CL615 Optimization Techniques: Term Project on Solution to quadratic assignment problems (QAP) using Ant Colony System

by
Ajay Shankar Bidyarthy
b.ajay@iitg.ernet.in
09012305
Department of Mathematics

November 11, 2012

Contents

List of Figures

Chapter 1

Solution to quadratic assignment problems (QAP) using Ant Colony System

1.1 Introduction

Quadratic assignment problems (QAPs) belong to the most difficult combinatorial optimization problems. The QAP was introduced by Koopmans and Beckmann in 1957 as a mathematical model for the location of indivisible economical activities.The quadratic assignment problem (QAP) is the problem of assigning n facilities to n locations so that the cost of the assignment, which is a function of the way facilities have been assigned to locations, is minimized.
We want to assign n facilities to n locations with the cost being proportional to the flow between the facilities multiplied by the distances between the locations plus, eventually, costs for placing the facilities at their respective locations. The objective is to allocate each facility to a location such that the total cost is minimized.

The QAP problem was first presented by Koopmans and Beckmann (1957) by the three matrices of dimension $n \times n$in the following form:

$$\begin{aligned} D = [d_{ih}] = &\quad \text{matrix of the distances (between location } i \text{ and location } h); \\ F = [f_{jk}] = &\quad \text{matrix of the flows (between activity } j \text{ and activity } k); \\ C = [c_{ij}] = &\quad \text{matrix of the assignment costs (of activity } j \text{ to location } i). \end{aligned} \tag{1.1}$$

Matrices D and F are integer-valued symmetrical matrices, the assignment cost c_{ij} of activity j to location i is usually ignored, as it does not make a significant contribution to the complexity of solving the problem.

After this construction, a permutation $\Pi : i- > \pi(i)$ can be introduced as a particular assignment of activity $j = \pi(i)$ to location $i (i = 1, 2, 3, ..., n)$.

The cost of transferring data between two activities can be expressed as the product of the distance between the locations to which the activities are assigned by the flow between the two activities, $d_{ih} f_{\pi(i)\pi(h)}$.

In order to solve the QAP we must find a permutation Π of the indices $1, 2, 3, ..., n$ which minimizes the local assignment cost:

$$\min\ z = \sum_{i,h=1}^{n} d_{ih} f_{\pi(i)\pi(h)} \tag{1.2}$$

The problem is formulated to show the quadratic nature of the objective function: solving the problem means identifying a permutation matrix $\mathbf{X}$ of dimension $n \times n$ (whose elements X_{ij} are 1 if activity j is assigned to location i and 0 in the other cases) such that:

$$\min z = \sum_{i,j=1}^{n} \sum_{h,k=1}^{n} d_{ih} f_{jk} X_{ij} X_{hk}$$

Subject to the following constraints fot the elements X_{ij}

$$\begin{aligned} &\sum_{i=1}^{n} X_{ij} = 1 \quad \text{for } j = 1, 2, 3, ..., n \\ &\sum_{j=1}^{n} X_{ij} = 1 \quad \text{for } i = 1, 2, 3, ..., n \\ &X_{ij} \in (0, 1) \quad \text{for } i, j = 1, 2, 3, ..., n \end{aligned} \tag{1.3}$$

These constraints identify the matrix $\mathbf{X}$ as belonging to set Π of the permutation matrices of order n. Please note that the QAP is a generalization of the Traveling Salesman Problem (TSP).

1.2 The ANT Colony System (ACS)

The quadratic assignment problem (QAP) is the problem of assigning n facilities to n locations so that the cost of the assignment, which is a function of the way facilities have been assigned to locations, is minimized.

