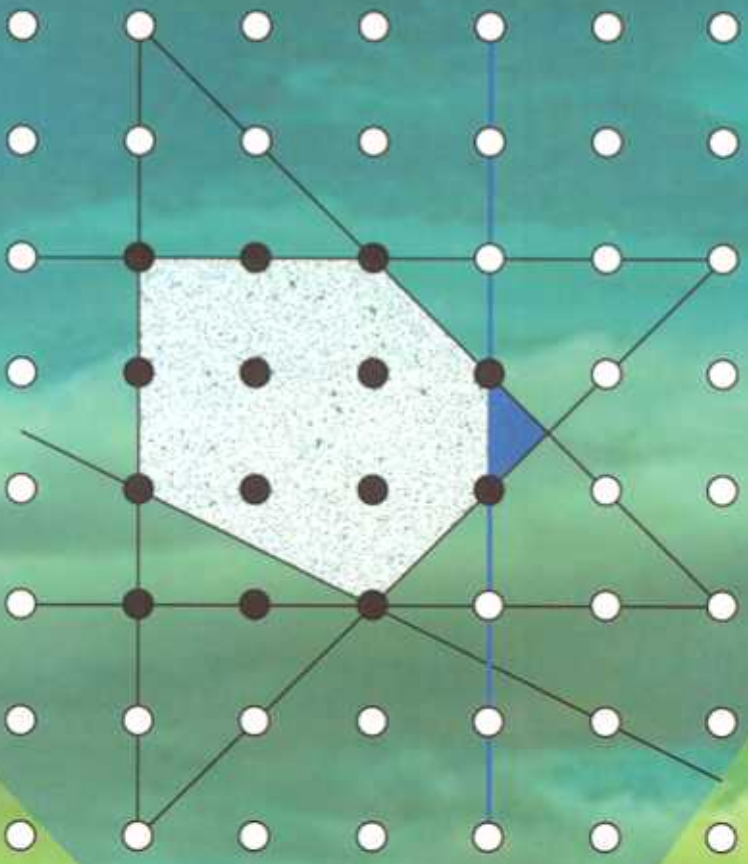
The Sharpest Cut

The Impact of Manfred Padberg and His Work



Edited by
Martin Grötschel

MPS-SIAM Series on Optimization

Chapter 17
The Steinberg Wiring Problem

Nathan W. Brixius[*] *and Kurt M. Anstreicher*[†]

It is clear that much more effort is needed and should be expended to solve this interesting riddle posed to combinatorial optimizers well over 35 years ago.
—M.W. Padberg and M.P. Rijal (1996)

MSC 2000. 90C27, 90C09, 90C10
Key words. Quadratic assignment problem, branch-and-bound, Gilmore-Lawler bound

17.1 Introduction

In a 1961 paper [44], Leon Steinberg described a "backboard wiring" problem that has resisted solution for 40 years. The problem concerns the placement of computer components so as to minimize the total amount of wiring required to connect them. In the particular instance considered by Steinberg, 34 components with a total of 2625 interconnections are to be placed on a backboard with 36 open positions. The geometry of the backboard is illustrated in Figure 17.1.

To formulate the wiring problem mathematically it is convenient to add two dummy components, with no connections to any others, so that the numbers of components and locations are both $n = 36$. Let a_{ik} be the number of wires that connect components i and k and b_{jl} be the "distance" between locations j and l on the backboard. (There are several possible choices for the b_{jl}. In his paper Steinberg considered using 1-norm, 2-norm, and

[*]Microsoft Corporation, Redmond, WA.
[†]Department of Management Sciences, University of Iowa, Iowa City, IA.

P01 •	P02 •	P03 •	P04 •	P05 •	P06 •	P07 •	P08 •	P09 •
P10 •	P11 •	P12 •	P13 •	P14 •	P15 •	P16 •	P17 •	P18 •
P19 •	P20 •	P21 •	P22 •	P23 •	P24 •	P25 •	P26 •	P27 •
P28 •	P29 •	P30 •	P31 •	P32 •	P33 •	P34 •	P35 •	P36 •

Figure 17.1. *Backboard for Steinberg problem.*

squared 2-norm distances between the backboard locations.) Let $x_{ij} = 1$ if component i is placed at location j on the backboard and $x_{ij} = 0$ otherwise. Doubling the objective, the problem can then be written in the form

$$\begin{aligned} \text{(SWP)} \qquad \min & \sum_{i,j,k,l} a_{ik} b_{jl} x_{ij} x_{kl} \text{ subject to} \\ & \sum_{j} x_{ij} = 1, \quad i = 1, \ldots, n, \\ & \sum_{i} x_{ij} = 1, \quad j = 1, \ldots, n, \\ & x_{ij} \in \{0, 1\}, \quad i, j = 1, \ldots, n. \end{aligned}$$

Note that the constraints of SWP are exactly that $X = \{x_{ij}\}$ is an $n \times n$-permutation matrix.

Steinberg devised a heuristic method to obtain (hopefully) a good solution for the wiring problem and applied it to the 2-norm and squared 2-norm versions of the problem. Most subsequent research has been directed to the 1-norm formulation.

In this chapter we describe the development of a branch-and-bound (B&B) algorithm to solve the 1-norm version of SWP to optimality. SWP is an example of a quadratic assignment problem (QAP), described in the next section. In Section 17.3 we describe lower-bounding schemes that have been proposed for QAP. In Section 17.4 we give some comparisons between bounds on SWP and similar problems, outline the construction of a complete B&B algorithm, and give computational results.

