More than Forty Years History of Developing Algorithms and Codes to Solve Integer Programming Problems

Kakuzo Iwamura*

Abstract

In this article, we will talk on some algorithmic developments for Knapsack Problem, Set-Covering/Set-Partitioning Problem. Then, we propose that finding an optimal/near optimal solution for a given Real-World input data of a NP-Complete/NP-Hard integer programming problem is great enough because any algorithm for such a problem shows exponential and heavy data dependent computing time.

1. Introduction

In 1970, I happened to get a job in Mitubishi Research Institute. There, I was ordered to make some programs in Operations Research. They were

1. Travelling salesperson problem (TSP),

2. Capacitated Facilities Location Programming (CFLP) Problem

and other statistical problems. Of the two, We devised an excellent computer program to solve the CFLP problem with another efficient heuristic one. Computational results of our computer programs can be found partly in Mukawa, H., Sensui, J., Iwamura, K. and J. Kase [1971]. As for the efficient heuristic algorithm for the CFLP Problem, the readers can consult Sorimachi, Y. [1970]. They worked very well for a Real-World input data.

On the contrary, we were not able to develope an efficient computer program to solve the TSP for a Real-World data. Through IBM 360/370, the fastest computer in 1970 in the world, our algorithm using a minimum spanning tree bound needed more than ten minutes for a TSP problem instance sized 30 cities. It meant it costed more than 300 thousand yen to solve a 30 city TSP and so a 100 city TSP was far from being able to be solved. Furthermore its computing time varied drastically as the problem instances changed. This was a great surprise for me. Even when I faithfully implemented an algorithm of Little, J. D. C. et al. [1963] with the most professional programming technique given by Mr. H. Mukawa, our computer

* *Mathematics, Josai University Sakado, Saitama 350-0295, Japan*

program showed both exponential computing time and its computing time dependency on each problem instance. We were shocked by its computational inefficiency and its unexpectable computing time. "Oh! What a shock. Little et al's algorithm got the prize because it solved a US state capital TSP problem. Yet, it showed a poor computing efficiency. A great difference from the CFLP Problem. But why?" This experience caused us a deep contradiction. It was so deep that our programmer group disappeared in two years.

After finding a job at the Dept. of Math., Josai University, I started a theoretical research activities in Integer Programming. I found a knapsack typed integer programming problem computationally greedily solvable through D. P. technique. Then, I changed my interest to Set-coverning/Set-Partitioning Problems. At that time, still now I think, there was a roumor that general algorithms for the linear integer programming problems such as Gomory's fractional integer programming algorithm was inefficient for Real-World data. So, it was the time to try to invent a specific algorithm to solve a specific problem such as the Set-Partitioning problem. Furthermore it had a wide application in transportation engineering. For example, air line crew scheduling, bus routing, truck dispatching and so on. I made a great efforts to develop an efficient computer algorithm to solve the Set-Partitioning Problem. I payed every attention to make the computer program as fast as possible. It amounted over 600 pages documents to devise three computer programs to solve the Set-Partitioning Problem. The interested readers can have information through Suzuki, H. and K. Iwamura [1979], Iwamura, K. and Y. Maeda [1977a, 1977b], Maeda, E. and K. Iwamura [1982]. Still, our computational experiments showed the same aspect as I had experienced in developing computer programs to solve the TSP. Exponential computing time and a heavy data dependent computing time. One data was solved by one algorithm efficiently while another data of the same size wasn't solved by another in one week through FACOM 230-38S. Of the three algorithms, I was not able to say that this one was always superior to that one.

2. Definitions

A traveling salesperson has to visit all the cities (n cities in all) for a given time period. A distance cost for any pair of cities i, j is c_{ij}, which are all given. He has to visit any city only once, yet he has to visit all of them. There are $(n-1)!$ different routes to do this. Each traveling route is called a *tour*. The cost of the tour is defined as the sum of the distance costs on the tour. Among all the possible (i. e., feasible) tours, he has to find the tour which has the minimum cost, which we call *optimal tour*, or *optimal solution* for a given input data of the Traveling Salesperson Problem.

