The Generalized Quadratic Assignment Problem

Chi-Guhn Lee*, Zhong Ma

Department of Mechanical and Industrial Engineering, University of Toronto,
Toronto, Ontario, M5S 3G8, Canada

Abstract

We study a generalization of the quadratic assignment problem (QAP) by allowing multiple equipments to be assigned at a single location as long as resources at the location permit. This problem arises in many real world applications such as facility location problem and logistics network design. We call the problem as the generalized quadratic assignment problem (GQAP) and show that this relaxation increases the complexity of the problem dramatically. We propose three linearization approaches and a branch and bound algorithm to optimally solve the GQAP. Computational studies have been conducted to demonstrate the performance of the proposed approaches.

Key words: quadratic assignment problem, location problem, logistics, linearization, branch and bound, integer programming.

1 Introduction

This research is motivated by a real world problem, where a manufacturing company in the Greater Toronto Area (GTA) has to decide where it should locate multiple equipments. The company owns four manufacturing sites in the GTA and has different sets of equipments in different sites. As a result, it has to move intermediate parts between the locations to perform a sequence of operations for each part type. The company hires a local transportation company to serve the transportation needs and pays on a basis of service time including travel times between locations and waiting time during loading.

* Corresponding author.

Email address: cglee@mie.utoronto.ca (Chi-Guhn Lee).
and unloading. The annual transportation cost has been a significant part of their total operating cost over the years and the company seeks ways to cut the cost dramatically. While the company makes efforts to reduce the cost by improving its shipping operations, it also, with a collaboration with the University of Toronto, tries to restructure the manufacturing system by replacing some of the equipments from one location to another.

The problem is to optimally assign $m$ equipments to $n$ locations without violating the space limitation at each location, where $(m \neq n)$. In this research we focus on the case of $(m > n)$, i.e., the number of equipments is more than the number of locations. However, our approaches are applicable to the other cases $(m \leq n)$ as well. An equipment assigned to a location will occupy a fixed amount of space and incurs a known installation cost, which is dependent of the location. Each equipment has to be assigned and can only be assigned to a single location. The total cost is the sum of installation and transportation costs. The transportation cost is to capture the cost associated with material movements between locations over a given planning horizon, which is proportional to the sum of all the products of flow volumes between pairs of equipments and travel times between pairs of locations. The transportation cost can then be obtained by multiplying the charge specified in the contract with the transportation company to the travel time. The goal is to find an assignment such that the sum of installation and transportation costs should be minimized.

The problem is almost the same as the quadratic assignment problem (QAP) except that multiple equipments can be assigned to a single location. Although the QAP has been studied extensively for the last half century, to our best knowledge, no research has been done on how to solve this particular type of the QAP, which is then named as the generalized quadratic assignment problem (GQAP) by us. In the GQAP, multiple equipments can be assigned to a single location as long as such assignment does not violate resource constraints. Christofides and Gerrard [1] did first use the abbreviation of GQAP in 1976, referring to the QAP in graphs. The GQAP, according to their definition, is a special type of the QAP, where only a subset of all possible assignments is allowed to be used. Therefore, the GQAP of Christofides and Gerrard is different from ours.

The QAP is NP-hard and even its approximation algorithms are NP-hard [2]. It has been known that it is impossible to solve the QAP with more than 22 locations and it is not practical to optimally solve the QAP with more than 15 locations due to extensive computation [2]. Extensive efforts to solve the QAP in the past can be categorized into three major approaches: (a) linearization, (b) branch and bound algorithm, and (c) cutting plane method [2]. Various linearization approaches to the QAP can be found in Kaufman and Broeckx [3], Frieze and Yadeger [4], and Padberg and Rijal [5]. While the performance of
branch and bound algorithms depends on the tightness of lower bound, the lack of good lower bounds is one of the major difficulties in branch and bound algorithms for the QAP. Currently, the most widely used lower bounds includes Gilmore-Lawler bound [6], linear relaxation, and semi-definite relaxation [7,8]. Cutting plane method has typically shown inferior in performance to branch and bound algorithms. Traditional cutting planes are based on the mixed integer liner programming. A different approach based on polyhedral cutting plane (or branch and cut method) also appeared [2,9].

Due to the limited size of the problem that can be solved optimally, heuristic approaches become widely used when the problems become large. Heuristic approaches include construction methods, improvement methods, limited enumeration and meta-heuristics such as tabu search, simulated algorithm, genetic algorithm and randomized search [2].

Some people define the QAP in a more general setting, where the number of equipments is less than the number of locations [10,11]. However, each location can accommodate no more than a single equipment, thereby no resource issues at locations. The GQAP that we introduce in the paper is structurally different problem from the QAP in this sense.

The remainder of the paper is organized as follows. Section 2 introduces the problem in a more formal way and discuss the complexity of the problem. Section 3 presents three linearization approaches to the GQAP with proofs for the equivalence of the linear programs to the GQAP. Section 4 describes a branch and bound algorithm to solve the GQAP and Section 5 reports computational studies on the linearization approaches and the branch and bound algorithm. Finally, Section 6 concludes the paper with a few words.

2 Problem Description and Complexity

The GQAP is a generalized problem of the QAP in that there is no restriction that one location can accommodate only a single equipment. Let

\[ M : \text{a set of equipments} = \{1, 2, \ldots, m\}, \]
\[ N : \text{a set of locations} = \{1, 2, \ldots, n\}, \]
\[ s_i : \text{space requirement of equipment} \ i \in M, \]
\[ S_k : \text{total available space at location} \ k \in N, \]
\[ c_{ik} : \text{the cost of installing equipment} \ i \text{ at location} \ k, \]
\[ q_{ij} : \text{the flow volume from equipment} \ i \text{ to} \ j \text{ in the planning horizon}, \]
\[ d_{kh} : \text{the distance between location} \ k \text{ and} \ h \ (=d_{hk}), \]
\[ v : \text{the travel cost per unit distance and per unit flow volume}. \]
Once equipments $i$ and $j$ are installed at location $k$ and $h$, there will be transportation cost $vq_{ij}d_{kh}$ during the given planning horizon. Therefore, the total cost is the sum of the installation and the transportation cost. The sum of space requirement of equipments assigned at location $k$ cannot exceed the total space available at location $k$ and an equipment must be assigned to a unique location. We will assume in the paper that the number of equipment is greater than the number of locations, although our approaches would work when such condition is violated.