17.2 Quadratic Assignment Problems

The general QAP, introduced by Lawler [30], has the form

$$\begin{aligned} \text{(QAP)} \qquad \min & \sum_{i,j,k,l} d_{ijkl} x_{ij} x_{kl} \text{ subject to} \\ & X \in \Pi, \end{aligned}$$

where Π denotes the set of $n \times n$-permutation matrices. The problem SWP is an example of a "symmetric Koopmans–Beckmann" QAP (KBP). The term "Koopmans–Beckmann" denotes that the objective coefficient for $x_{ij}x_{kl}$ has the product form $a_{ik}b_{jl}$, and "symmetric"

means that $a_{ij} = a_{ji}$ and $b_{ij} = b_{ji}$ for all i, j. The 1-norm, 2-norm, and squared 2-norm versions of SWP are now known as the ste36a, ste36c, and ste36b QAPs; these and all other problem names are taken from QAPLIB [11].

The QAP can be used to formulate a variety of interesting problems in location theory, manufacturing, data analysis, and other areas [9, 12, 40]. Unfortunately, the QAP is, typically, extraordinarily difficult to solve due to its size. Several well-known combinatorial optimization problems, such as the traveling salesman problem (TSP), can be formulated as QAPs, and therefore the QAP is NP-hard. However, while TSPs with thousands of cities are now tractable [6, 37], in general a QAP with $n = 30$ presents a formidable computational challenge. For example, the well-known nug30 problem, posed in 1968 [34], was only recently solved using the equivalent of approximately seven years of serial computation [2].

Because of the extreme difficulty of the QAP, many heuristic approaches have been proposed to generate what we hope are good quality solutions. These techniques include GRASP [39], genetic algorithms [16], simulated annealing [14], tabu search [43, 45], and ant systems [17]. The best known objective value for the 1-norm version of SWP, 9526, was first obtained in 1990 using a tabu search algorithm [43] and has been subsequently rediscovered many times. One permutation (assignment of components to locations) attaining this value is

$$(12, 19, 30, 11, 2, 3, 22, 20, 10, 21, 5, 4, 13, 15, 31, 32, 28, 29,$$
$$24, 14, 17, 18, 16, 9, 8, 7, 6, 23, 33, 34, 35, 25, 27, 26, 1, 36).$$

Note that in this assignment the two dummy components (numbers 35 and 36) are placed in corners of the grid that are diagonally opposite one another.

17.3 Solution Approaches for the Quadratic Assignment Problem

Most exact solution methods for the QAP have been of the B&B type. A key component in such algorithms is the choice of method used to obtain lower bounds. There are a variety of lower-bounding approaches for the QAP, some of which have been used successfully in complete B&B algorithms.

17.3.1 Gilmore–Lawler bound

The most widely used lower bound for the QAP is the Gilmore–Lawler bound (GLB) [18, 30]. Note that the objective in QAP can be written in the form

$$\sum_{i,j} \left(\sum_{k,l} d_{ijkl} x_{kl} \right) x_{ij}.$$

Let f_{ij} denote the solution value in the linear assignment problem (LAP)

$$\min \sum_{k,l} d_{ijkl} x_{kl} \text{ subject to}$$
$$X \in \Pi,\ x_{ij} = 1.$$

It is then clear that GLB := LAP(F) $\leq$ QAP, where LAP(F) denotes the LAP with cost matrix F, and for convenience we use the name of an optimization problem to also refer to its solution value. For the general QAP the computation of GLB requires the solution of $n^2 + 1$ LAPs. However, for a KBP the LAP associated with each f_{ij} is trivial to solve, and as a result F can be obtained in a total of only $O(n^3)$ operations.

Several successful B&B algorithms for the QAP have utilized the GLB [8, 10, 13, 33]. GLB-based algorithms have proved effective for problems up to about size $n = 24$, but for larger problems the growth in nodes may become overwhelming.

17.3.2 Eigenvalue and related bounds

A KBP, with an added linear term, can be written in the matrix form

$$\text{(KBP)} \quad \min_{X \in \Pi} \operatorname{tr}(AXB + C)X^T,$$

where $\operatorname{tr}(\cdot)$ denotes the trace of a matrix. When A and B are symmetric, a bound for the quadratic term can be based on the fact that $X \in \Pi \Rightarrow X \in \mathcal{O}$, where $\mathcal{O}$ denotes the set of orthogonal matrices: $\mathcal{O} = \{X \mid XX^T = I\}$. For a symmetric matrix A let $\lambda(A) \in \mathbb{R}^n$ denote the vector of eigenvalues of A, and for vectors u and v let $\langle u, v\rangle_-$ denote the "minimal product"

$$\langle u, v\rangle_- := \min_{\pi} \sum_{i=1}^{n} u_i v_{\pi(i)},$$

where $\pi(\cdot)$ is a permutation of $1, 2, \ldots, n$. It is easy to show that $\langle u, v\rangle_-$ is obtained by putting the components of one of the vectors in nondecreasing order, and the components of the other in nonincreasing order, before taking the inner product. It can then be shown [15] that

$$\min_{X \in \mathcal{O}} \operatorname{tr}(AXBX^T) = \langle \lambda(A), \lambda(B)\rangle_-, \tag{17.1}$$

and therefore

$$\langle \lambda(A), \lambda(B)\rangle_- + \text{LAP}(C) \tag{17.2}$$

is a valid lower bound for a symmetric KBP. Unfortunately, the basic eigenvalue bound (17.2) is too weak to be computationally useful. Various schemes for improving the bound have been considered [15, 19, 41]. The most promising of these appears to be the projected eigenvalue bound (PB) of [19]. The construction of PB is based on enforcing the row and column sum constraints on X, in addition to orthogonality. Let V be an $n \times (n-1)$ matrix whose columns are an orthonormal basis for the nullspace of $e^T = (1, 1, \ldots, 1)$, and let $D = C + (2/n)Aee^TB$. Then

$$\text{PB} := \langle \lambda(V^TAV), \lambda(V^TBV)\rangle_- + \text{LAP}(D) - \frac{1}{n^2}(e^TAe)(e^TBe).$$

As shown in [19], for many problems PB provides a good quality bound at modest computational cost.

