On strongly polynomial algorithms for some classes of quadratic programming problems*†

JADRANKA SKORIN-KAPOV‡

Abstract. *In this paper we survey some results concerning polynomial and/or strongly polynomial solvability of some classes of quadratic programming problems. The discussion on polynomial solvability of continuous convex quadratic programming is followed by a couple of models for quadratic integer programming which, due to their special structure, allow polynomial (or even strongly polynomial) solvability. The theoretical merit of those results stems from the fact that a running time (i.e. the number of elementary arithmetic operations) of a strongly polynomial algorithm is independent of the input size of the problem.*

Key words: *quadratic programming, polynomial algorithms, strongly polynomial algorithms, job scheduling problems*

Sažetak. **O strogo polinomijalnim algoritmima za neke klase problema kvadratnog programiranja.** *U ovom radu dajemo pregled nekih rezultata o polinomijalnoj i strogo polinomijalnoj rješivosti određenih klasa problema kvadratnog programiranja. Nakon diskusije o polinomijalnoj rješivosti problema kontinuiranog konveksnog kvadratnog programiranja, predočena su dva modela kvadratnog cjelobrojnog programiranja koji, zahvaljujući određenoj strukturi, dozvoljavaju polinomijalnu, pa čak i strogo polinomijalnu, rješivost. Teoretski je doprinos navedenih rezultata u činjenici da vrijeme (tj. broj elementarnih računskih operacija) strogo polinomijalnih algoritama ne ovisi o veličini ulaznih podataka.*

Ključne riječi: *kvadratno programiranje, polinomijalni algoritmi, strogo polinomijalni algoritmi, problemi raspoređivanja poslova*

AMS subject classifications: 90, 68

Received October 27, 1997

*The work was partially supported by the NSF grants DDM-8909206, DDM-9200292, DDM-937417, and SBR-9602021.

†Part of this paper is presented at the MATHEMATICAL COLLOQUIUM in Osijek organized by Croatian Mathematical Society – Division Osijek, May 23, 1997.

‡W. A. Harriman School for Management and Policy, State University of New York at Stony Brook, Stony Brook, NY 11794-3775, e-mail: jskorin@frost.har.sunysb.edu

1. Introduction

Many problems in economics, statistics and numerical analysis can be formulated as the optimization of a convex quadratic function over a polyhedral set. Moreover, some algorithms for solving large scale mathematical programming problems minimize a quadratic function over a polyhedral set as a subroutine (e.g. Held [17]). Several methods that are based on solving a quadratic programming subproblem to determine a direction of search were also suggested for optimization problems with nonlinear constraints (see for example Gill et al. [7]). The existence of efficient quadratic programming algorithms, and the fact that nonlinear functions can sometimes be accurately approximated by quadratic functions, led to development of approximation methods that make use of quadratic subproblems (e.g. Fletcher[5]). The above mentioned are just some of the reasons why quadratic programming arose as a very important part of the rich theory of Mathematical Programming. It can be viewed as a "bridge" between linear and (more difficult) nonlinear programming. Further, many real world problems require a formulation in which all or some of the variables are restricted to be integral. Since a quadratic objective function enables one to take into account the interactions between variables, many applications have natural representations as 0-1 quadratic programming problems (e.g. finance [23] and capital budgeting [22]). In general, the efficiency of an algorithm is very often measured via its *running time*, i.e. the number of arithmetic operations performed in it. To that end, let us introduce few definitions in order to define a polynomial and a strongly polynomial algorithm. The *dimension* of the problem is the number of data in the input of a given optimization problem. The *size* of a rational number is the length of its binary description (i.e. the number of bits needed to record a given number in a binary format). The size of a rational vector is the sum of sizes of its components. With these definitions, an algorithm is termed *polynomial* if it has running time polynomial in the dimension of the problem and in the input size and, when applied to rational input, the size of numbers occurring in it is polynomially bounded by the dimension of the problem and the size of input. An algorithm is *strongly polynomial* if its running time is polynomially bounded in the dimension of the input, independent of the size of the input. Further, when the algorithm is applied to rational input, then the size of the numbers occurring during the algorithm is polynomially bounded in the dimension of the input and the size of the input numbers. Note that a merit of a strongly polynomial algorithm lays in the fact that its running time is independent of the input size. This paper surveys some results dealing with polynomial solvability of quadratic programming problems. We first briefly discuss results leading in the direction of strongly polynomial solvability of a subclass of continuous convex quadratic programming problems, and its significance for algorithmic complexity of related $0-1$ quadratic programming problems. This is followed by presentation of two integer quadratic programming models which, due to their special structures, allow polynomial and/or strongly polynomial solvability. Finally, this survey is summarized and one direction for future research is outlined.