Just to avoid trivial cases, we will assume (a) there is no single location that is big enough to hold all the equipments or $\sum_{i \in M} s_i > S_k \ \forall k \in N$, (b) there is no equipment which is too big to be assigned to any location or $S_k \geq \max_i s_i, \ \forall k \in N$, and (c) all the locations are distinctive or $d_{jh} \neq 0, \ \forall j \neq h \in N$. We also assume that (d) the distance between locations is symmetric.

Let $A_M(L)$ be a set of strings of length $|M| (= m)$ using the digits in the ground set $N$. Then $m$-dimensional vector $\alpha \in A_M(N)$ represents an assignment and $\alpha(i)$ denote $i$-th component of vector $\alpha$. Then $\alpha(i) = k$ implies that equipment $i$ ($i$-th component in the ordered set $M$) is at location $k$.

**Definition 1** An assignment $\alpha \in A_M(L)$ is feasible if $\sum_{j \in \{i: \alpha(i) = k, i \in M\}} s_j \leq S_k, \ \forall k \in N$. Moreover, let $A_M^f(N|S)$, where $S = (S_1, \ldots, S_n)$, be a feasible set of strings of length $|M|$ with the ground set $N$ and available space $S_k$ at location $k \in N$.

A subset $A_M^f(N|S)$ is the set of all feasible assignments when a set of equipments $M$, a set of locations $N$, and a vector of available spaces $S$ are given. The total transportation cost of assignment $\alpha$ is the sum of the installation and the transportation costs, i.e.,

$$z = \sum_{i \in M} c_{i\alpha(i)} + v \sum_{i \in M} \sum_{j \in M} q_{ij} d_{\alpha(i)\alpha(j)}$$

The GQAP can be given in the following form.

$$z^* = \min_{\alpha \in A_M^f(L|S)} \sum_{i \in M} c_{i\alpha(i)} + v \sum_{i \in M} \sum_{j \in M} q_{ij} d_{\alpha(i)\alpha(j)}$$

where $z^*$ is the optimal objective value. Now we introduce an integer programming formulation of the GQAP by letting

$$x_{ik} = \begin{cases} 1, & \text{if equipment } i \text{ is assigned to location } k, \\ 0, & \text{otherwise.} \end{cases}$$

Then the GQAP can be formulated as the following integer program (IP) and
will be called as QP in the paper:

\[
\min \sum_{i=1}^{m} \sum_{k=1}^{n} c_{ik} x_{ik} + v \sum_{i=1}^{m} \sum_{j=1}^{m} \sum_{k=1}^{n} \sum_{h=1}^{n} q_{ij} d_{kh} x_{ik} x_{jh}
\]

s.t. \[\sum_{k=1}^{n} x_{ik} = 1, \quad \forall i \in \{1, \ldots, m\},\] \hspace{1cm} (1a)

\[\sum_{i=1}^{m} s_i x_{ik} \leq S_k, \quad \forall k \in \{1, \ldots, n\},\] \hspace{1cm} (1b)

\[x_{ik} \in \{0, 1\}, \quad \forall i, k.\] \hspace{1cm} (1c)

The IP formulation of the GQAP presented above is an extension of the Koopmans and Beckman formulation [12] for the QAP. The constraints (1a) ensure that each process can be and must be allocated once. The constraints (1b) make sure that the space limitation is not violated at each location.

The GQAP is a generalization of the QAP since, after grouping \(m\) equipments into \(n\) groups, the GQAP becomes the QAP where \(n\) groups are to be assigned to \(n\) locations. Some researchers defined the QAP as a problem of assigning \(n\) equipments to \(n\) or less locations, leaving some of locations empty. Adopting this definition of the QAP, among \(n\) groups of equipments, there can empty groups. This implies that we can group \(m\) equipments into 1 group, 2 groups, \(\ldots\), up to \(n\) groups. There are so many ways of doing so, hence the number of instances of the QAP that need to be solved to solve one instance of the GQAP is enormous, meaning that the complexity of the GQAP is significantly higher than that of the QAP. Notice that the number of equipments can be more than that of locations in such an argument.

**Theorem 2** The GQAP is strongly NP-hard.

**Proof:** To solve one instance of the GQAP, we need to solve one QAP for each possible way of grouping \(m\) equipments into \(n\) groups, where an empty group that has no equipment is allowed. Therefore, the GQAP is at least as complex as the QAP and it is known that QAP is strongly NP-hard [11]. \(\square\)

Generally speaking, as the space requirement becomes small relatively to the available space, there must be more ways of grouping the equipments, hence the GQAP becomes more complex. The maximum number of instances of the QAP that need to be solved to solve one instance of the GQAP is the number of ways of grouping \(m\) objects to \(n\) groups.
3 Linearization of the GQAP

There have been three major linearization approaches to the QAP in the past: Kaufman and Broeckx linearization [3], Frieze and Yadegar linearization [4], and Padberg and Rijal linearization [5]. All of them are mixed integer linear programming (MILP) formulations and based on the Koopmans and Beckmann formulation of the QAP. Since the GQAP is a generalization of the QAP, we try to extend the existing approaches to the QAP to the GQAP.