A quadratic programming bound (QPB) for KBP that is related to PB was devised in [4]. By construction QPB $\geq$ PB, and evaluating QPB requires the approximate solution of a convex quadratic program (QP) in the n^2 variables X. In [4] QPB was evaluated by solving the QP using an interior-point algorithm. This approach provides a very accurate solution,

but is too expensive to use in a B&B context. In [7] the Frank–Wolfe (FW) algorithm is used to approximately solve the QP associated with QPB. Although the asymptotic properties of the FW algorithm are known to be poor, this scheme is of interest in the context of QPB because the work on each iteration of the FW algorithm is dominated by the solution of an LAP. The resulting B&B algorithm exhibits state-of-the-art performance on many benchmark KPBs. In [2] the same QPB-based B&B algorithm, implemented using the "master-worker" distributed processing platform, obtains the first solution of several large problems including the nug30 QAP.

There has also been recent work devising bounds for KBP based on semidefinite programming. In [5] it is shown that there is a semidefinite programming interpretation for (17.1), and this interpretation is used in the derivation of QPB in [4]. Semidefinite programming bounds for KBP are described in [31] and [46]. In [3] it is shown that the basic semidefinite programming bound of [46] is closely related to PB. More complex semidefinite programming bounds described in [31] and [46] are also related to the linear programming bounds described below. These semidefinite programming bounds are often of excellent quality, but are obtained at a very high computational cost.

17.3.3 Linear programming and dual linear programming bounds

A large class of bounds for the QAP are related to linear programming relaxations of the problem. Defining new variables $y_{ijkl} = x_{ij}x_{kl}$ and dropping the integrality conditions results in a linear programming relaxation [1, 42]

$$
\begin{aligned}
\text{(LPQAP)} \quad \min \sum_{i,j,k,l} y_{ijkl} d_{ijkl} \text{ subject to} \\
\sum_{j} x_{ij} = 1, \quad i = 1, \ldots, n, \\
\sum_{i} x_{ij} = 1, \quad j = 1, \ldots, n, \\
\sum_{l} y_{ijkl} = x_{ij}, \quad i, j, k = 1, \ldots, n, \\
\sum_{k} y_{ijkl} = x_{ij}, \quad i, j, l = 1, \ldots, n, \\
y_{ijkl} = y_{klij}, \quad i, j, k, l = 1, \ldots, n, \qquad (17.3) \\
x_{ij} \geq 0, \quad y_{ijkl} \geq 0, \quad i, j, k, l = 1, \ldots, n.
\end{aligned}
$$

The symmetry constraints (17.3) imply that LPQAP can be formulated using variables $y_{ijkl}, i \leq k$. Additional variables can be eliminated using the facts that $y_{ijij} = x_{ij}$ for all i and j, $y_{ijil} = 0$ for all i and $j \neq l$, and $y_{ijkj} = 0$ for all $i \neq k$ and j for X feasible in QAP. Taken together, these observations allow for a reformulation of LPQAP as a linear programming problem with $n^2 + n^2(n-1)^2/2$ variables. Further analysis [38, Section 7.1] can be used to reduce the number of equality constraints required in LPQAP to $2n(n-1)^2 - (n-1)(n-2), n \geq 3$. For a symmetric problem like SWP, LPQAP can be formulated using $n^2 + n^2(n-1)^2/4$ variables and $n^2(n-2) + 2n - 1$ equality constraints, $n \geq 3$ [25], [38, Section 7.3]. For a symmetric problem with $n = 36$, for example, LPQAP can be written using 398,196 variables and 44,135 equality constraints. The solution of LPQAP

using an interior-point method was investigated in [42]. This approach produces excellent bounds for many problems, but appears to be prohibitively costly for implementation in a B&B algorithm.

It is known [1] that, if the symmetry conditions (17.3) are dropped, then the solution value in LPQAP is exactly GLB. It can also be shown [28] that many bounding schemes for QAP can be viewed as Lagrangian procedures that attempt to approximately solve the dual of LPQAP. Computationally, the most successful of these is a method motivated by the Hungarian algorithm for LAP, due to P. Hahn and coworkers [20, 21, 22]. The B&B code of Hahn et al. recently obtained the first solution of the kra30a QAP, a hospital layout problem dating from 1972 [23].

In [28] a dual linear programming procedure similar to that proposed in [20] is used to obtain a lower bound of 7860 for the 1-norm version of SWP. To our knowledge this is the best known lower bound for the problem.

17.3.4 The polyhedral approach

The polyhedral approach to QAP is based on investigating the convex hull of 0/1-valued solutions to the linear programming relaxation LPQAP. This line of research was initiated by Padberg and Rijal [38] and has been further developed by Kaibel and Jünger [24, 25, 26, 27]. The convex hull of 0/1-valued solutions of LPQAP is a face of the Boolean quadric polytope, studied in [36]. An essential element of the polyhedral approach is the characterization of valid inequalities that can be added to LPQAP to tighten the relaxation. In [38, Section 1.5] the polyhedral approach is applied to a linear programming relaxation of SWP. (The relaxation is similar to LPQAP, but is specialized for a symmetric KBP and also exploits sparsity of the matrix A.) Solution of the resulting linear program took approximately one month on a 50MHz Sun workstation and obtained a lower bound of 7794. This was the best known lower bound for the problem prior to the dual linear programming bound obtained in [28].

