Improving a Formulation of the Quadratic Knapsack Problem

Daniel J. Grainger, Adam N. Letchford[1]

Department of Management Science, The Management School, Lancaster University, Lancaster LA1 4YX, UK

Abstract

The Quadratic Knapsack Problem can be formulated, using an idea of Glover, as a mixed 0-1 linear program with only $2n$ variables. We present a simple method for strengthening that formulation, which gives good results when the profit matrix is dense and non-negative.

Keywords: quadratic knapsack problem, integer programming.

1 Introduction

The *Quadratic Knapsack Problem* (QKP) is the generalisation of the classical 0-1 knapsack problem obtained when the objective function is permitted to be quadratic. It takes the following form:

$$\max \left\{ x^T Q x : w^T x \leq c, x \in \{0,1\}^n \right\},$$

where $x \in \{0,1\}^n$ is the vector of decision variables, $Q = \{q_{ij}\} \in \mathbb{Z}^{n \times n}$ is the profit matrix, $w \in \mathbb{Z}_+^n$ is the weight vector and $c \in \mathbb{Z}_+$ is the knapsack capacity. Note that there is no need for a linear term in the objective, since $x_i^2 = x_i$ when x_i is binary.

Frequently, it is assumed that $q_{ij} \geq 0$ for all $i \neq j$. Sometimes it is also assumed that $q_{ii} \geq 0$ for all i. We refer the reader to Kellerer *et al.* [14] and Pisinger [15] for surveys on applications, formulations, relaxations and solution methods for the QKP.

Quadratic 0-1 problems like the QKP are often *linearised*, by using additional variables and constraints, to make them more amenable to solution by integer programming. The standard linearisation (Glover & Woolsey [12]) introduces an additional $\binom{n}{2}$ binary variables, each representing the product of a pair of original variables. This approach has been applied successfully to the QKP by, for example, Billionnet & Calmels [5], Billionet *et al.* [6], Caprara *et al.* [8] and Pisinger *et al.* [16].

[1]Corresponding author. Tel.: +44-1524-594719. Fax: +44-1524-844885. E-mail address: `a.n.letchford@lancaster.ac.uk`

In this paper, however, we are concentrating on a rather different linearisation method, proposed originally by Glover [10], that uses only n additional variables. We show that, when a QKP instance satisfies a certain property which often holds in practice, Glover's linearisation can be strengthened. Computational results show that this strengthening is useful when the profit matrix Q is dense and non-negative.

The structure of the paper is as follows. In Section 2, we present the linearisation of Glover and review various improvements that have been proposed in the literature. In Section 3, we present the new improvement procedure. Some computational results are given in Section 4.

2 Glover's Linearisation

In 1975, Glover [10] presented a linearisation technique for 0-1 linear programs with quadratic objectives, which involves the introduction of only n additional variables and $4n$ additional constraints. With our notation, and specialised to the QKP, the argument proceeds as follows. For each i, define the quantity

$$y_i = x_i \left(\sum_{j \neq i} q_{ij} x_j \right).$$

Note that the objective function of the QKP is equal to $\sum_{i=1}^n q_{ii} x_i + \sum_{i=1}^n y_i$. Now let L_i and U_i, respectively, be lower and upper bounds on the value that $\sum_{j \neq i} q_{ij} x_j$ can take in any QKP solution. Then, we add extra linking constraints to ensure that y_i takes the correct value when x_i takes the value 0 or 1. The resulting formulation is:

$$\begin{array}{lrll}
\max & \sum_{i=1}^n q_{ii} x_i + \sum_{i=1}^n y_i & & \\
\text{s.t.} & y_i \geq L_i x_i & (1 \leq i \leq n) & (1) \\
& y_i \leq U_i x_i & (1 \leq i \leq n) & (2) \\
& y_i \geq \sum_{j \neq i} q_{ij} x_j - U_i(1 - x_i) & (1 \leq i \leq n) & (3) \\
& y_i \leq \sum_{j \neq i} q_{ij} x_j - L_i(1 - x_i) & (1 \leq i \leq n) & (4) \\
& w^T x \leq c & & \\
& x \in \{0,1\}^n & & \\
& y \in \mathbb{R}^n. & &
\end{array}$$

To see that this formulation is valid, note that the first two sets of constraints ensure that $y_i = 0$ when $x_i = 0$, but do not restrict y_i when $x_i = 1$. Similarly, the next two sets of constraints ensure that $y_i = \sum_{j \neq i} q_{ij} x_j$ when $x_i = 1$, but do not restrict y_i when $x_i = 0$.