3.1 Frieze and Yadegar linearization

For the QAP with size of \( n \), Frieze and Yadegar first introduce \( n^4 \) Boolean variables \( y_{ikjh} \) by setting

\[
y_{ikjh} := x_{ik}x_{jh}, \quad \text{for } 1 \leq i, j, h, k \leq n.
\]

Applying the similar idea, we have the following MILP for the GQAP and name it as LPFY in the paper:

\[
\begin{align*}
\min & \quad \sum_{i=1}^{m} \sum_{k=1}^{n} c_{ik}x_{ik} + v \sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{k=1}^{n} \sum_{h=1}^{n} q_{ik}d_{jh}y_{ikjh} \\
\text{s.t} & \quad \sum_{k=1}^{n} x_{ik} = 1, \quad \forall i, \quad (2a) \\
& \quad \sum_{k=1}^{n} y_{ikjh} = x_{jh}, \quad \forall i, j, h, \quad (2b) \\
& \quad \sum_{h=1}^{n} y_{ikjh} = x_{ik}, \quad \forall i, j, k, \quad (2c) \\
& \quad \sum_{i=1}^{m} s_{i}x_{ij} \leq S_{j}, \quad \forall j, \quad (2d) \\
& \quad x_{ij} \in \{0, 1\}, \quad \forall i, j, \quad (2e) \\
& \quad 0 \leq y_{ikjh} \leq 1, \quad \forall i, j, h, k. \quad (2f)
\end{align*}
\]

**Remark 1** LPFY has \((m^2n^2)\) real variables, \((mn)\) binary variables, \((m^2n^2 + 2mn + m + n)\) constraints.

**Theorem 3** QP and LPFY are equivalent.

**Proof:** For a given \( x \) satisfying the constraints of QP given in (1a) through (1c) and \( y_{ikjh} = x_{ik}x_{jh} \), it is trivial to show that \((x, y)\) is feasible to LPFY and the values of objective function of LPFY and QP match. To show the other direction of the equivalency, we have to show that \((x, y)\) satisfies
\( y_{ikjh} = x_{ik}x_{jh} \) if \((x, y)\) is feasible to LPFY. Let \((x, y)\) be a feasible solution to LPFY. Then,

From (2b), \( \sum_{i=1}^{m} \sum_{k=1}^{n} y_{ikjh} = \sum_{i=1}^{m} x_{jh} = mx_{jh}, \forall j, h, \)

which implies, \( x_{jh} = 0 \Rightarrow \sum_{i=1}^{m} \sum_{k=1}^{n} y_{ikjh} = 0 \Rightarrow y_{ikjh} = 0, \forall k, h. \) (3)

From (2c), \( \sum_{j=1}^{m} \sum_{h=1}^{n} y_{ikjh} = \sum_{j=1}^{m} x_{ik} = mx_{ik}, \forall i, k, \)

which implies, \( x_{ik} = 0 \Rightarrow \sum_{j=1}^{m} \sum_{h=1}^{n} y_{ikjh} = 0 \Rightarrow y_{ikjh} = 0, \forall i, j. \) (4)

Let \( \varphi \in A_{M}(N|S) \) such that \( x_{i\varphi(i)} = 1 \) for \( i \in M \). Then we need to show that \( y_{i\varphi(i)k\varphi(k)} = 1 \) for \( i, k \in M \). We have

\[ \sum_{k=1}^{n} y_{ikjh} = y_{i\varphi(i)jh}, \forall i, j, h. \]

With \( h = \varphi(j) \), we have the following since \( \sum_{i=1}^{m} \sum_{k=1}^{n} y_{ikjh} = mx_{jh}, \forall j, h, \)

\[ \sum_{i=1}^{m} y_{i\varphi(i)j\varphi(j)} = m, \forall j. \]

Therefore, from (2f), we have that \( x_{ij} = x_{kh} = 1 \) implies \( y_{ikjh} = 1 \). This, along
with (3) and (4), shows the second part of the equivalency. \( \square \)

### 3.2 Kaufman and Broeckx linearization

Kaufman and Broeckx [3] derived a linearization of the QAP with \( n^2 \) Boolean variables, \( n^2 \) real variables and \( O(n^2) \) constraints. This formulation is probably the smallest linearization of the QAP in terms of the number of variables and constraints. We present a linearization of the GQAP, given below as LPKB,
based on the idea of Kaufman and Broeckx.

\[
\begin{align*}
\min & \quad \sum_{i=1}^{m} \sum_{k=1}^{n} \{c_{ik}x_{ik} + vy_{ik}\} \\
\text{s.t} & \quad \sum_{i=1}^{m} s_{ik}x_{ik} \leq S_k, \quad \forall k \\
& \quad \sum_{k=1}^{n} x_{ik} = 1, \quad \forall i \in M, \quad (5a) \\
& \quad w_{ik}x_{ik} + \sum_{j=1}^{m} \sum_{h=1}^{n} q_{ij}d_{kh}x_{jh} - y_{ik} \leq w_{ik}, \quad \forall i, k, \quad (5b) \\
& \quad x_{ik} \in \{0, 1\}, \quad \forall i, k, \quad (5c) \\
& \quad 0 \leq y_{ik}, \quad \forall i, k. \quad (5d)
\end{align*}
\]

where

\[
\begin{align*}
y_{ik} := x_{ik} \sum_{j=1}^{m} \sum_{h=1}^{n} q_{ij}d_{kh}x_{jh}
\end{align*}
\]

\[
\begin{align*}
w_{ik} = \sum_{j=1}^{m} \sum_{h=1}^{n} q_{ij}d_{kh}, \quad \forall i, k
\end{align*}
\]

**Remark 2** The LPKB has \((mn)\) real variables, \((mn)\) binary variables, \((mn + m + n)\) constraints.

The Kaufman and Broeckx linearization is the linearization with the least number of variables and constraints among the three linearization approaches discussed in the paper.

**Theorem 4** QP and LPKB are equivalent.

**Proof:** For a given feasible solution \(x\) to QP and \(y\) as defined in LPKB, it is straightforward to show that \((x, y)\) is feasible to LPKB and the values of objective function of QP and LPKB match. Now we are left to show the other direction of the equivalency. Let \((x, y)\) be a feasible solution to LPKB. Then,

\[
\begin{align*}
\text{From } (5c), \quad x_{jh} = 0 \Rightarrow y_{ik} \geq w_{ik}(x_{ik} - 1) + 0, \quad \forall j, h, \\
\text{From } (5c), \quad x_{ik} = 0 \Rightarrow y_{ik} \geq -w_{ik} + \sum_{j=1}^{m} \sum_{h=1}^{n} q_{ij}d_{kh}x_{jh}, \quad \forall i, k.
\end{align*}
\]

We should notice both \(w_{ij}(x_{ij} - 1)\) and \(-w_{ij} + \sum_{k=1}^{m} \sum_{h=1}^{n} q_{ik}d_{jh}x_{kh}\) are non-positive. Since \(y_{ij} \geq 0\) from \((5e)\) and LPKB is to minimize the sum of \(y_{ik}\), \(y_{ik}\) must be 0 when either \(x_{ik} = 0\) or \(x_{jh} = 0\).
Let $\varphi \in A^f_M(N|S)$ such that $x_{i\varphi(i)} = 1$ for $i \in M$. Then we need to show that

$$y_{i\varphi(i)} = \sum_{j=1}^{m} \sum_{h=1}^{n} q_{ij}d_{\varphi(i)h} = \sum_{j=1}^{m} q_{ij}d_{\varphi(i)\varphi(j)}$$

for $i, j \in M$.