Given n pairs of positive integers c_j, a_j and a positive integer b. *Knapsack Problem* is an integer programming problem,

maximize

$$\sum_{j=1}^{n} c_j x_j$$

subject to

$$\sum_{j=1}^{n} a_j x_j \leq b,\ x_j\text{: nonnegative integer } (1 \leq j \leq n) \tag{1}$$

If we change

$$x_j\text{: nonnegative integers} \tag{2}$$

to

$$x_j\text{: binary} \tag{3}$$

then we get *0-1 Knapsack Problem.* If we further change

$$\sum_{j=1}^{n} a_j x_j \leq b \tag{4}$$

into

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i (1 \leq i \leq m) \tag{5}$$

where a_{ij}, b_i are nonnegative integers, then we have *Multiple 0-1 Knapsack Problem.* Finally, if the constraints are

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i (1 \leq i \leq m),\ x_j\text{: nonnegative integer} \tag{6}$$

then we have *Unbounded Multiple Knapsack Problem*, which we called *Knapsack Typed Integer Programming* (K. Iwamura [1972, 1974]).

A Set-Covering Problem is another integer programming. We want to maximize

$$\sum_{j=1}^{n} c_j x_j$$

subject to

$$\sum_{j=1}^{n} a_{ij} x_j \geq 1 (1 \leq i \leq m),\ x_j\text{: binary} (1 \leq j \leq n). \tag{7}$$

Here, if we replace inequality by equality, then we get *Set-Partitioning Problem.* In the next section we will see algorithmic aspects of these problems with some computer programming features.

3. Knapsack Problem

Already in 1970' it was well recognized that Knapsack problem was easily solved to Real-World problem data. We imagined that the situation was same for knapsack typed integer programming problems (K. Iwamura [1972]). Its easiness contributed almost half of success of H. Mukawa, J. Sensyui, K. Iwamura and J. Kase [1971]. Another half of it was because *Capacitated Facilities Location Programming problem* contained a transportation problem as a subproblem. To solve a transportation problem is much faster than to solve a general linear programming problem.

We find that S. Martello and P. Toth's book, *Knapsack Problem,* published in 1990, is a fine one. They treat almost all types of knapsack problems. But they don't treat our Knapsack

Typed integer programming. They equip the book with some FORTRAN programs to solve some kinds of knapsack problems, although we can't see whether personal computers other than IBM PCs can read the programs. Using the terminology of S. Martello and P. Toth, Knapsack Typed integer programming is called *Unbounded Multiple Knapsack Problem.* Finally it is author's duty to ask the readers to take notice that they say that they have some imput data of some knapsack typed problems which they are not able to sove successfully.

As for the research activities carried out by other Japanese, I think that H. Suzuki and K. Iwamura [1979], H. Konno and H. Suzuki [1982] are still useful. M. Futakawa et al. [1995] (Max-Min Knapsack Problem) reported that their problem was able to be solved very quickly through their heuristic algorithm and its error ratio got lesser and lesser down to 0.001 percent as the problem size increased up to ten thousand. This fact was pointed out already in 1972 by J. G. Lührs [1972] for a simple Knapsack Problem. In 1970s researchers thought that knapsack problem was an easy one to devote ourselves in. In fact, they are still fruitful as the piles of the computational results in S. Martello and P. Toth [1990] show it to us. See also T.-C. Lai, M. L. Brandeau and S. Chiu [1994] and Y. Hayashi [1995]. Today, we can say that we will find an optimal /a near optimal solution for a Real-World data of Knapsack Problems by the Branch and Bound method within a reasonable amount of time.

4. Set-Covering/Set-Partitioning Problem

When we turn our attention to the Set-Covering and/or Set-Partitioning Problems, we realize they are far beyond our ability, particularly finding an optimal solution for a Real-World data of the Set-Partitioning problem. Therefore the author thinks it's benificial to state an algorithmic history of discrete optimization and integer programming.

