiplier-updating directions that form acute angles with directions toward the optimal multipliers without fully minimizing the relaxed problem, the Lagrangian relaxation and surrogate subgradient method [15] is presented next. In Subsection 2.2, the surrogate Lagrangian relaxation method is developed, and convergence of the method is proved without requiring the optimal dual value. Convergence rate of the method is discussed in Subsection 2.3. Subsection 2.4 discusses practical implementation aspects of the algorithm such as a constructive stepsize-setting procedure.

2.2.1. Mixed-Integer Programming and Lagrangian Relaxation

Consider a mixed-integer problem formulation:

$$\min_x f(x), \text{ subject to } g(x) \leq 0,\ x \in X, \tag{2.1}$$

where $x = (y, z)$, $y \in \mathbb{R}^{Nr}$, $z \in \mathbb{Z}^{Nz}$, and $X \subseteq \mathbb{R}^{Nr} \times \mathbb{Z}^{Nz}$, with $\mathbb{R}$ denoting the set of real numbers, $\mathbb{Z}$ denoting the set of integers, f: $X \rightarrow$ IR and g: $X \rightarrow \mathbb{R}^m$ are continuous and differentiable with respect to y. In addition, $g(x)$ satisfy the following assumptions:

Assumption 2.2.1 There exists a scalar M such that

$$||g(x)|| < M < \infty,\ \forall x \in X. \tag{2.2}$$

Assumption 2.2.2. Regularity Condition Gradient vectors of active inequality constraints with respect to y are linearly independent at a constrained local minimum $x^* = (y^*, z^*)$ of $f(x)$. □

The regularity Assumption 2.2.2 is needed only in the continuous subspace $\mathbb{R}^{Nr}$ to rule out possible irregularities, such as linear dependence of gradients of active constraints. In the discrete subspace $\mathbb{Z}^{Nz}$, regularity conditions are not needed [23].

When solving discrete optimization problems, the Lagrangian relaxation method has been used [15, 22] and shown to be especially powerful for solving separable programming problems. In the method, the constraints of (2.1) are relaxed and the Lagrangian function is formed by introducing a vector of Lagrange multipliers $\lambda^T = (\lambda_1, \ldots, \lambda_m) \in \mathbb{R}^m$:

$$L(\lambda, x) := f(x) + \lambda^T g(x). \tag{2.3}$$

The dual function, resulting from the minimization of the Lagrangian function (2.3), becomes

$$q(\lambda) := \min_{x \in X} L(\lambda, x), \tag{2.4}$$

and the dual problem is to maximize the concave non-smooth dual function [22]:

$$\max_{\lambda} q(\lambda), \text{ s.t. } \lambda \in \mathbb{R}^m,\ \lambda \geq 0. \tag{2.5}$$

When the original problem is integer or mixed-integer linear, the dual function is polyhedral concave and non-smooth [15]. In the subgradient method, to maximize the dual function, multipliers are updated according to:

$$\lambda^{k+1} = [\lambda^k + c^k g(x^k)]^+, \; k = 0, 1, \ldots, \tag{2.6}$$

where $[]^+$ denotes projection onto the positive orthant, $g(x^k)$ is the subgradient of the dual function $q(\lambda)$ at λ^k and c^k is a positive scalar stepsize. If equality constraints $h(x) = 0$ are present in the formulation, multipliers are updated according to (2.6) without projecting onto the positive orthant.

Since $q(\lambda)$ is convex, dual values are not greater than the optimal dual value $q^* := q(\lambda^*)$

$$q(\lambda^k) \le q^*. \tag{2.7}$$

Moreover, by the definition of subgradients, the following relationship holds:

$$q^* - q(\lambda^k) \le (\lambda^* - \lambda^k)^T g(x^k). \tag{2.8}$$

Both sides of the inequality (2.8) are nonnegative owing to the inequality (2.7). Therefore, subgradient directions from acute angles with directions toward λ^*, and distances between the current multipliers and the optimum λ^* can be decreased under the following condition on stepsizes [22]:

$$0 < c^k < \frac{2(q^* - q(\lambda^k))}{\left\| g(x^k) \right\|^2}, \; k = 0, 1, \ldots. \tag{2.9}$$

While q^* is unknown in practice, significant research has been done to guarantee convergence to λ^* by adaptively estimating q^*. For example, in the subgradient-level method [7], q^* can be adaptively adjusted based on such criteria as a sufficient ascent of a dual value or significant oscillation of the multipliers.

While subgradient directions are traditionally used to update multipliers, such directions may almost be perpendicular to directions toward λ^*, thereby leading to slow convergence. Moreover, while

optimization of the relaxed problem (2.4) is generally simpler than the optimization of the original problem (2.1)–(2.2), it can still be difficult when the relaxed problem is non-separable and NP-hard. This usually leads to difficulties of fully optimizing the relaxed problem (2.4) and computing corresponding subgradient directions of the dual function (2.4). Therefore, it is desirable to obtain multiplier-updating directions that form acute angles with directions toward λ^* in a computationally efficient manner and to show that multipliers are moving closer to λ^*.

To reduce computational requirements by not requiring the relaxed problem to be fully optimized, the Lagrangian relaxation and surrogate subgradient method has been developed in [15] for separable integer programming problems under the assumption that the constraint functions are $g(x) = Ax–b$. For our problem (2.1)-(2.2) under consideration, for any feasible solution $x^k \in X$ of the relaxed problem, the *surrogate dual* is defined following [15] as:

$$\tilde{L}(\lambda^k, x^k) := f(x^k) + (\lambda^k)^T \tilde{g}(x^k), \tag{2.10}$$

where

$$\tilde{g}(x^k) := g(x^k) \tag{2.11}$$

is the surrogate subgradient direction.

Since the relaxed problem is not fully optimized, surrogate dual values are generally above the dual surface and can be larger than q^*, thereby causing the violation of (2.7). As a result, surrogate subgradient directions may not form acute angles with directions toward λ^*, and divergence may occur.

To guarantee that surrogate subgradient directions form acute angles with directions toward λ^*, the relaxed problem has to be sufficiently optimized, such that surrogate dual values (2.10) satisfy the following *surrogate optimality condition*:

$$\tilde{L}(\lambda^k, x^k) < \tilde{L}(\lambda^k, x^{k-1}), \tag{2.12}$$

and stepsizes have to be sufficiently small

$$0 < c^k < \frac{q^* - \tilde{L}(\lambda^k, x^k)}{\left\| \tilde{g}(x^k) \right\|^2}, \; k = 0, 1, \ldots . \tag{2.13}$$

Under the assumption that constraints are $g(x) = Ax–b$, it has been proved in [15,17] that multipliers move closer to λ^* at every iteration when updated recursively:

$$\hat{\lambda}^{k+1} = \lambda^k + c^k \tilde{g}(x^k), \; k = 0, 1, \ldots \tag{2.14}$$

$$\lambda^{k+1} = [\hat{\lambda}^{k+1}]^+, \; k = 0, 1, \ldots \tag{2.15}$$

where x^k satisfy (2.12), and c^k satisfy (2.13).

In addition, it has been shown that the lower-bound property of a surrogate dual function is preserved

$$\tilde{L}(\lambda^k, x^k) < q^*, k = 0, 1, \ldots . \tag{2.16}$$

While convergence was proved [15, 17] when the constraints are $g(x) = Ax–b$, the proof in [15, 17] does not use or require linearity of $g(x)$ to establish convergence. Therefore, for general constraints $g(x)$ under the regularity condition of Assumption 2.1.2, multipliers converge to λ^* and the lower bound property (2.16) is preserved if multipliers are updated according to (2.14)-(2.15), and stepsizes satisfy (2.13).

When the original problem (2.1)–(2.2) is separable, that is, the objective function $f(x)$ and constraints $g(x)$ are of an additive form, the relaxed problem (2.4) can be separated into N_s individual subproblems. Within the surrogate subgradient framework, it is sufficient to optimize the relaxed problem (2.4) with respect to several subproblems ($<N_s$), subject to the surrogate optimality condition (2.12), to obtain surrogate subgradient directions. The accompanying computational effort is approximately $1/N_s$ per subproblem of the effort required to obtain subgradient directions.

When the original problem (2.1)–(2.2) is non-separable or difficult to decompose into individual subproblems, the relaxed problem (2.4), subject to the surrogate optimality condition (2.12), can also be optimized with a sizable efficiency gain as compared to the subgradient method to obtain surrogate subgradient directions by optimizing the relaxed problem with respect to selected decision variables, while keeping other decision variables fixed.

The major difficulty of the surrogate subgradient method is its convergence, since the upper bound on stepsizes (2.13) cannot be specified due to the unavailability of q^*. In practical implementations, estimates of q^* may violate (2.13), thereby leading to divergence.

2.2.2. The Surrogate Lagrangian Relaxation Method

In this section, the main theoretical contribution of this C, a new method is developed, and convergence to λ^* is proved without requiring q^*. In the method, the surrogate optimality condition (2.12) and the multiplier-updating formulas (2.14) and (2.15) will be used. To prove and guarantee convergence without requiring q^*, instead of the stepsizing formula (2.13), a new formula to set stepsizes will be developed, and convergence of multipliers (2.14)-(2.15) to λ^* will be proved. In addition, it will be proved that an interleaved method [14] with the new stepsizing formula also converges to λ^*.

The main idea is to obtain stepsizes such that distances between multipliers[1] at consecutive iterations decrease, i.e.,

$$\left\| \hat{\lambda}^{k+1} - \lambda^k \right\| = \alpha_k \left\| \hat{\lambda}^k - \lambda^{k-1} \right\|, \; 0 < \alpha_k < 1, \; k = 1, 2, \ldots . \tag{2.17}$$

The stepsizing formula satisfying (2.17) can be derived by using (2.14). Indeed, (2.14) and (2.17) imply

[1] Strictly speaking, when dealing with inequality constraints $g(x) \leq 0$, distances between multipliers and projections of multiples from the previous iteration are considered rather than distances between multipliers.

$$\left\| c^k \tilde{g}(x^k) \right\| = \alpha_k \left\| c^{k-1} \tilde{g}(x^{k-1}) \right\|, \; 0 < \alpha_k < 1, \; k = 1, 2, \ldots . \tag{2.18}$$

In the new method, stepsizes c^k satisfying (2.18) can always be uniquely obtained, unless norms of surrogate subgradients are zero[2]. Therefore, norms of surrogate subgradients are subject to a strict positivity requirement:

$$\left\| \tilde{g}(x^k) \right\| > 0 . \tag{2.19}$$

Since c^k and c^{k-1} are positive scalars[3], and norms of surrogate subgradients are strictly positive, (2.18) implies

$$c^k = \alpha_k \frac{c^{k-1} \left\| \tilde{g}(x^{k-1}) \right\|}{\left\| \tilde{g}(x^k) \right\|}, \; 0 < \alpha_k < 1, \; k = 1, 2, \ldots . \tag{2.20}$$

The combined multiplier-updating formula (2.14)-(2.15) and (2.17) can be viewed as a mapping from $\hat{\lambda}^k$ ($\in \mathbb{R}^M$) to $\hat{\lambda}^{k+1}$ ($\in \mathbb{R}^M$). Since the distances between multipliers at consecutive iterations always strictly decrease per (2.17), multipliers converge to a limit, and stepsizes approach zero. When $\{\alpha_k\}$ are too small, however, stepsizes can approach 0 too fast, and the algorithm may terminate prematurely. To avoid that, stepsizes (2.20) should be kept sufficiently large, and this can be achieved by keeping α_k sufficiently close to 1 as proved in the following theorem, the main result of this Chapter.

Theorem 2.2.3 Suppose that multiplier-updating directions satisfy the conditions (2.12) and (2.19) and constraints (2.2) satisfy the regularity condition of Assumption 2.2.2. If α_k satisfies the following conditions:

---

[2] In the subgradient method, zero-subgradient implies that the optimum is obtained, and the algorithm terminates with the optimal primal solution. In the surrogate subgradient method, zero-surrogate subgradient implies that only a feasible solution is obtained and the algorithm must proceed.

[3] Initial stepsize c^0 is initialized to be a positive scalar, therefore, stepsizes c^k, $k = 1, 2, \ldots$ satisfying (2.18) are positive.

$$\prod_{i=1}^{k} \alpha_i \to 0 \,, \tag{2.21a}$$

and

$$\lim_{k\to\infty} \frac{1-\alpha_k}{c^k} = 0 \,, \tag{2.21b}$$

then the mapping (2.14)-(2.15), with c^k satisfying (2.20) has a unique fixed point λ^*. □

This theorem is proved in three stages. In Stage 1, convergence to a unique fixed point (not necessarily λ^*) is proved under the condition (2.21a). In Stage 2, convergence to λ^* is proved by temporarily using q^* to establish a lower bound on stepsizes. In Stage 3, the proof is completed with an additional asymptotical condition (2.21b) without requiring q^*.

Proposition 2.2.4 With the stepsizing formula (2.20), the Lagrange multipliers (2.14)-(2.15) converge to a unique fixed point $\bar{\lambda} \equiv \lim_{k\to\infty} \lambda^k$ (not necessarily to λ^*), provided (2.21a) and the norm positivity requirement (2.19) hold. □

Proof From (2.20) it follows that

$$c^k = \prod_{i=1}^{k} \alpha_i \frac{c^0 \left\| \tilde{g}(x^0) \right\|}{\left\| \tilde{g}(x^k) \right\|}, \; k = 1, 2, \ldots . \tag{2.22}$$

Then by using (2.14) and (2.22), we get

$$\left\| \hat{\lambda}^{k+1} - \lambda^k \right\| = c^0 \left\| \tilde{g}(x^0) \right\| \prod_{i=1}^{k} \alpha_i \,. \tag{2.23}$$

Since projections are non-expansive, (2.23) can be written as an inequality

$$\left\| \lambda^{k+1} - \lambda^k \right\| \le c^0 \left\| \tilde{g}(x^0) \right\| \prod_{i=1}^{k} \alpha_i \,. \tag{2.24}$$

Since (2.21a) holds, $\left\| \lambda^{k+1} - \lambda^k \right\|$ approach zero.

To prove that multipliers converge to a unique fixed point, it will be proved that surrogate dual values approach dual values as $\left\|\lambda^{k+1} - \lambda^{k}\right\|$ become small for a sufficiently large iteration $k = L$. After that, the proof uses an argument similar to convergence results of the subgradient method with a diminishing stepsize rule as in [1, 2, 8, 9, 22].

To prove that surrogate dual values approach dual values, consider an arbitrary and a fixed value of multipliers λ at an arbitrary iteration M. A series of surrogate optimizations for the fixed value of λ, subject to the surrogate optimality condition (2.12), consecutively finds solutions $x^{M+1}, x^{M+2}, \ldots$ that satisfy

$$q(\lambda) < \ldots < \tilde{L}(\lambda, x^{M+2}) < \tilde{L}(\lambda, x^{M+1}) < \tilde{L}(\lambda, x^{M}), \tag{2.25}$$

until a dual value $q(\lambda)$ is reached. Given the discrete nature of the original problem (2.1)–(2.2), only a finite number of iterations in (2.25) is required to reach $q(\lambda)$, and $\tilde{L}(\lambda, x^{M+k_0}) = q(\lambda)$ for a positive number k_0. For example, when a problem has N_s subproblems, and one subproblem is optimized at a time, then $q(\lambda)$ is obtained within at most $k_0 = N_s$ iterations.

Following the same logic, when $\left\|\lambda^{k+1} - \lambda^{k}\right\|$ are sufficiently small, surrogate subgradient directions approach subgradient directions. Indeed, since $\left\|\lambda^{k+1} - \lambda^{k}\right\|$ in (2.24) converge to zero, there exists an iteration L and a positive finite number l such that the distance between λ^{L} and λ^{L+l} is sufficiently small such that values $q(\lambda^{L})$ and $q(\lambda^{L+l})$ belong to the same facet of the dual function $q(\lambda)$. As in (2.25), starting from an iteration L, a surrogate dual value $\tilde{L}(\lambda^{L}, x^{L})$ converges to a dual value $q(\lambda^{L+l})$ within a finite number of iterations l. Therefore, starting from iteration $L+l$, surrogate subgradient directions become subgradient directions. In the subgradient method with stepsizes approaching zero, multipliers converge to a fixed point: $\bar{\lambda} = \lim_{k \to \infty} \lambda^{k}$. □

The condition (2.21a) alone is not sufficient to guarantee convergence to λ^*, since stepsizes may approach 0 fast, thereby leading to a premature algorithm termination. To avoid that, stepsizes will be kept sufficient large by temporarily introducing q^*.

Proposition 2.2.5. Sufficient Condition for Convergence to λ^* With the stepsizing formula (2.20), condition (2.21a), and the norm positivity requirement (2.19), the Lagrange multipliers (2.14)-(2.15) converge to λ^* if there exist $\kappa > k$ for all k and stepsizes satisfy the following lower-bound condition:

$$\frac{q^* - \tilde{L}(\lambda^\kappa, x^\kappa)}{\left\|\tilde{g}(x^\kappa)\right\|^2} \le c^\kappa . \tag{2.26}$$

Proof To prove that the multipliers $\bar{\lambda}$ are optimal when stepsizes c^k satisfy conditions (2.20) and (2.26), and stepsize-setting parameters α^k satisfy (2.21a), the following equality is to be established

$$q(\bar{\lambda}) = q^* . \tag{2.27}$$

The lower-bound condition on stepsizes (2.26) leads to

$$q^* - \tilde{L}(\lambda^\kappa, x^\kappa) \le c^\kappa \left\|\tilde{g}(x^\kappa)\right\|^2 . \tag{2.28}$$

Conditions (2.21a) and (2.22) imply $c^k \to 0$. Since $\kappa > k$, then $c^\kappa \to 0$ as $k \to \infty$, and inequality (2.28) yields

$$q^* - \tilde{L}(\lambda^\kappa, x^\kappa) \le 0 . \tag{2.29}$$

According to Proposition 2.3, $\bar{\lambda} = \lim_{k\to\infty} \lambda^k$. Since $\kappa > k$, then $\bar{\lambda} = \lim_{\kappa\to\infty} \lambda^\kappa$ implies

$$q^* - \tilde{L}(\bar{\lambda}, x^\kappa) \le 0 . \tag{2.30}$$

From the inequality (2.25) it follows that $\tilde{L}(\bar{\lambda}, x^\kappa) \to q(\bar{\lambda})$. Therefore, by using (2.25) and (2.30) we get

$$q^* - q(\bar{\lambda}) \le 0 . \tag{2.31}$$

Therefore, $\bar{\lambda}$ maximizes the dual function, and $\bar{\lambda}$ is an optimum. □

Theorem 2.2 will now be proved by contradiction by using condition (2.21b) without requiring q^*. It will be shown that a condition contrary to (2.26) does not hold under condition (2.21b), thereby proving that multipliers converge to λ^*.

Proof of Theorem 2.2.3.

Proof The formal proof follows by a contradiction.

Step 1: Assuming that a condition contrary to Condition (2.26) holds, there exists κ such that for all $k \ge \kappa$,

$$c^k < \frac{q^* - \tilde{L}(\lambda^k, x^k)}{\left\| \tilde{g}(x^k) \right\|^2} . \tag{2.32}$$

Under the surrogate optimality condition (2.12) and the condition (2.32), surrogate subgradient directions form acute angles with directions toward λ^*, multipliers move closer to λ^*, the lower bound property (2.16) of the surrogate dual is preserved, and the following inequality holds [15, 17]:

$$0 < q^* - \tilde{L}(\lambda^k, x^k) \le (\lambda^* - \lambda^k)^T \, \tilde{g}(x^k) \le \left\| \lambda^* - \lambda^k \right\| \left\| \tilde{g}(x^k) \right\| . \tag{2.33}$$

From (2.32) and (2.33) it follows that

$$c^k < \frac{\left\| \lambda^* - \lambda^k \right\| \left\| \tilde{g}(x^k) \right\|}{\left\| \tilde{g}(x^k) \right\|^2} = \frac{\left\| \lambda^* - \lambda^k \right\|}{\left\| \tilde{g}(x^k) \right\|} . \tag{2.34}$$

Therefore, for all $k \ge \kappa$,

$$c^k \left\| \tilde{g}(x^k) \right\| < \left\| \lambda^* - \lambda^k \right\| . \tag{2.35}$$

In general, stepsizes satisfying (2.20) and (2.35) may not lead to convergence, since stepsizes c^k may decrease faster than distances between λ^* and λ^k, and multipliers may not reach λ^*.

Step 2: It will be proved that the condition (2.21b) ensures that stepsizes c^k decrease slower than distances between λ^* and λ^k, and that the inequality (2.35) is violated as a result.

Consider the inequality (2.35) at an iteration $\kappa + m$ (m>0)

$$c^{\kappa+m}\left\|\tilde{g}(x^{\kappa+m})\right\| < \left\|\lambda^* - \lambda^{\kappa+m}\right\|. \tag{2.36}$$

Since the inequality (2.32) holds by assumption, multipliers move closer to λ^*, and there exists $0 < \beta^{\kappa+m-1} < 1$ such that

$$\left\|\lambda^* - \lambda^{\kappa+m}\right\| = \beta^{\kappa+m-1}\left\|\lambda^* - \lambda^{\kappa+m-1}\right\|. \tag{2.37}$$

The value of β^k ($k \geq \kappa$) is the rate with which multipliers approach λ^*. When $\beta^k \sim 1$,[4] the contradiction will be established by showing that the left-hand side of (2.35) decreases slower than the right-hand side for sufficiently large values of α_k (< 1) as k increases. With the stepsizing formula (2.20) and with the equality (2.37), the inequality (2.36) becomes

$$\alpha_{\kappa+m-1}c^{\kappa+m-1}\left\|\tilde{g}(x^{\kappa+m-1})\right\| < \beta^{\kappa+m-1}\left\|\lambda^* - \lambda^{\kappa+m-1}\right\|. \tag{2.38}$$

Following the same logic, the inequality (2.38) can be inductively represented in the following way:

$$c^{\kappa}\left\|\tilde{g}(x^{\kappa})\right\| < \frac{\prod_{i=\kappa}^{\kappa+m-1}\beta_i}{\prod_{i=\kappa}^{\kappa+m-1}\alpha_i}\left\|\lambda^* - \lambda^{\kappa}\right\|,\ m > 0. \tag{2.39}$$

[4] When β^k<<1, the right-hand side of (35) decreases faster than the left-hand side as k increases. This leads to the contradiction, and the theorem is proved.

To arrive at the contradiction, given that the left-hand of (2.39) is positive, the right-hand side of (2.39) will be proved to be arbitrarily small under (2.21b) as m increases.

From (2.14)-(2.15), (2.37) and the non-expansive property of projections, it follows that

$$\beta^k := \frac{\left\|\lambda^* - \lambda^{k+1}\right\|}{\left\|\lambda^* - \lambda^k\right\|} \le \frac{\left\|\lambda^* - \lambda^k - c^k \tilde{g}(x^k)\right\|}{\left\|\lambda^* - \lambda^k\right\|}. \tag{2.40}$$

The right-hand side of (2.40) can be expanded in Taylor series around $c^k \to 0$, while keeping first two terms of the expansion by using the following relation:

$$\frac{\partial}{\partial c}\left\|h(c)\right\|_2 = \frac{h(c)^T \frac{\partial}{\partial c} h(c)}{\left\|h(c)\right\|_2}, \tag{2.41}$$

where h is a vector-valued function of c. Therefore,

$$\beta^k \le 1 - \frac{(\lambda^* - \lambda^k)^T \ \tilde{g}(x^k)\ c^k}{\left\|\lambda^* - \lambda^k\right\|^2} + O((c^k)^2). \tag{2.42}$$

Consider the following ratio:

$$\frac{1-\alpha_k}{1-\beta^k} \le \frac{1-\alpha_k}{c^k}\left(\frac{(\lambda^* - \lambda^k)^T \ \tilde{g}(x^k)}{\left\|\lambda^* - \lambda^k\right\|^2} - O(c^k)\right)^{-1}. \tag{2.43}$$

The second term of the product in the right-hand side of (2.43) is bounded. Indeed, from the relation (2.33) it follows that

$$\varepsilon < \frac{(\lambda^* - \lambda^k)^T \ \tilde{g}(x^k)}{\left\|\lambda^* - \lambda^k\right\|^2} \le \frac{\left\|\tilde{g}(x^k)\right\|}{\left\|\lambda^* - \lambda^k\right\|}, \tag{2.44}$$

for any small $\varepsilon > 0$.

For sufficiently small c^k, we can assume that $-\varepsilon/2 < O(c^k) < \varepsilon/2$, therefore,

$$\frac{\varepsilon}{2} < \frac{(\lambda^* - \lambda^k)^T \, \tilde{g}(x^k)}{\left\|\lambda^* - \lambda^k\right\|^2} - O(c^k) \,. \tag{2.45}$$

Assuming $\|\lambda^k - \lambda^*\| > \varepsilon > 0$, and $\|g(x)\| < M < \infty$, from (2.44) and (2.45) it follows that

$$0 < \frac{\varepsilon}{2} < \frac{(\lambda^* - \lambda^k)^T \, \tilde{g}(x^k)}{\left\|\lambda^* - \lambda^k\right\|^2} - O(c^k) < \frac{\left\|\tilde{g}(x^k)\right\|}{\left\|\lambda^* - \lambda^k\right\|} + \frac{\varepsilon}{2} < \frac{M}{\varepsilon} + \frac{\varepsilon}{2} < \infty \,. \tag{2.46}$$

Therefore, the reciprocal value in (2.43) is also bounded. On the other hand, if $\|\lambda^k - \lambda^*\| < \varepsilon$ for any small $\varepsilon > 0$, then $\lambda^k \to \lambda^*$, and the convergence is proved.

When the asymptotical condition (2.21b) holds, the right-hand side of (2.43) converges to zero as $k \to \infty$. Therefore, the left-hand side of (2.43) converges to zero

$$\frac{1 - \alpha_k}{1 - \beta^k} \to 0 \,. \tag{2.47}$$

To arrive at the contradiction, we need to show that, while the left-hand side of (2.39) is constant for a given κ, the right-hand side can be made arbitrarily small as $m \to \infty$, provided (2.47) holds. That is, it remains to be proved that, for any predetermined and arbitrarily small value $\varepsilon > 0$, there exists an iteration m_ε satisfying

$$\frac{\prod_{i=\kappa}^{\kappa+m_\varepsilon-1} \beta^i}{\prod_{i=\kappa}^{\kappa+m_\varepsilon-1} \alpha_i} < \varepsilon \,. \tag{2.48}$$

Based on (2.47), α_k approaches 1 faster than the entire expression in (2.47) approaches zero. Therefore, there exists an iteration N such that for any $n > N$ there exist a positive constant $\delta_n > 0$, and for an arbitrarily small positive $\varepsilon_1 > 0$ the following conditions hold:

$$\frac{1 - \alpha_n}{1 - \beta^n} < \varepsilon_1 \tag{2.49}$$

and

$$1-\alpha_n = \varepsilon_1^{1+\delta_n}\,. \tag{2.50}$$

From the inequality (2.49) it follows that

$$\frac{\beta^n}{\alpha_n} < \frac{1}{\alpha_n}\left(1+\frac{\alpha_n}{\varepsilon_1}-\frac{1}{\varepsilon_1}\right) = \frac{1}{\varepsilon_1}+\frac{1}{\alpha_n}\left(1-\frac{1}{\varepsilon_1}\right). \tag{2.51}$$

Based on (2.50), the inequality (2.51) becomes

$$\frac{\beta^n}{\alpha_n} < \frac{1}{\varepsilon_1}+\frac{1}{1-\varepsilon_1^{1+\delta_n}}\left(\frac{\varepsilon_1-1}{\varepsilon_1}\right) = \frac{1}{\varepsilon_1}\left(1+\frac{\varepsilon_1-1}{1-\varepsilon_1^{1+\delta_n}}\right) = \frac{1}{\varepsilon_1}\left(\frac{-\varepsilon_1^{1+\delta_n}+\varepsilon_1}{1-\varepsilon_1^{1+\delta_n}}\right) = \left(\frac{1-\varepsilon_1^{\delta_n}}{1-\varepsilon_1^{1+\delta_n}}\right). \tag{2.52}$$

Given that $\varepsilon_1>0$ is arbitrarily small, (2.52) becomes

$$\frac{\beta^n}{\alpha_n} < \left(\frac{1-\varepsilon_1^{\delta_n}}{1-\varepsilon_1^{1+\delta_n}}\right) \sim 1-\varepsilon_1^{\delta_n} < 1-\varepsilon_1^{1+\delta_n} = \alpha_n\,. \tag{2.53}$$

Therefore, given (2.21a),

$$\prod_{i=n}^{k}\frac{\beta^i}{\alpha_i} < \prod_{i=n}^{k}\alpha_i \to 0\,. \tag{2.54}$$

Thus, the inequality (2.48) is established for an arbitrary small value $\varepsilon > 0$ and iteration m_ε. Therefore, (2.39) becomes

$$c^\kappa \left\| \tilde{g}(x^\kappa) \right\| < \varepsilon \left\| \lambda^* - \lambda^\kappa \right\|. \tag{2.55}$$

Since $\varepsilon > 0$ is arbitrarily small, the inequality (2.55) does not hold for a fixed iteration κ. Therefore, the inequality (2.34) does not hold for $k = \kappa+m_\varepsilon$. This contradicts the assumption, and convergence to λ^* is proved. □

Based on the convergence results proved in Theorem 2.2.3, the following Corollary discusses the convergence of the interleaved method [14] developed for separable problems. In the method, Lagrange multipliers are updated after each subproblem is solved.

Corollary 2.2.6 The interleaved method converges with the novel stepsizing formula (2.20) provided the conditions (2.21a) and (2.21b) hold.

Proof The interleaved method is defined for separable problems. For such problems, after constraints are relaxed, the Lagrangian function can be represented in an additive form $L = L_1 + \ldots + L_{Ns}$, and the relaxed problem can be separated into N_s subproblems. To prove this Corollary, it is sufficient to show that the surrogate optimality condition (2.12) holds after one subproblem is solved. Indeed, after a subproblem i is solved to optimality, and x_i^k is obtained, then by definition of an optimum

$$L_i(\lambda^k, x_i^k) \le L_i(\lambda^k, \xi), \ \forall \xi \,. \tag{2.56}$$

Since the inequality (2.56) holds for all feasible ξ, it also holds for x^{k-1}

$$L_i(\lambda^k, x_i^k) \le L_i(\lambda^k, x_i^{k-1}) \,, \tag{2.57}$$

Since subproblems, other than i, are not optimized, the following equality holds

$$L_{-i}(\lambda^k, x_{-i}^k) = L_{-i}(\lambda^k, x_{-i}^{k-1}) \,, \tag{2.58}$$

where $x_{-i}^{k-1} = x_j^{k-1}$, $j = 1, \ldots, N_s$, $j \neq i$. Given that the problem is separable, and the Lagrangian is additive,

$$L(\lambda^k, x^k) \le L(\lambda^k, x^{k-1}) \,, \tag{2.59}$$

where $x^k = (x_1^{k-1}, \ldots, x_i^k, \ldots x_{Ns}^{k-1})$.