We know

$$y_{ik} = y_{i\varphi(i)} \geq w_{i\varphi(i)}(x_{i\varphi(i)} - 1) + \sum_{j=1}^{n} \sum_{h=1}^{n} q_{ij}d_{\varphi(i)h}x_{jh}$$

$$= w_{i\varphi(i)}(x_{i\varphi(i)} - 1) + \sum_{j=1}^{m} q_{ij}d_{\varphi(i)\varphi(j)}x_{j\varphi(j)} = \sum_{j=1}^{m} q_{ij}d_{\varphi(i)\varphi(j)}$$

Since the problem is to minimize the sum of $y_{ik}$,

$$y_{ik} = \sum_{j=1}^{m} q_{ij}d_{\varphi(i)\varphi(j)}$$

Combining all above results, then we can conclude the following:

$$y_{ik} = x_{ik} \sum_{j=1}^{m} \sum_{h=1}^{n} q_{ij}d_{kh}x_{jh}, \forall 1 \leq i \leq m, 1 \leq j \leq n.$$  

\[\square\]

### 3.3 A New Linearization (L3)

The third linearization is motivated from both of Frieze and Yadegar linearization [4] and Padberg and Rijal Linearization [5] of the QAP. Padberg and Rijal set $y_{iklj} = x_{ij} \cdot x_{kh}$ and present the following observations:

1. If $y_{iklj} = x_{ik}x_{lj}, \forall i, j, k, h$, then $y_{iklj} = x_{ij}x_{kh} = x_{kh}x_{ij} = y_{klij}$.
2. Each equipment can be assigned exactly once, that is, $y_{ikih} = x_{ik}x_{ih} = 0$.

These observations allow us to focus only on variables $y_{iklj}$ with $i < j$ if we introduce a restriction of $1 \leq i < j \leq m$. This motivates us to revise Frieze and Yadegar linearization [4] and reduce the number of decision variables and the number of constraints.

Let

$$y_{iklj} := x_{ik}x_{lj}, \forall i, j, k, h.$$
The Linear model for the GQAP is given by the following linear program L3:

\[
\begin{align*}
\min & \quad \sum_{i=1}^{m} \sum_{k=1}^{n} c_{ik} x_{ik} + v \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \sum_{k=1}^{n} \sum_{h=1}^{n} (q_{ij} d_{kh} + q_{ji} d_{hk}) y_{ikjh} \\
\text{s.t} & \quad \sum_{k=1}^{n} x_{ik} = 1, \quad \forall i \quad (7a) \\
& \quad \sum_{h=1}^{n} y_{ikjh} = x_{ik}, \quad \forall i, k, \text{ and } i < j \leq m \quad (7b) \\
& \quad \sum_{k=1}^{n} y_{ikjh} = x_{jh}, \quad \forall j, h, \text{ and } 1 \leq i < j \leq m \quad (7c) \\
& \quad \sum_{i=1}^{m} s_{i} x_{ik} \leq S_{k}, \quad \forall k \quad (7d) \\
& \quad x_{ik} \in \{0, 1\}, \quad \forall i, k \quad (7e) \\
& \quad 0 \leq y_{ikjh} \leq 1, \quad 1 \leq i < j \leq m, 1 \leq k, h \leq n. \quad (7f)
\end{align*}
\]

**Remark 3** The L3 has \(((m-1)^2 n^2 / 2)\) real variables, \(m n\) binary variables, \(((m-1)^2 n^2 + 2(m-1)^2 n + m + n)\) constraints.

**Theorem 5** The QP and L3 are equivalent.

**Proof:** The objective function of L3 is the same as that of QP as shown below:

\[
\begin{align*}
\sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \sum_{k=1}^{n} \sum_{h=1}^{n} (q_{ij} d_{kh} + q_{ji} d_{hk}) y_{ikjh} &= \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \sum_{k=1}^{n} \sum_{h=1}^{n} (q_{ij} d_{kh} + q_{ji} d_{hk}) y_{ikjh} \\
&= \sum_{i=1}^{m} \sum_{j=i+1}^{m} \sum_{k=1}^{n} \sum_{h=1}^{n} q_{ij} d_{kh} y_{ikjh} + \sum_{i=1}^{m} \sum_{j=i+1}^{m} \sum_{k=1}^{n} \sum_{h=1}^{n} q_{ji} d_{hk} y_{ikjh} \\
&= \sum_{i=1}^{m} \sum_{j=i+1}^{m} \sum_{k=1}^{n} \sum_{h=1}^{n} q_{ij} d_{kh} y_{ikjh} + \sum_{i=1}^{m} \sum_{j=i+1}^{m-1} \sum_{k=1}^{n} \sum_{h=1}^{n} q_{ij} d_{kh} y_{ikjh} \\
&= \sum_{i=1}^{m} \sum_{j=i+1}^{m} \sum_{k=1}^{n} \sum_{h=1}^{n} q_{ij} d_{kh} y_{ikjh}.
\end{align*}
\]

For a given feasible solution \(x\) to QP, let \(y_{ikjh} = x_{ik} x_{jh}\). Then it is trivial to show that \((x, y)\) is feasible to L3. The other direction of the equivalency can be shown by the same argument in the proof of Theorem 3. \(\square\)

4 **Branch and Bound approach:**

There are three major branch and bound algorithms for the QAP in terms of branching-rules: (1) single assignment methods, (2) pair assignment methods, and (3) relative positioning methods [9]. In the single assignment method,
only a single equipment is assigned to a location at a time, whereas, in a pair assignment method, a pair of equipments are assigned to a pair of locations. In the relative positioning method, a partial assignment, or an assignment of a subset of equipments, at each level are determined in terms of distances between locations, i.e. their relative positions. In the paper, we employ the first approach due to the space constraints that we have.