2. On polynomial solvability of convex quadratic programming

In 1979 Hačijan [15] proved that a linear programming problem is polynomially solvable. In the same year, Kozlov, Tarasov and Hačijan [21] extended this result and proved polynomial solvability of convex quadratic programming. The question whether linear programming is, in general, strongly polynomially solvable is still open. However, a significant step in that direction was provided by Tardos [25] who presented a polynomial algorithm for linear programs in which the number of arithmetic steps depends only on the size of the numbers in the constraint matrix, and is independent of the size of numbers on the right hand side and the cost coefficients. Granot and Skorin-Kapov [10] extended this result to strictly convex quadratic programming problems of the form

$$max\{c^T x - \frac{1}{2}x^T Dx : Ax \leq b, x \geq 0\}, \tag{1}$$

with D being a positive definite matrix. Their algorithm finds optimal primal and dual solutions to the quadratic programming problem (if they exist) by solving a sequence of simple quadratic problems (having zero right hand side vector and linear part of the objective function polynomially bounded by matrices A and D) via Kozlov et al.'s algorithm, and by checking feasibility of a linear system in time independent of the right hand side using Tardos' feasibility algorithm. Denoting by $T(A)$ the complexity of Tardos' feasibility algorithm, by $K(A, D)$ the complexity of Kozlov's et al.'s algorithm applied to modified quadratic subproblems, and by Δ the maximal absolute determinant over all square submatrices of the matrix $(D, A^T, -I)$, the main Granot and Skorin-Kapov's result was the following theorem:

Theorem 1. (Theorem 3.5 from reference [10]) *The quadratic programming algorithm applied to Problem (1) has running time polynomial in the size of the matrices A and D and independent of the sizes of the vectors c and b. It runs in* $O(n(2n+m)^3 + n(2n+m)\log(2n+m)\Delta + T(A) + nT(A,D) + nK(A,D))$ *time.*

As mentioned earlier, in many applications of quadratic programming, integrality of some or all of the variables is required. For example, problems of the type

$$max\{c^T x - \frac{1}{2}x^T Dx : Ax \leq b, x \in \{0,1\}\}, \tag{2}$$

arise naturally in finance [23] and capital budgeting [22]. Different approaches for solving the above problem can be found in the literature, including linearization methods in which the quadratic problem is transformed into a linear 0-1 or mixed-integer program (see Watters [26], Glover [8]). Algorithms based on branch and bound have been proposed by many authors (e.g. Laughunn [22], Hansen [16]). McBride and Yormark [24] gave an implicit enumeration algorithm in which at each node they solve a quadratic programming relaxation of a corresponding integer subproblem using Lemke-Howston's complementary pivoting algorithm. It is conceivable that a success of such an implicit enumeration algorithm depends on the efficiency of the quadratic programming algorithm used, and on the distance between optimal integer and corresponding continuous solutions. Although