If the inequality (2.59) is strict, then the Corollary is proved. If the inequality (2.59) holds as an equality, then the method proceeds by optimizing the next subproblem until the inequality (2.59) holds as a strict inequality. If, nevertheless, after solving all the subproblems, the surrogate optimality condition is

satisfied as an equality, this means that a surrogate dual value equals to a dual value, and a surrogate subgradient direction equals to a subgradient direction. This can happen if λ^{k-1} and λ^k belong to the same facet of the dual function, and subgradient directions at iterations k and k-1 are equal. At this point, the rest of the proof is identical to the results proved in Theorem 2.2 and Proposition 2.3. □

2.2.3. Convergence Rate of the Surrogate Lagrangian Relaxation Method

Following the general framework of standard subgradient methods [8], it is proved in Proposition 2.2.7 that when λ^k are not too close to λ^*, convergence rate is linear assuming that stepsizes c^k are sufficiently small.

Proposition 2.2.7. Under the Assumption 2.1.1 [8], the new method converges with a linear rate for sufficiently small stepsizes c^k, assuming there exists a scalar μ>0 that satisfies

$$q^* - \tilde{L}(\lambda^k, x^k) \geq \mu \left\| \lambda^* - \lambda^k \right\|^2, \; k = 0,1,\ldots, \tag{2.60}$$

and stepsizes c^k satisfy

$$0 < c^k < \frac{1}{2\mu}, \; k = 0,1,\ldots, \tag{2.61}$$

and

$$\frac{c^k \left\| \tilde{g}(x^k) \right\|^2}{\mu} < \left\| \lambda^* - \lambda^k \right\|^2, \; k = 0,1,\ldots. \tag{2.62}$$

Proof From the inequalities (2.33) and (2.60), it follows that

$$\left\| \lambda^* - \lambda^{k+1} \right\|^2 \leq \left\| \lambda^* - \lambda^k \right\|^2 (1 - 2c^k \mu) + (c^k)^2 \left\| \tilde{g}(x^k) \right\|^2. \tag{2.63}$$

Dividing both sides of (2.63) by $\left\| \lambda^* - \lambda^k \right\|^2$ yields

$$(\beta^k)^2 \equiv \frac{\left\|\lambda^* - \lambda^{k+1}\right\|^2}{\left\|\lambda^* - \lambda^k\right\|^2} \le (1 - 2c^k\mu) + \frac{(c^k)^2\left\|\tilde{g}(x^k)\right\|^2}{\left\|\lambda^* - \lambda^k\right\|^2}. \tag{2.64}$$

Intuitively, given that stepsizes are sufficiently small, and multipliers are sufficiently far from λ^*, the last term is negligibly small, and the convergence rate is linear with $\beta^k \approx \sqrt{(2.1\text{-}2c^k\mu)} < 1$. To determine the neighborhood of λ^*, to which the linear convergence can be guaranteed, the right-hand side of (2.64) should be less than 1, that is

$$(1 - 2c^k\mu) + \frac{(c^k)^2\left\|\tilde{g}(x^k)\right\|^2}{\left\|\lambda^* - \lambda^k\right\|^2} < 1. \tag{2.65}$$

Under the condition (2.62), the inequality (2.65) holds. □

Remark 2.2.8. Assumptions (2.60) and (2.62) imply the following assumption on stepsizes

$$c^k < \frac{q^* - \tilde{L}(\lambda^k, x^k)}{\left\|\tilde{g}(x^k)\right\|^2}. \tag{2.66}$$

The assumption (2.66) is the condition on the stepsizes (2.13) used in the convergence proof of the surrogate subgradient method [15]. As stated earlier, under the condition (2.66), the lower bound of the surrogate dual is preserved per (2.16), thereby implying that the left-hand side of (2.60) is positive, and μ>0 that satisfies (2.60) exists. In other words, under (2.62), assumptions (2.60) and (2.66) are equivalent. □

2.2.4. Practical Implementation of the Surrogate Lagrangian Relaxation Method

This subsection discusses practical implementation aspects of our method. A constructive rule for setting parameters α_k is developed in Proposition 2.8 and proved to satisfy conditions (2.21a) and (2.21b) required for convergence to λ^* without requiring q^*. Lastly, an algorithm of the method is presented.

Proposition 2.2.9 The stepsize-setting parameters α_k can be updated as follows to ensure that the multipliers converge to λ^*:

$$\alpha_k = 1 - \frac{1}{Mk^p},\ p = 1 - \frac{1}{k^r},\ M \geq 1,\ 0 < r < 1,\ k = 2,3,\ldots . \tag{2.67}$$

Proof Step 1: To show that stepsizes (2.22) converge to zero, it is sufficient to show that the following product converges to zero

$$\prod_{i=1}^{k} \alpha_i = \prod_{i=1}^{k} \left(1 - \frac{1}{Mi^p} \right). \tag{2.68}$$

For the ease of the proof, convergence of (2.68) to zero will be established by proving that the natural logarithm of (2.68) converges to -∞. After taking the logarithm of the product (2.68), it becomes the sum of the logarithms

$$\log\left(\prod_{i=1}^{k} \alpha_i \right) = \sum_{i=1}^{k} \log(\alpha_i) = \sum_{i=1}^{k} \log\left(1 - \frac{1}{Mi^p} \right). \tag{2.69}$$

Indeed, as $i \to \infty$, the first term of the Taylor series expansion of $\log\left(1 - \frac{1}{Mi^p} \right)$ is $-\frac{1}{Mi}$. Therefore, the sum (2.69) converges to -∞, and stepsizes (2.22) converge to zero.

Step 2: To show that condition (2.21b) of Theorem 2.1 holds, given (2.67), condition (2.21b) can be rewritten as

$$\frac{1 - \alpha_k}{c^k} \sim \frac{\frac{1}{Mk^p}}{\prod_{i=1}^{k} \left(1 - \frac{1}{Mi^p} \right)}. \tag{2.70}$$

As before, it will be shown that that the logarithm of (2.70) approaches -∞. Consider the logarithm of the right-hand side of (2.70)

$$\log\left(\frac{1}{Mk^{p}}\right)-\sum_{i=1}^{k}\log\left(1-\frac{1}{Mi^{p}}\right). \tag{2.71}$$

To prove the asymptotical condition (2.21b), it is sufficient to demonstrate that

$$\log\left(1-\frac{1}{Mk^{p}}\right)=o\left(\frac{1}{k}\log\left(\frac{1}{Mk^{p}}\right)\right). \tag{2.72}$$

Given that $p \to 1$ as $k \to \infty$, the following relation holds:

$$\lim_{k\to\infty}\frac{\log\left(1-\frac{1}{Mk^{p}}\right)}{\frac{1}{k}\log\left(\frac{1}{Mk^{p}}\right)}=\lim_{k\to\infty}\frac{\log\left(1-\frac{1}{Mk}\right)}{\frac{1}{k}\log\left(\frac{1}{Mk}\right)}. \tag{2.73}$$

Using the L'Hopital's rule leads to

$$\lim_{k\to\infty}\frac{\log\left(1-\frac{1}{Mk}\right)}{\frac{1}{k}\log\left(\frac{1}{Mk}\right)}=\lim_{k\to\infty}\frac{\frac{d}{dk}\log\left(1-\frac{1}{Mk}\right)}{\frac{d}{dk}\left(\frac{1}{k}\log\left(\frac{1}{Mk}\right)\right)}=\lim_{k\to\infty}\frac{k}{(1-Mk)(1-\log(Mk))}. \tag{2.74}$$

Applying L'Hopital's rule one more time yields

$$\lim_{k\to\infty}\frac{k}{(1-Mk)(1-\log(Mk))}=\lim_{k\to\infty}\frac{1}{k^{-1}-M\log(Mk)}=-\lim_{k\to\infty}\frac{1}{M\log(Mk)}=0. \tag{2.75}$$

As proved in the steps above, $c^k \to 0$, and the condition (2.21b) holds. Therefore, convergence of multipliers to λ^* is proved. □

The entire algorithm can be summarized in the following steps:

Step 0: Initialize multipliers λ^0, obtain x^0 by optimizing the relaxed problem, estimate $\hat{q}$ of q^* by using best heuristics available for a particular problem, initialize c^0 according to

$$c^0 = \frac{\hat{q} - \tilde{L}(\lambda^0, x^0)}{\left\| \tilde{g}(x^0) \right\|^2}. \tag{2.76}$$

Step 1: Update α_k, for example, by using (2.67). For given values (α_k, x^k), update stepsizes c^k according to (2.20). For given values (x^k, λ^k, c^k), update multipliers according to (2.14)-(2.15) to obtain λ^{k+1}.

Step 2: For the given λ^{k+1}, minimize the Lagrangian function until the surrogate optimality condition (2.12) is satisfied. As a special case, for separable problems, it is sufficient to optimize just one subproblem (Corollary 2.5).

Step 3: Check stopping criteria: CPU time, number of iterations, surrogate subgradient norm, distance between multipliers, etc. If stopping criteria are satisfied, then go to Step 4. Otherwise, go to Step 1.

Step 4: Obtain feasible solutions. Problem-specific heuristics may be used to obtain feasible costs while a dual value provides a lower bound on the optimal cost. A duality gap can then be calculated by using the best available feasible cost and the largest available dual value.

As proved before, at convergence of multipliers, a surrogate dual value converges to a dual value. If the algorithm is terminated before convergence, a dual value can be obtained by fully optimizing the relaxed problem. In Section 2.3, it will be demonstrated that owing to reduced computational requirements, the new method can obtain a better dual value, a better feasible cost and a lower duality gap as compared to other methods.

2.3. Numerical Testing

The purpose of this section is to compare the surrogate Lagrangian relaxation method with other methods that are used for optimizing non-smooth dual function such as the subgradient-level method and the incremental subgradient method. In Example 2.3.1, a small nonlinear (quadratic) integer problem is considered to demonstrate that, surrogate subgradient directions frequently form small acute angles with

directions toward the optimal multipliers, thereby alleviating the zigzagging issues that often accompany the subgradient method. In Example 2.3.2, linear integer generalized assignment problems are considered to demonstrate that the new method is capable of handling large separable optimization problems. It is then demonstrated that when simple heuristics are used to adjust relaxed problem solutions to obtain feasible costs, the method is capable of reducing the duality gap as compared to other methods such as the incremental subgradient method. In Example 2.3.3, nonlinear integer quadratic assignment problem is considered to demonstrate the quality of the method for optimizing non-separable non-smooth dual problems, and the method is compared with the subgradient-level method. The new method is implemented using CPLEX 12.2 on Intel® Xeon® CPU E5620 (2.12M Cache, 5.86 GT/s Intel® QPI) @ 2.40GHz (2.2 processors) and 36.00 GB of RAM.

Example 2.3.1. A Nonlinear Integer Problem Consider the following nonlinear integer optimization problem

$$\min_{\{x_1,x_2,x_3,x_4,x_5,x_6\}\in Z_+\cup\{0\}} \left\{0.5x_1^2 + 0.1x_2^2 + 0.5x_3^2 + 0.1x_4^2 + 0.5x_5^2 + 0.1x_6^2\right\} \tag{2.77}$$

$$s.t.\ \ 48 - x_1 + 0.2x_2 - x_3 + 0.2x_4 - x_5 + 0.2x_6 \le 0, \tag{2.78}$$

$$250 - 5x_1 + x_2 - 5x_3 + x_4 - 5x_5 + x_6 \le 0.$$

After constraints (2.78) are relaxed by the multipliers λ_1 and λ_2, respectively, the Lagrangian function becomes

$$\begin{aligned} L(x_1,x_2,x_3,x_4,x_5,x_6,\lambda_1,\lambda_2) &= 0.5x_1^2 + 0.1x_2^2 + 0.5x_3^2 + 0.1x_4^2 + 0.5x_5^2 + 0.1x_6^2 + \\ &\lambda_1(48 - x_1 + 0.2x_2 - x_3 + 0.2x_4 - x_5 + 0.2x_6) + \lambda_2(250 - 5x_1 + x_2 - 5x_3 + x_4 - 5x_5 + x_6). \end{aligned} \tag{2.79}$$

Given that the objective function and coupling constraints in (2.77)-(2.78) are of an additive form, the relaxed problem can be separated into six individual subproblems:

$$\min_{\{x_i\}\in Z_+\cup\{0\},\ i=1,\ldots,6} L(x_1,x_2,x_3,x_4,x_5,x_6,\lambda_1,\lambda_2) =$$
$$\min_{\{x_i\}\in Z_+\cup\{0\},\ i=1,\ldots,6} \{(0.5x_1^2 - \lambda_1 x_1 - 5\lambda_2 x_1) + (0.1x_2^2 + 0.2\lambda_1 x_2 + \lambda_2 x_2) +$$
$$(0.5x_3^2 - \lambda_1 x_3 - 5\lambda_2 x_3) + (0.1x_4^2 + 0.2\lambda_1 x_4 + \lambda_2 x_4) + (0.5x_5^2 - \lambda_1 x_5 - 5\lambda_2 x_5) +$$
$$(0.1x_6^2 + 0.2\lambda_1 x_6 + \lambda_2 x_6) + 48\lambda_1 + 250\lambda_2\}. \qquad (2.80)$$

To compare subgradient and surrogate Lagrangian relaxation methods, the stepsizing formula (2.20) is used to update the multipliers within both frameworks. The stepsize is initialized according to (2.76) by using an optimal value of the LP relaxation of (2.77)–(2.78), as an estimate of q^*. In the subgradient method, the relaxed problem (2.80) is optimized with respect to all $\{x_i\}$, $i = 1, \ldots, 6$. Since the relaxed problem (2.80) is separable, individual subproblems can be solved individually. In this example, three out of six subproblems are solved per iteration to obtain surrogate subgradient directions. The multipliers λ_1 and λ_2 are updated 18 iterations within the subgradient framework and 36 iterations within the surrogate Lagrangian relaxation framework. In both frameworks, each subproblem is solved 18 times. The trajectories of the multipliers are shown in Figs. 1 and 2.

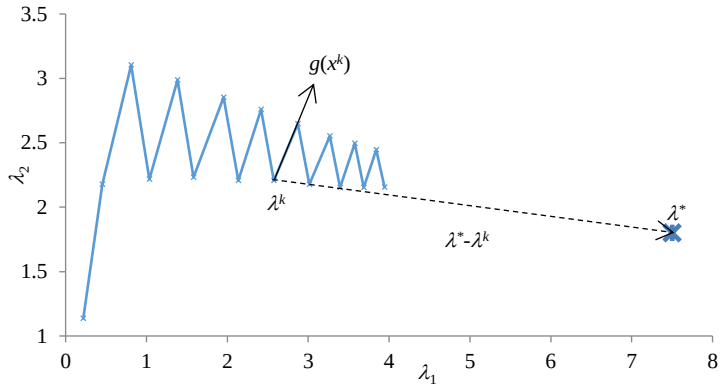


Figure 2.3.1. Trajectories of the multipliers using the subgradient method

Figure 2.3.1 demonstrates that the subgradient directions $g(x^k)$ are frequently almost perpendicular to the directions λ^*-λ^k toward λ^* (respective directions are shown in Figure 2.1 by solid and dashed arrows), and the multipliers zigzag causing slow convergence.

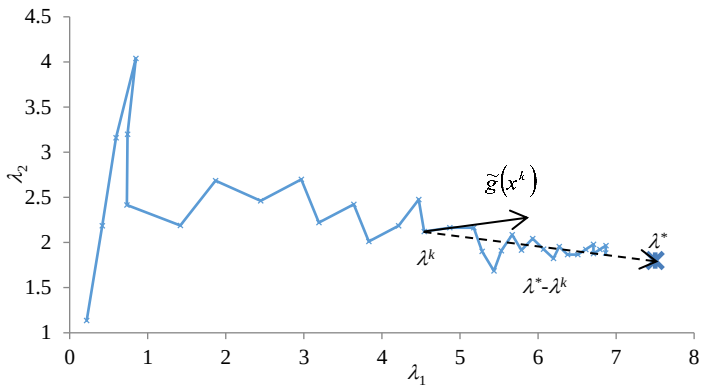


Figure 2.3.2. Trajectories of the multipliers using the surrogate subgradient method

In contrast, the surrogate directions $\tilde{g}(x^k)$ (shown by a solid arrow in Figure 2.3.1) are smoother and frequently form smaller angles with the directions λ^*-λ^k toward λ^* (shown by a dashed arrow in Figure 2.3.2), thereby alleviating zigzagging and leading to faster convergence.

Table 2.3.1. Comparison of the Subgradient and Surrogate Optimization Methods

Method	Number of iterations	CPU time (s)	Distance to the optimum
Subgradient method	18	2.75	3.574777
Surrogate optimization method	36	1.62	0.798711

Table 2.3.1 demonstrates that within the surrogate Lagrangian relaxation framework, the multipliers move closer to λ^* as compared to the multipliers updated by using subgradient directions, thereby reducing the number of iterations required for convergence. In addition, since the relaxed problem is not fully optimized in the new method, the surrogate subgradient directions are easier to obtain. This also leads to faster convergence in terms of the computation time.

Example 2.3.2. Generalized Assignment Problems. In generalized assignment problems, the total cost for assigning a given set of jobs to available machines is minimized. Each job is assigned to one machine, and the total processing time for all jobs assigned to a machine should not exceed the machine's time available. Mathematically, the generalized assignment problem is formulated in the following way:

$$\min_{x_{i,j}} \sum_{i=1}^{I}\sum_{j=1}^{J} g_{i,j}x_{i,j}\ ,\ \ x_{i,j} \in \{0,1\},\ g_{i,j} \geq 0,\ a_{i,j} \geq 0,\ b_j \geq 0\ , \tag{2.81}$$

$$s.t.\ \ \sum_{i=1}^{I} a_{i,j}x_{i,j} \leq b_j,\ j = 1,...,J\ , \tag{2.82}$$

$$\sum_{j=1}^{J} x_{i,j} = 1, i = 1,...,I\ , \tag{2.83}$$

where I is the number of jobs and J is the number of machines, $a_{i,j}$ is time required by machine j to perform job i and $g_{i,j}$ is cost for assigning job i to machine j. Capacity constraints (2.82) ensure that the total amount of time, required by the jobs to be performed on a given machine, does not exceed the machine j's time available b_j. Constraints (2.83) ensure that each job is to be performed on one and one machine only. For more details, refer to [24-31].

Since the objective function of (2.81) and constraints (2.82)-(2.83) are of an additive form, after relaxing constraints (2.83) by introducing the Lagrange multipliers, the problem is formulated in a separable form

$$\begin{aligned} &q(\lambda) = \min_{x_{i,j}} \sum_{j=1}^{I}\sum_{i=1}^{J}\left(g_{i,j} + \lambda_i\right)x_{i,j} - \sum_{i=1}^{J}\lambda_i\ ,\ \ s.t.\ \ \sum_{i=1}^{I} a_{i,j}x_{i,j} \leq b_j,\ j = 1,\ ...,\ J\ , \\ &x_{i,j} \in \{0,1\},\ g_{i,j} \geq 0,\ a_{i,j} \geq 0,\ b_j \geq 0\ . \end{aligned} \tag{2.84}$$

As proved in Corollary 2.2.6, optimization with respect to only one subproblem is sufficient to satisfy the surrogate optimality condition

$$\sum_{i=1}^{I}\left(\sum_{j=1}^{J} g_{i,j} x_{i,j}^{k+1} + \lambda_i^{k+1}\left(\sum_{j=1}^{J} x_{i,j}^{k+1} - 1\right)\right) < \sum_{i=1}^{I}\left(\sum_{j=1}^{J} g_{i,j} x_{i,j}^{k} + \lambda_i^{k+1}\left(\sum_{j=1}^{J} x_{i,j}^{k} - 1\right)\right). \tag{2.85}$$

As discussed earlier, the accompanying computational effort is approximately $1/J$ per subproblem compared to the effort required to fully optimize the relaxed problem and obtain subgradient directions.

Comparison to Standard Methods for Non-Smooth Optimization To demonstrate the quality of the surrogate Lagrangian relaxation method, it is compared to existing methods available for optimizing non-smooth dual functions, such as the simple subgradient method, the simple subgradient-level method, and the incremental subgradient method. The comparison to the last two methods is especially important, since they do not require q^* for convergence to λ^*.

The Simple Subgradient Method In the method [22], the relaxed problem (2.84) is fully optimized, and stepsizes are updated according to the following relation

$$0 < c^k < \alpha \frac{UB - q(\lambda^k)}{\left\| g(x^k) \right\|^2},\ 0 < \alpha < 2, \tag{2.86}$$

where *UB* is the best feasible cost available at iteration *k*.

The Simple Subgradient-Level Method In the method [7], the relaxed problem (2.84) is fully optimized, and stepsizes are updated according to the following relation

$$0 < c^k < \alpha \frac{q^{lev} + \delta_k - q(\lambda^k)}{\left\| g(x^k) \right\|^2},\ 0 < \alpha < 2. \tag{2.87}$$

The Incremental Subgradient Method In the method [8], each subproblem is solved to optimality. After each problem is optimized, multipliers are updated and stepsizes are updated, similarly to the subgradient-level method, according to

$$0 < c^k < \alpha \frac{q^{lev} + \delta_k - q(\lambda^k)}{n\left\|g(x^k)\right\|^2},\ 0 < \alpha < 2, \tag{2.88}$$

where n is the number of subproblems, q^{lev} is the best dual value obtained up until iteration k, and δ_k is a parameter that decreases by a factor of 2 every time a significant oscillation of multipliers is detected, that is when multipliers "travel" a distance exceeding a predetermined value B

$$\sigma_k > B, \tag{2.89}$$

where

$$\sigma_k = \sigma_{k-1} + \left\|\lambda^k - \lambda^{k-1}\right\|. \tag{2.90}$$

Once significant oscillations are detected, and condition (2.89) is satisfied, σ_k is reset to 0. For more information, refer to [7, 8].

For a fair comparison of the methods, each subproblem is solved exactly once per iteration. For example, within the subgradient method, minimization of the relaxed problem counts as one iteration. In the incremental subgradient method, one iteration is complete once each subproblem is solved exactly once. In the new method, 10, 2 and 1 subproblems were chosen to be solved for instances GAPd801600, GAPd201600, and GAPd15900[5], respectively. Therefore, for these instances the number of sub-iterations is 8, 10, and 15, respectively.

[5] For the GAP15900 instance, the implementation of the new method may resemble that of the interleaved method [14] since only one subproblem is optimized at a time. The important difference between the new method and the interleaved method is the stepsizing formula.

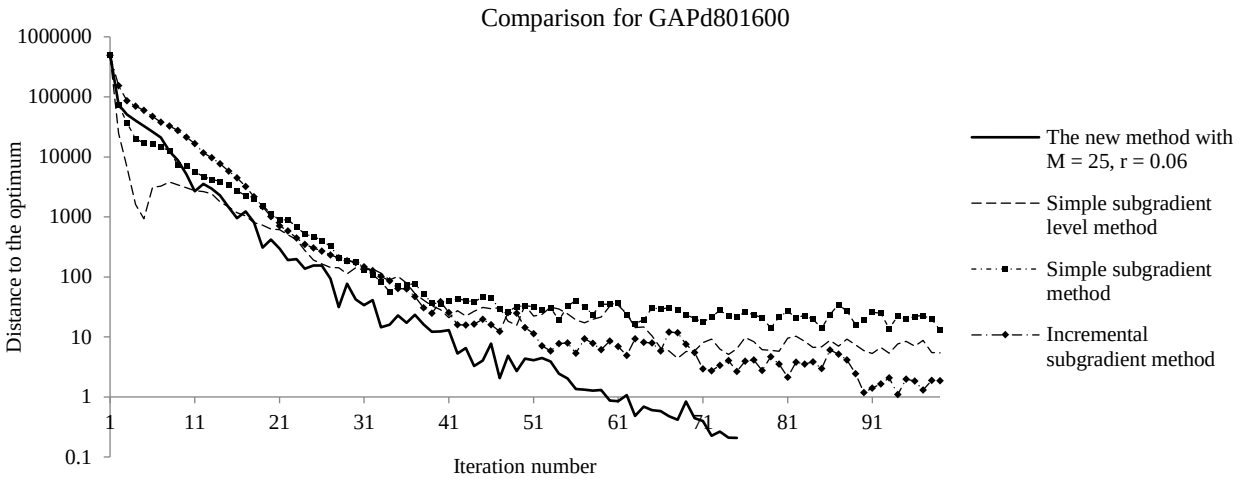


Figure 2.3.3. Comparison of the surrogate Lagrangian relaxation method with parameters $M = 25$ and $r = 0.06$ against: 1) the simple subgradient method with $\alpha = 1$; 2) the subgradient-level method with parameters $\delta_0 = 100000$ and $B = 1000$; 3) the incremental subgradient method with parameters $\delta_0 = 100000$ and $B = 1000$

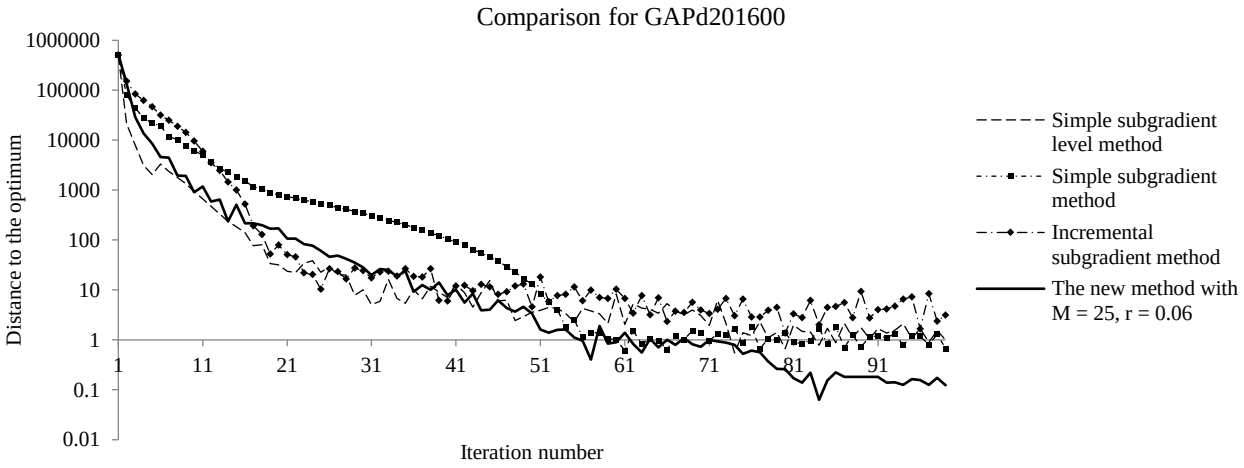


Figure 2.3.4. Comparison of the surrogate Lagrangian relaxation method with parameters $M = 25$ and $r = 0.06$ against: 1) the simple subgradient method with $\alpha = 1$; 2) the subgradient-level method with $\delta_0 = 100000$ and $B = 500$; 3) the incremental subgradient method with parameters $\delta_0 = 100000$ and $B = 1000$

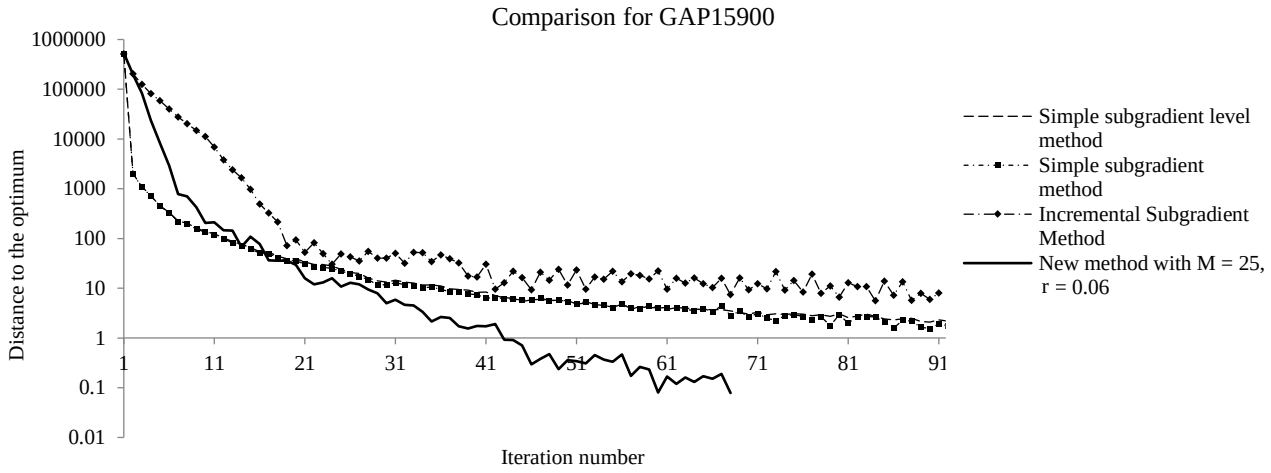


Figure 2.3.5. Comparison of the surrogate Lagrangian relaxation method with parameters $M = 25$ and $r = 0.06$ against: 1) the simple subgradient method with $\alpha=1$; 2) the subgradient-level method with parameters $\delta_0 = 50000$ and $B = 750$; 3) the incremental subgradient method with parameters $\delta_0 = 50000$ and $B = 750$

Figures 2.3.3-2.3.5 demonstrate performance of the surrogate Lagrangian relaxation method, as compared to subgradient methods.[6] Numerical results indicate that within the incremental subgradient framework, multipliers approach λ^* slowly, since stepsizes decrease to zero slowly. This happens because as stepsizes decrease, it takes more iterations for multipliers to "travel" distance B. This leads to slow convergence when multipliers move closer to λ^*. For a similar reason, convergence of the subgradient-level method can be slower as compared to the surrogate Lagrangian relaxation method. Since duality gaps of generalized assignment problems are typically small, a feasible cost can provide a reasonably good approximation of q^* within the simple subgradient method. However, convergence to λ^* does not occur. The following figure demonstrates a comparison of duality gaps obtained by the new method and the incremental subgradient method for the GAP d201600 instance.

[6] Performance of all methods in Figures 2.3-2.5 is tested by comparing distances to multipliers obtained by a subgradient method with non-summable stepsizes [22] after sufficiently many iterations (>20000).