As Glover himself noted, the values chosen for L_i and U_i affect the size of the feasible region of the LP relaxation, and therefore the upper bounds obtained in a branch-and-bound algorithm. One can set L_i and U_i to their best possible values by solving 0-1 knapsack problems (KPs). That is, one can set:

$$L_i := \min\left\{\sum_{j\neq i} q_{ij} z_j :\ w^T z \leq c,\ z \in \{0,1\}^n\right\}$$

and

$$U_i := \max\left\{\sum_{j\neq i} q_{ij} z_j :\ w^T z \leq c,\ z \in \{0,1\}^n\right\}.$$

These 0-1 KPs can be solved by a specialised dynamic programming or branch-and-bound algorithm, but, in practice, one can often solve them easily and quickly using a general-purpose ILP solver. Otherwise, a faster alternative is to solve the LP relaxations of the above 0-1 KPs, which can be done in linear time for each i (Balas & Zemel [3]).

As noted by Adams *et al.* [2], the constraints (1)-(4) can in fact be strengthened to:

$$y_i \geq L_i^1 x_i \qquad (1 \leq i \leq n) \tag{5}$$

$$y_i \leq U_i^1 x_i \qquad (1 \leq i \leq n) \tag{6}$$

$$y_i \geq \textstyle\sum_{j\neq i} q_{ij} x_j - U_i^0 (1 - x_i) \qquad (1 \leq i \leq n) \tag{7}$$

$$y_i \leq \textstyle\sum_{j\neq i} q_{ij} x_j - L_i^0 (1 - x_i) \qquad (1 \leq i \leq n), \tag{8}$$

where L_i^1 and U_i^1 are lower and upper bounds, respectively, on the value that $\sum_{j\neq i} q_{ij} x_j$ can take in any QKP solution such that $x_i = 1$, and L_i^0 and U_i^0 are lower and upper bounds, respectively, on the value that $\sum_{j\neq i} q_{ij} x_j$ can take in any QKP solution such that $x_i = 0$. The best values for these constants can again be computed exactly by solving 0-1 KPs, or approximately by solving LP relaxations.

Billionnet & Soutif [7] show that, if a good integer solution to the QKP is available, perhaps obtained with a heuristic algorithm, the upper and lower bounds L_i^1, U_i^1, L_i^0 and U_i^0 can be improved further, by solving a series of LPs. Their procedure cuts off some integer solutions to the QKP, but only those whose profit is worse than that of the known integer solution.

Our new strengthening procedure is similar in spirit to that of Billionnet & Soutif [7], in that it cuts off sub-optimal integer solutions of the QKP. However, it is much faster and simpler, since it does not require a good integer solution and does not require any LPs to be solved.

A few other ways to improve Glover's linearisation have been proposed in the literature; see for example Glover [11], Adams & Forrester [1] and Adams *et al.* [2]. For the sake of brevity, we do not describe them here.

3 The New Strengthening Procedure

As mentioned in the introduction, our new strengthening procedure is applicable only to QKP instances that satisfy a certain property. We will need the following two definitions.

Definition 1 *A feasible solution x^* to the QKP will be called* maximal *if changing the value of any variable from 0 to 1 causes the instance to become infeasible, i.e., if, for any i such that $x_i^* = 0$, we have $w^T x^* + w_i > c$.*

Intuitively speaking, the knapsack is 'fully packed' in a maximal solution.

Definition 2 *A QKP instance will be called* well-behaved *if there exists at least one optimal solution which is maximal.*

Intuitively, if a QKP instance is well-behaved, it means that the knapsack constraint is 'binding'.

We will show that, if we know that an instance is well-behaved, then we can exploit this fact to compute extremely tight lower bounds L_i^1 and L_i^0. Unfortunately, it seems likely to be $\mathcal{NP}$-hard to test if an instance is well-behaved. However, one can easily derive simple sufficient conditions for instances to be well-behaved. In particular, QKP instances in which all q_{ij} are non-negative are obviously well-behaved, since packing additional items into the knapsack cannot cause the profit to decrease.