4.1 The Upper Bound

We propose a greedy algorithm to find an initial solution to the GQAP, which will then provide an upper bound. Let $M$ be the set of all equipments, $M^u$ the set of all unassigned equipments, and $M^a$ be the set of all assigned equipments, then $M^a = M \setminus M^a$. The greedy algorithm is as follows:

**The Greedy Algorithm**

Begin with $s_i, S_k, d_{kh}, c_{ik}, q_{ij}, v, M^u := M, M^a := \emptyset$ and $N$;  
Order all locations from the best to the worst using the given criterion;  
Order all equipments from the best to the worst using the given criterion;  
While $M^u \neq \emptyset$ do  
Let $k$ be the best location in $N$ and $i$ the best equipment in $M^u$ such that $s_i \leq S_k$;  
Assign equipment $i$ to location $k$;  
$S_k = S_k - s_i$; $M^u := M^u \setminus \{i\}; M^a := M^a \cup \{i\}$;  
End of while  
The upper bound $z = v \sum_{i \in M^a} \sum_{j \in M^a} \sum_{k \in N} \sum_{h \in N} q_{ij}d_{kh}x_{ik}x_{jh} + \sum_{i \in M^a} \sum_{k \in N} c_{ik}x_{ik}$;  
End.

There are multiple alternatives for the criteria used in Step 3 and 4 in the greedy algorithm. Let

$$TD_k = \sum_{h \in N, h \neq k} d_{kh}, \forall k \in N \quad \text{and} \quad TF_i = \sum_{j \in M, j \neq i} (q_{ij} + q_{ji}), \forall i \in M.$$

Then we can think of the following criteria:

1. For locations,
   (a) $R_k = S_k$, order in non-ascending  
   (b) $R_k = TD_k/S_k$, order in non-descending  
   (c) $R_k = TD_k$, order in non-descending

2. For equipments,
   (a) $R_i = TF_i$, order in non-ascending  
   (b) $R_i = TF_i/s_i$, order in non-ascending  
   (c) $R_i = s_i$, order in non-descending
(d) \( R_i = \max_j q_{ij} \), order in non-ascending

We can make 12 different combinations of a criterion for equipment selection and a criterion for location selection. Since finding a heuristic solution using each of 12 greedy algorithms virtually takes no time, we apply all the combination of criteria to start with the best upper bound for the branch and bound algorithm, which will be presented shortly. Moreover, we use one more greedy algorithm that is based on the installation cost \( c_{ik} \). The criterion based on \( c_{ik} \) is not for ordering the equipments nor the locations but pairs of an equipment and a location since \( c_{ik} \) is defined for the pairs. According to this criterion, we order pairs of equipment \( i \) and location \( k \) in the increasing order of \( c_{ik} \) and perform assignment in the order, hoping the resulting installation cost to be reasonably small. This combination of criteria would work well when the installation cost dominates.

Once the branch and bound algorithm starts, it will produce partial assignments, where a subset of equipments are assigned. At each node in the branch and bound search tree, there will be a smaller scale problem defined for unassigned equipments and the locations with reduced available spaces. By applying the greedy algorithm with various criteria to the GQAP with unassigned equipment, we will get an upper bound for the partial problem, which can be added to the cost of the partial assignment at the node to get a new upper bound for the original GQAP. We will also get a new upper bound whenever the branch and bound search reaches a leaf node of the tree, that is the search finds a feasible full assignment. The initial upper bound will be constantly updated by such upper bounds throughout the search.

4.2 The Lower Bound

The generalized linear assignment problem (GLAP) provides a lower bound for the GQAP. The GLAP is defined for every pair of equipment \( i \) and location \( k \) as follows:

\[
\min_{\alpha \in A^i_M(L|S)} \sum_{j=1}^{m} v_{ij} d_{k\alpha(j)}, \forall i \in M, \forall k \in L.
\] (8)

Now let \( f_{ik} \) be the optimal objective function value of the GLAP. Then \( f_{ik} \) represents the minimum cost between equipment \( i \), which is fixed at location \( k \), and all the other equipments.

**Theorem 6** The optimal objective function value \( z^* \) of QP is bounded from below by the solution of the following GLAP

\[
z = \min_{\alpha \in A^i_M(L|S)} \sum_{i=1}^{m} (f_{i\alpha(i)} + c_{i\alpha(i)}).
\]
Proof: Let $e_{ikjh} = q_{ij}d_{kh}, \forall i, j \in M, \forall k, h \in N$ and $e_{ikih} = 0, \forall h \neq k$. We defined $f_{ik}$ as a solution of the GLAP given as follows.

$$\min_{\alpha} \sum_{j} v e_{ikjh(j)}$$

Therefore,

$$\sum_{i=1}^{m} f_{i\alpha(i)} = \sum_{i=1}^{m} \min_{\alpha} \sum_{j=1}^{m} v e_{i\alpha(i)j\alpha(j)} \leq \sum_{i=1}^{m} \sum_{j=1}^{m} v e_{i\alpha(i)j\alpha(j)}.$$

Hence the objective function value $z$ is bounded from below by

$$z = \min_{\alpha \in \mathcal{A}_m(L|S)} \sum_{i=1}^{m} (f_{i\alpha(i)} + c_{i\alpha(i)}) \leq \min_{\alpha \in \mathcal{A}_m(L|S)} \sum_{i=1}^{m} \left\{ \sum_{j=1}^{m} v e_{i\alpha(i)j\alpha(j)} + c_{i\alpha(i)} \right\} = z^*.$$

$\Box$

In the branch and bound algorithm, it is essential to compute a lower bound for a partial assignment. Suppose that we are given a partial assignment $\delta$ of $M^u$ and a vector of available spaces $S^u$ after allocating space to support the partial assignment $\delta$. Let

$$z(\delta, M^u) = \sum_{i \in M^u} c_{i\delta(i)} + v \sum_{i \in M^u} \sum_{j \in M^u} q_{ij}d_{\delta(i)\delta(j)}$$

Define

$$I_{ik} = \sum_{j \in M^u} v (q_{ij} + q_{ji})d_{j\alpha(j)x_{j\alpha(j)}}, \forall i \in M^u, \forall k \in N.$$ 

**Theorem 7** The optimal objective function value $z^*$ with a partial assignment $\delta$ on $M^u$ is bounded from below by

$$z^\delta = \min_{\alpha} \sum_{i \in M^u} \omega_{i\alpha(i)} + z(\delta, M^u),$$

where $\omega_{ik} = f_{ik} + I_{ik} + c_{ik}, \forall i \in M^u, \forall k \in N$.