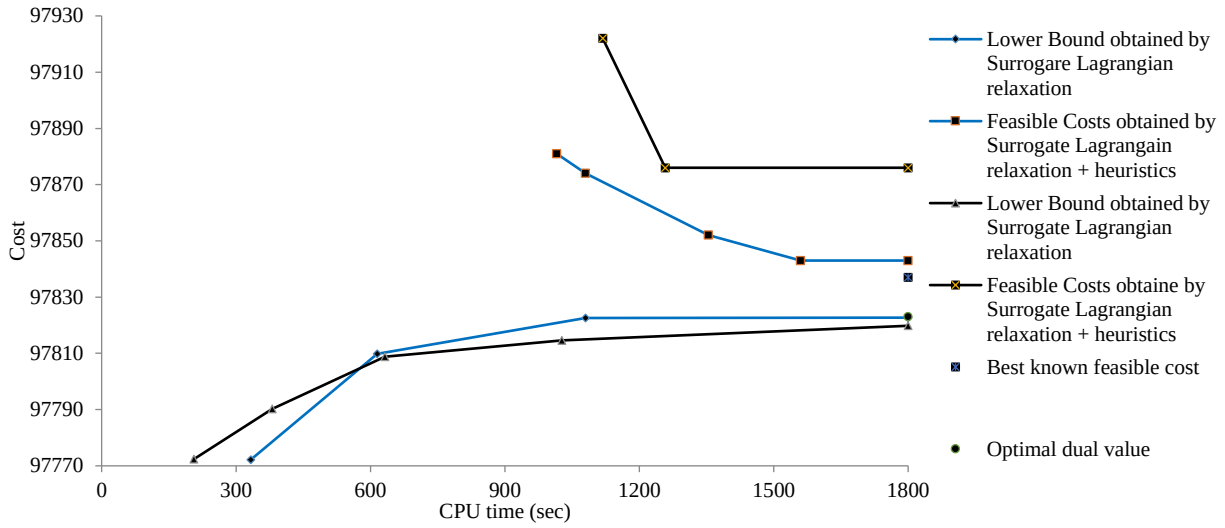


Figure 2.3.6. Comparison of the surrogate Lagrangian relaxation method with parameters M = 20 and r = 0.1 against the incremental subgradient method with parameters δ_0 = 500 and B = 15 for solving the GAPd201600 instance

As demonstrated in Figure 2.3.6, owing to the reduced computational effort, in the new method the dual value increases faster, and with the help of heuristics, feasible costs obtained are better, as compared to the incremental subgradient method. As a result, the duality gap obtained by using the new method is smaller than the gap obtained by using the incremental subgradient method.

Example 2.3.3. Quadratic Assignment Problems The objective of the Quadratic Assignment Problem (QAP) of order n is to find the best allocation of n facilities to n locations. Formulated in 1957 by [32], the problem has been applied to planning of buildings in university campuses [33], arrangement of departments in hospitals [34], scheduling parallel production lines [35], and ranking of archeological data [36]. It has also been shown that QAPs can be applied to the field ergonomics to solve the typewriter keyboard design problem [37]. Mathematically, the quadratic assignment problem can be formulated as an integer programming problem:

$$\min_{x_{i,j},x_{h,l}} \sum_{i,j=1}^{n} \sum_{h,l=1}^{n} d_{i,h} f_{j,l} x_{i,j} x_{h,l} \ , x_{i,j} \in \{0,1\},\ d_{i,h} \geq 0,\ f_{j,l} \geq 0\,, \tag{2.91}$$

$$s.t. \quad \sum_{i=1}^{n} x_{i,j} = 1,\ j = 1, ..., n\ , \tag{2.92}$$

$$\sum_{j=1}^{n} x_{i,j} = 1, i = 1, ..., n\ , \tag{2.93}$$

where n is the number of facilities and locations, $d_{i,h}$ is the distance between location i and location h, $f_{j,l}$ is the weight/flow between facility j and facility l (the net transfer of goods/supplies from facility j to l). Intuitively, two facilities with high flow should be built close to each other. Binary decision variables $x_{i,j}$ corresponds to facility i being placed in location j iff $x_{i,j} = 1$. Assignment constraints (2.92) and (2.93) ensure that one and one facility only can be assigned to a specific location.

The problem formulation (2.91)-(2.93) is non-separable because of the cross-product of decision variables in the objective function of (2.91). For a fair comparison of the methods, after the problem is linearized, branch-and-cut is used to obtain approximate solutions of the relaxed problem for the surrogate Lagrangian relaxation method and exact solutions of the relaxed problem for the subgradient-level method.

After relaxing constraints (2.93) by introducing Lagrange multipliers, the relaxed problem becomes:

$$\min_{x_{i,j}, x_{h,l}} \left\{ \sum_{i,j=1}^{n} \sum_{h,l=1}^{n} d_{i,h} f_{j,l} x_{i,j} x_{h,l} + \sum_{i=1}^{n} \lambda_i \left(\sum_{j=1}^{n} x_{i,j} - 1 \right) \right\},\ s.t.\ x_{i,j} \in \{0,1\} \text{ and (2.92).} \tag{2.94}$$

Since decision variables $x_{i,j}$ and $x_{h,l}$ are binary, feasible region for the product $x_{i,j} \cdot x_{h,l}$ consists of the four points: (0,0), (0,1), (1,0), and (1,1). Moreover, the product takes on the value of 1 if and only if both decision variables equal to 1. Based on this observation, the relaxed problem can be equivalently rewritten in a linear form as:

$$\min_{x_{i,j}, x_{h,l}, F_{i,j,h,l}} \left\{ \sum_{i,j=1}^{n} \sum_{h,l=1}^{n} d_{i,h} f_{j,l} F_{i,j,h,l} + \sum_{i=1}^{n} \lambda_i \left(\sum_{j=1}^{n} x_{i,j} - 1 \right) \right\},\ s.t.\ x_{i,j} \in \{0,1\},\ F_{i,j,h,l} \geq x_{i,j} + x_{h,l} - 1, \text{ and (2.92).} \tag{2.95}$$

To obtain subgradient and surrogate multiplier-updating directions, the linear problems formulation (2.95) is optimized by using branch-and-cut. In the subgradient method, the relaxed problem (2.95) is

fully optimized. In the surrogate Lagrangian relaxation method, the relaxed problem (2.95) is optimized approximately subject to the surrogate optimality condition:

$$\sum_{i,j=1}^{n}\sum_{h,l=1}^{n} d_{i,h} f_{j,l} F_{i,j,h,l}^{k+1} + \sum_{i=1}^{n} \lambda_i \left(\sum_{j=1}^{n} x_{i,j}^{k+1} - 1 \right) < \sum_{i,j=1}^{n}\sum_{h,l=1}^{n} d_{i,h} f_{j,l} F_{i,j,h,l}^{k} + \sum_{i=1}^{n} \lambda_i \left(\sum_{j=1}^{n} x_{i,j}^{k} - 1 \right). \quad (2.96)$$

In practice, the inequality (2.96) can be operationalized within the commercial solver CPLEX. Given initial values $x_{i,j}^{k}$ and $F_{i,j,h,l}^{k}$ as a warm MIP start, once branch-and-cut finds one solution that is strictly better than $x_{i,j}^{k}$ and $F_{i,j,h,l}^{k}$, optimization stops, the surrogate optimality condition is satisfied by definition, and surrogate multiplier-updating directions are computed by using $x_{i,j}^{k+1}$.

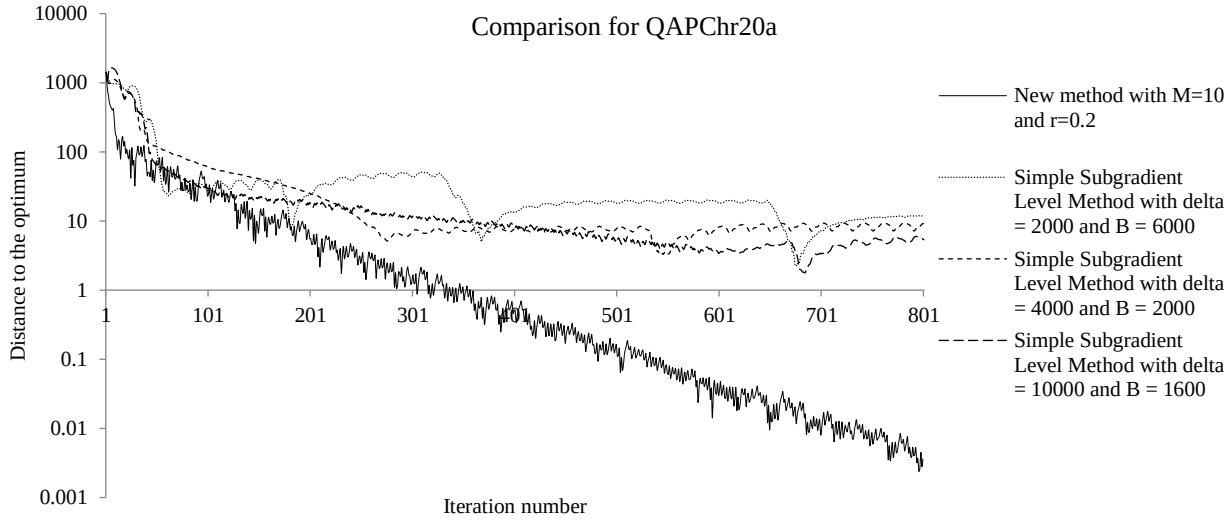


Figure 2.3.7. Comparison of the surrogate Lagrangian relaxation method with parameters M = 10 and r = 0.2 against the subgradient-level method with parameters: 1) δ_0 = 2000 and B = 6000; 2) δ_0 = 4000 and B = 2000, and 3) δ_0 = 10000 and B = 1600 for the QAPChr20a instance [38].

Figure 2.3.7 demonstrates performance comparison of surrogate Lagrangian relaxation and the subgradient-level method. In the surrogate Lagrangian method, multipliers converge to the optimum with tolerance 0.001 within 800 iterations. In the subgradient-level method, parameters δ_k and B can be chosen to ensure fast convergence within first 200 iterations. However, as stepsizes decrease, it can take many

iterations for multipliers to "travel" distance *B*, thereby leading to slow convergence as multipliers approach λ^*.

Example 2.3.4: Unit Commitment and Economic Dispatch problem with Combined Cycle units. The objective of the unit commitment and economic dispatch problem with conventional and combined cycle units is to commit units to satisfy the total system demand and reserve requirements by minimizing the total bid cost, consisting of the total cost on energy, spinning reserve and start-up cost while following transitions among combined cycle states. Transmission capacity, ramp-rate, and minimum up and down time constraints will not be considered for simplicity. Consider a market with *I* supply bids. Each bid corresponds to either a conventional unit, or to a combustion/steam turbine generator that comprises a combined cycle unit. Each bid indexed by *i* consists of energy and spinning reserve bidding prices c_i^E and c_i^S, startup costs S_i, maximum and minimum generation levels $p^E_{i,max}$ and $p^E_{i,min}$ and maximum levels for spinning reserve $p^S_{i,max}$. Energy bids can be modeled for each hour by up to ten blocks of energy with monotonically non-decreasing prices.

a. Generation Capacity Constraints

The energy and spinning reserve continuous decision variables corresponding to each bid *i* are denoted by $p_i^E(t)$ and $p_i^S(t)$, respectively. Energy and spinning reserve allocation status binary decision variables are denoted by $x_i^E(t)$ and $x_i^S(t)$. The relationship between these decision variables can be summarized as the following individual unit capacity constraints:

$$x_i^E(t)\,p_{i\min} \le p_i^E(t) \le x_i^E(t)\,p_{i\max}, \tag{2.97}$$

$$0 \le p_i^S(t) \le x_i^S(t)\,p_{i\max}^S, \tag{2.98}$$

$$p_i^E(t) + p_i^S(t) \le p_{i\max}. \tag{2.99}$$

The startup cost $S_i(t)$ is incurred if and only if the unit *i* has been turned "on" from an "off" state at hour *t*

$$u_i(t) \geq x_i^E(t) - x_i^E(t-1) . \tag{2.100}$$

b. Energy Demand and Reserve Requirement Constraints

The total power generated by units satisfying (2.97)-(2.99) should be equal to the system demand and satisfy total spinning reserve requirements $P^{DS}(t)$ at each hour t:

$$\sum_{i=1}^{I} p_i^E(t) = P^{DE}(t) , \tag{2.101}$$

$$\sum_{i=1}^{I} p_i^S(t) \geq P^{DS}(t) . \tag{2.102}$$

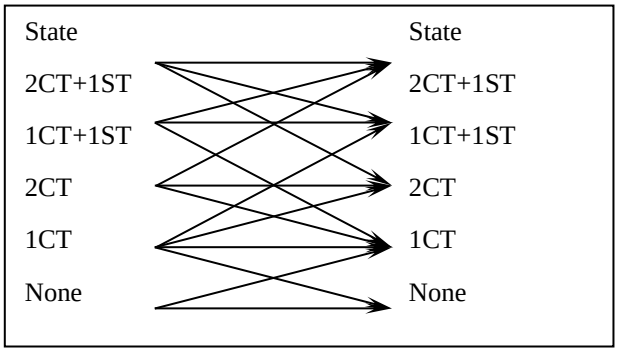


Figure 2.3.8. Transitions among the states in a combined cycle unit

c. Combined Cycle Transitions

Combined cycle unit can operate at multiple configurations of CTs and STs. However, transitions among configurations may be constrained. For example, steam turbines (ST) cannot be turned on if there is not enough heat from combustion turbines (CTs). Transition rules [1], [2] for a configuration 2CT+1ST are summarized in Figure 2.3.8.

Therefore, to model CC units, in addition to constraints (2.97)-(2.100), constraints that capture such transitions are required. Transitions can be modeled by using logical operators such as AND, OR, and $\Rightarrow$ (logical implication). For example, a transition from 1CT to 2CT can be modeled as

$$\begin{aligned}&\left(x_{CT1}(t\text{-}1)=1\ AND\ x_{CT2}(t\text{-}1)=0\ AND\ x_{ST}(t\text{-}1)=0\right)\ OR\\&\left(x_{CT2}(t\text{-}1)=1\ AND\ x_{CT1}(t\text{-}1)=0\ AND\ x_{ST}(t\text{-}1)=0\right)\Rightarrow\\&\left(x_{CT2}(t)=1\ AND\ x_{CT1}(t)=1\ AND\ x_{ST}\ (t)=0\right)\end{aligned} \tag{2.103}$$

Other transitions can be modeled in a similar fashion, and for brevity of explanation are not shown. Such logical expressions can be linearized using the following relations:

$$a_1 = 1 \text{ AND } a_2 = 0 \Leftrightarrow a_1 + 1 - a_2 = 2, \tag{2.104}$$

$$a = 0 \Rightarrow b = 0 \Leftrightarrow -M \cdot a \le b \le M \cdot a, \tag{2.105}$$

$$\begin{aligned}&a_1 \le b_1\ OR\ a_2 \le b_2 \Leftrightarrow\\&a_1 \le b_1 + M \cdot z_1;\ a_2 \le b_2 + M \cdot z_2;\ z_1 + z_2 = 1.\end{aligned} \tag{2.106}$$

where z_1 and z_2 are binary decision variables.

In addition, the output of a ST is typically no more than 50% of the total CT output within one CC unit. For example, for a configuration 2CT+1ST:

$$p_{ST}(t) \le \tfrac{1}{2}\left(p_{CT1}(t) + p_{CT2}(t)\right). \tag{2.107}$$

The objective of the UCED problem with conventional and CC unit is to minimize the total bid cost:

$$\sum_{i=1}^{I}\left(\sum_{t=1}^{T} c_i\left(p_i(t), t\right) + \sum_{t=1}^{T} S_i u_i(t)\right) \tag{2.108}$$

while satisfying all constraints (2.97)-(2.107).

The combination of SLR and B&C is implemented by using CPLEX 12.5.1 on Intel® Xeon® CPU E5620 (2.12M Cache, 5.86 GT/s Intel® QPI) @ 2.40GHz (2.2 processors) and 36.00 GB of RAM. Small- and medium-size instances are considered first to demonstrate that even in the presence of a few CC units, performance of B&C is poor as compared to performance of the combination of SLR and B&C. A large-scale UCED problem with 10 CC and 300 conventional units is then considered to demonstrate that the method is capable of efficiently solving large instances, and the new method is compared with branch-and-cut.

Table 2.3.2. Bid Data for Example 2.3.4

Unit		***Bid price (\$/MW)***	***Start up cost (\$)***	***Pmax (MW)***	***Pmin (MW)***
CC unit	CT1	30	1200	455	100
	CT2	34	1150	350	60
	ST	35	1100	300	50
Conventional units	1	37	1000	200	30
	2	40	350	350	40
	3	42	350	320	40
	4	45	300	240	30
	5	47	300	200	30
	6	55	180	190	20
	7	57	180	180	20
	8	58	175	170	20
	9	59	160	160	20
	10	60	250	150	20
	11	62	200	140	20
	12	63	150	130	20

Small and Medium-Sized UCED problems with Combined Cycle Units. UCED problems with several conventional and several CC units are considered. For simplicity, only energy product is considered. The bid data for an instance with 1 combined cycle and 12 conventional units is based on the data used in [13] and are shown in Table 2.3.2. System demands for each hour are shown in the following Table 2.3.3.

Table 2.3.3. Demand Data for Example 2.3.4

Hour	1	2	3	4	5	6	7	8	9	10	11	12
Demand	1667	1700	1713	1687	1767	1800	1787	1827	1900	1933	2333	2467
Hour	13	14	15	16	17	18	19	20	21	22	23	24
Demand	2533	2600	2800	2833	2933	3000	2733	2567	2267	1800	1533	1567

Other instances are created by sequentially adding CT2 and ST to each of the conventional units starting from unit 1, and the results are shown in Table 2.3.4.

Table 2.3.4. Results for Small and Medium-Sized UCED problems of Example 2.3.4

Number of CC units, number of conventional units	**Methods**					
	B&C			***SLR + B&C***		
	***Cost* ($)**	***CPU time* (*s*)**	***MIP Gap* (*%*)**	***Cost* ($)**	***CPU time* (*s*)**	***Duality Gap* (*%*)**
1, 12	1,956,032	0.51	0	1,956,032	12	0.081
2, 11	1,818,190	0.41	0	1,818,190	28	0.042
3, 10	1,767,335	2.25	0	1,767,385	28	0.037
4, 9	1,759,714	600	0.37	1,746,158	71	0.068
6, 7	1,762,870	600	1.18	1,760,860	77	0.1

Table 2.3.4 demonstrates that even as the number of CC units increases to 4 CC units, performance of B&C degrades.

Large-Scale UCED problem with Combined Cycle Units

In this example, a UCED problem with 300 conventional and 10 CC units is considered. For simplicity, only energy product is considered.

Table 2.3.5. Results for Large-Scale UCED of Example 2.3.4

Method	**Feasible Cost**	**Lower Bound**	**Gap (%)**	**CPU Time (min)**
B&C	50,260,500	45,305,200	9.859	30
SLR + B&C	49,894,806	49,879,027	0.032	5

This example demonstrates that the combination of SLR and B&C not only can efficiently solve the relaxed problem thereby reducing CPU time and ensuring a good lower bound, but also can obtain a good near-optimal cost.

Example 2.3.5. Unit Commitment and Economic Dispatch with Combined Cycle Units and Transmission Capacity Constraints. In this example, to demonstrate efficiency of the new method, the Unit Commitment and Economic Dispatch problem with combined cycle units [39]-[40] and transmission capacity constraints will be considered. The Unit Commitment and Economic Dispatch problem seeks to minimize the total cost consisting of the total generation and the total start-up costs by determining which generators to commit and deciding their generation levels that satisfy generator capacity, ramp-rate and minimum up- and down-time constraints [41]-[42] and following transitions among states of combined cycle units while meeting the demand P_i^D at each node i and satisfying transmission capacity $f_{l,\max}$ in each transmission line l. The constraints are formulated as follows:

Generation Capacity Constraints: Status of each bid[7] m_i (= 1, …, M_i) at node i (= 1, ..., I) indexed by (i,m_i) is modeled by binary decision variables $x_{(i,m_i)}(t)$: $x_{(i,m_i)}(t)$ = 1 indicates that the bid was selected, and $x_{(i,m_i)}(t)$ = 0 otherwise. If the bid is selected, energy $p_{(i,m_i)}(t)$ output should satisfy minimum/maximum generation levels:

$$x_{(i,m_i)}(t)\,p_{(i,m_i)\min} \le p_{(i,m_i)}(t) \le x_{(i,m_i)}(t)\,p_{(i,m_i)\max}. \tag{2.109}$$

The startup cost $S_{(i,m_i)}(t)$ is incurred if and only if the unit i has been turned an 'on' from an 'off' state at hour t

$$S_{(i,m_i)}(t) \ge S_{(i,m_i)}\left(x_{(i,m_i)}(t) - x_{(i,m_i)}(t-1)\right). \tag{2.110}$$

Ramp-rate constraints ensure that the increase/decrease in the output of a unit does not exceed a pre-specified ramp-rate within one hour.

Minimum up- and down-time constraints ensure that a unit must be kept online/offline for a pre-specified number of hours. Formulation of ramp-rate and minimum up- and down-time constraints can be found in [41].

Transitions within Combined Cycle Units: Combined cycle units can operate at multiple configurations of combustion turbines (CTs) and steam turbines (STs). However, transitions among configurations may be constrained. For example, steam turbines cannot be turned on if there is not enough heat from combustion turbines. Transition rules [39]-[40] for a configuration with two combustion turbines and one steam turbine (2.2CT+1ST) and their linear formulation can be found in [43]-[44].

Demand Constraints: Committed generators need to satisfy energy nodal load levels $P_i^D(t)$ either locally or by transmitting power through transmission lines. The total power generated should be equal to the system demand:

[7] Each bid corresponds to either a conventional unit, or to a combustion/steam turbine generator that comprises a combined cycle unit.

$$\sum_{i=1}^{I}\sum_{m=1}^{M_i} p_{(i,m)}(t) = \sum_{i=1}^{I} P_i^D(t)\,. \tag{2.111}$$

Power Flow Constraints: The power flow $f_{(b_1,b_2)}(t)$ in a line that connects nodes b_1 and b_2 can be expressed as a linear combination of net nodal injections of energy:

$$f_{(b_1,b_2)}(t) = \sum_{i=1}^{I} a^i_{(b_1,b_2)} \cdot \left(\sum_{m=1}^{M_i} p_{(i,m)}(t) - P_i^D(t) \right). \tag{2.112}$$

Power flows in a line are essentially a linear combination of nodal injections with weights being a_l^i, referred to as 'shift factors.'

Transmission Capacity Constraints: Power flows in any line cannot exceed the transmission capacity $f_{l\max}$ which for simplicity is set to be the same for each direction

$$-f_{(b_1,b_2)\max} \le f_{(b_1,b_2)}(t) \le f_{(b_1,b_2)\max}\,. \tag{2.113}$$

Objective Function. The objective of the UCED problem with conventional and combined cycle unit is to minimize the cost consisting on the total bid and start-up costs:

$$\sum_{i=1}^{I}\sum_{m=1}^{M_i}\left(\sum_{t=1}^{T} c_{(i,m_i)}\left(p_{(i,m_i)}(t),t\right) + \sum_{t=1}^{T} S_{(i,m_i)} \right) \tag{2.114}$$

while satisfying all constraints mentioned before.

Testing IEEE 30-bus system [45]. To test the new method, consider the IEEE 30-bus system that consists of 30 buses (I = 30) and 41 transmission lines (L = 41). The original data are modified so that each bus numbered 1 through 10 has exactly one combined cycle unit (M_i = 1), and each of the buses 11 and 12 has exactly one conventional generator.

To solve the problem, only nodal demand constraints (2.111) are relaxed and the relaxed problem becomes:

$$\sum_{i=1}^{I}\sum_{m=1}^{M_i}\left(\sum_{t=1}^{T} c_{(i,m_i)}\left(p_{(i,m_i)}(t),t\right)+\sum_{t=1}^{T} S_{(i,m_i)}u_{(i,m_i)}(t)\right)+\sum_{t=1}^{T}\lambda(t)\left(\sum_{i=1}^{I}\sum_{m=1}^{M_i} p_{(i,m)}(t)-\sum_{i=1}^{I} P_i^{\,D}(t)\right), \tag{45}$$

subject to all constraints mentioned before with the exception of nodal demand constraints (2.111).

A subproblem at iteration *k* can be written as:

$$\sum_{t=1}^{T} c_{(i,m_i)}\left(p_{(i,m_i)}(t),t\right)+\sum_{t=1}^{T} S_{(i,m_i)}u_{(i,m_i)}(t)+\sum_{t=1}^{T}\lambda^k(t)\left(\sum_{i=1}^{I}\sum_{m=1}^{M_i} p_{(i,m)}(t)\right), \tag{46}$$

subject to

$$-f_{(b_1,b_2)\max}\le\sum_{\substack{j=1\\ j\neq i}}^{I} a^{j}_{(b_1,b_2)}\cdot\left(\sum_{m=1}^{M_j} p^{k-1}_{(j,m)}(t)-P_j^{\,D}(t)\right)+a^{i}_{(b_1,b_2)}\cdot\left(\sum_{m=1}^{M_i} p_{(i,m_i)}(t)-P_i^{\,D}(t)\right)\le f_{(b_1,b_2)\max}. \tag{47}$$

Performance of the new method is compared to that of branch-and-cut, and the results are demonstrated in Figure 2.3.9.

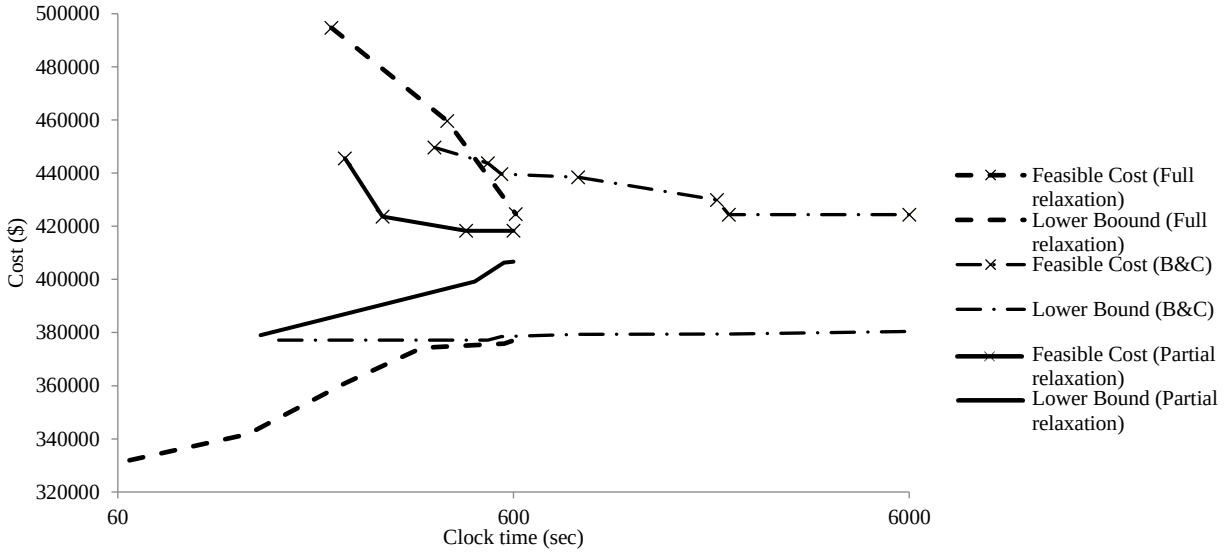


Figure 2.3.9. Comparison of the new method and branch-and-cut for the unit commitment problem

Figure 2.3.9 shows that without full decomposition, the new method obtains a good feasible solution within 10 minutes of clock time. Upon comparison with the integration of surrogate Lagrangian relaxation and branch-and-cut with full relaxation, the new method converges faster judging by the

quality of the lower bound, and the method obtains better feasible solutions. Performance of the method is also much better as compared to that of standard branch-and-cut.

2.4. Conclusions

The major breakthrough of this section is on the development of the novel Surrogate Lagrangian method and its convergence proof without requiring the optimal dual value and without fully optimizing the relaxed problem. Stepsizes that guarantee convergence without requiring the optimal dual value have been obtained. Under additional assumptions, convergence rate of the new method is proved to be linear. Also, at convergence of the multipliers, the new method generates a valid lower bound. Numerical results demonstrate that the method reduces computational requirements by reducing the effort required to obtain surrogate directions and by alleviating zigzagging of the multipliers. From the application point of view, an important extension of the method would be its combination with other methods in order to efficiently solve mixed-integer programming problems. In particular, the future work would be to prove that the method can be combined with branch-and-cut in order to efficiently solve mixed-integer linear programming problems by exploiting both separability and linearity, thereby resolving the difficulties that frequently accompany pure branch-and-cut.