the polynomiality of an algorithm can not always be identified with real world computational efficiency or practicality, it is an important theoretical result which leads the research efforts in the direction of constructing efficient problem oriented polynomial algorithms. In a related work dealing with proximity and sensitivity of integer and mixed-integer quadratic programs, Granot and Skorin-Kapov [11] have shown that for any optimal solution $\bar{z}$ for a given separable quadratic integer programming problem, there exists an optimal solution $\bar{x}$ for its continuous relaxation, such that $\|\bar{z} - \bar{x}\|_\infty \leq n\Delta(A)$, where n is the number of variables and $\Delta(A)$ is the largest absolute subdeterminant of the integer constraint matrix A. The extension to mixed-integer nonseparable quadratic case was also given. Further, Granot and Skorin-Kapov [9] have shown how to replace the objective function of a quadratic $0-1$ programming problem with n variables by an objective function with integral coefficients whose size is polynomially bounded by n, without changing the set of optimal solutions. This result assures that the running time of any algorithm for solving $0-1$ programming problems can be made independent of the size of the objective function coefficients. This since the equivalent problem can then be solved by e.g. an implicit enumeration algorithm in which at each node the continuous relaxation of the corresponding integer subproblem is solved in polynomial time independent of the size of the objective function coefficients. Although, in general, quadratic integer programming problems are NP-hard, there are some special cases solvable to optimality in a reasonable computational time. In the sequel we will outline two special cases of quadratic integer programming problems allowing for polynomial solvability. Both applications deal with job scheduling. We first present an application involving a *single* machine, followed by an application of job scheduling on *multiple* machines.

3. On polynomial solvability of the high multiplicity total weighted tardiness problem

Hochbaum, Shamir and Shantikumar [19] developed a polynomial algorithm for solving a job scheduling problem on a *single machine*. The problem consists of minimizing the total weighted tardiness for a large number of unit length jobs which can be partitioned into few sets of jobs having identical due dates and penalty weights. Let us denote by n the number of sets of unit jobs. Set i includes p_i jobs having the same due date d_i and penalty weight w_i. The number of unit jobs in a set (p_i) is called the multiplicity of that set, and $P = \sum_{i=1}^{n} p_i$ is the total number of unit jobs. The objective is to find an assignment of the unit jobs to the P distinct time intervals $(i-1, i], i = 1, ..., P$ which will minimize the total weighted tardiness. The weighted tardiness of each unit of type i scheduled at interval $(t-1, t]$ is $w_i \times max(t - d_i, 0)$. One can assume without loss of generality that $w_1 \geq w_2 \geq ... \geq w_n > 0$, and can permute the types of jobs so that $d_{\pi(1)} < d_{\pi(2)} < ... < d_{\pi(n)}$. The i-th due date interval is defined as $(d_{\pi(i)}, d_{\pi(i+1)})$ for $i = 0, ..., n$. The weight of job j in interval i, denoted by $w_j^{(i)}$, is defined to equal zero if job j is not tardy. Assume without loss of generality that all due dates are distinct, that is, there are no empty intervals. Denote $d_{\pi(0)} = 0$ and $d_{\pi(n+1)} = P$. Using this notation, the following integer programming formulation of the total weighted tardiness problem

was presented in [19]:

$$\min \sum_{i=0}^{n}\sum_{j=1}^{n}[w_j^{(i)}(d_{\pi(i)} - d_j + \frac{1}{2})x_{ij} + w_j^{(i)}(\frac{1}{2}x_{ij}^2 + x_{ij}\sum_{k|w_k^{(i)}\geq w_j^{(i)},k<j} x_{ik})] \quad (3)$$

s.t.

$$\sum_{i=0}^{n} x_{ij} = p_j, \ \ j = 1, ..., n, \quad (4)$$

$$\sum_{j=1}^{n} x_{ij} = d_{\pi(1+1)} - d_{\pi(i)}, \ \ i = 0, ..., n, \quad (5)$$

$$x_{ij} \geq 0 \text{ and integer}, \ \ \forall\, i, j. \quad (6)$$

In [19] this problem was solved in polynomial time, independent of the size of the multiplicities and the due dates, but depending on the penalty weights. Granot and Skorin-Kapov [13] developed algorithms to solve this problem in polynomial time which is *independent* of the sizes of the weights. The running time of their algorithms depends only on the dimension of the problem and on the size of the maximal difference between consecutive due dates. They suggest two alternatives: either solve a single related continuous quadratic program and then use a rounding procedure to obtain the required integral solution, or else solve a sequence of linear programming problems where the number of problems solved is bounded by the size of the maximal difference between two consecutive due dates. If the size of the maximal difference between consecutive due dates is polynomially bounded by the dimension of the problem, both alternatives from [13] will result with algorithms which run in *strongly polynomial* time. In other words, the proposed algorithms are strongly polynomial in the case when all due dates are "very big", that is, all are clustered close to the sum of all multiplicities.