**Proof:** Let $z$ be the value of the objective function to be minimized. Then,

$$z = v \sum_{i \in M^u} \sum_{k \in N} \sum_{j \in M^u} \sum_{h \in N} q_{ij}d_{kh}x_{ik}x_{jh} + v \sum_{i \in M^u} \sum_{k \in N} \sum_{j \in M^u} \sum_{h \in N} q_{ij}d_{kh}x_{ik}x_{jh}$$

$$+ v \sum_{i \in M^u} \sum_{k \in N} \sum_{j \in M^u} \sum_{h \in N} q_{ij}d_{kh}x_{ik}x_{jh} + v \sum_{i \in M^u} \sum_{k \in N} \sum_{j \in M^u} \sum_{h \in N} q_{ij}d_{kh}x_{ik}x_{jh}$$

$$+ \sum_{i \in M} \sum_{k \in N} c_{ik}x_{ik}.$$ 

Since the distance is symmetric, the third and the fourth terms can be rewrit-
\[v \sum_{i \in M^a} \sum_{k \in N} \left( \sum_{j \in M^a} \sum_{h \in N} (q_{ij} + q_{ji})d_{kh}x_{jh} \right) x_{ik} = \sum_{i \in M^a} \sum_{k \in N} I_{ik}x_{ik}.\]

Then,
\[
z = v \sum_{i \in M^a} \sum_{k \in N} \sum_{j \in M^a} \sum_{h \in N} q_{ij}d_{kh}x_{ik}x_{jh} + \sum_{i \in M^a} \sum_{k \in N} \left( v \sum_{j \in M^a} \sum_{h \in N} q_{ij}d_{kh}x_{jh} + I_{ik} \right) x_{ik} + \sum_{i \in M} \sum_{k \in N} c_{ik}x_{ik}
\geq z(\delta, M^a) + \sum_{i \in M^a} \sum_{k \in N} \left( \min_{h \in N} \left\{ v \sum_{j \in M^a} \sum_{h \in N} q_{ij}d_{kh}x_{jh} \right\} + I_{ik} \right) x_{ik}
+ \sum_{i \in M^a} \sum_{k \in N} c_{ik}x_{ik}
= z(\delta, M^a) + \sum_{i \in M^a} \sum_{k \in N} (f_{ik} + I_{ik} + c_{ik})x_{ik}.
\]

To minimize for the both sides:
\[
z^* \geq z(\delta, M^a) + \min_{i \in M^a} \min_{k \in N} \left\{ f_{ik} + I_{ik} + c_{ik} \right\} x_{ik}
= z(\delta, M^a) + \min_{\alpha} \sum_{i \in M^a} \omega_{\alpha(i)} = z^\delta.
\]

\[\square\]

Theorem 7 allows us to compute a lower bound at a node in the branch and bound search tree, that is a lower bound when a subset of the equipments are already assigned.

### 4.3 The Branching

Recall that \(f_{ik}\) is defined by Equation (8). Moreover, let \(F_{ik} = f_{ik} + c_{ik}\). Then \(F_{ik}\) is a lower bound of the cost incurred by the assignment of equipment \(i\) to location \(k\). Furthermore, define
\[
P_{ik} = \min_{h \in N} \{ F_{ih} | F_{ih} \geq F_{ik}, \text{ and } h \neq k \} - F_{ik}, \forall i \in M, \forall k \in N.
\]

\(P_{ik}\) equals to the minimum increase to the lower bound if equipment \(i\) is not assigned to location \(k\). In order to exclude as many subproblems as possible at an early stage of the search, we search for the maximum of all \(P_{ik}, \forall i \in M^a, \forall k \in N\) and branch out to two nodes: a node for assignment \((i \rightarrow k)\)
and a node for null-assignment \((i \not\rightarrow k)\). An assignment \((i \rightarrow k)\) indicates equipment \(i\) is assigned to location \(k\) and a null-assignment \((i \not\rightarrow k)\) indicates that equipment \(i\) is not assigned to location \(k\). Since we implement the depth-first search scheme, we go down to the leaf following “assignment” instead of “null-assignment” and nodes representing “null-assignment” will be branched out later.

We also experiment variations of the branching rule described above. Among other things we found the space requirement of equipments and the available space at locations play important roles in determining the performance of the search algorithm. We replace \(F_{ik}\) with any one of five alternatives as shown below.

\[
F_{ik} := \text{one of } \{ F_{ik}, F_{ik}s_i, \frac{F_{ik}}{s_i}, \frac{F_{ik}s_i}{S_k}, \frac{F_{ik}s_i}{S^u_k} \}.
\]

where \(S_k\) is the space available at location \(k\) before any assignment and \(S^u_k\) is the space available at location \(k\) after a partial assignment.

The branch and bound algorithm has been implemented in C++ using CPLEX callable library. We do not implement any specialized code to solve the GLAP, which plays a critical role in finding lower bounds for each branch and bound node. The lower bound, the upper bound, and the branching rules discussed so far have been used in the implementation of the branch and bound algorithm.

5 Computational Results and Empirical Study

In this section we present computational studies on the three linearization approaches and the branch and bound algorithm for the GQAP.

5.1 Comparison among Three Linearizations and the Branch and Bound

For comparison among the three linearization approaches and the branch and bound algorithm, we use AMPL/CPLEX version 8.1 on a desktop computer with an Intel Pentium III 677 MHz. Table 1 shows the result of the computational comparison. In Table 1 the first column shows the code given to individual problem instances. A code consists of three parts: a number before “X” representing the number of equipments, a number after “X” representing the number of locations, and the last two letters indicating more details of the instances as given below.