References

1. Ermoliev, Y. M.: Methods for solving nonlinear extremal problems. Cybernetics 2(4), 1–17 (1966)
2. Polyak, B. T.: A general method of solving extremum problems. Soviet Mathematics Doklady 8, 593–597 (1967)
3. Polyak, B. T: Minimization of unsmooth functionals. USSR Computational Mathematics and Mathematical Physics 9(3), 14–29 (1969) (in Russian)
4. Shor, N. Z.: On the rate of convergence of the generalized gradient method. Cybernetics 4(3), 79-80 (1968)

5. Shor, N. Z.: Generalized gradient methods for non-smooth functions and their applications to mathematical programming problems. Economic and Mathematical Methods 12(2), 337–356 (1976) (in Russian)
6. Goffin, J.-L.: On the finite convergence of the relaxation method for solving systems of inequalities. Operations Research Center Report ORC 71-36, University of California at Berkeley, (1971)
7. Goffin, J.-L. and Kiwiel, K.: Convergence of a simple subgradient level method. Mathematical Programming 85(1), 207–211 (1999)
8. A. Nedic and D. P. Bertsekas: Convergence rate of incremental subgradient algorithms. In: S. Uryasev and P. M. Pardalos (eds.): Stochastic Optimization: Algorithms and Applications, Kluwer Academic Publishers, pp. 263–304 (2000)
9. Nedic, A., and Bertsekas, D.: Incremental subgradient methods for nondifferentiable optimization. SIAM Journal on Optimization 56 (1), 109–138 (2001)
10. Nesterov, Y.: Smooth minimization of non-smooth functions. Mathematical Programming 103(1), 127–152 (2005)
11. Bertsekas, D. P.: Incremental gradient, subgradient, and proximal methods for convex optimization: A survey. LIDS Technical Report no. 2848, MIT, (2010)
12. Bertsekas, D. P.: Incremental proximal methods for large scale convex optimization. Mathematical Programming 129, 163–195 (2011)
13. Lemarechal, C., Nemirovskii A.S., and Nesterov, Y. E.: New variants of bundle methods. Mathematical Programming 69, 111-147 (1995)
14. Kaskavelis, C. A. and Caramanis, M. C.: Efficient Lagrangian relaxation algorithms for industry size job-shop scheduling problems. IIE Transactions 30(11), 1085–1097 (1998)
15. Zhao, X., Luh, P. B, and Wang, J.: Surrogate gradient algorithm for Lagrangian relaxation. Journal of Optimization Theory and Applications 100 (3), 699–712 (1999)

16. Luh, P. B., Blankson, W. E., Chen, Y., Yan, J. H., Stern, G. A., Chang, S. C. and Zhao, F.: Payment cost minimization auction for the deregulated electricity markets using surrogate optimization. IEEE Transactions on Power Systems 21 (2), 568–578 (2006)
17. Sun, T., Zhao, Q. C., and Luh, P. B.: On the surrogate gradient algorithm for Lagrangian relaxation. Journal of Optimization Theory and Applications 133(3), 413–416 (2007)
18. Chang, T. S.: Comments on "Surrogate gradient algorithm for Lagrangian relaxation." Journal of Optimization Theory and Applications 137(3), 691–697 (2008)
19. Bragin, M. A., Han, X., Luh, P. B., and Yan, J. H.: Payment cost minimization using Lagrangian relaxation and modified surrogate optimization approach. Proceedings of the IEEE Power Engineering Society, General Meeting, Detroit, Michigan, (2011)
20. Bragin, M. A., Luh, P. B., Yan, J. H., Yu, N., Han, X., and Stern, G. A.: An efficient surrogate subgradient method within Lagrangian relaxation for the payment cost minimization problem. Proceedings of the IEEE Power Engineering Society, General Meeting, San Diego, California, (2012)
21. Allen, E., Nelgason, R., Kennongton, J., and Shettym B.: A generalization of Polyak's convergence result for subgradient optimization. Mathematical Programming 37(3), 309-317 (1987)
22. Bertsekas, D. P.: Nonlinear Programming. Athena Scientific, Massachusetts (2008)
23. Wah, B. W., and Chen, Y. X.: Subgoal partitioning and global search for solving temporal planning problems in mixed space. International Journal of Artificial Intelligence Tools 13(4), 767-790 (2004)
24. Chu, P. C., and Beasley, J. E.: A genetic algorithm for the generalized assignment problem. Computers and Operations Research 24(1), 17–23 (1997)
25. Yagiura, M., Yamaguchi, T., and Ibaraki, T.: A variable depth search algorithm with branching search for the generalized assignment problem. Optimization Methods and Software, 10, 419-441 (1998)
26. Yagiura, M., Ibaraki, T., and Glover, F.: A path relinking approach with ejection chains for the generalized assignment problem. European Journal of Operational Research 169(2), 548–569 (2006)

27. Avella, P., Boccia, M., and Vasilyev, I.: A computational study of exact knapsack separation for the generalized assignment problem. Computational Optimization and Applications 45(3), 543–555 (2010)
28. Posta, M., Ferland, J. A., and Michelon, P.: An exact method with variable fixing for solving the generalized assignment problem. Computational Optimization and Applications 52(3), 629-644 (2012)
29. Özbakir, L., Baykasoglu, A., and Tapkan, P.: Bees algorithm for generalized assignment problem. Applied Mathematics and Computation, 215(11), 3782–3795 (2010)
30. Asahiro, Y., Ishibashi, M., and Yamashita, M.: Independent and cooperative parallel search methods for the generalized assignment problem. Optimization Methods and Software 18(2), 129–141 (2003)
31. Laguna, M., Kelly, J. P., Gonzalez-Velarde, J. L., and Glover, F.: Tabu search for the multilevel generalized assignment problem. European Journal of Operational Research 82, 176–189 (1995)
32. Koopmans, T. C. and Beckmann, M. J.: Assignment problems and the location of economic activities. Econometrica 25(1), 53-76 (1957)
33. Dickey, J. W. and Hopkins, J. W.: Campus building arrangement using TOPAZ. Transportation Research, 6, 59–68 (1972)
34. Elshafei, A. N.: Hospital layout as a quadratic assignment problem. Operations Research Quarterly 28, 167–179 (1977)
35. Geoffrion, A. M. and Graves, G. W.: Scheduling parallel production lines with changeover costs: Practical applications of a quadratic assignment/LP approach. Operations Research 24, 595–610 (1976)
36. Krarup, J. and Pruzan, P. M.: Computer-aided layout design. Mathematical Programming Study 9, 75–94 (1978)
37. Burkard, R. E. and Offermann, J.: Entwurf von Schreibmaschinentastaturen mittels quadratischer Zuordnungsprobleme, Zeitschrift fur Operations Research 21(4), 121–132 (1977) (in German)
38. Christofides, N. and Benavent, E.: An exact algorithm for the quadratic assignment problem. Operations Research 37(5), 760-768 (1989)

39. J. Alemany, D. Moitre, H. Pinto, F. Magnago, " Short-Term Scheduling of Combined Cycle Units Using Mixed Integer Linear Programming Solution," Energy and Power Engineering, Vol. 5, pp. 161-170, 2013

40. G. Anders (Principal Investigator), "Commitment Techniques for Combined Cycle Generating Units," Kinectrics Inc, Toronto, Canada, CEATI Report No. T053700-31-3, Dec. 2005.

41. Guan, X., Luh, P. B., Yan, H. and Amalfi, J. A.: An optimization-based method for unit commitment. Int. J. Elec. Power Energy Syst. 14(1), 9–17 (1992)

42. Guan, X., Luh, P. B., Yan, H., and Rogan, P. M.: Optimization-based scheduling of hydrothermal power systems with pumped-storage units. IEEE Trans. Power Syst. **9**(2), 1023-1031 (1994)

43. Bragin, M. A., Luh, P. B., Yan, J. H., and Stern, G. A.: Surrogate Lagrangian relaxation and branch-and-cut for unit commitment with combined cycle units. In: Proceedings of the IEEE Power and Energy Society, General Meeting, National Harbor, Maryland (2014)

44. Bragin, M. A., Luh, P. B., Yan, J. H., and Stern, G. A.: Novel exploitation of convex hull invariance for solving unit commitment by using surrogate Lagrangian relaxation and branch-and-cut. In: Proceedings of the IEEE PES GM, Denver, Colorado (2015)

45. Zhang, S., Song, Y., Hu, Z., & Yao, L. (2011). Robust optimization method based on scenario analysis for unit commitment considering wind uncertainties. Proceedings of the IEEE Power and Energy Society. General Meeting, Detroit, MI.

Chapter 3

Augmented Surrogate Lagrangian Relaxation for Mixed-Integer Programming Problems

For many important mixed-integer programming (MIP) problems, the goal is to obtain near-optimal solutions with quantifiable quality in a computationally efficient manner (within, e.g., 5, 10 or 20 minutes). An important subclass of MIP problems include "block-structured" [1] mixed-integer linear programming (MILP) problems, which can be viewed as multiple subsystems interconnected through system-wide coupling constraints. Examples include Generalized Assignment problems [2, 3], Location-Routing problems [4-7], and Unit Commitment and Economic Dispatch problems [8-14]. Linearity can be exploited by branch-and-cut [15-17], and separability can be exploited by Lagrangian relaxation [3, 18-22]. For example, after relaxing assignment constraints, Generalized Assignment Problems can be decomposed into machine subproblems [2, 3], and Unit Commitment and Economic Dispatch problems can be decomposed into unit subproblems after relaxing system demand constraints [8-14]. However, time required to obtain solutions with good quality by branch-and-cut and Lagrangian relaxation is frequently large [23-24]. Such difficulties arise because of fundamental difficulties as will be explained ahead.

Branch-and-Cut To solve MILP problems, branch-and-cut [15-17] attempts to obtain the "convex hull" by using "facet-defining" cuts. Then the optimal solution is obtained at one of the vertices of the convex hull. One way to obtain facets of the convex hull is by using "lift-and-project" cuts [25-32]. Another way to obtain facets of the convex hull is by using Gomory cuts [33-38]. Cuts are created after linearly combining constraints and then rotating and shifting resulting hyperplanes. When solving block-

structured MILP problems, the method does not exploit "block structures," and constraints within one block are handled globally thereby affecting the solution process of the entire problem and leading to slow convergence. When the convex hull has complicated facial structures, facet-defining cuts are difficult to obtain, and the method relies mostly on time-consuming branching operations [39-40]. These difficulties have been vividly demonstrated for unit commitment problems with combined cycle units that arise in power systems [41-42].

Lagrangian Relaxation and Surrogate Lagrangian Relaxation The "block structure" can be exploited within Lagrangian relaxation, and complexity of resulting subproblems and associated convex hulls is drastically reduced. However, the method may require significant computational requirements and suffer from slow convergence because frequently used subgradient methods [8-14, 43-50] requires solving all subproblems to update Lagrange multipliers. These computational difficulties have been overcome by the surrogate subgradient method without solving all subproblems [51]. Resulting surrogate subgradient directions are smoother and form smaller acute angles toward the optimal multipliers as compared to subgradient directions, thereby alleviating zigzagging and reducing the number of iterations required for convergence. However, convergence proof and practical implementations of the method critically require the knowledge of the optimal dual value, thereby leading to major convergence difficulties. These difficulties have been overcome by our recently-developed Surrogate Lagrangian Relaxation method [52]. The difficulty of the method is that levels of constraint violations may not reduce fast enough.

Method of Multipliers (*"Augmented Lagrangian Relaxation"*) To accelerate the reduction of violation levels of relaxed constraints, the method of multipliers (referred to as "Augmented Lagrangian relaxation" (ALR)) [53-57] was proposed in the late 1960s by Hestenes [53] and Powell [54] whereby constraint violations are penalized by introducing quadratic penalty terms. Under assumptions of convexity and smoothness of the objective function and constraints, convergence has been established [55]. The ALR method has been used for MILP [57] problems and it has been shown duality gap approaches zero. However, the associated computational effort may be large because of the combinatorial nature of the

problems. While Augmented Lagrangian relaxation has been one of the fastest methods, because of the introduction of quadratic penalty terms, the relaxed problem is nonlinear and non-separable.

In this Section, to solve MIP problems to obtain near-optimal solutions with quantifiable quality and within strict time limits, a novel solution methodology is developed. In Section 3.1, the surrogate Augmented Lagrangian relaxation methodology is developed based on our recent Surrogate Lagrangian Relaxation with major improvements on convergence through the introduction of quadratic penalty terms as motivated by the fast convergence of Augmented Lagrangian relaxation.

In Section 3.2, when specializing to solving MILP problems by using the surrogate Augmented Lagrangian relaxation method, existing linearization methods and their difficulties are presented. To preserve fast convergence characteristics of surrogate Augmented Lagrangian relaxation while exploiting linear solvers, the augmented relaxed problem is linearized through the novel V-shape linearization scheme that preserves positions of minima.

In Section 3.3, when further specializing to solving block-structured MILP problems, the linearized relaxed problem is decomposed into smaller subproblems with exponentially reduced complexity as compared to the original problem thereby drastically reducing computational requirements. Moreover, analytical subproblem solutions are obtained thereby making the reduction of computational requirements even more drastic.

In Section 3.4, key steps of the algorithm for solving MIP and MILP problems will be provided together with an explanation of the process of obtaining of feasible solutions and the development of novel guidelines for selecting penalty coefficients. Computational efficiency of the new method is improved by exploiting the novel observation that after multipliers are updated, subproblem constraints do not change. Correspondingly, subproblem convex hulls never change. In an ideal situation, if such invariant convex hulls are obtained and kept, subproblems become LP problems and can be solved very efficiently without further cutting

In Section 3.5, efficiency of whole-problem and subproblem cuts is discussed. In particular, it is proved that under fairly reasonable assumptions, efficiency of subproblem cuts is equivalent to that of whole problem cuts.

In Section 3.6, it is demonstrated numerically that convergence of the new method is more stable as compared to standard subgradient methods. When solving large-scale MILP problems whereby convex hulls are difficult to obtain such as generalized assignment problems and unit commitment problems with combined cycle units, it is demonstrated the new method is robust and much more efficient as compared to frequently-used branch-and-cut and our recent surrogate Lagrangian relaxation, and represents a major step forward to solve difficult MILP problems.

3.1. Mixed-Integer Programming and Surrogate Augmented Lagrangian Relaxation

After presenting a mixed-integer programming (MIP) problem, motivated by fast convergence of Augmented Lagrangian relaxation (ALR) [53-57], *surrogate Augmented Lagrangian relaxation* (SALR) will be developed whereby complexity involved in solving relaxed problems will be reduced by requiring the "surrogate subgradient condition," which guarantees that surrogate subgradient directions form acute angles with directions toward optimal multipliers. Moreover, surrogate directions are smooth, zigzagging is alleviated, and under appropriate choices of stepsizes convergence is guaranteed.

3.1.1. Mixed-Integer Programming

Consider a MIP problem in a general form:

$$\min_{x,y} f(x,y) \text{ , subject to } g(x,y) \le 0 \text{ , } (x,y) \in \Omega, \tag{3.1}$$

where Ω is a bounded and non-empty subset of $\mathbb{R}^p \times \mathbb{Z}^n$ and functions $f(x, y)$ and $g(x, y)$ satisfy the following assumption.

Assumption 3.1.1. (Boundedness) Function $f(x, y)$ is bounded from below and $g(x, y)$ is bounded from above and below.

3.1.2. Augmented Lagrangian Relaxation

After converting inequality constraints $g(x, y) \leq 0$ into equality constraints by introducing non-negative continuous slack variables $z \in \mathbb{R}^m, z \geq 0$, the *Augmented Lagrangian function* can be written as:

$$L_{c^k}(x, y, z, \lambda) = f(x, y) + \lambda^T (g(x, y) + z) + \frac{c^k}{2} \|g(x, y) + z\|^2, \ (x, y) \in \mathbb{R}^p \times \mathbb{Z}^n, \ z \in \mathbb{R}^m, z \geq 0, \quad \lambda \in \mathbb{R}^m, \ z \geq 0. \tag{3.2}$$

Here $\{c^k\}$ are nonnegative scalar penalty coefficients. The *augmented relaxed problem* is defined as:

$$\min_{x,y,z} L_{c^k}(x, y, z, \lambda), \ (x, y) \in \Omega, \ \lambda^T \in \mathbb{R}^m. \tag{3.3}$$

The *augmented dual function* that results from solving the relaxed problem (3.3) can be defined as:

$$q_{c^k}(\lambda) := \min_{x,y,z} L_{c^k}(x, y, z, \lambda), \ (\mathrm{x}, y) \in \Omega, \ z \in \mathbb{R}^m, z \geq 0, \lambda^T \in \mathbb{R}^m. \tag{3.4}$$

3.1.3. Surrogate Augmented Lagrangian Relaxation

Generally, the minimization in (3.3) can be time-consuming, and as a result, the traditionally used subgradient method may require significant computational effort. Moreover, subgradient methods frequently suffer from zigzagging of multipliers. To alleviate these difficulties, the complexity is reduced by requiring the satisfaction of the following surrogate optimality condition [52, p. 178]:

$$\tilde{L}_{c^k}\left(x^k, y^k, z^k, \lambda^k\right) < \tilde{L}_{c^k}\left(x^{k-1}, y^{k-1}, z^{k-1}, \lambda^k\right), \tag{3.5}$$

where $\tilde{L}_{c^k}\left(x^k, y^k, z^k, \lambda^k\right)$ is the *surrogate augmented dual value* defined for a feasible solution (x^k, y^k, z^k) of (3.3) as:

$$\tilde{L}_{c^k}\left(x^k, y^k, z^k, \lambda^k\right) := f(x^k, y^k) + \left(\lambda^k\right)^T \left(\tilde{g}(x^k, y^k, z^k)\right) + \frac{c^k}{2}\left\|\tilde{g}(x^k, y^k, z^k)\right\|^2 . \tag{3.6}$$

Surrogate subgradient directions are defined for a feasible solution (x^k, y^k, z^k) of (3.3) as:

$$\tilde{g}(x^k, y^k, z^k) := g(x^k, y^k) + z^k , \tag{3.7}$$

If solution (x^k, y^k, z^k) satisfies (3.5), then directions (3.7) form acute angles with directions toward λ_c^*. Therefore, under appropriate choice of stepsizes s^k, multipliers move toward the λ_c^* when updated in the following way:

$$\lambda^{k+1} = \lambda^k + s^k \tilde{g}(x^k, y^k, z^k) . \tag{3.8}$$

Convergence of the multiplier-updating scheme (3.8) under condition (3.5) will be discussed in the following subsection 3.1.4.

3.1.4. Convergence of Surrogate Augmented Lagrangian Relaxation

To guarantee convergence, diminishing stepsizes will be computed following Bragin, et al [52] based on the notion of contraction mapping as:

$$s^k = \alpha_k \frac{s^{k-1}\left\|\tilde{g}\left(x^{k-1}, y^{k-1}, z^{k-1}\right)\right\|}{\left\|\tilde{g}\left(x^k, y^k, z^k\right)\right\|}, \; 0 < \alpha_k < 1 . \tag{3.9}$$

A specific formula for setting α_k to guarantee convergence without requiring the optimal dual value is:

$$\alpha_k = 1 - \frac{1}{Mk^\rho}, \; \rho = 1 - \frac{1}{k^r}, \; M \geq 1, \; 0 < r < 1 . \tag{3.10}$$

It will be assumed that the penalty coefficients are constant $c^k = c$. The result can be summarized in the following Theorem.

Theorem 3.2.1. (Convergence of SALR). *Suppose that multipliers are updated per (3.8), surrogate optimality condition (3.5) is satisfied, stepsizes are set according to (3.9)-(3.10) and penalty coefficients are constant. If α_k satisfies the following conditions:*

$$\prod_{i=1}^{k} \alpha_i \to 0\,, \tag{3.11}$$

and

$$\lim_{k\to\infty} \frac{1-\alpha_k}{s^k} = 0\,, \tag{3.12}$$

then multipliers converge to λ_c^* that maximize the following augmented dual function:

$$q_c(\lambda) = \min_{x,y,z} L_c(x, y, z, \lambda),\ (x, y) \in \Omega \subset \mathbb{R}^p \times \mathbb{Z}^n,\ z \in \mathbb{R}^m, z \geq 0,\ \textit{and}\ \lambda^T \in \mathbb{R}^m. \tag{3.13}$$

The goal of the proof is to satisfy the result of Polyak [58] whereby convergence was established with diminishing and non-summable stepsizes, and Theorem will be proved using three Propositions. In Proposition 3.1.3, it will be proved that stepsizes (3.9)-(3.10) with condition (3.11) satisfy the diminishing property. Moreover, under condition (3.11), surrogate dual values (3.6) approach dual values and multipliers converge to a unique limit (not necessarily optimal). In Proposition 3.1.4 an asymptotical representation of stepsizes (3.9) will be derived under condition (3.12). In Proposition 3.1.5, it will be proved that stepsizes satisfy the non-summability property required in [58] for convergence to the optimum.

Proposition 3.1.3. *With the stepsizing formula (3.9)-(3.10), the Lagrange multipliers (3.8) converge to a unique limit (3.not necessarily optimal), provided the surrogate optimality condition (3.5) and condition (3.11) hold.*

Proof Since stepsizes (3.9) are updated recursively, they can be represented as:

$$s^k = \prod_{i=1}^{k} \alpha_i \frac{s^0 \left\| \tilde{g}\left(x^0, y^0, z^0\right) \right\|}{\left\| \tilde{g}\left(x^k, y^k, z^k\right) \right\|} \tag{3.14}$$

Therefore, under condition (3.11), the diminishing property of stepsizes follows immediately. To prove convergence with diminishing stepsizes, it needs to be shown that under the surrogate optimality condition (3.5), surrogate dual values (3.6) approach dual values. Indeed, consider a fixed value of

multipliers λ and a fixed value of penalty coefficients c. Then after an iteration N, a series of surrogate optimizations will lead to

$$...<\tilde{L}_c\left(x^{N+2},y^{N+2},z^{N+2},\lambda\right)<\tilde{L}_c\left(x^{N+1},y^{N+1},z^{N+1},\lambda\right)<\tilde{L}_c\left(x^{N},y^{N},z^{N},\lambda\right). \tag{3.15}$$

Because of the discrete nature of the problem, the dual value will be reached after a finite number of iterations:

$$q_c(\lambda)\le...<\tilde{L}_c\left(x^{N+2},y^{N+2},z^{N+2},\lambda\right)<\tilde{L}_c\left(x^{N+1},y^{N+1},z^{N+1},\lambda\right)<\tilde{L}_c\left(x^{N},y^{N},z^{N},\lambda\right). \tag{3.16}$$

Since stepsizes (3.14) approach zero because of (3.11), the same argument will hold when stepsizes are sufficiently small. Therefore, after certain iteration, convergence of the method will essentially follow that of a subgradient method with diminishing stepsizes. □

Proposition 3.1.4. *An asymptotic representation of stepsizes (3.9)-(3.10) under condition (3.12) is:*

$$s^k=\frac{\left(\prod_{j=0}^{k-1}\eta^j\right)s^0}{1+s^0\sum_{i=0}^{k-1}\left(\left(\prod_{j=0}^{i}\eta^j\right)K^{i+1}\right)}, \text{ where } \eta^k=\frac{\left\|g\left(x^k,y^k,z^k\right)\right\|}{\left\|g\left(x^{k+1},y^{k+1},z^{k+1}\right)\right\|}. \tag{3.17}$$

Proof A series $\left\{\frac{1-\alpha_k}{s^k}\right\}$ in (3.12) with non-negative terms is convergent to zero. Therefore, (3.12) can be written as

$$1-\alpha_k=K^k s^k, K^k\to 0, \tag{3.18}$$

where $\{K^k\}$ is a non-negative series approaching zero. Also, it is natural to assume that each term is bounded

$$K^k\le C^K. \tag{3.19}$$

At iteration 1, from (3.18) stepsize setting parameter α_1 can be expressed as:

$$\alpha_1=1-K^1s^1. \tag{3.20}$$

From the stepsizing formula (3.9) and expression for α_1 in (3.20), stepsize s^1 at iteration 1 can be expressed as:

$$s^1 = \eta^0\left(1 - K^1 c^1\right)c^0, \tag{3.21}$$

The equation (3.21) is linear in terms of s^1, and solving (3.21) for s^1 one gets:

$$s^1 = \frac{\eta^0 c^0}{1+\eta^0 K^1 s^0}. \tag{3.22}$$

To derive the general asymptotic expression for s^k, consider an expression for s^2:

$$s^2 = \frac{\eta^1 s^1}{1+\eta^1 K^2 s^1} = \frac{\eta^1 \dfrac{\eta^0 s^0}{1+\eta^0 K^1 s^0}}{1+\eta^1 K^2 \dfrac{\eta^0 s^0}{1+\eta^0 K^1 s^0}} = \frac{\eta^1\eta^0 s^0}{1+\eta^0 K^1 s^0 + \eta^1\eta^0 K^2 s^0}. \tag{3.23}$$

In a similar fashion, the expression for s^3 can be obtained as:

$$s^3 = \frac{\eta^2 s^2}{1+\eta^2 K^3 s^2} = \frac{\eta^2 \dfrac{\eta^1\eta^0 s^0}{1+\eta^0 K^1 s^0 + \eta^1\eta^0 K^2 s^0}}{1+\eta^2 K^3 \dfrac{\eta^1\eta^0 s^0}{1+\eta^0 K^1 s^0 + \eta^1\eta^0 K^2 s^0}} = \frac{\eta^2\eta^1\eta^0 s^0}{1+\eta^0 K^1 s^0 + \eta^1\eta^0 K^2 s^0 + \eta^2\eta^1\eta^0 K^3 s^0}. \tag{3.24}$$

Following the pattern, an expression for s^k can be inductively represented as in (3.17). □

Proposition 3.1.5. (Non-summability of (3.17)) *Stepsizes (3.17) are non-summable.*

Proof To prove the non-summability of stepsizes, consider the following sum:

$$\sum_{k=1}^{\infty} s^k = \sum_{k=1}^{\infty}\left(\frac{\left(\prod_{j=0}^{k-1}\eta^j\right)s^0}{1+s^0\sum_{i=0}^{k-1}\left(\left(\prod_{j=0}^{i}\eta^j\right)K^{i+1}\right)}\right). \tag{3.25}$$

In the following, to prove the non-summability of (3.17), a lower bound of the right-hand side of (3.25) will be derived and proved to be infinite thereby implying that the summation in (3.25) is infinite. To derive the lower bound, Assumption 3.1.1 leading to the boundedness of subgradient norms and boundedness of K^k in (3.19) will be used.

First, the boundedness of $\prod_{j=0}^{k-1}\eta^j$ from above follows from Assumption 2.1. Indeed, since the domain Ω is bounded and non-empty, the norms of constraints are bounded:

$$\prod_{j=0}^{k-1}\eta^j=\prod_{j=0}^{k-1}\frac{\left\|g\left(x^j,y^j,z^j\right)\right\|}{\left\|g\left(x^{j+1},y^{j+1},z^{j+1}\right)\right\|}=\frac{\left\|g\left(x^0,y^0,z^0\right)\right\|}{\left\|g\left(x^k,y^k,z^k\right)\right\|}>C^\eta>0 \text{ for all } k. \tag{3.26}$$

Second, the boundedness of $\left(\prod_{j=0}^{i}\eta^j\right)K^{i+1}$ follows from the definition of η^k and

$$\left(\prod_{j=0}^{i}\eta^j\right)K^{i+1}\le\left(\prod_{j=0}^{i}\eta^j\right)C^K=C^K\prod_{j=0}^{i}\frac{\left\|g\left(x^j,y^j,z^j\right)\right\|}{\left\|g\left(x^{j+1},y^{j+1},z^{j+1}\right)\right\|}=C^K\frac{\left\|g\left(x^0,y^0,z^0\right)\right\|}{\left\|g\left(x^{j+1},y^{j+1},z^{j+1}\right)\right\|}<C<\infty \text{ for all } i. \tag{3.27}$$

The summation in the denominator of (3.25) can be estimated by using (3.27) as

$$\sum_{i=0}^{k-1}\left(\left(\prod_{j=0}^{i}\eta^j\right)K^{i+1}\right)<\sum_{i=0}^{k-1}C=C\cdot k\,. \tag{3.28}$$

Therefore, (3.25) can be estimated by using (3.26) and (3.27) as

$$\sum_{k=1}^{\infty}s^k>\sum_{k=1}^{\infty}\left(\frac{C^\eta\cdot s^0}{1+s^0\cdot C\cdot k}\right)=\frac{C^\eta}{C}\sum_{k=1}^{\infty}\left(\frac{1}{\frac{1}{C\cdot s^0}+k}\right). \tag{3.29}$$

By using the change of variables $k=\kappa-\left\lfloor\frac{1}{C\cdot s^0}\right\rfloor-1$, the last summation can be represented as:

$$\frac{C^\eta}{C}\sum_{k=1}^{\infty}\left(\frac{1}{\frac{1}{C\cdot s^0}+k}\right)=\frac{C^\eta}{C}\sum_{\kappa=2+\left\lfloor\frac{1}{C\cdot s^0}\right\rfloor}^{\infty}\left(\frac{1}{\frac{1}{C\cdot s^0}+\kappa-\left\lfloor\frac{1}{C\cdot s^0}\right\rfloor-1}\right). \tag{3.30}$$

The fractional part of any number is between 0 and 1, therefore,

$$0\le\frac{1}{C\cdot s^0}-\left\lfloor\frac{1}{C\cdot s^0}\right\rfloor\le 1\,. \tag{3.31}$$

From (3.31) is follows that

$$-1\le\frac{1}{C\cdot s^0}-\left\lfloor\frac{1}{C\cdot s^0}\right\rfloor-1\le 0\,. \tag{3.32}$$

Therefore, each term in the right-hand side of (3.30) can be estimated as:

$$\frac{1}{\frac{1}{C\cdot s^0}+\kappa-\left\lfloor\frac{1}{C\cdot s^0}\right\rfloor-1}\geq\frac{1}{\kappa}. \tag{3.33}$$

Based on inequalities (3.30)-(3.33), inequality (3.29) can be written as:

$$\sum_{k=1}^{\infty}s^k>\sum_{k=1}^{\infty}\left(\frac{s^0}{1+s^0\cdot C\cdot k}\right)=\frac{C^\eta}{C}\sum_{k=1}^{\infty}\left(\frac{1}{\frac{1}{C\cdot s^0}+k}\right)>\frac{C^\eta}{C}\sum_{\kappa=2+\left\lfloor\frac{1}{C\cdot s^0}\right\rfloor}^{\infty}\left(\frac{1}{\kappa}\right)=\infty. \tag{3.34}$$

Therefore, stepsizes (3.9)-(3.10) that satisfy conditions (3.11-12) approach zero and are non-summable. □

Proof of Theorem 3.1.2 follows from Propositions 3.1.3-3.1.5 and is summarized below.

Proof of Theorem 3.1.2 As proved in Proposition 3.1.3, the SALR method behaves as the subgradient method under condition (3.11). In Propositions 3.1.4-3.1.5 it is proved that stepsizes are non-summable under condition (3.12). Therefore, following the classical result of Polyak [58], the multipliers within the SALR method converge to λ_c^*. □

Proposition 3.1.6. (Rate of Convergence of SALR) *The rate of convergence of the SALR method is linear outside of a sphere centered at λ_c^* with the radius of the sphere defined in the following inequality:*

$$\sqrt{\frac{s^k\left\|\tilde{g}\left(x^k,y^k,z^k\right)\right\|^2}{\mu}}\leq\left\|\lambda^k-\lambda_c^*\right\|, \tag{3.35}$$

assuming that the positive constant μ exists such that

$$q_c\left(\lambda_c^*\right)-\tilde{L}_c\left(x^k,y^k,z^k,\lambda^k\right)\geq\mu\left\|\lambda^k-\lambda_c^*\right\|^2. \tag{3.36}$$

Also, stepsizes are assumed to be sufficiently small

$$0<s^k<\frac{1}{2\mu}. \tag{3.37}$$

Proof Steps of the proof follow closely those of Proposition 2.5 in [52] provided to the surrogate Lagrangian relaxation. □

Within SALR, because of penalization of constraint violations it is expected that the norm of surrogate subgradients will reduce faster than within SLR. Therefore, the sphere centered at λ_c^* outside of

which linear convergence is possible will have a smaller radius within SALR per (3.35). Therefore, within SALR multipliers can get closer to the optimum with a linear rate.

3.2. Surrogate Augmented Lagrangian Relaxation and V-shape Linearization for MILP Problems

In this section, a general formulation of an MILP problem will be presented in subsection 3.2.1. To achieve linearity of augmented relaxed problems, current linearization methods and associated difficulties will be presented in subsection 3.2.2. In subsection 3.2.3, to preserve fast convergence of Augmented Lagrangian relaxation while exploiting linear solvers, the augmented relaxed problem is linearized through the novel V-shape linearization scheme that preserves positions of minima. In subsection 3.2.4, convergence of the resulting combination of surrogate Augmented Lagrangian relaxation and branch-and-cut (3.SALR+B&C) will be proved.