The polyhedral approach to discrete optimization has resulted in very successful *branch-and-cut* algorithms for particular discrete optimization problems such as TSP [37, 6]. Branch-and-cut algorithms typically invest a large amount of time generating valid inequalities, and resolving subproblems, in an effort to reduce branching to a minimum. The development of branch-and-cut algorithms for QAP is still in its infancy, but recent results [24] indicate that the methodology promises to become a general purpose solution method.

17.4 Solving the Steinberg Problem

In this section we consider applying a B&B algorithm to solve the 1-norm version of SWP to optimality. In Table 17.1 we give the values for a number of different bounds applied to the problem. In the table "Sum" is the trivial bound obtained from the fact that there are 2625 interconnections[1] between components and all distances are at least one. TDB, the triangle decomposition bound of [29], is a parametric strengthening of PB that can be applied to problems with distance matrices arising from 1-norms on grids. QPB is computed using

[1]The number of interconnections is given as 2620 in [44]. This appears to be due to an error in computing the sum of the entries in row/column 29 of the matrix A; see [44, Figure 1].

Table 17.1. *Bounds for 1-norm wiring problem.*

Bound	Value	Gap
Dual-LP	7860	17%
Polyhedral	7794	18%
GLB	7124	25%
TDB	6997	27%
Sum	5250	45%
QPB	−10294	208%
PB	−11700	223%

500 FW iterations (see [7] for details), and all gaps are computed relative to the best-known value of 9526. It is clear that PB and the related QPB perform very poorly. The performance of GLB is reasonable, and although the dual linear programming and polyhedral bounds are better, the computational cost of these bounds is many orders of magnitude higher than that of GLB. The computation to obtain TDB is also much greater than that required for PB or GLB.

It is well known that eigenvalue bounds can be negative on instances of KBP for which zero is a trivial lower bound. In [29] it is suggested that this poor performance may be related to sparsity of the flow matrix A. In Figure 17.2 we give the sparsity (fraction of zero components) and coefficient of variation (CV, equal to the standard deviation of the components divided by their mean) for the flow matrices from a number of grid-based KBPs from QAPLIB [11]. It is clear that ste36a is very sparse, with a high CV. In the context of heuristics for QAP, CV is often termed "flow dominance" [17] and has been used as an algorithm control parameter.