4. On strongly polynomial solvability of a class of quadratic scheduling problems

A study by Hochbaum and Shamir [19] has considered a variety of high multiplicity problems on a single machine. Granot, Skorin-Kapov and Tamir [14] have analyzed the following *multimachine*, high multiplicity scheduling problem. Suppose that for $j = 1, ..., n$, there are d_j unit time jobs of type j available for processing at time $t = 0$ (in this formulation d_j is the multiplicity of type j). Each one of these d_j jobs is associated with a weight factor, r_j, and it must be processed in its entirety by one of m (parallel) machines. The job types are ordered according to their weight factors and renamed so that $r_1 > r_2 > ... > r_n > 0$. The machines may differ in their processing rates, as well as in their availability. Specifically for i, $i = 1, ..., m$, we assume that it takes the i-th machine t_i time units to process any unit time job. The i-th machine is released at time $t = q_i$, and is available for processing only in the time interval $[q_i, q_i + c_i]$. The objective is to schedule the set of $\sum_{j=1}^{n} d_j$ jobs and sequence them on the m machines so that the total weighted flow time of the jobs is minimized. The problem becomes NP-hard if the jobs are allowed to be of

variable length, even when $d_j = 1$, $j = 1, ..., n; m = 2$, and the two machines are identical (see for example Garey and Johnson, 1979). Letting x_{ij} be the number of unit time jobs of type j scheduled to machine i, the problem can be formulated as a nonseparable quadratic integer programming problem with transportation type constraints as follows:

$$\min \sum_{i=1}^{m}\sum_{j=1}^{n} q_i r_j x_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{n} t_i r_j \sum_{k=1}^{x_{ij}} (\sum_{l=1}^{j-1} x_{il} + k) \tag{7}$$

s.t.

$$\sum_{i=1}^{m} x_{ij} = d_j, \ \ j = 1, ..., n, \tag{8}$$

$$\sum_{j=1}^{n} x_{ij} = c_i, \ \ i = 1, ..., m, \tag{9}$$

$$x_{ij} \geq 0 \text{ and integer}, \ \ \forall i, j. \tag{10}$$

We first show how to replace the nonseparable objective function in this problem by an equivalent separable one. Denoting by $u_{ik} = \sum_{t=1}^{k} x_{it}$, Problem (7-10) can be equivalently written as

$$\min \sum_{i=1}^{m}\sum_{j=1}^{n} (q_i + \frac{1}{2}t_i) r_j x_{ij} + \frac{1}{2}\sum_{i=1}^{m} t_i \sum_{j=1}^{n} (r_j - r_{j+1}) u_{ij}^2 \tag{11}$$

s.t.

$$\sum_{i=1}^{m} x_{ij} = d_j, \ \ j = 1, ..., n, \tag{12}$$

$$\sum_{j=1}^{n} x_{ij} = c_i, \ \ i = 1, ..., m, \tag{13}$$

$$\sum_{j=1}^{k} x_{ij} - u_{ik} = 0, \ \ i = 1, ..., m; \ k = 1, ..., n, \tag{14}$$

$$x_{ij} \geq 0 \text{ and integer}, \ \ \forall i, j. \tag{15}$$

Note that the nonnegativity as well as the integrality of the u_{ij} variables follows directly from the nonnegativity and the integrality of the x_{ij} variables and hence is not included. If we further denote by $u_{i0} \equiv 0$, $i = 1, ..., m$, we obtain $x_{ij} = u_{ij} - u_{ij-1}$, $j = 1, ..., n$. Since $u_{in} = c_i$, $i = 1, ..., m$, using the fact that $\sum_{j=1}^{n} d_j = \sum_{i=1}^{m} c_i$, and further denoting $\sum_{i=1}^{m} u_{i1} = d_1 \equiv D_1$ and $\sum_{i=1}^{m} u_{ij} = \sum_{i=1}^{j} d_j \equiv D_j$ for $j = 1, ..., n$, one obtains an equivalent problem to Problem (7-10) in terms of the $m(n-1)$ u variables. For simplicity we denote by $p_i \equiv \frac{1}{2}t_i$. The equivalent formulation is then:

$$\min \sum_{j=1}^{n-1} \frac{1}{2}(r_j - r_{j+1}) \sum_{i=1}^{m} \frac{1}{t_i}(t_i u_{ij} + p_i)^2 \tag{16}$$

s.t.