3.2.1. Mixed-Integer Linear Programming

An MILP problem can be formulated following [24] as:

$$\min_{x,y} \left\{ d^x x + d^y y \right\}, \text{ subject to } Ax + Ey \le b, \ (x, y) \in \Omega, \tag{3.38}$$

where Ω is a subset of $\mathbb{R}^p \times \mathbb{Z}^n$, with $\mathbb{R}$ denoting the set of real numbers and $\mathbb{Z}$ the set of integers. Vectors x and y are $p\times1$ and $n\times1$ column decision vectors, respectively. Matrices A and E are have dimensions $m\times p$ and $m\times n$, d^x is a $1\times p$ vector, d^y is a $1\times n$ vector, b is an $m\times1$ column vector. To existence of solutions to (3.38) will be guaranteed per following Assumptions.

Assumption 3.2.1 (Boundedness) The domain of (3.38) Ω is bounded and non-empty.

Assumption 3.2.2 (Full rank) Matrices A and E are full-rank.

3.2.2. Augmented Lagrangian Relaxation

Following subsection 3.1, the *Augmented Lagrangian function* is formed by converting inequality constraints in (3.38) into equality constraints by introducing non-negative continuous slack variables $z \in \mathbb{R}^m, z \geq 0$, introducing real-valued multipliers $\lambda^T = (\lambda_1, \ldots, \lambda_m) \in \mathbb{R}^m$, and penalizing violations of relaxed constraints:

$$L_{c^k}\left(x, y, z, \lambda^k\right) = d^x x + d^y y + \left(\lambda^k\right)^T \left(Ax + Ey - b + z\right) + \frac{c^k}{2}\left\|Ax + Ey - b + z\right\|^2. \tag{3.39}$$

Here $\{c^k\}$ are nonnegative scalar penalty coefficients. At every iteration, the following *augmented relaxed problem* is solved:

$$\min_{x,y,z} L_{c^k}\left(x, y, z, \lambda^k\right),\ (x, y) \in \Omega \subset \mathbb{R}^p \times \mathbb{Z}^n,\ z \in \mathbb{R}^m, z \geq 0, \text{ and } \lambda^T \in \mathbb{R}^m. \tag{3.40}$$

The major difficulty is that (3.40) is nonlinear because of quadratic penalty terms that include squares and cross-products of decision variables. Therefore, the problem (3.40) cannot be solved by using branch-and-cut. To present existing linearization method and their drawbacks, a simple example will be considered.

Example 3.2.1 (Difficulties of Existing Linearization Approaches) Consider a simple MILP example:

$$\min_{x,y}(3x + 4y),\ s.t.\quad x + y = 3,\ x, y \in \Omega \subset \mathbb{R} \times \mathbb{Z}, \text{ and } \Omega = [0,10] \times [0,10]. \tag{3.41}$$

Following (3.39), the relaxed problem corresponding to (3.41) is:

$$\min_{x,y}\left(3x + 4y + \lambda(x + y - 3) + \frac{c}{2}(x + y - 3)^2\right),\ x, y \in \Omega \subset \mathbb{R} \times \mathbb{Z}, \text{ and } \Omega = [0,10] \times [0,10]. \tag{3.42}$$

Two linearization approaches will be presented together with their pros and cons.

A simple approach to linearize the penalty function at iteration *k* is by fixing certain decision variables at their values at iteration *k*-1:

$$\min_{x,y}\left(3x+4y+\lambda(x+y-3)+\frac{c}{2}(x+y-3)\left(x^{k-1}+y^{k-1}-3\right)\right),\ x,y\in\Omega\subset\mathbb{R}\times\mathbb{Z},\ \text{and}$$
$$\Omega=[0,10]\times[0,10]. \tag{3.43}$$

However, because the objective function is linear, solutions will always be at boundaries of the domain Ω. This will lead to jumping of solution values from one extreme to another, resulting in zigzagging of multipliers and slow convergence. As a result, convergence characteristics will be drastically affected through the linearization (3.43). To avoid this situation, deviations of solutions from previous values are typically penalized as:

$$\min_{x,y}\left\{3x+4y+\lambda(x+y-3)+cx\left(x^{k-1}+y^{k-1}-3\right)+\frac{\eta}{2}\left(x-x^{k-1}\right)^2+cy\left(x^{k-1}+y^{k-1}-3\right)+\frac{\eta}{2}\left(y-y^{k-1}\right)^2\right\},$$
$$x,y\in\Omega\subset\mathbb{R}\times\mathbb{Z},\ \text{and}\ \Omega=[0,10]\times[0,10]. \tag{3.44}$$

The problem (3.44) is still nonlinear and cannot be solved by using MILP solvers such as branch-and-cut.

To exploit linear solvers when solving the augmented relaxed problem (3.40) while preserving fast convergence characteristics of the SALR method, the novel V-shape linearization will be developed in the following subsection 3.2.3.

3.2.3. V-shape Linearization

To avoid the above-mentioned difficulties, a novel three-step V-shape linearization process of the entire problem (3.40) is introduced. In this way, it will be ensured that solutions of the linearized relaxed problem and solutions of (3.40) are the same by design thereby preserving convergence characteristics of SALR. In the first step, a convex extension of (3.40) without bounds on (x, *y*, *z*) and without integral restrictions on *y* will be considered:

$$\min_{x,y,z} L_{c^k}\left(x,y,z,\lambda^k\right),\ (x,y)\in\mathbb{R}^p\times\mathbb{R}^n,\ z\in\mathbb{R}^m,\ \text{and}\ \lambda^T\in\mathbb{R}^m. \tag{3.45}$$

A minimum of (3.45) with respect to one decision variable at a time is evaluated after taking a partial derivative. Since the objective function in (3.45) is quadratic its partial derivatives are linear, and the

minimum of (3.45) is a linear function of other variables. In the second step, this linearity is exploited to construct piecewise-linear V-shape functions with the same minima of (3.45) through the use of absolute values with slopes appropriately defined, and the V-shape function with the same minimum as that of (3.45) is constructed by taking a linear combination of individual V-shape functions. Because the domain Ω in (3.40) is bounded, y is discrete and z is non-negative, solutions to (3.45) will be projected onto respective feasible sets. In the third step, each absolute function is linearized following [59, p. 28], and projections of solutions will be linearized using big-M inequalities.

Step 1: Evaluation of Minima. To obtain the minimum of (3.45) with respect to each variable, derivations will be first shown for x_i, the i^{th} component of x. For convenience of derivations, after grouping terms containing x, (3.39) can be represented as:

$$\frac{c^k}{2}x^T A^T Ax + \left(c^k x^T A^T (Ey - b + z) + x^T A^T \lambda^k + x^T \left(d^x\right)^T\right) + \left(\frac{c^k}{2}\left\|Ey - b + z\right\|^2 + d^y y + \left(\lambda^k\right)^T (Ey - b + z)\right). \quad (3.46)$$

The last term in parentheses does not contain variables x and will be omitted. Remaining terms containing x_i are:

$$\left(\frac{c^k}{2}x_i^2 A_i^T A_i + c^k \sum_{j \neq i} x_i A_i^T A_j x_j\right) + c^k x_i A_i^T (Ey - b + z) + x_i A_i^T \lambda^k + x_i d_i^x . \quad (3.47)$$

Here A_i is the i^{th} column of matrix A and d_i^x is the i^{th} component of d^x. The expression in (3.47) can be rewritten as:

$$\frac{c^k}{2}x_i^2 A_i^T A_i + x_i\left(c^k \sum_{j \neq i} A_i^T A_j x_j + c^k A_i^T (Ey - b + z) + A_i^T \lambda^k + d_i^x\right). \quad (3.48)$$

Because matrix A is full rank by Assumption 3.2.2, the inverse of $A_i^T A_i$ exists, and the minimum of (3.48) can be obtained after taking a partial derivative of (3.48) with respect to x_i, equating the derivative to zero and solving for x_i:

$$x_{i,\min} = -\frac{1}{2c^k A_i^T A_i}\left(c^k \sum_{j\neq i} A_i^T A_j x_j + c^k A_i^T (Ey - b + z) + A_i^T \lambda^k + d_i^x \right). \tag{3.49}$$

Since the Augmented Lagrangian function (3.39) is quadratic, all its partial derivatives are linear and the minimum (3.49) is a linear function. However, because the domain Ω is bounded, (3.49) need to be projected onto it. The resulting projected minimum can be denoted as $[x_{i,\min}]_\Omega$. In a special case of simple bounds such as $x_i \in [x_i^l, x_i^u]$, the projection can be written as:

$$\left[x_{i,\min}\right]_\Omega = \min\left\{\max\left\{x_{i,\min}, x_i^l\right\}, x_i^u\right\}. \tag{3.50}$$

In a similar fashion, minima with respect to y_i and z_i can be obtained as $[y_{i,\min}]_\Omega$, $[z_{i,\min}]_+$ where $y_{i,\min}$, $z_{i,\min}$ are defined as:

$$y_{i,\min} = -\frac{1}{2c^k E_i^T E_i}\left(c^k \sum_{j\neq i} E_i^T E_j y_j + c^k E_i^T (Ax - b + z) + E_i^T \lambda^k + d_i^y \right), \text{ and } z_{i,\min} = -(Ax + Ey - b)_i . \tag{3.51}$$

Since decision variables y_i are discrete, $y_{i.\min}$ in (3.51) will not be the actual minimum of (3.40). In Figure 3.2.1, it is shown that the discrete valued minimum y_i^* does not coincide with the minimum $y_{i.\min}$ of the convex extension (3.45). In the following Step 2, this issue will be resolved by constructing V-shape function with the same positions of discrete as well as continuous minima as those of (3.40).

Step 2: Construction of V-shape Functions. Piece-wise linear functions with minima $[x_{i,\min}]_\Omega$, $[y_{i,\min}]_\Omega$, $[z_{i,\min}]_+$ are constructed by using absolute-value functions as:

$$a_i^x \left| x_i - \left[x_{i,\min}\right]_\Omega \right|, \; a_i^y \left| y_i - \left[y_{i,\min}\right]_\Omega \right| \text{ and } a_i^z \left| z_i - \left[z_{i,\min}\right]_+ \right|, \tag{3.52}$$

where

$$a_i^x = \frac{c^k}{2}\left| x_i^{k-1} - \left[x_{i,\min}^{k-1}\right]_\Omega \right|, \; a_i^y = \frac{c^k}{2}\left| y_i^{k-1} - \left[y_{i,\min}^{k-1}\right]_\Omega \right|, \text{ and } a_i^z = \frac{c^k}{2}\left| z_i^{k-1} - \left[z_{i,\min}^{k-1}\right]_+ \right|. \tag{3.53}$$

V-shape functions of x_i and y_i are shown in Figure 1. Because of symmetry of the V-shape functions (3.52), integral solutions y_i and corresponding minima of (3.40) and (3.52) are the same. Since positions of minima in (3.52) are $[x_{i,\min}]_\Omega$, $[y_{i,\min}]_\Omega$, $[z_{i,\min}]_+$ regardless of other variables, a linear combination of

functions in (3.52) will have the same minimum as that of (3.39). The minimization of the piece-wise linear function is:

$$\min_{x,y,z}\left\{\sum_{i=1}^{p} a_i^x \left| x_i - \left[x_{\min,i}\right]_\Omega \right| + \sum_{i=1}^{n} a_i^y \left| y_i - \left[y_{\min,i}\right]_\Omega \right| + \sum_{i=1}^{m} a_i^z \left| z_i - \left[z_{\min,i}\right]_+ \right|\right\}, (x, y) \in \Omega \subset \mathbb{R}_+^p \times \mathbb{Z}_+^n, z \in \mathbb{R}_+^m, \tag{3.54}$$

and $\lambda^T \in \mathbb{R}^m$.

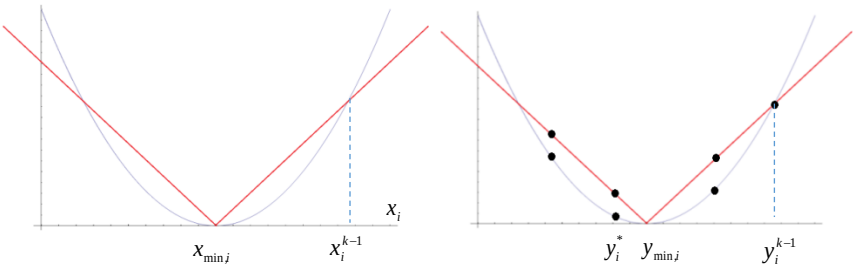


Figure 3.2.1. Illustrations of the augmented Lagrangian relaxation (3.39) (parabola) and the V-shape function (absolute value) (3.52). Discrete values in the domain of y_i are denoted by black dots.

Step 3: Linearization of V-shape Functions. The objective function of (3.54) is nonlinear because of absolute value functions and operators $[]_\Omega$ that project solutions onto Ω. The linearization of the absolute value function will be performed following [59, p. 28], and the linearization of projection operators will be performed by using big-M inequalities.

The idea behind linearization of absolute function can be summarized in a simple problem:

$$\min_x |x|, \tag{3.55}$$

which can be linearized as

$$\min_{x,Q} Q, s.t. x \le Q, x \ge -Q. \tag{3.56}$$

Therefore, (3.54) can be linearized as:

$$\min_{x,y,z}\left\{\sum_{i=1}^{p} a_i^x Q_i^x + \sum_{i=1}^{n} a_i^y Q_i^y + \sum_{i=1}^{m} a_i^z Q_i^z\right\}, \tag{3.57}$$

$$s.t. \ -Q_i^x \le x_i - \left[x_{\min,i}\right]_\Omega \le Q_i^x, -Q_i^y \le y_i - \left[y_{\min,i}\right]_\Omega \le Q_i^y, -Q_i^z \le z_i - \left[z_{\min,i}\right]_+ \le Q_i^z. \tag{3.58}$$

The problem (3.57)-(3.58) is still nonlinear because of projection operators. The linearization concept will be explained by converting the operator $[]_+$ in terms of the max-function in the last inequality of (3.58), and then the max-function will be linearized by using big-*M* inequalities. After rearranging terms, the last inequality of (3.58) becomes:

$$\max\left\{z_{i,\min},0\right\} \le z_i + Q_i^z,\ z_i - Q_i^z \le \max\left\{z_{i,\min},0\right\}. \tag{3.59}$$

Within the first inequality, maximum of $z_{i,\min}$ and 0 has to be less than the right-hand side. Therefore, both $z_{i,\min}$ and 0 have to be less than the right-hand side

$$z_{i,\min} \le z_i + Q_i^z, 0 \le z_i + Q_i^z,\ z_i - Q_i^z \le \max\left\{z_{i,\min},0\right\}. \tag{3.60}$$

Within the last inequality, either $z_{i,\min}$ or 0 has to be greater than the left-hand side. This can be captured using an OR operator as:

$$z_{i,\min} \le z_i + Q_i^z, 0 \le z_i + Q_i^z,\ \left(z_i - Q_i^z \le z_{i,\min}\ \text{OR}\ z_i - Q_i^z \le 0\right). \tag{3.61}$$

Within (3.61), only one of the constraints within the parentheses needs to be satisfied, while the other constraint is redundant. The logical constraints can, therefore, be linearized through the introduction of two binary variables $\{\zeta_{z,1}, \zeta_{z,2}\}$ as:

$$z_{i,\min} \le z_i + Q_i^z, 0 \le z_i + Q_i^z,\ z_i - Q_i^z \le z_{i,\min} + M\cdot\zeta_{z,1},\ z_i - Q_i^z \le M\cdot\zeta_{z,2}, \zeta_{z,1} + \zeta_{z,2} = 1, \tag{3.62}$$

where *M* is a "big-*M*" parameter. By using the simple example, the above ideas will be illustrated.

Example 3.2.1 (Continued) **Step 1: Evaluation of a Minimum.** In this step, following (3.45), a convex extension of (3.42) is defined as:

$$\min_{x,y}\left(3x+4y+\lambda(x+y-3)+\frac{c}{2}(x+y-3)^2\right),\ x,y\in\mathbb{R}_+\times\mathbb{R}_+. \tag{3.63}$$

Position of the minimum can be found by first taking a partial derivative of the objective function in (3.63) as:

$$\frac{\partial\left(3x+4y+\lambda(x+y-3)+\frac{c}{2}(x+y-3)^2\right)}{\partial x}=3+\lambda+c(x+y-3). \tag{3.64}$$

A position of the minimum can be found by setting the expression in (3.64) to zero and solving for *x*. Therefore, without restrictions $x\in[0,10]$, the minimum of (3.63) is:

$$x_{\min}=\left(-\frac{3+\lambda-3c+cy}{c}\right). \tag{3.65}$$

To satisfy the non-negativity restriction of *x*,[8] the feasible minimum of (3.63) is

$$x^{+}_{\min}(\lambda,y)=\max\{x_{\min}(\lambda,y),0\}. \tag{3.66}$$

In a similar fashion, the *y*-minimum of (3.63) is:

$$y^{+}_{\min}(\lambda,x)=\max\{y_{\min}(\lambda,x),0\}=\max\left\{-\frac{4+\lambda-3c+cx}{c},0\right\}. \tag{3.67}$$

Step 2: Construction of V-shape Functions. A piece-wise linear function with the minimum (3.66) is constructed using an absolute value function:

$$\frac{c}{2}a\left|x-x^{+}_{\min}(\lambda,y)\right| \ . \tag{3.68}$$

By the properties of quadratic and absolute value function, the function (3.68) has the same position of the minimum with respect to *x* as the objective function of (3.63). In a similar fashion, a piece-wise linear with the minimum (3.67) can be constructed:

$$\frac{c}{2}b\left|y-y^{+}_{\min}(\lambda,x)\right| \ . \tag{3.69}$$

The construction of a piece-wise linear function with same positions of minima as those of (3.63), can be performed by taking a linear combination of (3.68) and (3.69) as:

---

[8] The other boundary on *x* ($x\leqslant 10$) can be treated in the same way and for the simplicity will not be presented.

$$\frac{c}{2}a\left|x-x^{+}_{\min}(\lambda,y)\right|+\frac{c}{2}b\left|y-y^{+}_{\min}(\lambda,x)\right| . \tag{3.70}$$

where *a* and *b* are nonnegative slopes which can be defined following (3.53).

Step 3: Linearization of V-shape Functions. To linearize (3.70), absolute value functions will be linearized first following (3.55-56) as

$$\min_{x,y,Q_x,Q_y}\left\{\frac{c}{2}a\cdot Q_x+\frac{c}{2}b\cdot Q_y\right\},\ x,y\in\Omega\subset\mathbb{R}_+\times\mathbb{Z}_+,\ \Omega=[0,10]\times[0,10],\ Q_x,Q_y\in\mathbb{R}, \tag{3.71}$$

$$s.t.\ -Q_x\le x-x^{+}_{\min}(\lambda,y)\le Q_x,\ -Q_y\le y-y^{+}_{\min}(\lambda,x)\le Q_y. \tag{3.72}$$

Then max functions in (3.72) can be linearized following the steps of (3.59)-(3.62).

3.2.4. Convergence of SALR+B&C with V-shape Linearization

Following the framework of Section 3.1, the relaxed problem (3.40) will be solved subject to the simple "surrogate optimality condition" [52, p. 178]:

$$\tilde{L}_c\left(x^k,y^k,z^k,\lambda^k\right)<\tilde{L}_c\left(x^{k-1},y^{k-1},z^{k-1},\lambda^k\right), \tag{3.73}$$

where $\tilde{L}_c\left(x^k,y^k,z^k,\lambda^k\right)$ is the surrogate dual function defined for a feasible solution of (3.57) as

$$\tilde{L}_c\left(x^k,y^k,z^k,\lambda^k\right)=d^x x^k+d^y y^k+\left(\lambda^k\right)^T\left(A^0x^k+E^0y^k-b^0+z^k\right)+\frac{c^k}{2}\left\|A^0x^k+E^0y^k-b^0+z^k\right\|^2. \tag{3.74}$$

Condition (3.73) ensures that surrogate subgradient directions $\tilde{g}(x^k,y^k,z^k)=A^0x^k+E^0y^k-b^0+z^k$ form acute angles with the direction toward λ^*. To guarantee convergence, diminishing stepsizes will be computed per (3.9)-(3.10), and multipliers will be updated as:

$$\lambda^{k+1}=\lambda^k+s^k\tilde{g}\left(x^k,y^k,z^k\right). \tag{3.75}$$

In the following Theorem 3.1.2, convergence will be established first by proving that solutions to the V-shape linearized problem (3.57)-(3.58) satisfy the condition (3.73), and then the proof will follow that of Theorem 3.1.2.

Theorem 3.2.3: Convergence of SALR+B&C. *Suppose that multipliers are updated per (3.75), surrogate optimality condition (3.73) is satisfied, stepsizes are set according to (3.9)-(3.10) and penalty coefficients are constant $c^k = c$. If α_k satisfies condition (3.11)-(3.12), then multipliers converge to λ_c^* that maximize the following dual function:*

$$q_c(\lambda) = \min_{x,y,z} L_c(x, y, z, \lambda),\ (x, y) \in \Omega \subset \mathbb{R}^p \times \mathbb{Z}^n,\ z \in \mathbb{R}^m,\ z \geq 0,\ \text{and } \lambda^T \in \mathbb{R}^m. \tag{3.76}$$

Proof By construction of V-shape function, solutions to (3.57)-(3.62) are the same as solutions to (3.40). Moreover, each monotonic segment of V-shape functions corresponds to a monotonic segment of the Augmented Lagrangian function (3.40) with respect to each variable. These features can be seen in Figure 3.2.1. This piece-wise monotonicity allows establishing the following. Suppose a solution (x^k, y^k, z^k) to (3.57)-(3.62) corresponds to a value that is lower than the value of the objective function evaluated at (x^{k-1}, y^{k-1}, z^{k-1}). Then, owing to the same monotonicity, the solution (x^k, y^k, z^k) will correspond to a value of (3.39) that is lower than the value of (3.39) at (x^{k-1}, y^{k-1}, z^{k-1}). Therefore, the surrogate optimality condition is satisfied. The rest of the proof follows that of Theorem 3.1.2. □

3.3. Combination of SALR and Branch-and-Cut for Block-Structured MILP Problems

Many large systems are created by connecting multiple subsystems through system-wide coupling constraints, and such systems are frequently formulated as "block-structured" [1] MILP whereby matrices *A* and *E* is singly bordered block diagonal [1] whereby *A* and *E* can be partitioned into several parts: A_0 and E_0, and A_i and E_i, $i = 1,\ldots,I$

$$A = \begin{pmatrix} A^0 & & & \\ A_1 & 0 & . & 0 \\ 0 & A_2 & . & 0 \\ . & . & . & . \\ 0 & 0 & . & A_I \end{pmatrix}, E = \begin{pmatrix} E^0 & & & \\ E_1 & 0 & . & 0 \\ 0 & E_2 & . & 0 \\ . & . & . & . \\ 0 & 0 & . & E_I \end{pmatrix}, \ A^0 = \begin{pmatrix} A_1^0 & \ldots & A_I^0 \end{pmatrix}, E^0 = \begin{pmatrix} E_1^0 & \ldots & E_I^0 \end{pmatrix}, \tag{3.77}$$

where A^0 and E^0 are full rank $m_0 \times p$ and $m_0 \times n$ matrices, A^0_j and E^0_j are $1 \times p$ and $1 \times n$ vectors, and A_i and E_i are $m_i \times p_i$ and $m_i \times n_i$ matrices respectively such that $n_1 + \ldots + n_I = n$, $p_1 + \ldots + p_I = p$. Accordingly, vectors b, d^x and d^y can be partitioned as:

$$b = \left(b^0, b_1, ..., b_I\right)^T, \; d^x = \left(d_1^x, ..., d_I^x\right) \text{ and } \; d^y = \left(d_1^y, ..., d_I^y\right), \tag{3.78}$$

where b^0 and b_i are $m_0 \times 1$ and $m_i \times 1$ column vectors. Therefore, constraints in (3.38) can be split into system-wide constraints

$$A^0 x + E^0 y \le b^0, \tag{3.79}$$

and subsystem constraints

$$A_i x_i + E_i y_i \le b_i, i = 1, ..., I \; . \tag{3.80}$$

Moreover, decision column vectors x and y can be partitioned into pair-wise disjoint sub-column-vectors stacked on top of one another: $x = (x_1, x_2, \ldots, x_I)$, and $y = (y_1, y_2, \ldots, y_I)$. The problem (3.38) can be represented as:

$$\min_{x,y} \left\{d^x x + d^y y\right\}, (x, y) \in \Omega \subset \mathbb{R}^p \times \mathbb{Z}^n. \tag{3.81}$$

$$s.t. \; A^0 x + E^0 y \le b^0, \tag{3.82}$$

$$A_i x_i + E_i y_i \le b_i, i = 1, ..., I \; . \tag{3.83}$$

To solve problem (3.81)-(3.83), the V-shape linearization will be used, and the resulting problem will be decomposed into smaller subproblems with reduced complexity as compared to the original problem. Subproblem i is created by fixing decision variables at values (x^{k-1}, y^{k-1}, z^{k-1}) obtained at previous iteration k-1 associated with blocks other than block i as:

$$\min_{x,y,z} \left\{ \sum_{i=1}^{p_i} a_i^x \left| x_i - \left[x_{\min,i}\right]_\Omega \right| + \sum_{i=1}^{n_i} a_i^y \left| y_i - \left[y_{\min,i}\right]_\Omega \right| + \sum_{i=1}^{m_0} a_i^z \left| z_i - \left[z_{\min,i}\right]_+ \right| \right\}, \; s.t., \; A_i x_i + E_i y_i \le b_i, i = 1, ..., I \; , (x, y) \in \Omega \subset \mathbb{R}_i^p \times \mathbb{Z}_i^n, \; z \in \mathbb{R}^{m_0}, z \ge 0, \text{ and } \lambda^T \in \mathbb{R}^{m_0}, \tag{3.84}$$

where

$$x_{i,\min} = -\frac{1}{2c^k \left(A_i^0\right)^T A_i^0}\left(c^k \sum_{j \neq i} \left(A_i^0\right)^T A_j^0 x_j^{k-1} + c^k \left(A_i^0\right)^T (E^0 y^{k-1} - b^0 + z^{k-1}) + \left(A_i^0\right)^T \lambda^k + d_i^x \right), \tag{3.85}$$

$$y_{i,\min} = -\frac{1}{2c^k \left(E_i^0\right)^T E_i^0}\left(c^k \sum_{j \neq i} \left(E_i^0\right)^T E_j^0 y_j^{k-1} + c^k \left(E_i^0\right)^T (A^0 x^{k-1} - b^0 + z^{k-1}) + \left(E_i^0\right)^T \lambda^k + d_i^y \right), \text{ and} \tag{3.86}$$

$$z_{i,\min} = -(A^0 x^{k-1} + E^0 y^{k-1} - b^0)_i \,. \tag{3.87}$$

Subproblem (3.84) is smaller in size, complexity is reduced as compared to that of the original problem and can be solved with drastically reduced effort. Moreover, positions of minima (3.85)-(3.87) can be obtained analytically thereby making computational improvements even more drastic. Since problem (3.81)-(3.83) is a special case of (3.38), under assumption that the surrogate optimality condition is satisfied, convergence results established in Theorem 3.2.3 also apply to the problem (3.81)-(3.83).

3.4. Key Steps, Implementation and Practical Considerations for MIP and MILP Problems

In subsection 3.4.1, key steps of algorithms for MIP and MILP problems will be provided. In subsection 3.4.2, a discussion about obtaining feasible solutions will be discussed. In subsection 3.4.3, difficulties associated with solving one subproblem at a time will be illustrated through a simple example. To resolve these difficulties, practical considerations to adaptively adjust penalty coefficients within the SALR+B&C method of subsection 3.3 for block-structured MILP problems will be developed. In subsection 3.4.4, the exploitation of the convex hull invariance to increase computational efficiency will be explained.

3.4.1. Key Steps of the Algorithm for MIP Problems

The key steps can be summarized as follows:

Step 0: Initialize multipliers λ^0,[9] solve the augmented relaxed problem (3.3) subject to the surrogate optimality condition (3.5) to obtain (x^0, y^0, z^0) and initialize stepsizes s^0.

Step 1: Update α_k, then for given values $(\alpha_k, x^k, y^k, z^k)$, update s^k according to (3.9)-(3.10), and update multipliers λ^{k+1} per (3.8).

Step 2: For given λ^{k+1}, solve the augmented relaxed problem (3.3) subject to the surrogate optimality condition (3.5).

Step 3: Check stopping criteria such as the CPU time, number of iterations, surrogate subgradient norm, etc. If satisfied, go to Step 5. Otherwise, go to Step 1.

Step 4: Search for feasible solutions. If a solution is found, go to Step 5. Otherwise, go to Step 1.

Step 5: Check duality gap. A duality gap can be calculated by using the best available feasible cost and the largest available dual value. If duality gap is satisfactory, then Stop. Otherwise go to Step 1.

The difference of the algorithm for MILP problems of Section 3.2 is that instead of the relaxed problem (3.3) subject to condition (3.5), the V-shaped linearized problem (3.57)-(3.62) subject to (3.73) is solved in Steps 0 and 2.

3.4.2. Obtaining Feasible Solutions MIP and MILP Problems

The process of obtaining feasible solutions is generally problem-dependent since each problem may have its own subsystem and constraint structures. When the relaxed problem or subproblems are solved, solutions are typically feasible with respect to the relaxed problems, but these solutions may not satisfy relaxed constraints. To satisfy these constraints, some or all integer decision variables are fixed at y_i^k, the

[9] Multiplier initialization is typically problem-dependent. For example, within Generalized Assignment problems, multipliers can be initialized following [67].

most recent values obtained by solving subproblems. This can be operationalized by creating simple constraints on some or all integer variables such as

$$y_i = y_i^k \tag{3.88}$$

The adjustment of continuous decision variables is operationalized by solving the original problem (3.38) subject to (3.88).

3.4.3. Practical Considerations of Block-Structured MILP Problems

Based on theoretical results of previous sections, SALR+B&C method converges assuming the surrogate optimality condition is satisfied. However, based on our testing experience, as penalty coefficients increase, the method may not converge because as levels of constraint violations reduce too fast, feasibility is emphasized and the optimality may be compromised. The reason for the lack of convergence is the violation of surrogate optimality condition as will be illustrated in the following example.

Example 3.4.1 (3.Example 3.2.1 continued) Consider the problem (3.71) with $\lambda^0 = 10$, and $c = 2000$ and suppose that one subproblem is solved at a time. Subproblem y can be written as

$$\min_{y,Q_y}\left\{\frac{c}{2}b\cdot Q_y\right\},\ s.t.\ -Q_y \le y - y^+_{\min}\left(\lambda, x^{k-1}\right) \le Q_y\,. \tag{3.89}$$

However, because y is discrete, the deviations from $y^+_{\min}\left(\lambda, x^k\right)$ are penalized so much that decision variable values cannot move from one value to another, and this implies that a solution y^k cannot be found that satisfies surrogate optimality condition as a strict inequality. Consequently, if the solution of y subproblem never updates, then the solution to the following x-subproblem

$$\min_{x,Q_x}\left\{\frac{c}{2}a\cdot Q_x\right\},\ s.t.\ -Q_x \le x - x^+_{\min}\left(\lambda, y^k\right) \le Q_x\,, \tag{3.90}$$

will not be updated and will not satisfy the surrogate optimality condition.