The importance to solve the integer programming problems by computers were well acknowleged by operations researchers late 1950s (G. B. Danzig [1963]). R. E. Gomory's "Outline of an Algorithm for Integer Solutions to Linear Programs" appeared in Bull. Amer. Math. Soc. in 1958. Gomory developed two algorithms to solve general linear integer programming problem. They are called "Fractional integer programming algorithm" and "All integer algorithm" (T. C. Hu [1970]). In R. S. Garfinkel and G. L. Nemhauser [1972], they are called "the method of integer forms" and "dual all integer algorithm". In their book are showed Young's primal all-integer algorithm, too. These algorithms are all based on cutting plane methods because they add a new cutting plane constraint from iteration to iteration. In the early 70s, the author got a rumour that a cutting plane method generated so many cut constraints that the algorithms based on it wasn't able to go further because of both storage and computing time limit. And so, practitioners thought that a general algorithm to solve general integer linear programming problem were useless and helpless. The rumour was in fact a truth. We were able to see it at page 380 in Garfinkel and Nemhauser [1972], although there were some problems and some problem instances that could be solved within a reasonable time bound.

Then came a wave of research activities in integer programming (IP) by the Branch and

Bound method. This time, we picked up a specific integer programming problem such as Knapsack Problem, Travelling Salesman (Salesperson) Problem, Set Covering Problem, Vehicle Routing Problem, Set Partitioning Problem and so on. Yet some Japanese researchers, Prof. H. Watanabe at Tokyo University, Dr. H. Naruhisa and Dr. E. Maeda at Nippon Univac Research Center suggested that their computational experiences do not coincide with the results reported in journals. So, we carried out computational experiments of the Set Partitioning Problem with three algorithms. The first one is from E. Balas and M. W. Padberg [1975, 1972], the second one is from R. S. Garfinkel and G. L. Nemhauser [1969], the third one is from J. F. Pierce and J. S. Lasky [1973]. These were selected on the basis of their algorithmic efficiency, independent of implimentation easiness of their algorithms. Furthermore we gave every improvement to each of the three with as much programming techniques as we were able to pay. To each problem instance, we applied MPS/X to compare the efficiency of the three so that we could draw a fare judgement for the efficiencies of them three. Up to problem size 100 rows and 200 columns, densities varying from 68% to 3.4% every algorithm was able to get an optimal solution to all problem instances on FACOM 230-38S except E. Balas and M. W. Padberg's one. But when we tried 200 rows and 2000 columns problem instance, then MPS/X stopped computing because it had used up all the external memory. E. Balas and M. W. Padberg's algorithm was unsuccessful because it demanded too much memory size for its Column Generating Procedure. This drawback was also observed by E. Balas's student Prof. Geritssen. The most promising algorithm of J. F. Pierce and J. S. Lasky continued computing for more than 70 hours, i.e., more than 7 days, each day with 10 hours. Finally we were asked to give up computing by our University Computing Center. So, we re-ran the same problem instance with a new counter variable in the program to see how many subproblems it created. It was not a million but more than a billion. But why so many? This is, still at present, the computational difficulty every NP-Complete/NP-Hard Problem has. In this case, we easily saw that even the computations like

$$z \leftarrow z + c_j$$

and

$$z \leftarrow z - c_j$$

were meaningless because single or double floating point numbers mechanism could not maintain computational accuracy. Declaring z, c_j Real 8 byte isn't safe enough, i. e., one has to keep costs c_j and z all in integers. Even if we had improved the original algorithm so that it should re-calculate objective function value each time it got a new feasible solution, it was still incapable of treating an infeasible input problem data. At the worst we hoped that J. F. Pierce and J. S. Lasky's algorithm would over-perform the others, but in fact, it wasn't true. No algorithm showed such uniform efficiency over the others. We were heavily shocked that we once more got the same result as in the case of Travelling Salesman problem. Exponential computing time and its heavy data dependency. In 1977. Garfinkel and Nemhauser sent us a letter that they had no source program because they had a business campany make an assembler program and so they had no right to send us a source program list. In their paper,

they wrote that they made a source program in FORTRAN! But we thought they were sincere, anyway they answered us. N. Christofides [1974–75] just sent us his new coming book with no letter, no source program. In these cases, we offered them our source program in exchange for theirs to solve the Set Partitioning Problem. E. Balas also wrote to us to try some Set Partitioning data of low density, from 1.5% down to 1%. Surprizingly enough, in Balas and Padberg' paper "On the set covering problem: II An algorithm for set partitioning", *Operations Research 23 (1975)*, 74–90, there appeared no sentences that their algorithm was devoted to low density Set Partitioning Problem. We confirmed low density data 3.7% caused density 57% when their algorithm got to the Block Privoting Procedure for the enlarged table. Devising a branch and bound algorithm enthusiastically to solve a specific IP problem lasted about 15 years.