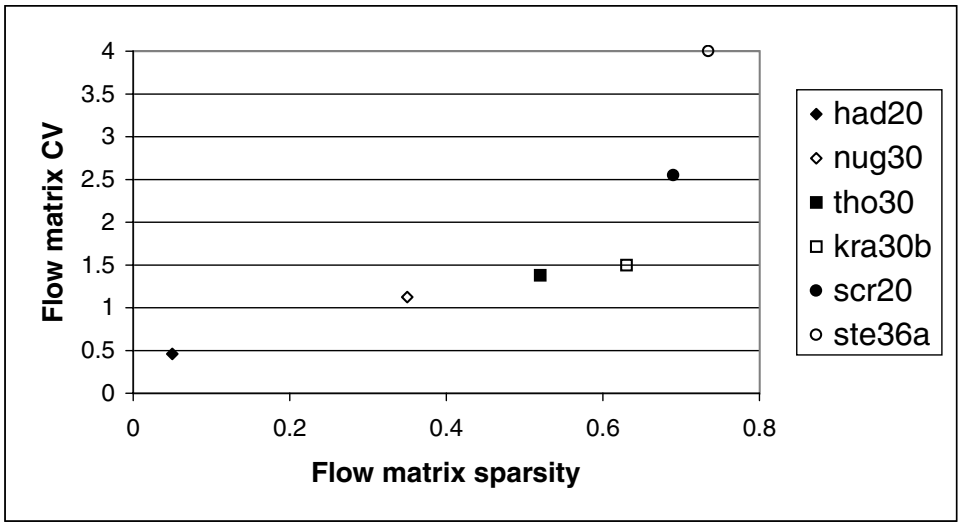


Figure 17.2. *Characteristics of flow matrices of grid-based QAPs.*

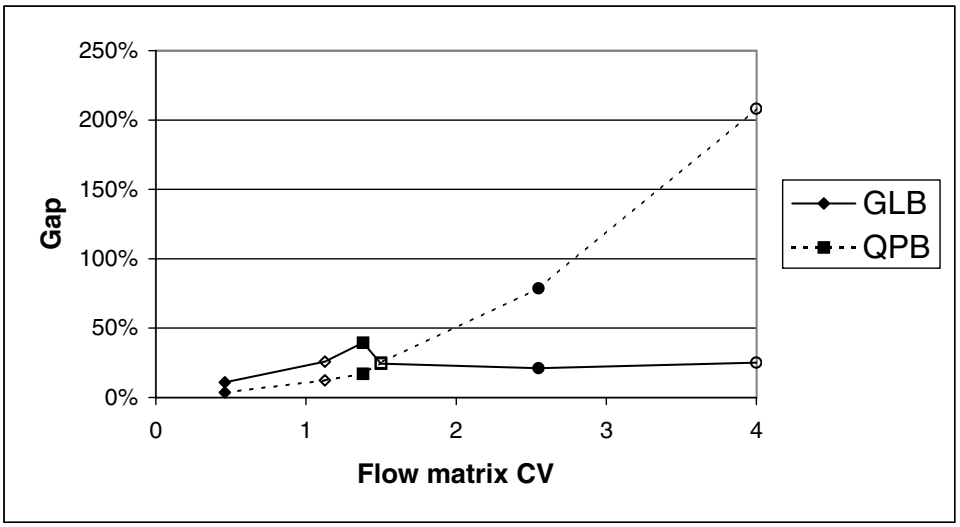


Figure 17.3. *Gaps for bounds on grid-based QAPs.*

In Figure 17.3 we give the gaps for GLB and QPB for the same problems considered in Figure 17.2. The markers used to denote the problems are the same as in Figure 17.2. The strong relationship between CV and the quality of QPB is evident. It is worth noting that the most successful applications of QPB reported in [2, 7] correspond to problems with relatively low CV values, such as had20, nug30, tho30, and kra30b. For had20, the problem with the lowest CV, solution using QPB is faster than the GLB-based algorithm of [8] by a factor of over 3000, after adjusting for hardware differences [7]. On the other hand, the equivalent time to solve scr20 using QPB is about a factor of 2.2 times that required in [8]. These observations suggest that QPB might not be a good candidate for the solution of ste36a, and consequently we consider the application of a GLB-based B&B algorithm.

17.4.1 Branching rules

As described in Section 17.3.1, the value of GLB for a QAP is obtained from LAP(F), where F is first derived from the original problem data. Associated with the solution of LAP(F) is a nonnegative reduced-cost matrix U such that

$$F \bullet X = z^* + U \bullet X$$

for any X with $Xe = X^T e = e$, where $F \bullet X = \mathrm{tr}(FX^T)$ and z^* is the solution value in GLB. (If X^* solves LAP(F), then $X^* \bullet U = 0$.) It follows that, if v is the value of a known solution to QAP, then

$$z^* + u_{ij} > v \Rightarrow x_{ij} = 0 \tag{17.4}$$

in any optimal solution X of QAP.

The use of (17.4) to eliminate children in the course of branching was introduced in [33], and it has been employed in many subsequent papers. Mautor and Roucairol [33]

also introduced *polytomic branching*, where at any node candidate children are obtained by either (*row branching*) fixing one facility and assigning it to all available locations, or (*column branching*) fixing one location and assigning to it all available facilities. In our implementation we use polytomic row and column branching. We consider two branching rules, Rules 2 and 4, that are motivated by similar QPB-based branching rules from [7]. For simplicity we describe the rules here as they would be implemented at the root node, using row branching. The problem associated with an arbitrary node in the B&B tree is a lower dimensional QAP, on which the implementation of the rules is very similar. Let $N = \{1, 2, \ldots, n\}$.

Rule 2. Branch on the row i that produces the smallest number of children. In the event of a tie, choose the row with the largest value $\sum_{j \in N_i'} u_{ij}$, where $N_i' = \{j \in N \mid z + u_{ij} < v\}$.

Note that the set N_i' in Rule 2 consists exactly of the child problems $x_{ij} = 1$ that *cannot* be eliminated. Rule 2 is an extension of the branching rule used in [33] and is effective in reducing the size of the tree on small problems. Close to the root on larger instances, however, the information provided by the reduced-cost matrix U may be insufficient to make good branching decisions. Consequently, we consider obtaining more information about the effect of setting $x_{ij} = 1$ before actually deciding where to branch.

Rule 4. Let I denote the set of rows having the NBEST highest values of $\sum_{j \in N} u_{ij}$. For each $i \in I$, $j \in N$, compute the GLB z^{ij} for the QAP obtained by setting $x_{ij} = 1$. Let U^{ij} be the reduced-cost matrix associated with z^{ij}. Let v^{ij} be the maximal row sum of U^{ij}, and let $w^{ij} = (|N| - 1)z^{ij} + v^{ij}$. Branch on the row i having the highest value of $\sum_{j \in N} w^{ij}$.

In the context of B&B algorithms Rule 4 is an example of a *strong branching rule* [32]. Because of the use of the U^{ij} matrices, Rule 4 can also be viewed as a *look-ahead procedure* that tries to maximize the bounds two levels deeper in the tree.

In addition to the elimination of children based on bounds, described above, redundant children can be eliminated using symmetry of the grid on which the distance matrix B is based (see Figure 17.1). For example, the children of the root node can be based on assignments $x_{ij} = 1$, $j \in J_1 = \{1{:}5, 10{:}14\}$, regardless of the choice of i. (For integers $m < n$ we use $m{:}n$ to denote the collection of integers k with $m \leq k \leq n$.) In addition, if at any node the current assignments are all to locations contained in the set $J_2 = \{5, 14, 23, 32\}$, then the children can be restricted to $x_{ij} = 1$, $j \in J_3 = \{1{:}5, 10{:}14, 19{:}23, 28{:}32\}$, regardless of the choice of i. In cases where symmetry can be exploited we use row branching, with the index set N in Rules 2 and 4 replaced by a suitable $J \subset N$. In all *other* cases Rules 2 and 4 are implemented so as to consider column branching as well as row branching, with only minor modifications required. (For example, in Rule 2 we choose the row or column that produces the least number of children.)

17.4.2 Computational results

We implemented a GLB-based B&B algorithm, using the branching rules described above, to solve the 1-norm SWP. As in [2] the choice of branching rule to apply at a given node is

Table 17.2. *Branching strategy used to solve ste36a.*

Rule	Depth	Gap	NBEST
4a	5	0.35	36
4b	6	0.30	10
2	50	0.00	–

determined by depth in the tree and the relative gap. The relative gap for a node is defined to be

$$g = \frac{v - z'}{v - z_0},$$

where v is the incumbent value, z' is the lower bound at the current node, and z_0 is the root lower bound. The exact branching strategy used is given in Table 17.2. At a given node the rules are scanned from the top down until a rule is found whose maximum depth is greater than or equal to the node's depth, and whose minimum gap is less than the node's relative gap. The B&B tree was traversed using depth-first search.