To satisfy all conditions for convergence within Theorems 3.1.2 and 3.2.3, surrogate optimality condition needs to be satisfied. If this condition is not satisfied after solving all subproblems, then penalty coefficients need to reduce. However, this may potentially take many iterations to find out. A simpler heuristic rule developed here requires that penalty terms are limited from above:

$$\frac{c}{2}\left\|A^0x+E^0y-b^0+z\right\|^2 \le M^c, \tag{3.91}$$

where M^c at every iteration can be selected as subproblem cost. At the same time it is desirable that penalty coefficients increase to sufficiently penalize constraint violations, and this can be ensured by the following inequality:

$$m^c \le \frac{c}{2}\left\|A^0x+E^0y-b^0+z\right\|^2, \tag{3.92}$$

where m^c has a value smaller than M^c. When c^k is large and (3.91) is violated, penalty coefficients should decrease

$$c^{k+1} = \beta c^k, \beta < 1. \tag{3.93}$$

When c^k is small and (3.92) is violated, constraint violations are not sufficiently penalized, and penalty coefficients should increase

$$c^{k+1} = \beta c^k, \beta > 1. \tag{3.94}$$

Parameters M^c, m^c and c^k should also be decreased when the surrogate optimality condition is violated:

$$M^{c,k+1} = \delta M^{c,k},\ m^{c,k+1} = \delta m^{c,k} \text{ and } c^{k+1} = \delta c^k. \tag{3.95}$$

Because of several differences in the implementation of SALR+B&C for block-structured MILP problems, key steps for the algorithm of Section 4 are provided separately below:

Step 0: Initialize λ^0, M^c, m^c solve the linearized subproblem i to obtain (x_i^0, y_i^0, z^0) and initialize s^0.

Step 1: Update α_k, then for given values $(\alpha_k, x_i^k, y_i^k, z^k)$, update s^k according to (3.9)-(3.10), and update multipliers λ^{k+1} per (3.75).

Step 2: Select the next subproblem (either sequentially or randomly) and solve it by using branch-and-cut with a MIP start.

Step 3: Check whether (3.93)-(3.94) are satisfied, and adjust c^{k+1} according to (3.93)-(3.94).

Step 4: Check stopping criteria. If satisfied, go to Step 4. Otherwise, go to Step 1.

Step 5: Search for feasible solutions . If a solution is found, go to Step 5. Otherwise, go to Step 1.

Step 6: Check duality gap. A duality gap can be calculated by using the best available feasible cost and the largest available dual value. If duality gap is satisfactory, then Stop. Otherwise go to Step 1.

3.4.4. Novel Exploitation of the Convex Hull Invariance

With drastically reduced complexity, subproblem "convex hulls" are much easier to obtain as compared to the "convex hull" of the original problem, and subproblems are handled locally without affecting the solution process of the entire problem. However, during the entire iterative process, subproblems need to be solved several times, and the overall computational effort may still be significant. With a novel observation, this difficulty will be alleviated by exploiting the fact that multipliers affect subproblem objective functions without affecting subproblem constraints or subproblem "convex hulls." Consequently, cuts remain valid throughout the entire iterative process. This invariant nature of subproblem convex hulls can be exploited by obtaining and keeping convex hulls. Once obtained, such invariant convex hulls can be kept and solving subproblems in subsequent iterations reduces to solving LP problems with ease. Even if the convex hull cannot be efficiently obtained, cuts generated by branch-and-cut can still remain valid and can be reused to significantly reduce computational effort involved in solving subproblems. To justify this claim, discussion on the performance of cuts is important. In particular, cuts generated by branch-and-cut when solving the original problem ("whole-problem cuts") will be discussed and compared with cuts obtained by branch-and-cut when solving subproblems ("subproblem cuts") in the following subsection 3.5.

3.5. Comparison of Subproblem and Whole-Problem Cuts

The aim of this section is to compare subproblem and whole-problem cuts, and this is achieved by considering an important class of cuts – Gomory cuts. First, after the general idea behind creation of Gomory cuts is presented, the Theorem comparing subproblem Gomory cuts and whole-problem Gomory cuts will be formulated in subsection 3.5.1, and conclusions about cuts of other types will be drawn. Then, the intuition behind Gomory cuts supporting the Theorem will be presented in subsection 3.5.2.

3.5.1. Comparison of Subproblem and Whole-Problem Gomory Cuts

The general goal behind creating cuts is to obtain facets of the convex hull by cutting off LP regions enclosed by constraints (38) without cutting off feasible solutions. Ideas behind Gomory cuts are rotation of hyperplanes corresponding to constraints (38) to make them parallel to the facets of the convex hull, and shifting of these planes to move them as close to the facets as possible [68-71]. Within fractional Gomory cuts [72], to improve flexibility of rotation and shifting, aggregation through a linear combination of constraints (38) by using real-valued vectors μ is also performed. However, rotation, shifting and aggregation by themselves may not be efficient to obtain facets of the convex hull. To improve efficiency, so-called "disjunctive cuts" are created. The idea is to first remove LP regions inside the LP polyhedron enclosed by constraints (38) thereby separating an entire LP space into two disjoint sets. This procedure enables more aggressive rotation and shifting of cuts based on each of the sets independently thereby leading to even higher flexibility of rotation and shifting, and allowing cutting larger LP regions. Disjunctive cuts are then created by taking a linear combination of the two sets. Other types of Gomory cuts, considered in [72], are more specific versions of fractional Gomory cuts, and their construction follows the same logic as provided above.

Subproblem Gomory cuts that are generated by using branch-and-cut when solving subproblems are based on first aggregating constraints (80) for a particular subproblem i, and then by using rotation, shifting and disjunction as explained in the previous paragraph. Since subproblems are solved without

system-wide coupling constraints (79), for the comparison between subproblem and whole-problem Gomory cuts constraints (79) will be excluded from consideration. Therefore, the following assumption will be used:

Assumption 3.5.1. Without considering system-wide coupling constraints, coefficients of aggregation for whole-problem Gomory cuts are: $\mu = (0, \mu_1, \ldots, \mu_I)$.

In the following, after defining areas that are cut off by whole-problem and subproblem cuts outside subproblem convex hull for particular and for all possible coefficients of aggregation μ, the Theorem that compares these areas will then be formulated.

Definition 3.5.2. Whole-Problem Cuts.

Let coefficients of aggregation for whole-problem Gomory that satisfy Assumption 3.5.1 be μ=(0, μ_1, …, μ_I).

Let $\omega = (\omega_1, \ldots, \omega_I)$, where ω_i is an area outside subproblem i convex hull that is cut off by a whole-problem Gomory cut for a particular μ.

Let $\Omega = (\Omega_1, \ldots, \Omega_I)$, where Ω_i is the total area outside subproblem i convex hull that is cut off by whole-problem Gomory cuts, which is a union of ω_i for all possible values of μ.

Definition 3.5.3. Subproblem Cuts.

Let coefficients of aggregation for subproblem i Gomory cuts be μ_i.

Let ξ_i be an area outside subproblem i convex hull that is cut off by a subproblem i Gomory cut for a particular μ_i.

Let Ξ_i is the total area outside subproblem i convex hull that is cut off by subproblem i Gomory cuts outside subproblem i convex hull, which is a union of ξ_i for all possible values of μ_i.

Intuitively, flexibility of rotation and shifting of whole-problem Gomory cuts is greater as compared to subproblem Gomory cuts thereby generally leading Ω_i to be larger than Ξ_i. Under an additional condition on a particular vector μ, regions ω_i are equal to ξ_i as will be stated in the following Theorem.

Theorem 3.5.4. Comparison of Subproblem and Whole-Problem Gomory Cuts.

The area Ξ_i is a subset of Ω_i.

Moreover, if for a particular i, μ_i and μ the following condition holds:

$$\left\lfloor \sum_{j=1}^{I} \mu_j b_j - \lfloor \mu_i b_i \rfloor \right\rfloor = 0, \tag{3.96}$$

then areas ω_i and ξ_i are equal.

Proof of the Theorem is given in Appendix A.

In practice, there are several ways to select μ [69. 73]. By the Theorem above, while for all possible μ, whole-problem Gomory cuts are more efficient as compared to subproblem Gomory cuts, practical performance of cuts is roughly equivalent.

Remark 3.5.5. If the convex hull of subproblem i can be obtained, the area Ξ_i is exactly the same as Ω_i.

Corollary 3.5.6. The Theorem holds for other cuts that require aggregation of constraints.

Proof: Proof follows the same logic as that of the Theorem 3.5.4.

Remark 3.5.7. For cuts that are based on individual subproblem constraints without using aggregation, subproblem and whole-problem cuts are equivalent.

Examples below present rotation, shifting, disjunction, and aggregation thereby intuitively presenting ideas behind the Theorem.

3.5.2. Illustration of Theorem 3.5.4

Example 3.5.1: For illustration purposes, consider the following constraint based on (3.38):

$-3.5x_1 - 6x_2 \le -2.5$, $x_1, x_2 \in \{0,1\}$, or in terms of (3.38), $Ax \le b$, with A=(–3.5,–6), E=0, b=–2.5. (3.97)

Figure 3.5.2.a shows feasible points, the LP region shaded grey, and the convex hull enclosed by thick red solid lines. The plane defined by (3.97) can be rotated by changing slopes of x_1 and x_2 while not cutting off any portion of the convex hull. One way to perform this is by truncating coefficients of A by using a "floor" ($\lfloor\_\rfloor$) operator. The resulting plane becomes (Figure 3.5.2.b):

$-4x_1 - 6x_2 \le -2.5$, or in terms of (3.38), $\lfloor A \rfloor x \le b$. (3.98)

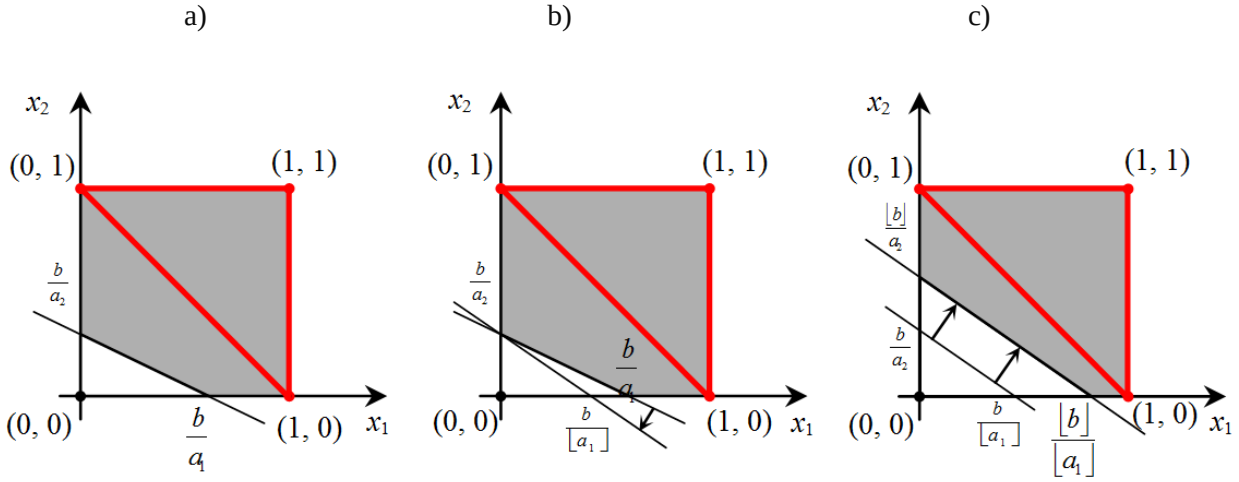


Figure 3.5.2. a) Original constraint (3.97) and LP region that satisfies (3.97), b) Cut (3.98) obtained by rotation, c) Cut (99) obtained by rotation and shifting. Convex hull is marked by red solid thick lines.

Since rotation is achieved by truncating the left-hand side thereby making it smaller, inequality (3.98) is valid for a general problem. For this particular example, the cut (3.98) resulting from the rotation forms a smaller angle with the convex hull facet that connects points (0, 1) and (1, 0), but does not become parallel to the facet.

For integer solutions, the left-hand side of (3.98) is integer while the right-hand side is fractional thereby leading to a fractional slack. This slack is removed by truncating the right-hand side of (3.98) thereby shifting the cut closer to the convex hull as shown in Figure 2.c, and the resulting cut becomes:

$-4x_1 - 6x_2 \le -3$, or in terms of (3.38), $\lfloor A \rfloor x \le \lfloor b \rfloor$. (3.99)

The amount of rotation and shifting achieved by truncation is predetermined by *A* and *b*, respectively, and is limited. To improve flexibility of rotating and shifting, the entire equation (3.38) can be pre-multiplied by a real-valued vector μ before performing rotation and shifting. In this particular example, equation (3.99) is pre-multiplied by 1/6:

$$-\frac{4}{6}x_1 - x_2 \le -\frac{3}{6}. \quad (3.100)$$

Because the floor operator is not an affine operator, truncation of the left-hand side of (3.100) leads to a cut:

$$-x_1 - x_2 \le -\frac{3}{6}, \tag{3.101}$$

which generally is not equivalent to (3.99). As a result, the angle of rotation is different from that of (99) as shown in Figure 3.5.3.b. Similarly, the truncation of the right-hand side of (101) leads to shifting by a different amount as compared to (3.99):

$$-x_1 - x_2 \le -1. \tag{3.102}$$

Generally, the final inequality based on (3.38) becomes $\lfloor \mu A \rfloor x \le \lfloor \mu b \rfloor$ when E=0. In this example, inequality (3.102) cuts off larger amounts as compared to (3.99). Moreover, it defines a facet of the convex hull as shown in Figure 3.5.3.c.

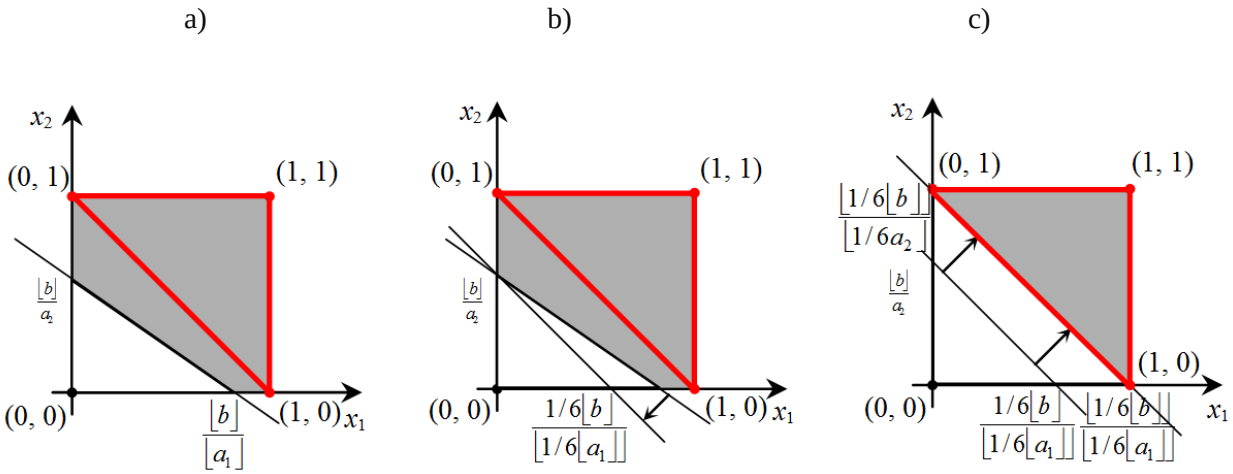


Figure 3.5.3. *a*) Constraint (3.100) and the corresponding LP region, *b*) Cut (101) obtained by rotation, *c*) Cut (3.102) obtained by rotation and shifting. Convex hull is marked by red solid thick lines.

The convex hull is obtaied by using a multiplier 1/6. In the following, disjunctive cuts will be presented and the facet-defining cut (3.102) will be obtained without using multipliers. The idea to obtain the convex hull is to first remove the LP region $x_2 \in (0,1)$ without removing feasible points thereby

creating a union of two disjoint sets shown in Figure 3.5.4.b. Then, cuts are aggressively rotated and shifted with respect to each set, and in this example such cuts define integer vertices of each set are defined as shown in Figure 3.5.4.c. Then the resulting sets are linked, and in this example vertices of each set are connected thereby defining the facet of the convex hull as shown in Figure 3.5.4.d.

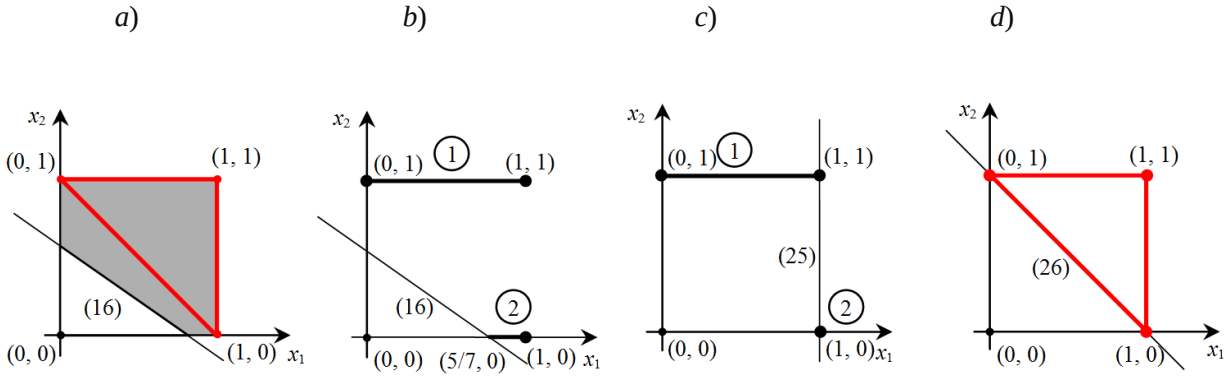


Figure 3.5.4. *a*) Constraint (3.97), the corresponding LP region, and the convex hull, *b*) Constraint (3.97) and disjoint LP regions satisfying (3.97) and (3.103) – (3.104), *c*) Definition of an integer vertex satisfying $x_2 \le 0$, *d*) Disjunctive cut (3.107). Convex hull is marked by red solid thick lines.

Mathematically, LP region is removed by using the following disjunction:

$$x_2 \ge 1 . \tag{3.103}$$

OR

$$x_2 \le 0 . \tag{3.104}$$

Two LP regions satisfying (3.97) and (3.103) – (3.104) are marked in Figure 4.b by thick solid lines. The first region includes two feasible points (0,1), (1,1) and a line segment connecting them. This region is a convex hull of the points (0,1) and (1,1). The second region includes points (5/7,0), (1,0) and a line segment connecting them. The convex hull of feasible points satisfying (3.104) consist of the single point (1,0). The LP region outside this convex hull can be cut off by aggressively rotating and shifting of (3.97) without worrying about the first set described by $x_2 \ge 1$. The rotation is achieved by eliminating x_2 from (3.97) by using $x_2 \le 0$ from (3.104):

$$-3.5x_1 \le -2.5 \Rightarrow -x_1 \le -\frac{5}{7}; \tag{3.105}$$

and shifting is achieved by applying the floor operator ($\lfloor\_\rfloor$) to the right-hand side of (105):

$$x_1 \ge 1. \tag{3.106}$$

Finally, integer vertices of each set are connected. In particular, a line connecting vertices (0,1) and (2.1,0) is a linear combination of them and is described by $x_1 + x_2$. Therefore, all the points satisfy the following cut:

$$x_1 + x_2 \ge 1, \tag{3.107}$$

which is exactly the same facet-defining cut as (3.102).

3.6. Numerical Testing

The new method is implemented in CPLEX 12.6.0, and is tested on an Intel® Xeon® CPU E5620, with 12M Cache, two 2.40 GHz processors, 36.00 GB of RAM, and Windows 7. In Example 3.6.1, it is demonstrated that the surrogate Augmented Lagrangian relaxation (SALR) method converges faster as compared to the standard subgradient method. In Example 3.6.2, by considering generalized assignment problems it is demonstrated that the combination of surrogate Lagrangian relaxation and branch-and-cut with the exploitation of subproblem convex hull invariance is more computational efficient as compared to the standard branch-and-cut. In Example 3.6.3, by considering unit commitment problems with combined cycle units it is demonstrated that SALR+B&C is more computationally efficient as compared to standard branch-and-cut.

Example 3.6.1. A small example with inequality constraints. The purpose of this example is to demonstrate the convergence of the surrogate Augmented Lagrangian relaxation method compared to that of the subgradient-level method. To achieve this goal, the following small and relatively simple integer nonlinear programming example subject to linear inequality constraints is considered:

$$\min_{\{x_1,x_2,x_3,x_4,x_5,x_6\}\in Z_+\cup\{0\}} \left\{0.5x_1^2 + 0.1x_2^2 + 0.5x_3^2 + 0.1x_4^2 + 0.5x_5^2 + 0.1x_6^2\right\} \tag{3.108}$$

$$s.t.\ \ 48 - x_1 + 0.2x_2 - x_3 + 0.2x_4 - x_5 + 0.2x_6 \le 0\,,\ 250 - 5x_1 + x_2 - 5x_3 + x_4 - 5x_5 + x_6 \le 0. \tag{3.109}$$

Within the SALR, after the relaxation of constraints (3.109), penalization of constraint violations, and performing V-shape linearization, one subproblem is solved at a time. As a result, multiplier-updating directions are fairly smooth and the method converges fast without much of zigzagging as shown in Figure 3.6.1.

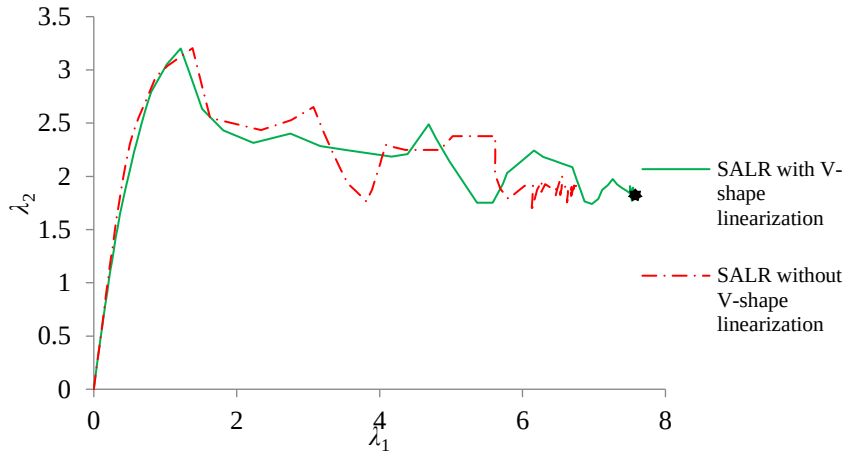


Figure 3.6.1. Trajectories of multipliers for Example 3.6.1. Optimum is shown by a star.

As demonstrated in Figure 3.6.1, within the SALR without V-shape linearization, multipliers zigzag more and multipliers do not converge because the satisfaction of the surrogate optimality condition may not be guaranteed.

In Figure 3.6.2, the comparison of SALR with the subgradient level method will be demonstrated. Because within the subgradient-level method all subproblems need to be solved, subgradient direction change drastically and multipliers zigzag. Moreover, slow convergence is exacerbated because of the need to adaptively adjust the estimate of the optimal dual value.

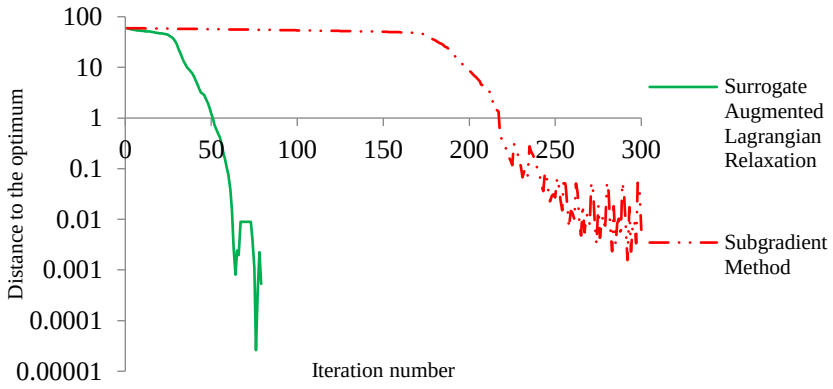


Figure 3.6.2. Comparison of convergence of SALR and the subgradient-level method.

Example 3.6.2. Generalized Assignment Problems. The purpose of this example is to demonstrate the exploitation of the subproblem convex hull invariance within the combination of the SLR method with branch-and-cut. The method is implemented in CPLEX 12.6.0 by using C/C++ interfaces within Microsoft Visual Studio 2013. CPLEX C Concert Technology is used to extract, save and load CPLEX-generated cuts through the use of callable libraries. By considering randomly generated problem instances, robustness of the combination of SALR with branch-and-cut and the combination of SLR with branch-and-cut will be tested by solving respective subproblems by using CPLEX. The problem will then also be solved by using SALR+B&C, and performance will be compared with the standard branch-and-cut.

Problem Formulation. Generalized Assignment Problems minimize the total cost for assigning given jobs to available machines. Each job is assigned to one machine, and the total time required by all jobs assigned to a machine should not exceed the machine's time available [1, 20, 65-67]. The problem is formulated in the following way:

$$\min_{x_{i,j}} \sum_{i=1}^{I}\sum_{j=1}^{J} g_{i,j}x_{i,j}\ , x_{i,j} \in \{0,1\},\ g_{i,j} \geq 0,\ a_{i,j} \geq 0,\ b_j \geq 0\ , \tag{3.110}$$

$$s.t.\ \sum_{i=1}^{I} a_{i,j}x_{i,j} \leq b_j\ , j = 1,...,J\ , \tag{3.111}$$

$$\sum_{j=1}^{J} x_{i,j} = 1, i = 1,...,I\ , \tag{3.112}$$

where I is the number of jobs and J is the number of machines, $a_{i,j}$ is time required by job i to be performed on machine j and $g_{i,j}$ is cost for assigning job i to machine j. Capacity constraints (3.111) ensure that the total amount of time required by the jobs to be performed on machine j does not exceed its time available b_j. Assignment constraints (3.112) ensure that each job is to be performed on exactly one machine.

Exploitation of subproblem convex hull invariance within the combination of SLR and branch-and-cut. After relaxing system-wide coupling assignment constraints and decomposing the resulting relaxed problems, a subproblem j can be written as:

$$\min_{x_{i,j}} \left\{ \sum_{i=1}^{I} g_{i,j} x_{i,j} + \sum_{i} \lambda_i x_{i,j} \right\}, \; x_{i,j} \in \{0,1\}, \; g_{i,j} \geq 0, \; a_{i,j} \geq 0, \; b_j \geq 0 . \tag{3.113}$$

$$s.t. \quad \sum_{i=1}^{I} a_{i,j} x_{i,j} \leq b_j . \tag{3.114}$$

In an ideal situation, subproblem convex hulls are much simpler than the convex hull of the original problem, and they can be efficiently obtained and kept. In this situation, there is little or no overhead and all the cuts can be reused. In practice, however, when using CPLEX, the overhead that is involved in accessing all the cuts that are generated by branch-and-cut may be large. This is because many of the things that made our implementation possible needed to be buildup on top of the existing callable libraries. However, because CPLEX is not open, this implementation creates an overhead.

Exploitation of the convex hull invariance. In an ideal situation, subproblem convex hulls are much simpler than the convex hull of the original problem, and they can be efficiently obtained and kept. In this situation, there is little or no overhead and all the cuts can be reused. In practice, however, when using CPLEX, the overhead that is involved in accessing all the cuts that are generated by branch-and-cut may be large. This is because many of the things that made our implementation possible needed to be buildup on top of the existing callable libraries. However, because CPLEX is not open, this implementation creates an overhead.

To demonstrate how retained cuts affect the CPU time, cuts are saved and retained every N (=1, 2, …, 15) iterations (one iteration is complete after solving all the subproblems exactly once). The results are summarized in Figure 3.6.3.

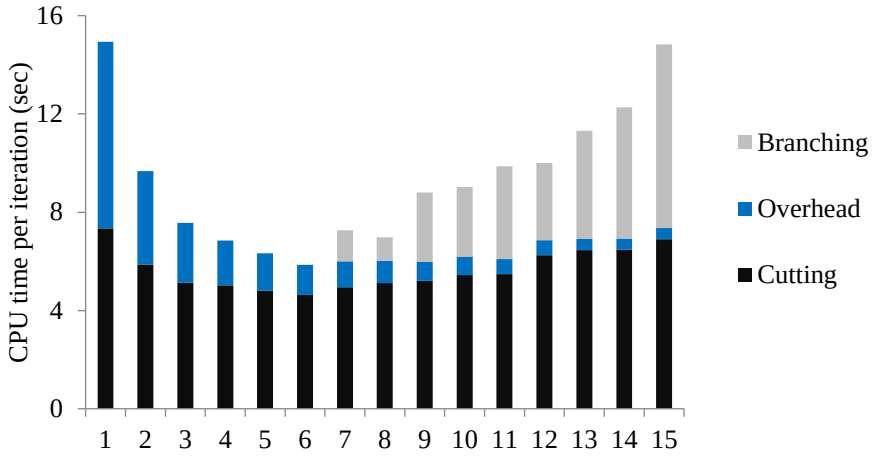


Figure 3.6.3. CPU time per iteration for the Generalized Assignment Problem with 20 machines and 1600 jobs (d201600) depending on the frequency of generating/retaining of cuts

Figure 3.6.3 provides a numerical validation of the idea that with cuts saved and retained, solving subproblems is still much easier than starting from scratch because presumably the LP region enclosed by subproblem constraints and retained cuts is smaller as compared to the LP region enclosed by subproblem constraints without cuts. However, as can be seen from Figure 3.6.3, when cuts are saved less frequently, the benefit of saved cuts is reduced, and to solve the problem, branching operations are used.

Efficiency of SALR+B&C. To demonstrate the efficiency of SALR+B&C, problem instances of category D from OR-Library [2, 21, 64] will be considered. Specifically, two large instances with 20 machines and 1600 jobs (d201600), and with 80 machines and 1600 jobs (d801600) will be tested. Figures 4 and 5 provide a solution progress chart versus total running time within the combination of SALR+B&C and within standard branch-and-cut.