5. Some Algorithmic Prospect to solve Real-World NP-Complete Problems

For almost 25 years up to today, there appeared lots of theoretical papers on the Complexity Theory. We can see its developments in M. R. Garey and D. S. Johnson [1979], A. V. Aho, J. E. Hopcroft and J. D. Ullman [1974] and T. Ibaraki [1994]. Yet, at present we can not think that 1970's integer programming problems are completely analyzed and explained through the Turing Machine based Complexity Theory. But the results of M. Li and P. M. B. Vitanyi [1989], Kobayashi [1994] coincide with the computational experiences in the Integer Programming. They say the following;

There is an input data distribution such that for any algorithm to solve a specific NP-Complete problem, its mean time complexity and its worst time complexity are of the same order.

In Y. Saruwatari, R. Hirabayashi and N. Nisida [1992], M. Kiuchi, Y. Shinano, Y. Saruwatari and R. Hirabayashi [1995], they report some problem instances which can not be solved within a suitable amount of computing time.

Concluding these experiences, the author thinks that we are in the third stage of the integer programming where an algorithm of some IP problem which has done well for just one Real-World input data should be appreciated appropriately. *Let's try to solve one Real-World input data for which we have to find an optimal/near optimal solution by any means.* Don't try to get a great amount of computational experiments so that we can ascertain that our new algorithm can solve some NP-Complete problem within such and such amount of time, which Dr. E. Maeda asked us in early 1970s. If one has found a precious optimal/near optimal solution for the Real-World imput data, then what else would be demanded by the other people. Never forget that when input data changes, so drastically does the computing time to get an optimal/near optimal solutions.

If we limit our problems to discrete optimization ones which has a fine structure like matroid and/or greedoid, then there might be some problems for which we will find a polynomial time algorithm. Minimum Spanning Tree, Shortest Parth, Flows in Network (R. S. Garfinkel and G. L. Nemhauser [1972], M. Iri [1969b], M. Iri et al. [1986]) fall all in this class.

So are optimization problems on Submodular Polyhedron (K. Iwamura [1995a,b], S. Fujishige [1991]). We point out that the theory of Submodular System can be traced back to the work of M. Iri [1984, 1979, 1969a, 1968]. There appears the notion of principal partitioning, some time apparently, some time veiled and unseen, in the theory of Submodular System. The author would like to say that dual supermodular polyhedron naturally comes out through principal partitioning and submodular system's poset. It's also noteworthy that a primaldual type theorem independent from LP duality is a useful one in combinatorial optimization. To see how Submodular Polyhedron came from network flow, the reader would be advised to consult M. Iri, S. Fujishige and T. Ooyama [1986]. For more developments of submodular functions, one can have a look at K. Murota [1995, 1996] (convexity), S. Fujishige [1991] (network flow) and M. Nakamura [1998] (principal partitioning).

The reference book [1981] written by T. Ibaraki is one of the best books that treats structural classification of discrete optimization problems from dynamic programming framework. Discrete decision processes and sequential decision processes(sdp). S. Iwamoto [1987] and M. Sniedovich [1992] will be eye-opening to those who want to know how powerful the Dynamic Programming is. One can catch the heart of the theory through T. Ibaraki [1973], too. In K. Iwamura [1993] it is revealed that greedy algorithm over a given greedoid can be captured within a framework of sequential decision processes. And so, the author would like to re-state here that notion of geedoids is a very wonderful one. Today, we can have its whole view by the book written by B. Korte, L. Lovász and R. Schrader [1991]. The author believes that greedoidal point of view would let the researchers in discrete optimization have a clear and better understanding for their problems at hand with efficient algorithms.