The solution of the problem required a total of approximately 7.75×10^8 nodes in the B&B tree. The best known value of 9526 was verified as being optimal. In Figure 17.4 we give the number of nodes at each level of the tree. Note the logarithmic scale for the y-axis. Subproblems at level 33 of the tree correspond to QAPs of dimension three, which were solved by enumeration. The solution required approximately 186 hours of CPU time on a single 800 MHz Pentium III PC. (Based on a direct comparison this machine is approximately 40% faster on our application than the HP™ 9000 model C3000 used in [7].) In Figure 17.5 we give the cumulative CPU time (in hours) expended for the nodes

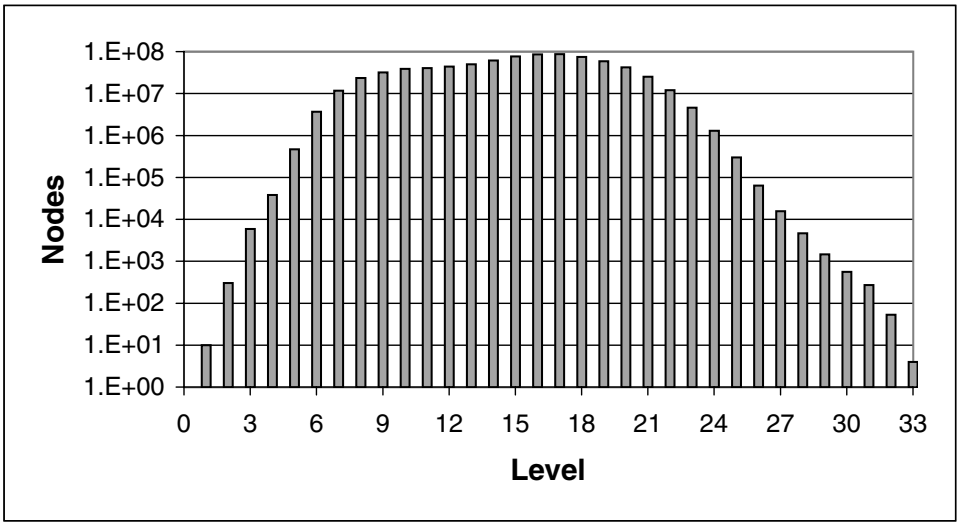


Figure 17.4. *Distribution of nodes in solution of ste36a.*

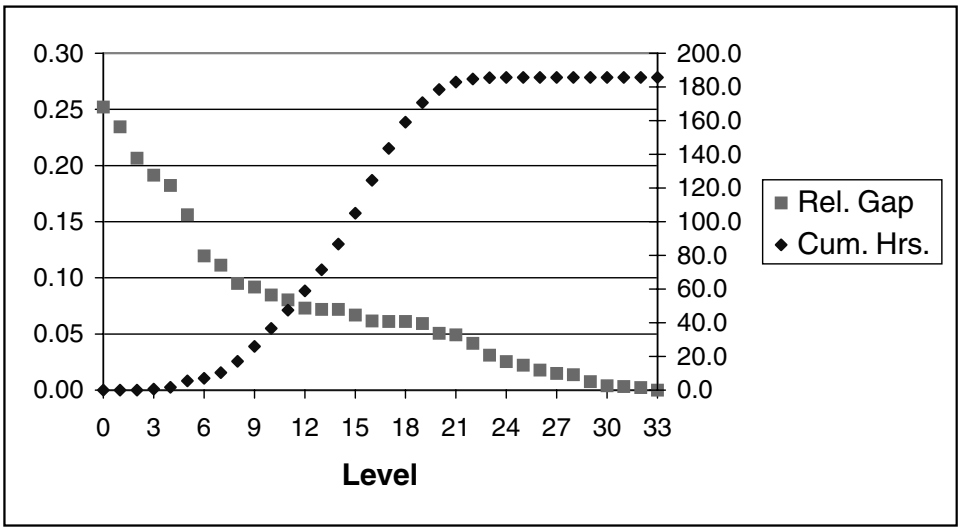


Figure 17.5. *Relative gap and cumulative time in solution of ste36a.*

up to each level of the tree. In the figure we also give the gap to optimality at each level, computed using the minimum bound obtained at that level. From the figure it is clear that it is relatively inexpensive to reduce the gap to about one-half its initial value. (The worst bound for a level 6 node was 8388, corresponding to a gap of 12%. The cumulative time to process all nodes at levels 0 to 6 is about 7 hours.)

To evaluate the effect of using Rule 4 at the top of the B&B tree we also ran the algorithm using only Rule 2, through level 7. In Table 17.3 we give comparitive statistics for the nodes through level 8 obtained from the solution run, and the run using only Rule 2. "L" denotes the level in the B&B tree. The "Fthm" and "Elim" columns report the fraction of nodes fathomed and the fraction of potential children of unfathomed nodes eliminated, respectively. "Gap" is the average gap to the optimal value for nodes on a given level,