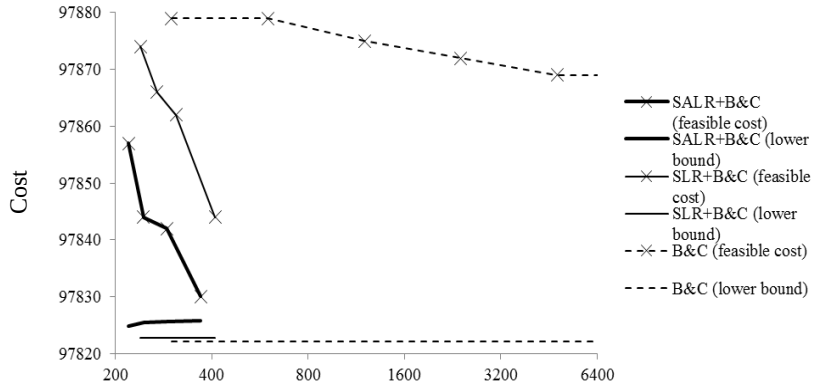


Figure 3.6.4. Comparison of SALR+B&C against the combination of SLR and branch-and-cut and standard branch-and-cut for the Generalized Assignment problem d201600 with 20 machines and 1600 jobs

As shown in Figure 3.6.4, the combination of SLR and branch-and-cut obtains solutions with better feasible costs as compared to standard branch-and-cut. Moreover, SALR+B&C further improves the quality of solutions, and the best cost obtained within the method after further improves the quality of solutions, and the best cost obtained within the method after 370 seconds is 97830, which is better as compared 97837, which, to the best of our knowledge, is the smallest value reported in the literature ([2] and [21]).

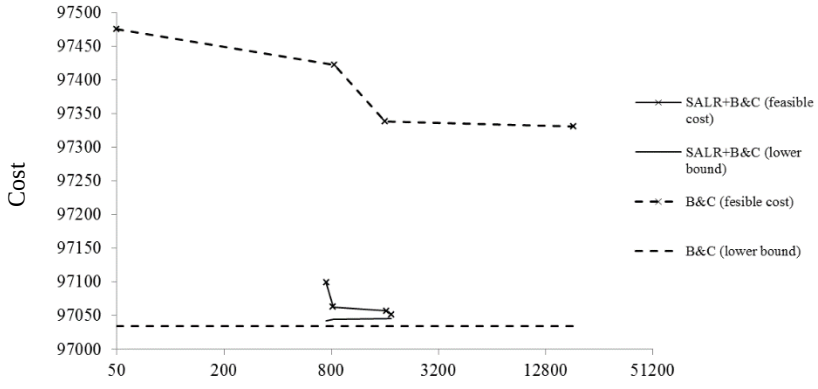


Figure 3.6.5. Comparison of SALR+B&C against the combination of SLR and branch-and-cut and standard branch-and-cut for the Generalized Assignment problem d801600 with 80 machines and 1600 jobs

The problem instance with 80 machines and 1600 jobs is the largest instance from OR-Library [64]. Because of the difficulties explained earlier, feasible solutions obtained by standard branch-and-cut are worse in quality and in terms of running time as compared to the combination of surrogate Lagrangian relaxation and branch-and-cut as demonstrated in Figure 3.6.5. The best feasible cost obtained within SALR+B&C is 97052, which matches the best result obtained in [2] and [21].

Robustness of SALR+B&C. To test robustness, 30 Monte Carlo simulations are performed after slightly perturbing parameters b_j for the problem instance with 20 machines and 1600 jobs (d201600). The stopping criterion is a 0.05% gap.

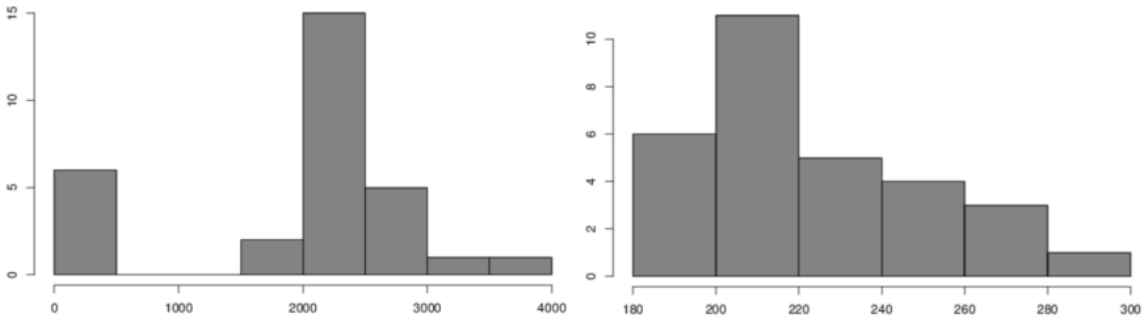


Figure 3.6.6. Histogram showing performance of a) standard branch-and-cut, b) SALR+B&C for solving GAP with 20 machines and 1600 jobs

As shown in Figure 3.6.6a, 6 instances out of 30 were solved within 500 seconds, indicating that on occasion branch-and-cut can obtaining good cuts and solve the problem fast. However, for the remaining 24 instances, the CPU time was at least 1500 seconds, and it was as high as 4000 seconds indicating that

for most problems, standard branch-and-cut was not able to solve the problem through cutting, and frequently large number of branching operations were required thereby leading to poor performance.

To test SALR+B&C, the stopping criterion was chosen 0.025% Gap, and the results are shown in Figure 3.6.6b. As shown in Figure 3.6.6b, all the instances were solved with the desired accuracy within 300 seconds, and the average time spent peaks abound 210 seconds.

Example 3.6.3. Unit Commitment and Economic Dispatch with Combined Cycle Units. Unit Commitment and Economic Dispatch (UCED) minimizes the total generation cost associated with independent generators to meet the system demand while satisfying generator capacity, ramp-rate and minimum up- and down-time constraints [7-8] by determining which generators to commit and deciding their generation levels. In a sense, system demand constraints are system-wide and coupling constraints with respect to individual generators. While combined cycle units are more efficient as compared to conventional units, transitions among states of a combined cycle unit complicate UCED problems [69-70] and pose major computational challenges for branch-and-cut. A detailed description of the problem formulation including modeling combined cycle transitions and their linearization can be found in [42-43]. For simplicity, transmission capacity constraints are not considered.

Performance of SALR+B&C. Performance of SALR+B&C will be demonstrated by using unit commitment problems with 300 conventional and 40 combined cycle units. Within SALR+B&C, the choice of the penalty parameters and stepsizes is independent. Therefore, penalty parameters can be increased more aggressively thereby efficiently penalizing constraint violations and leading to a fast and stable convergence. SALR+B&C will be tested using three different parameters of c^0: 10^{-3}, 10^{-5}, and 10^{-7}. Parameters M^c and m^c will be chosen to be 10^6 and 10^4, respectively. The parameter δ in (95) is chosen to be 10^{-1}, and the parameter β in (93)-(94) is chosen to be 1.05. The CPU time stopping criteria are chosen to be 10 and 20 minutes. The results are summarized in Table 3.6.1.

Table 3.6.1. Results for SALR+B&C for unit commitment with 300 conventional and 40 combined cycle unit after 10 and 20 minutes of CPU time

CPU time							
	20 minutes			10 minutes			
c^0	Feasible Cost	Lower Bound	Gap (%)	Feasible Cost	Lower Bound	Gap (%)	Value of c^k at convergence
10^{-3}	8,634,154	8,594,001	0.465	8,731,362	8,594,001	1.5731	0.00638
10^{-5}	8,635,736	8,604,477	0.3619	8,774,375	8,583,555	2.1747	0.01020
10^{-7}	8,644,031	8,592,093	0.6008	8,771,324	8,581,282	2.1666	0.00614

The robustness of SALR+B&C will also be tested against M^c and m^c introduced in (46). The method will be tested using five different parameters of m^c: 10^3, 10^4, 10^5, 10^6, and 10^7. Parameter c^0 is chosen to be 10^{-6}. The parameter δ in (94) is chosen to be 10^{-1} and the parameter β in (93)-(94) is chosen to be 1.05. The CPU time stopping criteria are chosen to be 10 and 20 minutes. Table 3.6.2 summarizes the results.

Table 3.6.2. Results for SALR+B&C for unit commitment with 300 conventional and 40 combined cycle unit after 10 and 20 minutes of CPU time

	CPU time					
	20 minutes			10 minutes		
m^c	Feasible Cost	Lower Bound	Gap (%)	Feasible Cost	Lower Bound	Gap (%)
10^3	8,647,352	8,575,445	0.8315	8,755,807	8,568,757	2.1363

10^4	8,635,736	8,604,477	0.3619	8,774,375	8,583,555	2.1747
10^5	8,637,736	8,591,635	0.5337	8,661,205	8,591,635	0.8032
10^6	8,638,326	8,616,888	0.2481	8,686,405	8,616,888	0.8003
10^7	8,657,772	8,615,513	0.4881	8,698,247	8,615,513	0.9511

When m^c is small (10^3-10^4), the penalty coefficients increase slowly. As a result, constraint violations decrease slowly because they are not sufficiently penalized. As a result convergence is slower, and this leads to higher duality gaps.

Comparison with branch-and-cut. To compare SALR+B&C with branch-and-cut, the best results obtained by each method are selected.

Table 3.6.3. Comparison of SALR+B&C, and standard branch-and-cut after 20 minutes

SALR+B&C			B&C		
Feasible Cost ($)	Lower Bound ($)	Gap (%)	Feasible Cost ($)	Lower Bound ($)	Gap (%)
8,635,736	8,604,477	0.36	N/A	8,077,345	N/A

As demonstrated in Table 3 and also in Figure 7, after 20 minutes, SALR+B&C obtained a solution with a corresponding duality gap 0.36%. Standard branch-and-cut was unable to obtain any feasible solution within 20 minutes.

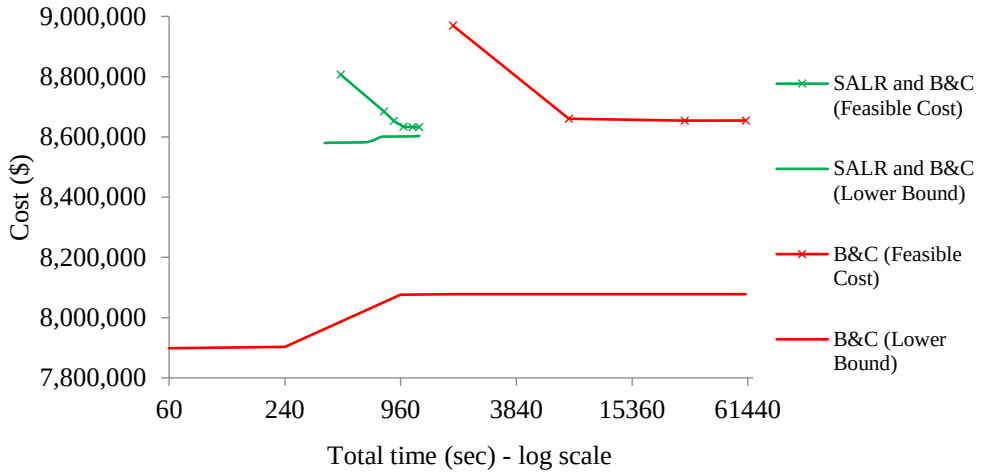
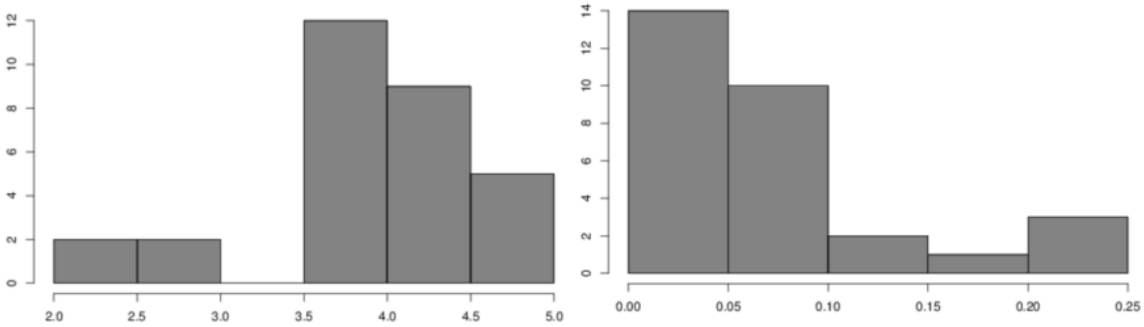


Figure 3.6.7. Comparison of SALR+B&C and standard branch-and-cut

Robustness of SALR+B&C. To test robustness, unit commitment problem with 300 conventional and 20 combined cycle units is considered and 30 Monte Carlo simulations are performed after slightly perturbing system demand for each hour, and the stopping criterion is a 10 minutes within SALR+B&C. For standard branch-and-cut, the stopping criterion is 1 hour. The robustness results are shown in Figure 3.6.8.



MIP gap (%) Duality gap (%)

Figure 3.6.8. Histogram showing performance of a) standard branch-and-cut, b) SALR+B&C for solving unit commitment problem with 300 conventional and 20 combined cycle units

As shown in Figure 3.6.8, the SALR+B&C method is very robust and consistent because for all the instances considered, the duality gap is never above 0.25%. In contrast, performance of branch-and-cut is

significantly affected by the initial data, and for the same set of problems, the MIP gap obtained is within the much wider range from 2% to 5%.

3.6. Conclusions

In this section, to solve MIP and MILP problems, a novel solution methodology is developed based on our recent Surrogate Lagrangian Relaxation with major convergence improvements following Augmented Lagrangian relaxation. When specializing to MILP problems, to preserve fast convergence while exploiting linear solvers, the novel V-shape linearization scheme is introduced. When further specializing to block-structured MILP, subproblem solutions can be obtained analytically, and these solutions are effectively coordinated. When solving large-scale problems for which facet-defining cuts are difficult to obtain, the new method is robust and much more efficient as compared to frequently-used branch-and-cut and our recent surrogate Lagrangian relaxation. As a future direction, since subproblem convex hulls are much simpler than that of the original problem, if tight subproblem formulations can be obtained, solving subproblems will be possible without cutting and branching operations. Then with effective coordination of subproblem solutions by SALR, our capabilities to solve difficult MIP and MILP problems will be advanced in a major way.

References

1. Giovanni, F. and Gentile, G.: Zero-lifting for integer block structured problems. Journal of combinatorial optimization 7(2) 161-167 (2003)
2. Posta, M., Ferland, J. A., and Michelon, P.: An exact method with variable fixing for solving the generalized assignment problem. Comput. Optim. Appl. 52(3), 629–644 (2012)
3. Fisher, M. L.: The Lagrangian relaxation method for solving integer programming problems. Management Sci. 27(1), 1-18 (1981)
4. Nagy, G., and Salhi, S.: Location-routing: Issues, models and methods. Eur. J. Oper. Res. 177(2), 649-672 (2007)

5. Prins, C., Prodhon, C., Ruiz, A., Soriano, P., and Wolfler Calvo, R.: Solving the capacitated location-routing problem by a cooperative Lagrangean relaxation-granular tabu search heuristic. Transp. Sci. 41(4), 470-483 (2007)
6. Belenguer, J.-M., Benavent, E., Prins, C., Prodhon, C., and Wolfler-Calvo, R.: A branch-and-cut algorithm for the capacitated location routing problem. Comp. Oper. Res. 38(6), 931–941 (2010)
7. Kohl, N. and Madsen, O. B. G.: An Optimization Algorithm for the Vehicle Routing Problem with Time Windows Based on Lagrangian Relaxation. Oper. Res. 45, 395–406 (1997)
8. Guan, X., Luh, P. B., Yan, H. and Amalfi, J. A.: An optimization-based method for unit commitment. Int. J. Elec. Power Energy Syst. 14(1), 9–17 (1992)
9. Guan, X., Luh, P. B., Yan, H., and Rogan, P. M.: Optimization-based scheduling of hydrothermal power systems with pumped-storage units. IEEE Trans. Power Syst. 9(2), 1023-1031 (1994)
10. Zhuang, F. and Galiana, F.D.: Towards a More Rigorous and Practical Unit Commitment by Lagrangian Relaxation. IEEE Trans. Power Syst., 3, 763–773 (1988)
11. Zhai, Q. and Guan, X.: Unit commitment with identical units: Successive subproblem solving method based on Lagrangian relaxation. IEEE Trans. Power Syst., 17(4), 1250–1257 (2002)
12. Weerakorn, O., and Petcharaks, N.: Unit commitment by enhanced adaptive Lagrangian relaxation. IEEE Trans. Power Syst., 19(1), 620-528 (2004)
13. Jimenez, N., and Conejo, A.: Short-Term Hydrothermal Coordination by Lagrangian Relaxation: Solution of the Dual Problem. IEEE Trans. Power Syst., 14, 89–95 (1999)
14. Virmani, S., Adrian, E. C., Imhof, K. and Mukherjee, S.: Implementation of a Lagrangian relaxation based unit commitment problem. IEEE Trans. Power Syst., 4(4), 1373-1380 (1989)
15. Padberg, M., and Rinaldi, G.: A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. SIAM Review 33(1), 60-100 (1991)
16. Hoffman, K. L., and Padberg, M.: Solving airline crew scheduling problems by branch-and-cut. Mgmt. Sci. 39(6), 657-682 (1993)

17. Balas, E., Ceria, S., and Cornuéjols, G.: Mixed 0-1 Programming by lift-and-project in a branch-and-cut framework. Mgmt. Sci. 42(9), 1229-1246 (1996)
18. Everett, H.: Generalized Lagrange multiplier method for solving problems of optimum allocation of resources. Oper. Res. 11(3), 399-417 (1963)
19. Held, M., and Karp, R. M.: The traveling salesman problem and minimum spanning trees. Oper. Res. 18(6), 1138-1162 (1970)
20. Held, M., and Karp, R. M.: The traveling salesman problem and minimum spanning trees: Part II. Math. Program. 1(1), 6-25 (1971)
21. Fisher, M. L.: Optimal solution of scheduling problems using Lagrange multipliers: Part I. Oper. Res. 21(5), 1114-1127 (1973)
22. Geoffrion, A. M.: Lagrangean relaxation for integer programming, Springer Berlin Heidelberg 82-114 (1974)
23. Özyurt, Z., and Aksen, D.: Solving the multi-depot location-routing problem with Lagrangian relaxation. Extending the horizons: Advances in computing, optimization, and decision technologies. Springer US, 125-144 (2007)
24. Avella, P., Boccia, M., and Vasilyev, I.: A computational study of exact knapsack separation for the generalized assignment problem. Comput. Optim. Appl. 45(3), 543–555 (2010)
25. Balas, E., Ceria, S., and Cornuejols, G.: A Lift-and-Project Cutting Plane Algorithm for Mixed 0-1 Programs Math. Program. 58, 295-324 (1993)
26. Balas, E.: Disjunctive Programming. Annals of Discrete Mathematics, 5, 3-51 (1979)
27. Balas, E.: Recent Advances in Lift-and-Project, Technical Report, GSIA, Carnegie Mellon University (1997)
28. Balas, E. and Jeroslow, R.G.: Strengthening Cuts for Mixed Integer Programs. Eur. J. Oper. Res 4, 224–234 (1980)

29. Ceria, S. and Pataki, G.: Solving Integer and Disjunctive Programs by Lift-and-Project. In R.E. Bixby, E.A. Boyd, and R.Z. Rios-Mercado (eds.), IPCO VI, Lecture Notes in Computer Science, 1412. Springer, 271–283 (1998).
30. Balas, E.: A modified lift-and-project procedure. Math. Program. 79, 19–32 (1997)
31. Balas, E., and Perregaard, M.: Lift-and-project for mixed 0-1 programming: recent progress. Discrete Applied Mathematics, 123, 129–154 (2002)
32. Giandomenico, M., Rossi, F., and Smriglio. S.: Strong lift-and-project cutting planes for the stable set problem. Math. Program. Ser. A, 141(1–2), 165–192 (2013)
33. Ceria, S., Cornuéjols, G., and Dawande, M.: Combining and strengthening Gomory cuts. E. Balas and J. Clausen, eds. Proc. 4th IPCO Conference, Copenhagen, Denmark. Springer-Verlag, 438-451, (1995)
34. Caprara, A., and Fischetti, M.: {0,1/2}-Chvátal-Gomory cuts. Math. Program. 74(3), 221-235 (1996)
35. Eisenbrand, F.: Gomory-Chvátal cutting planes and the elementary closure of polyhedra. Doctoral Dissertation, Universität des Saarlandes, http://www2.math.uni-paderborn.de/fileadmin/Mathematik/AG-Eisenbrand/publications/diss.pdf (2000)
36. Aliev, I., and Letchford, A. N.: Iterated Chvátal-Gomory cuts and the geometry of numbers. arXiv preprint arXiv:1306.6031 (2013)
37. Gomory, R. E.: An algorithm for the mixed integer problem (No. RAND-P-1885). RAND Corp, Santa Monica, CA (1960)
38. Gomory, R. E.: Solving linear programming problems in integers. Combinatorial Analysis, 10, 211-215 (1960)
39. Land, A. H. and Doig, A. G.: An automatic method of solving discrete programming problems. Econometrica 28(3), 497-520 (1960)
40. Mitchell, J. E.: Branch-and-cut. Wiley encyclopedia of operations research and management science. John Wiley & Sons, Inc (2010)

41. Bragin, M. A., Luh, P. B., Yan, J. H., and Stern, G. A.: Surrogate Lagrangian relaxation and branch-and-cut for unit commitment with combined cycle units. In: Proceedings of the IEEE Power and Energy Society, General Meeting, National Harbor, Maryland (2014)
42. Bragin, M. A., Luh, P. B., Yan, J. H., and Stern, G. A.: Novel exploitation of convex hull invariance for solving unit commitment by using surrogate Lagrangian relaxation and branch-and-cut. In: Proceedings of the IEEE PES GM, Denver, Colorado (2015)
43. Ermoliev, Y. M.: Methods for solving nonlinear extremal problems. Cybernetics 2(4), 1–14 (1966)
44. Polyak, B. T.: A general method of solving extremum problems. Sov. Math. 8, 593–597 (1967)
45. Shor, N. Z.: On the rate of convergence of the generalized gradient method. Cybernetics 4(3), 79-80 (1968)
46. Shor, N. Z.: Generalized gradient methods for non-smooth functions and their applications to mathematical programming problems. Econ. Math. Methods 12(2), 337–356 (1976) (in Russian)
47. Brannlund, U., Lindberg, P. O., Niu, A. and Nilsson, J. E.: Railway timetabling using Lagrangian relaxation. Transportation Sci., 32. 358-369 (1998)
48. Fisher, M. L.: An applications oriented guide to Lagrangian relaxation. Interfaces 15(2), 10-21 (1985)
49. Sherali, H. D. and Choi, G.: Recovery of Primal Solutions When Using Subgradient Optimization Methods to Solve Lagrangian Duals of Linear Programs. Operations Research Letters, 19, 105-113 (1996)
50. Mulvey, J. M. and Crowder, H. P.: Cluster analysis: an application of Lagrangian relaxation. Management Sci. 25, 329-340 (1979)
51. Zhao, X., Luh, P. B, and Wang, J.: Surrogate gradient algorithm for Lagrangian relaxation. J. Optim. Theory Appl. 100(3), 699–712 (1999)
52. Bragin, M. A., Luh, P. B., Yan, J. H., Yu, N., and Stern, G. A.: Convergence of the surrogate Lagrangian relaxation method, J. Optim. Theory Appl. 164(1), 173-201 (2015)
53. Hestenes, M. R.: Multiplier and gradient methods. J. Optim. Theory Appl. 4(5), 303-320 (1969)

54. Powell, M. J. D.: A method for nonlinear constraints in minimization problems. In: Optimization, (R. Fletcher, ed.), Academic Press (1969)
55. Bertsekas, D. P.: Nonlinear Programming, 3rd Edition, Athena Scientific, Belmont, MA (2016)
56. Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundations and Trends in Machine Learning. 3(1), 1–122 (2010)
57. Feizollahi, M. J., Ahmed, S, and Sun, A.: Exact augmented Lagrangian duality for mixed integer linear programming. Mathematical Programming, 1-23, 2016
58. Polyak, B. T.: Minimization of unsmooth functionals. USSR Comput. Math. Math. Phys. 9(3), 14–29 (1969). (in Russian)
59. Luenberger, D. G.: Linear and Nonlinear Programming. Addison-Wesley, Reading, MA (1984)
60. Goffin, J.-L., and Kiwiel, K.: Convergence of a simple subgradient level method. Math. Program. 85(1), 207–211 (1998)
61. Yagiura, M., Yamaguchi, T., and Ibaraki, T.: A variable depth search algorithm with branching search for the generalized assignment problem. Optim. Methods Softw. 10(2), 419–441 (1998)
62. Yagiura, M., Ibaraki, T., and Glover, F.: A path relinking approach with ejection chains for the generalized assignment problem. Eur. J. Oper. Res. 169(2), 548–569 (2006)
63. Özbakir, L., Baykasoglu, A., and Tapkan, P.: Bees algorithm for generalized assignment problem. Appl. Math. Comput. 215(11), 3782–3795 (2010)
64. Beasley, J. E.: Or-library: distributing test problems by electronic mail. Journal of Operational Research Society, 41(11), 1069–1072 (1990)
65. Anders, G. (Principal Investigator): Commitment techniques for combined cycle generating units, Kinectrics Inc, Toronto, Canada, CEATI Report No. T053700-31-3 (2005)
66. Alemany, J., Moitre, D., Pinto, H., and Magnago, F.: Short-term scheduling of combined cycle units using mixed integer linear programming solution. Energy Power Eng. 5(2), 161–170 (2013)

67. Fisher, M. L., Jaikumar, R., and Van Wassenhove, L. N.: A multiplier adjustment method for the generalized assignment problem. Mgmt. Sci., 32(9), 1095-1103 (1986)
68. Chvátal, V.: Edmonds polytopes and a hierarchy of combinatorial problems. Discr. Math. **4**(4), 305-337 (1973)
69. Ceria, S., Cornuéjols, G., and Dawande, M.: Combining and strengthening Gomory cuts. E. Balas and J. Clausen, eds. Proc. 4th IPCO Conference, Copenhagen, Denmark. Springer-Verlag, 438-451, (1995)
70. Caprara, A., and Fischetti, M.: {0,1/2}-Chvátal-Gomory cuts. Math. Program. **74**(3), 221-235 (1996)
71. Eisenbrand, F.: Gomory-Chvátal cutting planes and the elementary closure of polyhedra. Doctoral Dissertation, Universität des Saarlandes, http://www2.math.uni-paderborn.de/fileadmin/Mathematik/AG-Eisenbrand/publications/diss.pdf (2000)
72. Padberg, M.: Classical cuts for mixed-integer programming and branch-and-cut. Ann. Oper. Res. 139(1), 321–352 (2005)
73. Cornuéjols, G.: Valid inequalities for mixed integer linear programs. Math. Program. **112**(1), 3-44 (2008)

Appendix A. Proof of Theorem 3.5.4.

Theorem 3.5.4. Comparison of Subproblem and Whole-Problem Gomory Cuts.

The area Ξ_i is a subset of Ω_i.

Moreover, if for a particular i, μ_i and μ the following condition holds:

$$\left\lfloor \sum_{j=1}^{I} \mu_j b_j - \lfloor \mu_i b_i \rfloor \right\rfloor = 0, \tag{A1}$$

then areas ω_i and ξ_i are equal. □

Proof: Consider constraints (3.38):

$$Ax + Ey \le b. \tag{A2}$$

Coefficients of aggregation are μ = (0, μ_1, …, μ_I). In a sense, only subsystem constraints will be aggregated. After introducing non-negative slack variables s, the aggregated constraint becomes:

$$\mu Ax + \mu Ey + \mu s = \mu b. \tag{A3}$$

The equality (A3) can be equivalently written as:

$$\lfloor \mu A \rfloor x - \lfloor \mu b \rfloor = -(\mu Ax - \lfloor \mu A \rfloor x + \mu Ey + \mu s - \mu b + \lfloor \mu b \rfloor). \tag{A4}$$

When solving pure ILP problems, rotation and shifting can be performed after a pre-multiplication by μ. However, when solving MILP problems, because of the presence of continuous variables, shifting through truncation of b may cut off continuous feasible solutions. Therefore, disjunction is performed together with rotation and shifting. For simplicity of the proof, left-hand side of (A4) will be considered to create a disjunction:

$$\lfloor \mu A \rfloor x - \lfloor \mu b \rfloor \le 0, \tag{A5}$$

and

$$\lfloor \mu A \rfloor x - \lfloor \mu b \rfloor \ge 1. \tag{A6}$$

Whole-Problem Gomory Inequalities. Since subsystems have disjoint domains, (A5) can be represented as

$$\sum_{j=1}^{I} \lfloor \mu_j A_j \rfloor x_j \le \left\lfloor \sum_{j=1}^{I} \mu_j b_j \right\rfloor. \tag{A7}$$

To determine areas that are cut off by (A7) outside subsystem *i* convex hull, owing to disjoint nature of subproblems, all x_j's other than x_i can be set to zero, and (A7) becomes:

$$\lfloor \mu_i A_i \rfloor x_i \le \left\lfloor \sum_{j=1}^{I} \mu_j b_j \right\rfloor. \tag{A8}$$

Subproblem Gomory Inequalities. A disjunction of subproblem *i* inequalities can be constructed by using the same logic as in (A5) – (A6), and the resulting disjunction is:

$$\lfloor \mu_i A_i \rfloor x_i \le \lfloor \mu_i b_i \rfloor, \tag{A9}$$

and

$$\lfloor \mu_i A_i \rfloor x_i \ge \lfloor \mu_i b_i \rfloor + 1. \tag{A10}$$

Comparison. Left-hand sides of (A8) and (A10) are the same. To compare (A8) and (A0) for all possible μ, their right-hand sides will be compared. Owing to higher flexibility in which whole-problem inequalities can be shifted as compared to subproblem inequalities, and the following inequality holds:

$$\left\lfloor \sum_{j=1}^{I} \mu_j b_j \right\rfloor \le \lfloor \mu_i b_i \rfloor, \text{ for all } \mu. \tag{A11}$$

Therefore, the total area cut off by (A5) outside subproblem *i* convex hull is at least as large as the total area cut off by (A9). Similarly, (A8) cuts off larger area as compared to (A10).

Gomory cuts are constructed by aggressive rotation and shifting based on each set from a disjunction, and then by connecting these sets. Also, subproblem *i* Gomory cuts cut off the total area Ξ_i, and whole-problem Gomory cuts cut off the total area Ω_i. Since subproblem *i* Gomory cuts are based on a disjunction (A9) – (A11) that cuts off smaller areas, then Ξ_i is a subset of Ω_i. The logic of the proof will remain the same if *Ey* and *s* are considered based on the right-hand side of (A4).

Equivalence of Subproblem and Whole-Problem Cuts. To determine performance of subproblem and whole-problem Gomory cuts for particular μ under condition (A1), consider the following equality:

$$0 = \left\lfloor \sum_{i=1}^{I} \mu_i b_i - \lfloor \mu_j b_j \rfloor \right\rfloor = \left\lfloor \sum_{i=1}^{I} \mu_i b_i \right\rfloor - \lfloor \mu_j b_j \rfloor, \text{ for a particular } \mu. \tag{A12}$$

Therefore, in the subproblem *i* space, (A5) is the same as (A9). As a result, under condition (A1), whole-problem and subproblem *i* Gomory cuts are the same, and areas ω_i and ξ_i are equal.