If we do indeed know that an instance is well-behaved, then we can in theory set

$$L_i^1 := \min \left\{ \sum_{j \neq i} q_{ij} z_j : \ w^T z \leq c, \ z \in \{0,1\}^n, \ z_i = 1, \ z \text{ maximal} \right\}$$

and

$$L_i^0 := \min \left\{ \sum_{j \neq i} q_{ij} z_j : \ w^T z \leq c, \ z \in \{0,1\}^n, \ z_i = 0, \ z \text{ maximal} \right\}.$$

According to Kellerer [13], however, computing these values exactly is itself $\mathcal{NP}$-hard. So we recommend instead using more easily computable lower bounds on these values. We will need the following two lemmas:

Lemma 1 *If x^* is maximal, then $w^T x^* \geq c - w_{\max} + 1$, where $w_{\max} = \max_{1 \leq i \leq n} w_i$.*

Proof. If $w^T x^* \leq c - w_{\max}$, then another item can be inserted into the knapsack without exceeding the knapsack capacity, i.e., there exists some i such that $x_i^* = 0$ and $w^T x^* + w_i \leq c$. Therefore x^* is not maximal. $\square$

Lemma 2 *If x^* is maximal and $x_i^* = 0$, then $w^T x \geq c - w_i + 1$.*

Proof. If $w^T x^* \leq c - w_i$ and $x_i^* = 0$, then we can insert item i into the knapsack (i.e., change x_i^* from 0 to 1) without exceeding the knapsack capacity. Therefore x^* is not maximal. $\square$

From this we get the following theorem:

Theorem 1 *If a QKP instance is well-behaved, we can set*

$$L_i^1 := \min \left\{ \sum_{j \neq i} q_{ij} z_j : \; c - w_{\max} + 1 \leq w^T z \leq c, \; z \in \{0,1\}^n, \; z_i = 1 \right\}$$

and

$$L_i^0 := \min \left\{ \sum_{j \neq i} q_{ij} z_j : \; c - w_i + 1 \leq w^T z \leq c, \; z \in \{0,1\}^n, \; z_i = 0 \right\},$$

without losing at least one optimal integer solution.

Proof. From the above discussion, these values are lower bounds on the value taken by $\sum_{j \neq i} q_{ij} x_j$ in any maximal QKP solution such that $x_i = 1$ and $x_i = 0$, respectively. By definition, when a QKP instance is well-behaved, there is at least one optimal solution that is maximal. $\square$

The knapsack-like problems described in the theorem can be solved by dynamic programming in $\mathcal{O}(nc)$ time for each i. Just as in the previous section, a more attractive alternative may be to solve the LP relaxation of these problems, which can again be done in linear time.

4 Computational Experiments

In this section we give the results of some computational experiments.

Our test problems were constructed in the following way. The profits q_{ij} were random integers taken uniformly from $[1, 100]$, the weights w_i were random integers taken uniformly from $[1, 50]$, and the capacity was set at $\lfloor \sigma(a)/2 \rfloor$. These instances are a subclass of the instances proposed by Gallo *et al.* [9], which have become standard in the literature [14, 15]. Note that they are well-behaved, and therefore our method can be applied.

All computations were conducted on a 2.8GHz Pentium 4 PC with 512 Mb RAM running under Microsoft Windows XP Professional (Version 2002). The code was written in C using Microsoft Visual Studio.net and called on routines from Version 9.1 of the ILOG CPLEX Callable Library.

In the following tables, we report results for four variants of Glover's formulation:

- Formulation F0: the 'trivial' formulation in which, for all i, L_i is set to 0 and U_i is set to $\sum_{j \neq i} q_{ij} x_j$.
- Formulation F1: the formulation suggested by Glover [10], in which L_i is set to 0 as before, but U_i is computed by solving a 0-1 KP.
- Formulation F2: the formulation suggested by Adams *et al.* [2], in which L_i^0 and L_i^1 are set to 0, but U_i^0 and U_i^1 are computed by solving two 0-1 KPs.
- Formulation F3: the new formulation, in which U_i^0, U_i^1, L_i^0 and L_i^1 are computed by solving four 0-1 KPs.

As mentioned above, one can construct variants of F1 to F3 by solving the LP relaxations of the 0-1 KPs, but this makes little difference in practice. (The slight gain in speed in the pre-processing phase is usually offset by a slight increase in the time taken to solve the resulting MIPs.)