Table 17.3. *Comparison of solution run with Rule 2 only.*

	Rule 2 only				Solution run			
L	Nodes	Gap	Fthm	Elim	Nodes	Gap	Fthm	Elim
0	1	2402.0	0.000	0.000	1	2402.0	0.000	0.000
1	10	2280.5	0.000	0.000	10	1953.7	0.000	0.054
2	318	1770.3	0.003	0.069	301	1218.4	0.000	0.421
3	9941	1239.8	0.080	0.549	5869	697.1	0.070	0.787
4	136112	727.7	0.209	0.580	38263	542.5	0.320	0.441
5	1445612	594.7	0.336	0.535	465182	354.3	0.404	0.569
6	13832243	438.4	0.445	0.629	3703103	260.8	0.579	0.752
7	85562934	322.4	0.546	0.613	11627541	183.0	0.641	0.806
8	436142577	266.5			23549921	132.2	0.730	0.821

computed using the lower bound inherited from the parent node. From the table it is clear that the use of Rule 4 at the top levels has an enormous effect on the subsequent evolution of the tree. Note that using only Rule 2 increases the number of nodes on level 8 by a factor of over 18. Moreover, the average gap for these level 8 nodes is approximately doubled, suggesting that the number of nodes at deeper levels will continue to worsen substantially compared to the solution run. We believe that the time to solve ste36a using only Rule 2 would be at least a factor of 100 higher than the time obtained here using Rules 4 and 2 together. Further evidence of the value of Rule 4 is provided by the results of a preliminary solution run that used Rule 4 only on levels 0, 1, and 2 of the tree. This earlier run required more than double the nodes (1.79×10^9) and time (435 hours) of the final solution run reported here.

It is interesting to compare some characteristics of the B&B tree for ste36a with the solution of nug30 obtained in [2]. For example, statistics like "Fthm" and "Elim" are substantially better near the top of the tree for ste36a than for nug30. On the other hand, the node distribution for ste36a, as shown in Figure 17.4, is much "flatter" than the corresponding distribution for nug30. Although the peak number of nodes is modest compared to the solution of nug30, there are 14 levels (8–21) where the number of nodes is within a factor of 5 of the peak number (8.7×10^7, on level 17). In the B&B tree for the nug30 problem only 6 levels had node counts within a factor of 5 of the peak (2.66×10^9, on level 10). We conclude that, while the use of the GLB with strong branching is effective in limiting the growth of the B&B tree for ste36a, there is still room for improvement in the overall time required to solve the problem.

After this chapter was written, we learned of a previously unreleased technical report by M. Nyström [35] that describes the solution of the ste36b/c problems. Nyström used a distributed B&B algorithm based on the GLB, implemented on twenty-two 200 MHz Pentium Pro CPUs. The serial time to solve the ste36b/c instances on one of these CPUs is estimated to be approximately 60 days and 200 days, respectively. (The time for ste36c is substantially higher because this problem was solved using an initial incumbent value of $+\infty$.)

Bibliography

[1] W.P. Adams and T. Johnson. Improved linear programming based lower bounds for the quadratic assignment problem. In P. Pardalos and H. Wolkowicz, editors, *Quadratic Assignment and Related Problems*, volume 16 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 43–77. AMS, Providence, RI, 1994.

[2] K. Anstreicher, N. Brixius, J.-P. Goux, and J. Linderoth. Solving Large Quadratic Assignment Problems on Computational Grids. *Mathematical Programming Series* B, 91:563–588, 2002.

[3] K.M. Anstreicher. Eigenvalue bounds versus semidefinite relaxations for the quadratic assignment problem. *SIAM Journal on Optimization*, 11:254–265, 2000.

[4] K.M. Anstreicher and N.W. Brixius. A new bound for the quadratic assignment problem based on convex quadratic programming. *Mathematical Programming*, 89:341–357, 2001.

[5] K. Anstreicher and H. Wolkowicz. On Lagrangian relaxation of quadratic matrix constraints. *SIAM Journal on Matrix Analysis and Applications*, 22:41–55, 2000.

[6] D. Applegate, R. Bixby, V. Chvátal, and W. Cook. On the solution of traveling salesman problems. *Documenta Mathematica*, Extra Volume III:645–656, 1998.

[7] N.W. Brixius and K.M. Anstreicher. Solving quadratic assignment problems using convex quadratic programming relaxations. *Optimization Methods and Software*, 16:49–68, 2001.

[8] A. Brüngger, A. Marzetta, J. Clausen, and M. Perregaard. Solving large-scale QAP problems in parallel with the search library ZRAM. *Journal of Parallel and Distributed Computing*, 50:157–169, 1998.

[9] R.E. Burkard, E. Çela, P.M. Pardalos, and L.S. Pitsoulis. The quadratic assignment problem. In D.-Z. Du and P.M Pardalos, editors, volume 3 of *Handbook of Combinatorial Optimization*, pages 241–337. Kluwer Academic, Boston, 1998.

[10] R.E. Burkard and U. Derigs. *Assignment and Matching Problems: Solution Methods with Fortran Programs*, volume 184 of *Lecture Notes in Economics and Mathematical Systems*. Springer-Verlag, Berlin, 1980.

[11] R.E. Burkard, S.E. Karisch, and F. Rendl. QAPLIB—a quadratic assignment problem library. *Journal of Global Optimization*, 10:391–403, 1997. See also www.opt.math.tugraz.ac.at/qaplib.

[12] E. Çela. *The Quadratic Assignment Problem: Theory and Algorithms*. Kluwer, Dordrecht, Boston, 1998.

[13] J. Clausen and M. Perregaard. Solving large quadratic assignment problems in parallel. *Computational Optimization and Applications*, 8:111–127, 1997.

[14] D.T. Connolly. An improved annealing scheme for the QAP. *European Journal of Operational Research*, 46:93–100, 1990.

[15] G. Finke, R.E. Burkard, and F. Rendl. Quadratic assignment problems. *Annals of Discrete Mathematics*, 31:61–82, 1987.

[16] C. Fleurent and J.A. Ferland. Genetic hybrids for the quadratic assignment problem. In P. Pardalos and H. Wolkowicz, editors, *Quadratic Assignment and Related Problems*, volume 16 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 173–187. AMS, Providence, RI, 1994.

[17] L.M. Gambardella, É.D. Taillard, and M. Dorigo. Ant colonies for the quadratic assignment problem. *Journal of the Operational Research Society*, 50:167–176, 1999.