E: equipments have a wide range of space requirement,
F: equipments have roughly the same space requirement,
<table>
<thead>
<tr>
<th>Code</th>
<th>Frize-Yadeger</th>
<th>Kaufman-Broeckx</th>
<th>L3</th>
<th>Branch &amp; Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CPU</td>
<td>Tree</td>
<td>CPU</td>
<td>Tree</td>
</tr>
<tr>
<td>10X5EG</td>
<td>39.2</td>
<td>623</td>
<td>12.2</td>
<td>5994</td>
</tr>
<tr>
<td>10X5EH</td>
<td>49.8</td>
<td>666</td>
<td>21.4</td>
<td>10566</td>
</tr>
<tr>
<td>10X5FG</td>
<td>51.6</td>
<td>566</td>
<td>5.2</td>
<td>12449</td>
</tr>
<tr>
<td>10X5FH</td>
<td>96.2</td>
<td>1776</td>
<td>26.5</td>
<td>13666</td>
</tr>
<tr>
<td>12X3EG</td>
<td>5.4</td>
<td>133</td>
<td>2.0</td>
<td>1231</td>
</tr>
<tr>
<td>12X3EH</td>
<td>17.6</td>
<td>585</td>
<td>8.4</td>
<td>5565</td>
</tr>
<tr>
<td>12X3FG</td>
<td>19.6</td>
<td>1066</td>
<td>6.3</td>
<td>3305</td>
</tr>
<tr>
<td>12X3FH</td>
<td>17.3</td>
<td>621</td>
<td>14.8</td>
<td>10566</td>
</tr>
<tr>
<td>12X4EG</td>
<td>31.8</td>
<td>501</td>
<td>11.2</td>
<td>5604</td>
</tr>
<tr>
<td>12X4EH</td>
<td>102.9</td>
<td>1372</td>
<td>29.6</td>
<td>16409</td>
</tr>
<tr>
<td>12X4FG</td>
<td>88.5</td>
<td>2037</td>
<td>55.3</td>
<td>32717</td>
</tr>
<tr>
<td>12X4FH</td>
<td>82.0</td>
<td>925</td>
<td>5.9</td>
<td>1935</td>
</tr>
<tr>
<td>12X5EG</td>
<td>119.1</td>
<td>1313</td>
<td>65.4</td>
<td>20238</td>
</tr>
<tr>
<td>12X5EH</td>
<td>567.9</td>
<td>5177</td>
<td>20.6</td>
<td>4445</td>
</tr>
<tr>
<td>12X6</td>
<td>651.5</td>
<td>2769</td>
<td>18.2</td>
<td>4214</td>
</tr>
<tr>
<td>12X8</td>
<td>1091.0</td>
<td>1833</td>
<td>1333.9</td>
<td>240724</td>
</tr>
<tr>
<td>14X6</td>
<td>480.6</td>
<td>1467</td>
<td>52.5</td>
<td>12449</td>
</tr>
<tr>
<td>14X7</td>
<td>1670.2</td>
<td>4308</td>
<td>172.1</td>
<td>30921</td>
</tr>
<tr>
<td>14X8</td>
<td>2717.6</td>
<td>1755</td>
<td>256.4</td>
<td>23866</td>
</tr>
<tr>
<td>14X9</td>
<td>3759.1</td>
<td>2252</td>
<td>7881.7</td>
<td>871127</td>
</tr>
<tr>
<td>15X6</td>
<td>1614.1</td>
<td>5308</td>
<td>109.4</td>
<td>21183</td>
</tr>
<tr>
<td>15X7</td>
<td>1971.3</td>
<td>2569</td>
<td>658.7</td>
<td>121059</td>
</tr>
<tr>
<td>15X8</td>
<td>4056.1</td>
<td>2136</td>
<td>397.5</td>
<td>49240</td>
</tr>
<tr>
<td>16X6</td>
<td>1815.7</td>
<td>1915</td>
<td>630.3</td>
<td>237272</td>
</tr>
<tr>
<td>16X7</td>
<td>22423.7</td>
<td>17461</td>
<td>988.7</td>
<td>106803</td>
</tr>
<tr>
<td>5X24</td>
<td>17.3</td>
<td>57</td>
<td>8.5</td>
<td>3977</td>
</tr>
<tr>
<td>5X30</td>
<td>20.0</td>
<td>75</td>
<td>32.5</td>
<td>8845</td>
</tr>
</tbody>
</table>

G: locations have a wide range of available space,
H: locations have roughly the same available space.

in which all v, q_{ij}, d_{kh} and c_{ik} are the same in all combinations of E, F, G, and H. The second column of Table 1 shows the CPU times in seconds and the number of nodes in the branch and bound search tree.

We make the following observations:

1. L3 and Kaufman-Broeckx linearization perform better than Frieze-Yadeger linearization and the branch and bound algorithm. L3 appears more often than Kaufman-Broeckx linearization as the best performer when we ignore
Table 2
The quality of initial upper bound (frequencies for intervals not shown are all zeros.)

<table>
<thead>
<tr>
<th>Gap</th>
<th>Frequency</th>
<th>Gap</th>
<th>Frequency</th>
<th>Gap</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0%, 5%)</td>
<td>9</td>
<td>(20%, 25%)</td>
<td>5</td>
<td>(45%, 50%)</td>
<td>1</td>
</tr>
<tr>
<td>(5%, 10%)</td>
<td>12</td>
<td>(25%, 30%)</td>
<td>3</td>
<td>(50%, 55%)</td>
<td>0</td>
</tr>
<tr>
<td>(10%, 15%)</td>
<td>11</td>
<td>(30%, 35%)</td>
<td>2</td>
<td>(55%, 60%)</td>
<td>0</td>
</tr>
<tr>
<td>(15%, 20%)</td>
<td>10</td>
<td>(40%, 45%)</td>
<td>1</td>
<td>(60%, 65%)</td>
<td>1</td>
</tr>
</tbody>
</table>

less than one second difference in CPU time. L3 also seems to use less memory than Kaufman-Broeckx linearization: L3 uses 0.86% to 44.59% of memory used by Kaufman-Broeckx linearization except for only a single case (12X6), where it uses 112.33%. This implies that L3 is more memory efficient and can possibly solve bigger problems.

(2) Although the branch and bound algorithm appears most frequently as the worst performer, it outperforms Frieze-Yadeger linearization in large problem instances (larger instances than 12X4).