Table 1 shows, for various values of n and for each of the four formulations, the percentage gap between the optimum and the upper bound obtained from solving the LP relaxation. Each figure is an average taken over 20 random instances.

Tables 2 and 3 show, for the same instances and the same formulations, the average number of branch-and-bound nodes taken to solve the problems to optimality and the average time (in seconds) taken. A missing entry indicates that the branch-and-bound run was aborted due to excessive running times or memory requirements.

The improvement due to each of the strengthening procedures is readily apparent in these tables. Moreover, the results show clearly that the new method works very well for these random QKP instances. More generally, we expect the method to be most useful when the majority of the profits are positive, since L_i^0 and L_i^1 can be expected to be reasonably close to U_i^0 and U_i^1, respectively, in such cases.

Acknowledgement: The research of the second author was supported by the Engineering and Physical Sciences Research Council under grant `EP/D072662/1`.

References

[1] W.P. Adams & R.J. Forrester, A simple recipe for concise mixed 0-1 linearizations, Oper. Res. Lett. 33 (2005) 55-61.

[2] W.P. Adams, R.J. Forrester & F.W. Glover, Comparisons and enhancement strategies for linearizing mixed 0-1 quadratic programs, Discr. Opt. 1 (2004) 99-120.

[3] E. Balas & Zemel, An algorithm for large zero-one knapsack problems, Oper. Res. 28 (1980) 1130–1154.

[4] R.E. Bellman, Dynamic Programming, Princeton University Press, 1957.

[5] A. Billionnet & F. Calmels, Linear programming for the 0-1 quadratic knapsack problem, Eur. J. Opl Res. 92 (1996) 310–325.

[6] A. Billionnet, A. Faye & É. Soutif, A new upper bound for the 0-1 quadratic knapsack problem, Eur. J. Opl Res. 112 (1999) 664–672.

[7] A. Billionet & É. Soutif, Using a mixed integer programming tool for solving the 0-1 quadratic knapsack problem, INFORMS J. on Comp. 16 (2004) 188–197.

[8] A. Caprara, D. Pisinger & P. Toth, Exact solution of the quadratic knapsack problem, INFORMS J. on Computing 11 (1999) 125–137.

[9] G. Gallo, P.L. Hammer & B. Simeone, Quadratic knapsack problems, Math. Prog. Study 12 (1980) 132–149.

[10] F. Glover, Improved linear integer programming formulations of nonlinear integer problems, Management Science 22 (1975) 455-460.

[11] F. Glover, An improved MIP formulation for products of discrete and continuous variables, J. Inform. Optim. Sci. 5 (1984) 469-471.

[12] F. Glover & E. Woolsey, Converting the 0-1 polynomial programming problem to a 0-1 linear program, Oper. Res. 22 (1974) 180–182.

[13] H. Kellerer, Private communication, 2005.

[14] H. Kellerer, U. Pferschy & D. Pisinger, Knapsack Problems, Springer-Verlag, Berlin, 2004.

[15] D. Pisinger, The quadratic knapsack problem – a survey, Discr. Appl. Math. 155 (2007) 623-648.

[16] D. Pisinger, A.B. Rasmussen & R. Sandvik, Solution of large-sized quadratic knapsack problems through aggressive reduction, INFORMS J. on Comp. 19 (2007) 280–290.

n	F0	F1	F2	F3
20	17.81	9.95	7.76	6.40
40	14.28	6.87	5.92	4.66
60	13.97	6.47	5.82	4.51
80	13.54	5.36	4.91	3.70
100	11.91	5.21	4.83	3.57
120	11.89	4.59	4.29	3.13

Table 1: Average percentage gaps for the upper bounds.

n	F0	F1	F2	F3
20	187	120	85	69
40	11773	3945	2461	1644
60	132400	72819	71176	25199
80	1354184	178399	128100	33163
100	-	776312	323839	107992
120	-	-	-	140260

Table 2: Number of branch-and-bound nodes.

n	F0	F1	F2	F3
20	0.18	0.12	0.10	0.09
40	9.05	3.86	2.22	1.59
60	193.24	71.13	78.33	55.88
80	2524.89	319.94	306.22	75.44
100	-	1630.13	849.26	209.88
120	-	-	-	357.40

Table 3: Total branch-and-bound time (seconds).