[18] P.C. Gilmore. Optimal and suboptimal algorithms for the quadratic assignment problem. *Journal of the Society for Industrial and Applied Mathematics*, 10:305–313, 1962.

[19] S.W. Hadley, F. Rendl, and H. Wolkowicz. A new lower bound via projection for the quadratic assignment problem. *Mathematics of Operations Research*, 17:727–739, 1992.

[20] P.M. Hahn and T. Grant. Lower bounds for the quadratic assignment problem based upon a dual formulation. *Operations Research*, 46:912–922, 1998.

[21] P.M. Hahn, T. Grant, and N. Hall. A branch-and-bound algorithm for the quadratic assignment problem based on the Hungarian method. *European Journal of Operational Research*, 108:629–640, 1998.

[22] P.M. Hahn, W.L. Hightower, T.A. Johnson, M. Guignard-Spielberg, and C. Roucairol. *Tree Elaboration Strategies in Branch and Bound Algorithms for Solving the Quadratic Assignment Problem*. Technical report, Systems Engineering, University of Pennsylvania, Philadelphia, 1999.

[23] P.M. Hahn and J. Krarup. A hospital facility layout problem finally solved. *The Journal of Intelligent Manufacturing,* 5/6:487–496, 2001.

[24] M. Jünger and V. Kaibel. Box-inequalities for quadratic assignment polytopes, *Mathematical Programming,* 91:175–197, 2001.

[25] M. Jünger and V. Kaibel. On the SQAP-polytope. *SIAM Journal on Optimization*, 11:444–463, 2000.

[26] V. Kaibel. Polyhedral combinatorics of quadratic assignment problems with less objects than locations. In R.E. Bixby, E.A. Boyd, and R.Z. Ríos-Mercado, editors, *Integer Programming and Combinatorial Optimization*, volume 1412 of *Lecture Notes in Computer Science*, pages 409–422. Springer-Verlag, Berlin, 1998.

[27] V. Kaibel. Polyhedral methods for the QAP. In P.M. Pardalos and L. Pitsoulis, editors, *Nonlinear Assignment Problems*. Kluwer Academic, Dordrecht, Boston, 2000.

[28] S.E. Karisch, E. Çela, J. Clausen, and T. Espersen. A dual framework for lower bounds of the quadratic assignment problem based on linearization. *Computing*, 63:351–403, 1999.

[29] S.E. Karisch and F. Rendl. Lower bounds for the quadratic assignment problem via triangle decompositions. *Mathematical Programming*, 71:137–152, 1995.

[30] E.L. Lawler. The quadratic assignment problem. *Management Science*, 9:586–599, 1963.

[31] C.-J. Lin and R. Saigal. *On Solving Large-Scale Semidefinite Programming Problems—A Case Study of Quadratic Assignment Problem*. Technical report, Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI, 1997.

[32] J.T. Linderoth and M.W.P. Savelsbergh. A computational study of branch and bound search strategies for mixed integer programming. *INFORMS Journal on Computing*, 11:173–187, 1999.

[33] T. Mautor and C. Roucairol. A new exact algorithm for the solution of quadratic assignment problems. *Discrete Applied Mathematics*, 55:281–293, 1994.

[34] C.E. Nugent, T.E. Vollman, and J. Ruml. An experimental comparison of techniques for the assignment of facilities to locations. *Operations Research*, 16:150–173, 1968.

[35] M. Nyström. *Solving Certain Large Instances of the Quadratic Assignment Problem: Steinberg's Examples*. Technical report, Department of Computer Science, California Institute of Technology, Pasadena, CA, 1999.

[36] M. Padberg. The Boolean quadratic polytope: Some characteristics, facets, and relatives. *Mathematical Programming*, 45:139–172, 1989.

[37] M. Padberg and G. Rinaldi. A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM Review*, 33:60–100, 1991.

[38] M.W. Padberg and M.P. Rijal. *Location, Scheduling, Design and Integer Programming*. Kluwer Academic, Boston, 1996.

[39] P.M. Pardalos, L.S. Pitsoulis, and M.G.C. Resende. A parallel GRASP implementation for the quadratic assignment problem. In A. Ferreira and J. Rolim, editors, *Parallel Algorithms for Irregularly Structured Problems—Irregular '94*, pages 111–130. Kluwer Academic, Dordrecht, Netherlands, 1995.

[40] P.M. Pardalos, F. Rendl, and H. Wolkowicz. The quadratic assignment problem: A survey and recent developments. In P.M. Pardalos and H. Wolkowicz, editors, *Quadratic Assignment and Related Problems*, volume 16 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 1–42. AMS, Providence, RI, 1994.

[41] F. Rendl and H. Wolkowicz. Applications of parametric programming and eigenvalue maximization to the quadratic assignment problem. *Mathematical Programming*, 53:63–78, 1992.

[42] M.G.C. Resende, K.G. Ramakrishnan, and Z. Drezner. Computing lower bounds for the quadratic assignment problem with an interior point algorithm for linear programming. *Operations Research*, 43:781–791, 1995.

[43] J. Skorin-Kapov. Tabu search applied to the quadratic assignment problem. *ORSA Journal on Computing*, 2:33–45, 1990.

[44] L. Steinberg. The backboard wiring problem: A placement algorithm. *SIAM Review*, 3:37–50, 1961.

[45] É.D. Taillard. Robust taboo search for the quadratic assignment problem. *Parallel Computing*, 17:443–455, 1995.

[46] Q. Zhao, S.E. Karisch, F. Rendl, and H. Wolkowicz. Semidefinite programming relaxations for the quadratic assignment problem. *Journal of Combinatorial Optimization*, 2:71–109, 1998.