The author thinks neuro-computing framework, genetic algorithm, simulated annealing, tabu search, fuzzy computing framework, life span method and so on are all considered suitable in overcoming both exponential computing time and heavy data dependency. K. Iwamura and B. Liu [1996], B. Liu and K. Iwamura [1996, 1997], K. Tagawa, D. Okada, Y. Kanzaki, K. Inoue and H. Haneda [1998] and J. Xie and W. Xing [1998] are some successful examples in this direction.

References

1. A. V. Aho, J. E. Hopcroft and J. D. Ullman [1974]: *The Design and Analysis of computer algorithms*, Addison-Wesley.
2. E. Balas and M. W. Padberg [1972]: On the Set-Covering Problem, *ORSA*, Vol. 20, 1152–1161.
3. E. Balas and M. W. Padberg [1975]: On the Set-Covering Problem: II An Algorithm for Set-Partitioning, *ORSA*, Vol. 23, 74–90.
4. N. Christofides and S. Korman [1974–75]: A computational survey of methods for the set covering problem, *Management Science 21*, 591–599.
5. G. B. Dantzig [1963]: *Linear Programming and Extensions*, Princeton University Press.
6. S. Fujishige [1991]: *Submodular functions and optimization*, Elsevier Science Publishers.
7. M. Futakawa, T. Yamada and S. Kataoka [1995]: A heuristic algorithm to solve Max-Min knapsack problem, *Abstracts of the 1995 Fall National Conference of Operations Research Society of Japan*, 246–247.

8. M. R. Garey and D. S. Johnson [1979]: *Computers and Intractability A Guide to the Theory of NP-Completeness,* Freeman, San Francisco.
9. R. S. Garfinkel and G. L. Nemhauser [1969]: The Set Partitioning Problem: Set Covering with Equality Constraints, *Operations Research*, Vol. 17, 848–856.
10. R. S. Garfinkel and G. L. Nemhauser [1972]: *Integer Programming,* John Wiley and Sons, Inc.
11. Y. Hayashi [1995]: An efficient algorithm to solve the 0–1 knapsack problem when $J = 2^n - 1$, *Abstracts of the 1995 Fall National Conference of ORSJ,* 244–245 (in Japanese).
12. A. J. Hoffman [1963]: On simple linear programming problems, *Convexity.* American Mathematical Society.
13. T. C. Hu [1970] *Integer Programming and Network Flows*, Addison-Wesley
14. H. Ibaraki and M. Fukushima [1993]: *Saitekikano syuhou* (in Japanese), Kyouritusyuppan.
15. T. Ibaraki [1994]: Combinatorial Optimization and Its Complexity (in Japanese) Jinkouchinougakkaishi, Vol. 9 No. 3, 335–341.
16. T. Ibaraki [1973]: Solvable classes of discrete dynamic programming *J. of Mathematical Analysis and Applications,* Vol. 43, 642–693.
17. T. Ibaraki [1981]: *Theory of Combinatorial Optimization* (in Japanese), Coronashya.
18. A. Ikegami and A. Niwa [1995]: The Vehicle Routing Problem with Time Constraints, *J. of the Operations Research Society of Japan,* Vol. 38 No. 1, 107–123.
19. Y. Ikura [1976]: Private communication
20. M. Iri [1968]: An algebraic approach to the problem of topological degrees of freedom of a network, *Trans. Inst. Electronics and Communication Engineers of Japan,* Vol. 51A, No. 5, 180–187.
21. M. Iri [1969a]: The Maximum-Rank Minimum-Term-Rank Theorem for the Pivotal Transforms of a Matrix, *Linear Algebra and Its Applications,* Vol. 2, 427–446.
22. M. Iri [1969b]: *Network Flows, Transportation and Scheduling-Theory and Algorithms,* Academic Press, New York
23. M. Iri [1979]: A review of recent work in Japan on principal partitions of matroids and their applications, *Annals of the New York Academy of Sciences,* Vol. 319, 306–319.
24. M. Iri [1984]: Structural Theory for the Combinatorial Systems Characterized by Submodular Functions, in *W. R. Pulleyblank (ed.): Progress in Combinatorial Optimization,* Academic Press, 197–219.
25. M. Iri, S. Fujishige and T. Ohoyama [1986]: *Graph-Network-Matroid,* Sangyohotoshyo (in Japanese).
26. S. Iwamoto [1987]: *Dynamic Programming,* Kyushyudaigaku-Shuppankai (in Japanese).
27. K. Iwamura [1972]: A solution procedure of an all integer linear programming appearing in CFLP (based on branch and bound), (in talk), *Research Meeting of Mathematical Programming, Operations Research Society of Japan.*
28. K. Iwamura [1974]: On Some Theorems of Knapsack Functions, *Journal of the Operations Research Society of Japan,* Vol. 17, No. 4, 173–183.
29. K. Iwamura and Y. Maeda [1977a]: Core Saving Criterion for the MCGP of the Set-Partitioning Algorithm proposed by Balas and Padberg, Working Paper.
30. K. Iwamura and Y. Maeda [1977b]: A Computational Experiment of the Set Partitioning Algorithm proposed by Balas and Padberg, Working Paper.
31. K. Iwamura [1988]: Contracting Greedoids and a Rado-Hall Type Theorem, *Institut für Ökonometrie und Operations Research,* Universität Bonn.
32. K. Iwamura and Y. Deguchi [1992]: An inverse problem for the two-dimensional discrete constant spring system, *Suurikaisekikennkyujo-koukyuuroku 798*, 219–225.
33. K. Iwamura [1993]: Discrete Decision Process Model Involves Greedy Algorithm Over Greedoid, *Journal of Information and Optimization Sciences,* Vol. 14, 83–86.
34. K. Iwamura and G. Nakayama [1994]: An inverse problem for two-dimensional elastic body, *Research Report of Faculty of Sciences, Josai University*, Vol. 2, 25–31.