$$\sum_{i=1}^{m} u_{ij} = D_j, \quad j = 1, ..., n-1, \tag{17}$$

$$u_{ij-1} \leq u_{ij} \leq c_i, \quad i = 1, ..., m;\ j = 1, ..., n-1, \tag{18}$$

$$u_{ij} \quad \text{integer}, \quad i = 1, ..., m;\ j = 1, ..., n-1. \tag{19}$$

Granot, Skorin-Kapov and Tamir [14] show how to solve the problem in strongly polynomial time with the solution depending only on the order of the r_j's, but not on their actual values. To that end, consider first the relaxation of (16-19) obtained by omitting the constraints $u_{ij-1} \leq u_{ij}$, $i = 1, ..., m$; $j = 1, ..., n-1$. Since $(r_j - r_{j+1})$ is a positive constant for $j = 1, ..., n-1$, the solution to the relaxed problem can be found by solving $(n-1)$ independent quadratic knapsack problems of the form

$$\min \sum_{i=1}^{m} \frac{1}{t_i}(t_i u_{ij} + p_i)^2 \tag{20}$$

s.t.

$$\sum_{i=1}^{m} u_{ij} = D_j, \tag{21}$$

$$0 \leq u_{ij} \leq c_i, \quad i = 1, ..., m, \tag{22}$$

$$u_{ij} \quad \text{integer}, \quad i = 1, ..., m. \tag{23}$$

The right hand side coefficients, $D_1, D_2, ..., D_{n-1}$ corresponding to problems for $j = 1, ..., n-1$ respectively, form a monotonically increasing sequence. Therefore, it follows from the validity of the general, the so called 'marginal allocation' or 'incremental' algorithm (see Ibaraki and Katoh [20], section 4.2), that for each j, $j = 1, ..., n-2$, if $\{u^*_{ij}\}, i = 1, ..., m$, is an optimal solution to the knapsack problem j, there exists an optimal solution $\{u^*_{ij+1}\}, i = 1, ..., m$, to the problem for $j+1$ such that $u^*_{ij} \leq u^*_{ij+1}, i = 1, ...m$. In particular, for $j = 1, ..., n-1$, these solutions $\{u_{ij}\}, i = 1, ..., m$, satisfy the relaxed constraints, and therefore constitute an optimal solution to Problem (16-19). These solutions were inductively generated by using the linear time algorithm of Brucker [4] to solve the continuous relaxations, and then using the $O(m)$ rounding scheme given in Ibaraki and Katoh [20 (Theorem 4.6.2, page 76), to obtain the optimal integral solution. The above algorithm translates basically to the following optimal procedure: consider first the job type with the largest weight and schedule all jobs of this type optimally by solving a related quadratic knapsack problem; adjust the available time interval on the machines and continue the process until all jobs are scheduled. This discussion implies the following result.

Theorem 2. (Theorem 1 from reference [14]) *Problem (7-10) can be solved in $O(mn)$ time. Moreover, the solution depends on the order of the r_j's but is independent of their magnitudes. If we include the preprocessing time, i.e. the time to sort the weights r_j in a decreasing order, the total computational complexity is $O(mn + n \log n)$.*

Granot, Skorin-Kapov and Tamir [14] further develop a parametric algorithm to efficiently solve the *identical processing rate* case (i.e. the case when $p_{ij} = 1, w_{ij} = w_j; i = 1, ..., m; j = 1, ..., n$). The parametric algorithm can be implemented in $O(m \log m)$ time and in $O(m)$ space by applying conventional data structures (see e.g. Aho, Hopcroft and Ullman [1]). This led to the following result.