Chapter 4

Distributed and Asynchronous Unit Commitment and Economic Dispatch

The drastic increase of generation uncertainty associated with intermittent renewables as well as demand uncertainties associated with behind-the-meter generation create major challenges in power system operations. The current centralized unit commitment and economic dispatch (UCED) approach used by Independent System Operators (ISOs) will not be flexible enough in the future when numbers of nodes and units are large and levels of uncertainties are high. In this section, a distributed and asynchronous framework and the corresponding solution methodology are presented. In the method, unit subproblems are solved locally in an asynchronous manner, and price-based coordination is performed by an ISO. Since prices are updated based on unit solutions obtained using prices of different vintages, convergence may not be guaranteed. To overcome this difficulty, our recent surrogate Lagrangian relaxation (SLR) is distributed, and conditions on price convergence are innovatively established through a Lyapunov energy function of the distance from current prices to the optimum. To consider the effects of asynchronous unit solutions, an upper bound of the energy function is shown to approach zero. Numerical testing on the UCED problem with 1000 units without transmission capacity constraints indicates that the method is robust and fast. To further explore the scalability of asynchronous coordination, a simplified UCED problem with 10,000 units each with generation capacity constraints only indicates that the method converges, is robust and more efficient as compared to ADMM.

4.1. Introduction

The drastic increase of generation uncertainty associated with intermittent renewables as well as demand uncertainties associated with behind-the-meter generation creates challenges in power system operations. According to NY-ISO's annual report [1], the penetration of renewables reached 23% in 2015 and the proposed increase of wind penetration during 2016 is from 1.7 to 3.7 GW. Because of intermittent nature of renewables, the generation uncertainty increases with the increase of levels of renewable penetration. The demand uncertainty can result from the so-called "load defection" [2, 3] whereby participants produce their own power or purchase it directly from suppliers. While customers tend to use more solar power at a distribution level, this also contributes to the demand uncertainty at the nodes of transmission network. According to NY-ISO's annual report [1], such behind-the-meter PV generation will increase form 1 GW in 2015 to almost 3 GW in 2025. Because ISOs do not see such behind-the-meter generation, demand uncertainties will be difficult to handle. Generation and demand uncertainties need to be accounted for while committing units days or hours ahead within the unit commitment and economic dispatch (UCED) problem, an important problem solved by ISOs on a daily basis. However, the current centralized UCED approach used by ISOs will not be flexible enough in the future when numbers of nodes and units are large and levels of uncertainties are high.

In this Chapter, to resolve uncertainty issues that ISOs will face in the future, a distributed and asynchronous framework for MILP problems and the corresponding solution methodology is established whereby unit subproblems are solved locally in an asynchronous manner, and price-based coordination is performed by an ISO. In Section II, existing distributed and asynchronous methods for MILP problems will be reviewed. Most asynchronous methods such as ADMM [e.g., 4] require convexity of the problem and cannot be easily extended for MILP problems. While traditional MILP methods are typically not asynchronous, the novel methodology will be based on the Lagrangian relaxation method, and this method will also be reviewed together with the recent surrogate Lagrangian relaxation (SLR) [5]. Within standard Lagrangian relaxation, the relaxed problem is fully optimized and dual values are on the dual

surface. Therefore, subgradient directions form acute angles with directions toward the optimal multipliers, and convergence can be guaranteed. However, the method may suffer from zigzagging of multipliers and convergence can be slow. This difficulty was overcome within our recent SLR method [5]. Without fully optimizing the relaxed problem, resulting "surrogate" subgradient directions are smooth and zigzagging is alleviated. However, because surrogate dual values are no longer on the dual surface, subgradient directions may not form acute angles with directions toward the optimal multipliers. Nevertheless, under the simple "surrogate optimality condition," surrogate dual values get close enough to the dual surface to guarantee that surrogate subgradients also form acute angles with directions toward the optimum. Within SLR, stepsizes approach zero thereby guaranteeing convergence, and at the same time stepsizes remain large enough to reach the optimum and avoid premature algorithm termination [5].

In subsection 4.3, a novel distributed and asynchronous framework will be presented whereby subproblems are solved using the latest available values of multipliers, and wait until updated values of multipliers arrive. The coordinator will wait for subproblem solutions to arrive for a pre-specified amount of time before updating multipliers.

In Section 4.4, under simplifying assumptions, a deterministic version of unit commitment subject to unit-wise and system-wide demand constraints will be briefly presented. After relaxing system-wide constraints, the relaxed problem can be decomposed into subproblems, and solving time of subproblems will be assumed to be stochastic.

In Section 4.5, the corresponding solution methodology will be presented. The difficulty is that surrogate subgradient directions may not form acute angles with directions toward the optimal multipliers. Moreover, the difficulty is exacerbated because subproblem solutions are updated using multipliers of different vintages. As a result, multipliers may not strictly approach the optimum. To overcome this

difficulty, it will be assumed that within a finite number of coordinator iterations[10], all subproblem solutions arrive to the coordinator at least once. Therefore, multipliers may move away from the optimum only a limited distance, at worst. As such, there is an upper bound on the Layapunov energy function of distance from current multipliers to the optimum. Moreover, after solving all subproblems at least once, multipliers will be on or close the dual surface, and multipliers will move toward the optimum. Because stepsizes approach zero, the upper bound of the Layapunov function will decrease. At the same time, since stepsizes remain large enough the premature algorithm termination is avoided and the upper bound decreases to zero.

In Section 4.6 it is demonstrated numerically, the new method is efficient to solve the UCED problem with 1000 units and to provide a tight lower bound. Moreover, based on a simplified UCED problem with 10000 units it is shown that the new method converges, is robust and more efficient as compared to ADMM.

4.2. Literature Review

After reviewing distributed and asynchronous approaches for MILP problems, decomposition and coordination methods will also be reviewed. The remaining difficulties associated with the asynchronous nature will be presented.

4.2.1. Distributed and Asynchronous Methods

Distributed and asynchronous methods have been used for solving MILP as well as continuous problems. While an effective practical asynchronous scheme was performed for an MILP slack matching problem [6, 7], theoretical convergence has not been established. Particular power systems applications of distributed and asynchronous methods used to solve LP problems include DC optimal power flow [8]

[10] Coordinator may choose to update prices with consistent periodicity (e.g., every 10 seconds) using subproblem solutions that arrived during this period.

and multi-objective stochastic economic dispatch problem [9]. A powerful method to solve continuous and convex problems consensus optimization [10, 11] has been asynchronous ADMM, and the method has been shown to converge with the rate is similar to synchronous ADMM and with the linear rate [4]. However, without the diminishing property of stepsizes ADMM cannot be easily extended for MILP problems the method typically does not converge [12].

4.2.2. Decomposition and Coordination Methods

Subgradient method. One method that was traditionally used to solve MILP problems including unit commitment and economic dispatch is Lagrangian relaxation [13, 14]. After relaxing system-wide coupling constraints such as system demand constraints, the relaxed problem is separable and it can be decomposed into individual subproblems. Within standard Lagrangian relaxation, multipliers are updated by using subgradient methods after solving all the subproblems thereby ensuring that dual values are always on the dual surface. While the convergence proof requires the knowledge of optimal dual value, since subgradients form acute angles with directions toward the optimal multipliers, the optimal dual value can be adaptively estimated to make sure that multipliers are getting closer to the optimum and convergence in practice can be guaranteed. However, subgradient directions are frequently almost perpendicular to the direction toward the optimal multipliers. As a result, multipliers suffer from zigzagging thereby leading to very slow convergence.

Surrogate Lagrangian relaxation. Our recently-developed surrogate Lagrangian relaxation [5] overcomes zigzagging as well as convergence difficulties by solving one or few subproblems at a time thereby ensuring that "surrogate" subgradient directions do not change drastically and are smooth. Within the method, multipliers are mostly above the dual surface and surrogate subgradient directions generally do not form acute angles with directions toward the optimal multipliers. To guarantee convergence, the relaxed problem needs to be sufficiently optimized subject to the simple "surrogate optimality condition." Essentially, when sufficiently close to the dual surface, it has been shown that surrogate subgradient directions also form acute angles with directions toward the optimal multipliers, and typically such angles

are more acute as compared to those within the subgradient method and the zigzagging is alleviated. Moreover, with small acute angles multipliers will approach the optimum faster thereby reducing the number of iterations required for convergence. More importantly, convergence the optimum does not require the knowledge of the optimal dual value. This was achieved with a constructive process in which distances between Lagrange multipliers at consecutive iterations decrease, and as a result, stepsizes approach zero and multipliers converge to a unique limit. At the same time, stepsizes are kept sufficiently large to avoid premature algorithm termination.

4.3. Distributed and Asynchronous Framework

Suppose there are I distributed subproblem solvers, and one coordinator. Each subproblem and reports to the coordinator, who will then iteratively coordinate subproblems solutions by updating prices. Within the distributed and asynchronous framework, it will be assumed that after each subproblem is solved, its solution is communicated to the coordinator, and the subproblem solver wait until prices λ^k are updated. For example, in Figure 1 subproblem 4 will wait until updated price λ^k is available, and the wait time is schematically shown by a dashed line. After prices are updated, subproblem solvers will use the latest available values of prices. The coordinator will gather solutions that arrive during a pre-specified amount of time (e.g. 10 seconds) before updating multipliers. If no solutions arrive, the iteration is skipped. For example, as shown in Figure 1, at iteration k, the coordinator received subproblem 1, 4, I+1 and I's solutions and the coordinator can then update prices without waiting for other solutions to arrive. As a result, the probability of arrival of solutions to the coordinator is uncertain and can be modeled by assuming a certain distribution. Generally, there is a communication delay, but for simplicity it will be ignored.

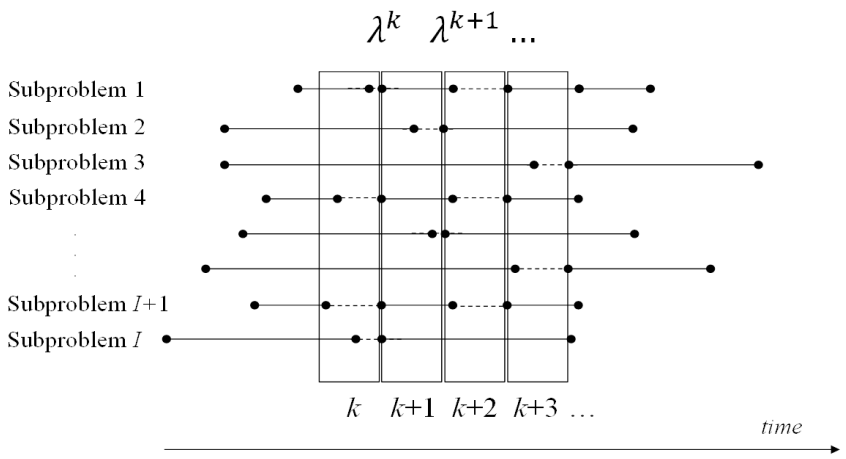


Figure 4.3.1. Illustration of distributed and asynchronous framework. Subproblem solve time is shown by a horizontal solid line, and dots indicate start and end of subproblem solve time.

4.4. Problem Description and Asynchronous Surrogate Lagrangian Relaxation (DA-SLR)

Consider I units and one coordinator. The task of the coordinator is to minimize the total cost consisting of energy and start-up costs:

$$\sum_{i=1}^{I}\sum_{t=1}^{T} c_i\left(p_i(t),t\right)+\sum_{i=1}^{I}\sum_{t=1}^{T} S_i(t), \tag{4.1}$$

subject to unit-wise constraints such as generation capacity, ramp-rate, and minimum up- and down-time constraints [15-17] and the following system-demand constraints:

$$\sum_{i=1}^{I} p_i(t) = P^D(t). \tag{4.2}$$

After relaxing constraints (4.2) by introducing Lagrange multipliers λ(t), which can be viewed as prices, solution of the following price-based unit commitment problem is delegated to unit i:

$$\sum_{t=1}^{T} c_i\left(p_i(t),t\right)+\sum_{t=1}^{T} S_i(t)+\sum_{t=1}^{T}\lambda(t)p_i(t), \tag{4.3}$$

subject to unit-wise constraints such as generation capacity, ramp-rate, and minimum up and down-time constraints. Each such subproblem is linear and can be solved by using branch-and-cut locally by each unit solver.

Within the SLR framework [5], it is sufficient to solve one subproblem (4.3) to satisfy the "surrogate optimality condition" to ensure that surrogate subgradient directions form acute angles with direction toward λ^*. After one subproblem is solved, multipliers are updated and the next subproblem is solved. In the following, a distributed and asynchronous framework will be established whereby several subproblems may be solved at the same time, and multipliers update occurs without waiting for all subproblem solutions to arrive to the coordinator.

As reviewed in Section II, within the surrogate Lagrangian relaxation framework, surrogate dual values are mostly above the dual surface, and surrogate subgradient directions will be denoted at coordinator iteration k as $\tilde{g}^k$ and will be obtained based on levels of system demand constraint violations as:

$$\tilde{g}^k = \sum_{i=1}^{I} p_i^{k_i}(\lambda^{k_i}) - P^D. \tag{4.4}$$

Multipliers will then be updated as:[11]

$$\lambda^{k+1} = \lambda^k + c^k \tilde{g}^k,\ k = 0,1,.... \tag{4.5}$$

Here k_i ($\leq k$) is the coordinator iteration number when subproblem i was solved based on λ^{k_i}. In the following Section 4.5, it will be proved that the multipliers (4.5) will converge to the optimum.

4.5. Convergence of DA-SLR

As reviewed in Section II, the major difficulty of the SLR method was that surrogate directions (4.4) may not form acute angles with directions toward the optimal multipliers. Within the distributed and asynchronous framework, this difficulty is compounded because solutions are obtained using multipliers $\lambda(t)$ of different vintages as illustrated in Figure 4.5.1. If some of the subproblems are not solved frequently enough, surrogate directions will not be sufficiently updated. As a result, multipliers

[11] Here and later the argument t is dropped for clarity of explanation.

N_1 N_2 N_3 N_4

may travel away from λ* for a large number of iterations thereby potentially leading to divergence. This effect of asynchronous unit solutions is illustrated in Figure 4.5.1 whereby between iterations N_1 and N_2 multipliers move away from the optimum and the energy function increases. As a result, multipliers (4.5) may not converge to the optimum, or convergence will be slow.

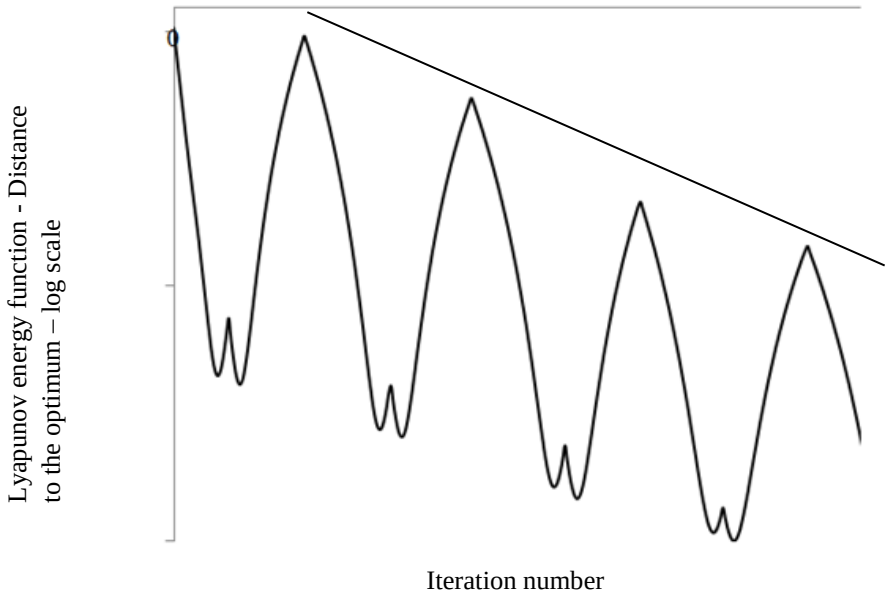


Figure 4.5.1. Illustration of convergence. The upper bound on distances from λ^k to λ^* is shown by a straight line.

To overcome this issue the following will be assumed first:

Assumption 4.5.1: Within a finite number of coordinator iterations κ, all subproblems will be solved at least once.

This assumption is similar in nature to "bounded delay" assumption [18] and to "bounded staleness" assumption [19]. This assumption is reasonable within the framework of Section III because it is expected that subproblems are much smaller in size and complexity and can be solved within a reasonable time.

By using this Assumption 4.5.1 together with the diminishing nature of stepsizes of the SLR method, multipliers will converge to the optimum. The result is summarized in the following Theorem:

Theorem: Suppose that distances between multipliers at consecutive iterations decrease as:

$$\left\|\lambda^{k+1}-\lambda^{k}\right\| < \alpha_k \left\|\lambda^{k}-\lambda^{k-1}\right\|, 0<\alpha_k<1. \tag{4.6}$$

Parameters α_k satisfy

$$\prod_{i=1}^{k}\alpha_i \to 0,\ k\to\infty, \tag{4.7}$$

thereby implying that by the contraction mapping, there exists a limit of λ^k. Moreover, stepsizes remain large enough per

$$\frac{1-\alpha_k}{c^k}\to 0,\ k\to\infty. \tag{4.8}$$

Then, if Assumption 1 holds, multipliers will be confined around λ^* per

$$\left\|\lambda^{*}-\lambda^{k}\right\| < B^k. \tag{4.9}$$

Moreover, the bounds B^k will approach zero thereby implying that multipliers approach λ^*.

Sketch of Proof:

Step 1: Increase of Lyapunov energy function. As argued before, the effect of asynchronous unit solutions may lead to the increase of Layapunov energy function. Because of Assumption 4.5.1, all subproblems will be solved within a finite number of iterations κ, and as a result, multipliers may travel away from λ^* a finite distance.

Step 2: Decrease of Lyapunov energy function. After solving all subproblems at least once, surrogate dual values will be on the dual surface or very close to it. Therefore, owing to the properties of subgradient directions and properties of surrogate subgradient directions as reviewed in Section 4.2, these directions will form acute angles with the direction toward λ^*, multipliers will start approaching λ^*. Also, per (4.8) it is guaranteed that stepsizes remain large enough thereby ensuring that the algorithm will not terminate prematurely and multipliers will make significant progress toward λ^*. On Figure 4.5.2, it is shown that a significant progress is made between iterations N_2 and N_3.

Step 3: Decrease of upper bounds on Lyapunov function. Since distances between multipliers at consecutive iterations decrease per (4.6) and the constant κ in Assumption does not change, multipliers will move away from λ^* by smaller amounts in future iterations. In Figure 4.5.2, it is illustrated, for

example, that distance travelled between iterations N_3 and N_4 is smaller than the distance travelled between iterations N_1 and N_2. As a result, the bounds within which multipliers travel will tighten round λ^*.

Step 4: Decrease of upper bounds to zero. On the larger scale, owing to (4.8) stepsizes remain large enough thereby ensuring that this process or tightening the bound in (4.9) will repeat periodically and infinitely often thereby leading to convergence.

In the following Section 4.6, efficiency, coordination aspects and scalability of the new method will be tested.

4.6. Numerical Testing

The new method is implemented by using CPLEX 12.6.0 on 64-bit windows by using a laptop with the processor Intel® CoreTM i7-4910MQ CPU @ 2.90GHz, 16 GB of RAM. The distributed processing will be simulated by assuming that CPU times of each distributed processor follow specified statistical distribution.

Example 4.6.1: UCED with 1000 units.

Consider a UCED problem with 1000 units with linear energy prices. The novel DA-SLR method will be compared with the alternate direction method of multipliers (ADMM). It is assumed that all solving times of distributed processors are independent and follow identically distributed truncated normal distributions. Parameters of the normal distributions are chosen in a way that κ in the assumption of Section 4.5 is κ = 3000. The results are shown in the following Figure 4.6.1.

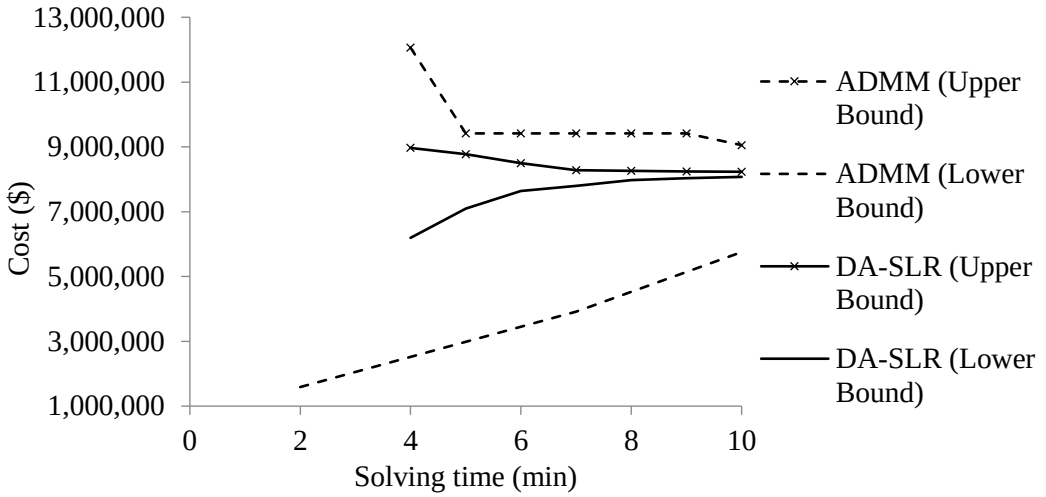


Figure 4.6.1. Results on UCED with 1000 units: Comparison of DA-SLR and ADMM

Within the DA-SLR the duality gap after 10 minutes (time required to solve subproblems + time required to update multipliers) is less than 1% while within ADMM, the duality gap is 46%. When solving the non-convex problem (2.1), DA-SLR has a diminishing property of stepsizes per (7) which per Section V allow to prove that upper bounds on distances from λ^k to λ^* tighten, which also implies that the lower bound provided within the method tightens. According the results in Figure 3, this lower bound tightens fast. In contrast, ADMM typically does not converge [9] when solving MILP problems such as unit commitment, because stepsizes within ADMM do not have a diminishing property, which is required for the optimization of associated non-smooth dual functions.

Example 4.6.2: Robustness testing by considering a simplified UCED problem with 10,000 units.

To test robustness of the coordination aspect of the method developed in this Chapter, consider simplified 2-hour UCED problem with 10,000 units each only with generation capacity constraints and 2 coupling demand constraints. Assume that solving time of each subproblem is uniformly distributed and the probabilities of arrival of solutions to the coordinator are uniformly distributed. Assume that the coordinator waits for 10 subproblem solutions to arrive before updating multipliers. The method was run 5 times and trajectories of multipliers are shown in the following Figure 4.6.2.

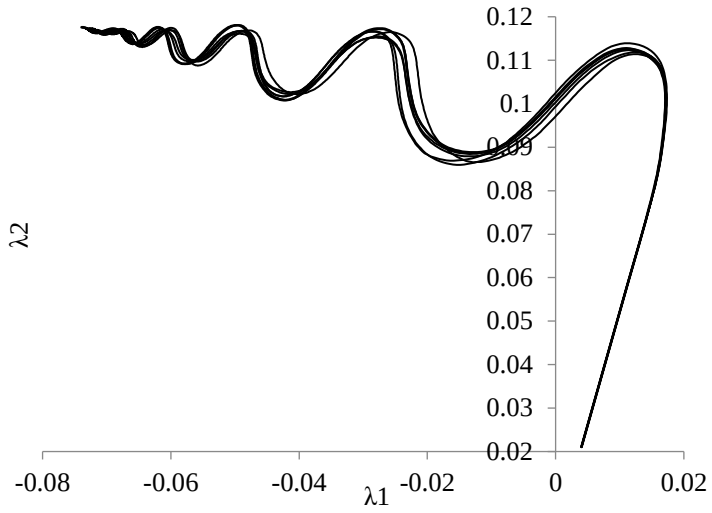


Figure 4.6.2. Trajectories of multipliers for Example 4.6.2 (robustness)

As shown in Figure 4.6.2, trajectories of multipliers are consistent with each other. In the following Figure 4.6.3, a comparison of convergence of DA-SLR and ADMM is performed and it is demonstrated that that the upper bound on the on distances from λ^k to λ^* approaches zero, and as a result multipliers approach the dual optimum within DA-SLR.

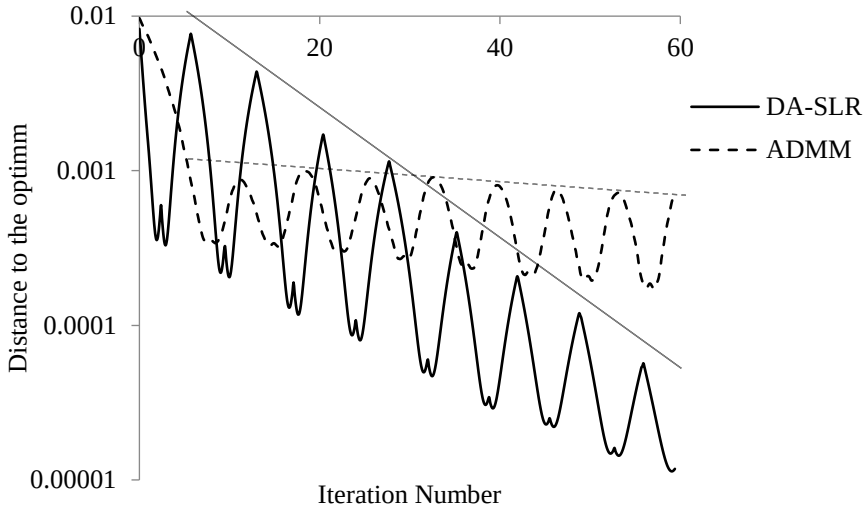


Figure 4.6.3. Comparison of DA-SLR with ADMM for Example 4.6.2. Upper bounds on distances to the optimum are shown by straight lines

As demonstrated in Figure 4.6.3, the upper bound on the distance to the optimum decreases much faster within the novel DA-SLR method as compared to that of ADMM.

4.7. Conclusion

This Chapter addresses the issue that the growing increase in generation and demand uncertainties brings to power systems operations by developing the distributed and asynchronous framework whereby solutions to price-based unit subproblems are coordinated by the coordinator such as an ISO asynchronously. Conditions on price convergence are innovatively established through upper bounds on a Lyapunov energy function and by proving that these bounds approach zero. It is demonstrated numerically that the new method is robust and converges fast. The method can also be extended for problems with transmission capacity constraints by coordinating nodal subproblem solutions using DA-SLR. Also, the method can also be used for distribution problems with a very large number of subproblems associated, for example, with PV solar panels.

References

1. https://www.iso-ne.com/static-assets/documents/2016/03/2016_reo.pdf
2. http://www.nyiso.com/public/webdocs/media_room/publications_presentations/Power_Trends/Power_Trends/2016-power-trends-FINAL-070516.pdf
3. The Economics of Grid Defection, Rocky Mountain Institute, February 2014.
4. Shi, Wei, Qing Ling, Kun Yuan, Gang Wu, and Wotao Yin. “On the linear convergence of the ADMM in decentralized consensus optimization.” IEEE Transactions on Signal Processing 62, no. 7 (2014): 1750-1761.
5. Bragin, Mikhail A., Peter B. Luh, Joseph H. Yan, Nanpeng Yu, and Gary A. Stern. “Convergence of the surrogate Lagrangian relaxation method.” Journal of Optimization Theory and Applications 164, no. 1 (2015): 173-201.

6. Beerel, Peter A., Andrew Lines, Mike Davies, and Nam-Hoon Kim. “Slack matching asynchronous designs.” In 12th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC'06), pp. 11-pp. IEEE, 2006.
7. Gill, Gennette, Vishal Gupta, and Montek Singh. “Performance estimation and slack matching for pipelined asynchronous architectures with choice.” In2008 IEEE/ACM International Conference on Computer-Aided Design, pp. 449-456. IEEE, 2008.
8. Dall'Anese, Emiliano, Hao Zhu, and Georgios B. Giannakis. “Distributed optimal power flow for smart microgrids.” IEEE Transactions on Smart Grid4, no. 3 (2013): 1464-1475.
9. Fu, Yimu, Mingbo Liu, and Licheng Li. “Multiobjective Stochastic Economic Dispatch With Variable Wind Generation Using Scenario-Based Decomposition and Asynchronous Block Iteration.” IEEE Transactions on Sustainable Energy 7, no. 1 (2016): 139-149.
10. Zhang, Ruiliang, and James T. Kwok. “Asynchronous Distributed ADMM for Consensus Optimization.” In ICML, pp. 1701-1709. 2014.
11. Nishihara, Robert, Laurent Lessard, Benjamin Recht, Andrew Packard, and Michael I. Jordan. “A General Analysis of the Convergence of ADMM.” arXiv preprint (2015).
12. Boyd, Stephen, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. “Distributed optimization and statistical learning via the alternating direction method of multipliers.” Foundations and Trends® in Machine Learning 3, no. 1 (2011): 1-122.
13. Guan, Xiaohong, Peter B. Luh, Houzhong Yan, and J. A. Amalfi. “An optimization-based method for unit commitment.” International Journal of Electrical Power & Energy Systems 14, no. 1 (1992): 9-17.
14. Guan, Xiaohong, Peter B. Luh, Houzhong Yan, and Peter Rogan. “Optimization-based scheduling of hydrothermal power systems with pumped-storage units.” IEEE Transactions on Power Systems 9, no. 2 (1994): 1023-1031.
15. Rajan, Deepak, and Samer Takriti. “Minimum up/down polytopes of the unit commitment problem with start-up costs.” IBM Res. Rep (2005).

16. Bragin, Mikhail A., Peter B. Luh, Joseph H. Yan, and Gary A. Stern. “An efficient approach for solving mixed-integer programming problems under the monotonic condition.” Journal of Control and Decision 3, no. 1 (2016): 44-67.
17. Bragin, Mikhail A., Peter B. Luh, Joseph H. Yan, and Gary A. Stern. “Novel exploitation of convex hull invariance for solving unit commitment by using surrogate Lagrangian relaxation and branch-and-cut.” In 2015 IEEE Power & Energy Society General Meeting, pp. 1-5. IEEE, 2015.
18. Ho, Qirong, James Cipar, Henggang Cui, Seunghak Lee, Jin Kyu Kim, Phillip B. Gibbons, Garth A. Gibson, Greg Ganger, and Eric P. Xing. “More effective distributed ml via a stale synchronous parallel parameter server.” In Advances in neural information processing systems, pp. 1223-1231. 2013.
19. Zhang, Ruiliang, and James T. Kwok. “Asynchronous Distributed ADMM for Consensus Optimization.” In ICML, pp. 1701-1709. 2014.