35. K. Iwamura and T. Nakayama [1994]: Drawing a Tree on Parallel Lines, Proc. of International Workshop on Intelligent Systems and Innovative Computations—The 6th Bellman Continuum—, 235-238.
36. K. Iwamura [1995]: Lexicographically optimal base of a submodular system with respect to a weight vector, *Journal of Information and Optimization Sciences,* Vol. 16, 49-60.
37. K. Iwamura [1995]: Primal Dual Algorithms for the Lexicographically Optimal Base of a Submodular Polyhedron and its Relation to a Poset Greedoid, *Journal of Systems Science and Systems Engineering,* Vol. 4 No. 3, 193-203.
38. K. Iwamura and B. Liu [1996]: A Genetic Algorithm for Chance Constrained Programming, *Journal of Information and Optimization Sciences,* Vol. 17 No. 2, 409-422.
39. K. Iwamura, Y. Deguchi and N. Okada [1998]: A Remark on Solving the Set-Partitioning Problem by Dual All Integer Algorithm, *Journal of Systems Science and Systems Engineering,* Vol. 7 No. 3, 363-367.
40. M. Kiuchi, Y. Shinano, Y. Saruwatari and R. Hirabayashi [1995]: Heiretu-Bunnshigenteihou wo motiita youryoutuki edajyunkairomonndai no gemmitukaihou Part 2 (in Japanese), Faculty of Engineering, Tokyo Science University.
41. C. M. Klein [1995]: A Submodular approach to discrete dynamic programming, *European Journal of Operations Research,* Vol. 80, 147-155.
42. K. Kobayashi [1992]: On existence of non- "a priori" malign measures, Faculty of Science, Tokyo Institute of Technology.
43. H. Konno and H. Suzuki [1982]: *Integer Programming and Discrete Optimization* (in Japanese), Nikkagiren-shuppansha.
44. B. Korte, L. Lovász and R. Schrader [1991]: *Greedoids,* Springer.
45. M. Kubo [1995]: Meta heuristics, *in Discrete Structures and Algorithms IV (ed. K. Murota, in Japanese)*, 171-230, Kindaikagakusha.
46. M. Kuroda [1976]: Private communication
47. T. -C. Lai, M. L. Brandeau and S. Chiu [1994]: An Approach for Worst Case Analysis of Heuristics: Analysis of a Flexible 0-1 Knapsack Problem, *J. of the Operations Research Society of Japan,* Vol. 37 No. 3, 197-210.
48. E. L. Lawler, J. K. Lenstera, A. H. G. Rinnooy Kan, and D. B. Shmoys [1987]: *The Traveling Salesman Problem,* John Wiley & Sons.
49. M. Li and P. M. B. Vitanyi [1989]: A Theory of Learning Simple Concepts Under Simple Distributions and Average Case Complexity for the Universal Distribution, *Proc. of the 30th FOCS,* 34-39.
50. S. Lin [1965]: Computer solutions of the traveling salesman problem, *Bell System Tech. J.,* Vol. 44, 2245-2269.
51. S. Lin and B. W. Kernighan [1973]: An Effective Heuristic Algorithm for the Traveling-salesman problem, *Operations Research,* Vol. 21, 498-516.
52. J. D. C. Little, K. J. Murty, D. W. Sweeney and C. Karel [1963]: An Algorithm for the Traveling Salesman Problem, *Operations Research,* Vol. 11, 979-989.
53. B. Liu and K. Iwamura [1997]: Modelling Stochastic Decision Systems Using Dependent-Chance Programming, *European Journal of Operational Research,* Vol. 101, 193-203.
54. B. Liu and K. Iwamura [1996]: A note on Chance Constrained Programming with Fuzzy Coefficients, *Fuzzy Sets and Systems,* Vol. 100, 229-233.
55. J. G. Lührs [1972]: Ein Einschliessungssatz zum Knapsack-problem, *Computing,* Vol. 9, 101-105.
56. E. Maeda and K. Iwamura [1982]: Set Partitioning Problem and Set Covering Problem, H. Konno and H. Suzuki(eds.) *Integer Programming and Combinatorial Optimization,* Nikkagiren-shuppansha (in Japanese).
57. S. Martello and P. Toth [1990]: *Kanpsack Problems,* John Wiley & Sons.
58. Y. Mineno [1978]: Private communication