(3) The L3 is much more powerful than Frieze-Yadeger linearization. The computing times of L3 range from 10.13% to 40.89% of the times of Frieze-Yadeger linearization.

(4) Not a single linearization approach consistently outperforms the branch and bound algorithm. The branch and bound algorithm also shows the highest stability in terms of CPU time and memory consumption.

5.2 Performance Analysis of the Branch and Bound Algorithm

Numerical analysis that will be presented in this section is based on an implementation of the branch and bound algorithm using C++ and CPLEX version 6.5 on a desktop computer with a Pentium III 1 GHz.

Impact of the Initial Upper Bound

The quality of the initial upper bound is critical for the preformation of the branch and bound algorithm. We compare the initial upper bound with optimal solution of 60 problem instances. Some initial upper bounds are pretty tight, the greed solution is optimal in 1 case, while we could not find any feasible solution using the greedy algorithm in Section 4.1 in 4 cases. Among 55 problem instances where the greedy algorithm finds upper bounds, upper bounds are less than 1% above optimal solutions in 4 cases, less than 5% in 9 cases, less than 10% in 21 cases, and so on. The complete statistics are given in Figure ??, where the x-axis represents the gap between upper bound and optimal solution in per cent and the y-axis represents the frequencies. Excluding 5 cases where the greedy algorithm cannot find any feasible solution or finds an optimal solution, the average gap between the upper bound and the optimal
solution is 19.95%.

Impact of the Branching Rule

The branching rules are implemented using five different definitions of $F_{ik}$ as discussed in Section 4.3. Our experience with the five definitions shows that $F_{ik}s_i$ performs the best in terms of the size of the branch and bound tree and the computational time, followed by $\frac{F_{ik}s_i}{S_k}$, then by $\frac{F_{ik}s_i}{S_k}$, then by $F_{ik}$, and the last $F_{ik}$.

Impact of Space Requirements

Problem instances with high variance in the space requirement and in the available space (i.e., problem instances with codes ending with EG) tend to be solved more efficiently than the other types.

Impact of Transportation Cost Coefficient: $v$

The transportation cost coefficient $v$ acts as a weight controller between the installation cost (linear cost) and the transportation cost (quadratic cost) over a given planning horizon. If $v = 0$, the cost only includes the installation cost and the problem becomes the generalized linear assignment problem, which is significantly easier than the GQAP. If $v = \infty$ or $c_{ik} = 0, \forall i \in M, \forall k \in N$, the problem becomes the pure generalized quadratic assignment problem. This implies the computational efforts required to solve the GQAP should increase as the value of $v$ increases from 0.

As shown in Table 3, we solve the problem instance C12X4FH for various values of $v$ in order to demonstrate the impact of the value of $v$ on the complexity of the problem. Table 3 clearly demonstrates that $v$ greatly affects the difficulty of the problem. This also shows the complexity comparison between the linear and the quadratic assignment problem.

6 Conclusions

In this paper, we define a new class of problems called the generalized quadratic assignment problem (GQAP), which is a generalization of the quadratic assignment problem (QAP). We show that the GQAP is significantly more complex than the QAP, which has been known to be one of the most challenging combinatorial optimization problems in the last half century. The GQAP has
Table 3
The effect of $v$ on the performance of the Branch and Bound algorithm

<table>
<thead>
<tr>
<th>$v$</th>
<th>Optimal value</th>
<th># of B&amp;B nodes</th>
<th># of CPLEX calls</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>401,500</td>
<td>196</td>
<td>3296</td>
</tr>
<tr>
<td>0.0000001</td>
<td>401,500</td>
<td>213</td>
<td>3475</td>
</tr>
<tr>
<td>0.00001</td>
<td>401,503.9</td>
<td>218</td>
<td>3514</td>
</tr>
<tr>
<td>0.001</td>
<td>401,888.2</td>
<td>216</td>
<td>3485</td>
</tr>
<tr>
<td>0.01</td>
<td>405,382</td>
<td>209</td>
<td>3432</td>
</tr>
<tr>
<td>0.1</td>
<td>440,320</td>
<td>221</td>
<td>3698</td>
</tr>
<tr>
<td>0.5</td>
<td>595,600</td>
<td>265</td>
<td>5012</td>
</tr>
<tr>
<td>1</td>
<td>779,900</td>
<td>286</td>
<td>6314</td>
</tr>
<tr>
<td>2</td>
<td>1,138,300</td>
<td>346</td>
<td>8563</td>
</tr>
<tr>
<td>3</td>
<td>1,496,700</td>
<td>460</td>
<td>11287</td>
</tr>
<tr>
<td>5</td>
<td>2,213,500</td>
<td>583</td>
<td>14542</td>
</tr>
<tr>
<td>8</td>
<td>3,279,800</td>
<td>836</td>
<td>19026</td>
</tr>
<tr>
<td>10</td>
<td>3,971,000</td>
<td>1193</td>
<td>24007</td>
</tr>
<tr>
<td>15</td>
<td>5,699,000</td>
<td>1129</td>
<td>22433</td>
</tr>
<tr>
<td>20</td>
<td>7,411,800</td>
<td>1200</td>
<td>23345</td>
</tr>
<tr>
<td>50</td>
<td>17,625,000</td>
<td>1222</td>
<td>24341</td>
</tr>
<tr>
<td>100</td>
<td>34,599,900</td>
<td>1189</td>
<td>23040</td>
</tr>
<tr>
<td>200</td>
<td>68,473,900</td>
<td>1178</td>
<td>22982</td>
</tr>
<tr>
<td>500</td>
<td>170,095,900</td>
<td>1141</td>
<td>22208</td>
</tr>
<tr>
<td>1000</td>
<td>339,465,900</td>
<td>1190</td>
<td>22732</td>
</tr>
<tr>
<td>5000</td>
<td>1,694,425,900</td>
<td>1196</td>
<td>22672</td>
</tr>
<tr>
<td>$C_{ik} = 0, \forall i, k$</td>
<td>338,740</td>
<td>1011</td>
<td>19360</td>
</tr>
</tbody>
</table>

numerous applications whenever multiple equipments (or objects) can be assigned to a single location (or assignee) subject to availability of required resources. We present three linearization approaches and a branch and bound algorithm to solve the GQAP along with numerical studies.

References