Theorem 3. (Theorem 2 from reference [14]) *The total time needed to solve the identical rates version of Problem (7-10) is $O(m \log m + n \log n)$. If the job types are already sorted by the weight factor, the identical rates version of Problem (7-10) is solvable in $O(min(m^2 + n; (m + n) \log m))$ time.*

Some additional applications and extensions, as well as the relationship between the above model and the problem of scheduling n different types of jobs of variable processing lengths on m parallel machines to minimize the total unweighted flow time, are also discussed in [14].

5. Summary and directions of future research

In this paper we have surveyed some polynomial algorithmic aspects of certain classes of quadratic continuous and integer programming problems. The mentioned results include the first strongly polynomial algorithm for a certain class of strictly convex quadratic programming problems [10], some proximity results for integer quadratic programs [11], and simultaneous approximations of objective function coefficients for integer quadratic programs [9]. More recent work includes results on polynomial solvability of a class of nonseparable quadratic integer programming problem with transportation type constraints, with applications to scheduling jobs on a single machine (the high multiplicity total weighted tardiness problem [13]), and scheduling jobs on parallel machines (the problem of scheduling n different types of unit jobs on m parallel machines so as to minimize the total weighted flow time of the jobs [14]). For illustration, more details were provided concerning this last work. Finally, we mention an application of this formulation which has motivated our study (see [12]). It is the following transportation sequencing problem. Suppose that after the completion of a certain manufacturing process which takes place at a plant, n types of products are being released. Let d_j be the number of units of product type j released. The products need now to be transferred for additional processing at any one of m stations. The distance between station i and the plant is q_i units of time, while c_i is the total processing capacity at station i. Each product has a different carrying cost rate which increases in time. Let r_j be the carrying cost rate of one unit of product j per one unit of time. Let t_i denote the processing rate at station i (which is independent of the product). The overall objective is to find for each $j = 1, ..., n$; $i = 1, ..., m$, quantities of product type j to be processed through

station i so as to minimize the total carrying (i.e. transportation and processing) cost. As an example consider the following simplified toxic waste disposal problem faced by a plant that generates n types of toxic wastes. Each toxic waste of type j has an impact on the environment which increases in time at a given rate, say r_j. After all wastes are generated they need to be transported for disposal at one of m given stations with disposal rate t_i at station i. The problem of disposing of all generated wastes while minimizing the total impact on the environment can be cast as an instance of the above described model.

Future research would include identification of optimization problems arising in *communication networks* which could be solved to optimality in polynomial or strongly polynomial time. Possibilities include examples of specially structured scheduling problems arising in: (1) multiprocessor computer architectures; and (2) communication networks with multiaccess channels (communication channels that can be accessed by several users, but only one user can transmit at any time). In the latter case, one can divide the communication resource of the channel into a portion used for packet transmission, and another used for reservation messages that coordinate the packet transmission. Therefore, the time is divided into data intervals (where actual data is transmitted) and reservation intervals (for scheduling transmissions from several packet streams). (See Bertsekas and Gallager [3].) Barcaccia and Bonuccelli [2] considered the time slot assignment problem (TSA) for variable bandwidth switching systems. An example of such system is a satellite-switched time division multiple access system in which there are different traffic rates for different zones covered by the satellite spot beams, in order to make more efficient use of the system by minimizing bandwidth wastes and increasing system's throughput. Time division multiple access (TDMA) communication needs to exhibit control of the user's access to the common communication medium, so that the transmission is collision-free. At the same time, TDMA communication has to be as efficient as possible. A measure od efficiency can be the speed of transferring the user's traffic. Barcaccia and Bonuccelli [2] have proposed polynomial algorithm for finding minimal length time slot assignment. An open question, we want to concentrate on, is whether there exists a strongly polynomial algorithm for this problem.

Acknowledgment

The research outlined in this survey was partially supported by National Science Foundation grants DDM-8909206, DDM-9200292, DDM-937417, and SBR-9602021.

References

[1] A. V. Aho, J. E. Hopkroft, J. D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, 1974.

[2] P. Baraccia, A. Bonuccelli, *Polynomial time optimal algorithms for time slot assignment of variable bandwidth systems*, IEEE/ACM Transaction on Networking **2**(1994), 247–251.

[3] D. Bertsekas, R. Gallager, *Data Networks*, Prentice Hall, 1987.