59. H. Mukawa, J. Sensui, K. Iwamura and J. Kase [1971] : An Algorithm to solve the Capacitated Facilities Location Programming Problem, *Mitsubishi Research Institute* (in Japanese).
60. K. Murota [1995] : *Convexity and Steinitz's Exchange Property,* RIMS-1023, Research Institute for Mathematical Sciences, Kyoto Univ.
61. K. Murota [1996] : *Discrete Convex Analysis,* RIMS-1065, Research Institute for Mathematical Sciences, Kyoto Univ.
62. M. Nakamura [1988] : Structural Theorems for Submodular Functions, Plymatroids and Plymatroid Intersections, *Graphs and Combinatorics,* Vol. 4, 257-284.
63. J. F. Pierce and J. S. Lasky [1973] : Improved Combinatorial Programming Algorithms for a Class of All-Zero-One Integer Programming Problems, *Management Science,* Vol. 19, 528-543.
64. G. Reinelt [1994] : *The Traveling salesman,* Springer.
65. Y. Saruwatari, R. Hirabayashi and N. Nishida [1992] : Node Duplication Lower Bounds for the Capacitated Arc Routing Problem, *J. of the Operations Research Society of Japan,* Vol. 35 No. 2, 119-133.
66. A. Schrijver [1986] : *Theory of Linear and Integer Programming,* Wiley.
67. M. Sniedovich [1992] : *Dynamic Programming,* Dekker.
68. Y. Sorimachi [1970] : Optimal Planning of Distribution Center, *Suurikagaku,* September 23-279 (in Japanese).
69. H. Suzuki and K. Iwamura [1979] : Knapsack problem and Set partitioning (covering) problem, *Opereishonzu-risaachi, in Japanese,* June 1979, 359-368.
70. X. Tang and J. Gu [1996] : Soft System Approach to Management Support System Development, A lecture at the Dept. Math., Josai University.
71. K. Tagawa, D. Okada, Y. Kanzaki, K. Inoue and H. Haneda [1998] : Distance Based Hybrid Genetic Algorithm for Symmetric and Asymmetric Traveling Salesman Problems, *Beijing Mathematics,* Vol. 4 No. 2, 42-49.
72. J. Xie and W. Xing [1998] : Incorporating Domain-Specific Knowledge into Evolutionary Algorithms, *Beijing Mathematics,* Vol. 4 No. 2, 131-139.
73. U. Zimmermann [1981] : *Linear and Combinatorial Optimization in Ordered Algebraic Structures,* North Holland.