[4] P. Brucker, *An Algorithm for quadratic knapsack problems*, Operations Research Letters **3**(1984), 163–166.

[5] R. Fletcher, *An algorithm for solving linearly constrained optimization problems*, Mathematical Programming **2**(1972), 133–165.

[6] M. R. Garey, D. S. Johnson, *Computers and Intractability: a Guide to the Theory of NP-completeness*, Freeman, San Francisco, 1979.

[7] P. E. Gill, W. Murray, M. A. Saunders, M. H. Wright, *QP-Based Methods for Large Scale Nonlinearly Constrained Optimization*, in: Nonlinear Programming **4** (Mangasarian, Meyer and Robinson, Eds.), Academic Press, New York, 1981, 455–460.

[8] F. Glover, *Improved linear integer programming formulations of nonlinear integer programs*, Management Science **22**(1975), 455–460.

[9] F. Granot, J. Skorin-Kapov, *Simultaneous approximation in quadratic integer programming*, Operations Research Letters **8**(1989), 251–255.

[10] F. Granot, J. Skorin-Kapov, *Towards a strongly polynomial algorithm for strictly convex quadratic programs: An extension of Tardos' algorithm*, Mathematical Programming **46**(1990), 225–236.

[11] F. Granot, J. Skorin-Kapov, *Some proximity and sensitivity results in quadratic integer programming*, Mathematical Programming **47**(1990), 259–268.

[12] F. Granot, J. Skorin-Kapov, *Strongly polynomial solvability of a nonseparable quadratic integer program with applications to toxic waste disposal*, in: Conference Proceedings KOI'92 (V. Bahovec, Lj. Martić and L. Neralić, Eds.) Zagreb, 1992, 95–105.

[13] F. Granot, J. Skorin-Kapov, *On polynomial solvability of the high multiplicity total weighted tardiness problem*, Discrete Applied Mathematics **41**(1993), 139–146.

[14] F. Granot, J. Skorin-Kapov, A. Tamir, *Using quadratic programming to solve high multiplicity scheduling problems on parallel machines*, Algorithmica **17**(1997), 100–110.

[15] L. G. Hačijan, *A polynomial algorithm in linear programming*, Soviet Math. Dokl. **20**(1979), 191–194.

[16] P. Hansen, *Quadratic Zero-One Programming by Implicit Enumeration*, in: Numerical Methods in Nonlinear Optimization (F.A. Lootsma, Ed.), Academic Press, 1972, 265–278.

[17] M. Held, P. Wolfe, H. Crowder, *Validation of subgradient optimization*, Mathematical Programming **6**(1974), 62–88.

[18] D. S. Hochbaum, R. Shamir, *Strongly polynomial algorithms for the high multiplicity scheduling problem*, Operations Research **39**(1991), 648–653.

[19] D. S. Hochbaum, R. Shamir, J. G. Shantikumar, *A polynomial algorithm for an integer quadratic transportation problem*, Mathematical Programming **55**(1992), 359–371.

[20] T. Ibaraki, N. Kaoth, *Resource Allocation Problems: Algorithmic Approaches*, The MIT Press, 1988.

[21] M. K. Kozlov, S. P. Tarasov, L. G. Hačijan, *Polynomial solvability of convex quadratic programming*, Soviet Math. Dokl. **20**(1979), 1108–1111.

[22] D. J. Laughhunn, *Quadratic binary programming with applications to capital budgeting problems*, Operations research **10**(1970), 454–467.

[23] J. Lintner, *The valuation of risk assets and the selection of risky investments in stock portfolios and capital budgets*, Rev. Econ. Statist. **47**(1965), 13–37.

[24] R. D. Mcbride, J. S. Yormark, *Finding All Solutions for a Class of Parametric Quadratic Integer Programming Problems*, Management Science **26**(1980), 795–884.

[25] E. Tardos, *A strongly polynomial algorithm to solve combinatorial linear programs*, Operations Research **34**(1986), 250–256.

[26] L. G. Watters, *Reduction of integer polynomial problems to zero-one linear programming problems*, Operations Research **15**(1967), 1171–1174.