1.2.1 How Ants Find a Shortest Path

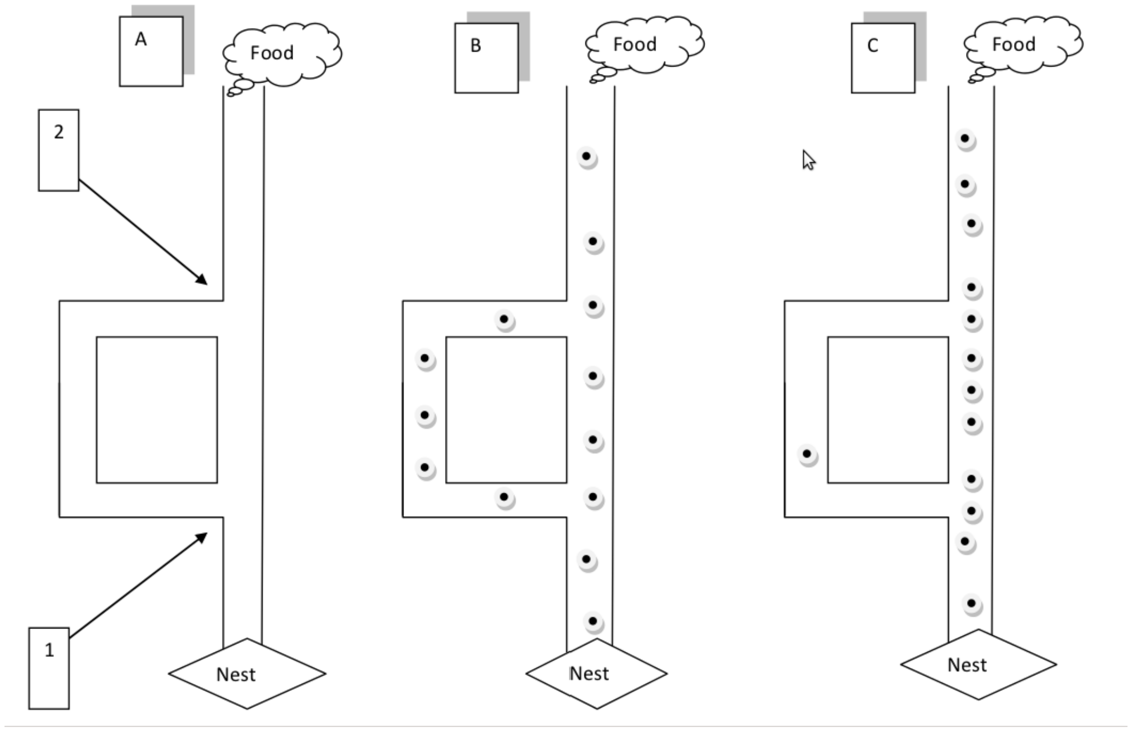


Figure 1.1: How ants exploit pheromone to find a shortest path

In A ants arrive at decision point 1 and 2. In B some ants choose the right path and some the left path. The choice is random. As the ants move at a constant similar speed, the ants that choose the right path will arrive at their destination quicker than those who choose the left path. Since pheromone accumulates where more ants travel, the shortest path begins to be the main choice of the ants in C. shortly all ants will choose the shorter, and more efficient path.

1.2.2 ACS pheromone configuration

Figure 2 shows an example of a trial configuration found by ACS in a network optimization problem. Line thickness reflects pheromone concentration. Heavier thickness indicates heavier concentration. Some edges are strongly marked, while others are weakly are weakly marked. Strongly marked edges are most likely to be part of the best solution found by the system. However, the weakly marked adges point to potential alternative solution. By relating these solution, local minimum solutions may be avoided. In the case of dynamic problems, where edges and nodes are constantly being modified or added, an alternative suite of weaker solutions may become strong ones. It may be possible that dynamic problems may become strong condidates for ACS methods.

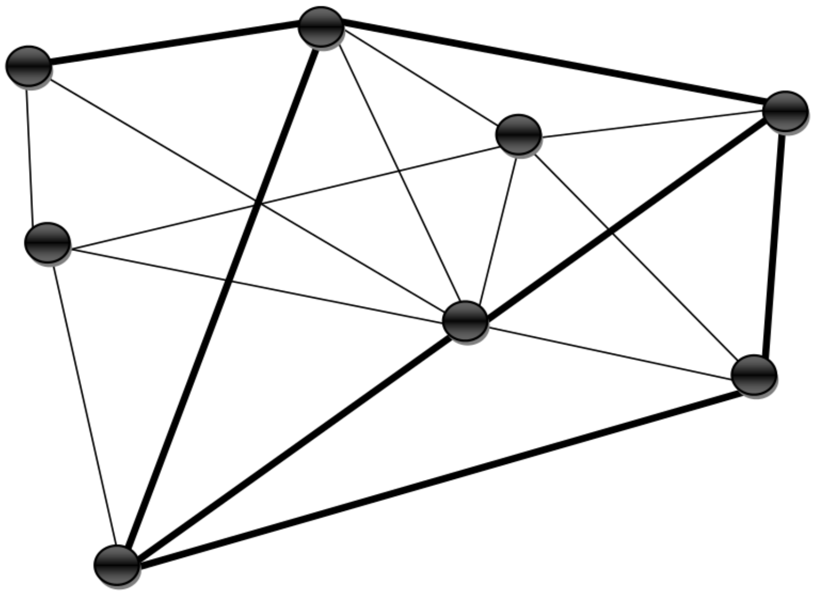


Figure 1.2: How ants exploit pheromone to find a shortest path

1.2.3 ACS Algorithm

According to our construction of QAP, we have a flow matrix, f, whose (i, j) element represents the flow between facilities i and j, and a distance matrix, d, whose (i, j) element represents the distance between locations i and j. A vector $p(j)$ represents the location to which facility j is assigned. This vector is a permutation of the numbers $\{1, 2, ..., n\}$.

Assumed that ants are traveling on a graph from a city r to the next city s. Each edge (r, s) on this graph has a cost (length), $\delta(r, s)$, and a pheromone measure, $\tau(r, s)$, which is updated every time an ant walks over this edge. Each ant builds a complete tour, by choosing its next city according to a probabilistic transition rule. The transition rule implies that better odds are given to cities connected by short edges with high amounts of pheromone. Figure 2 shows conceptually how the pheromone concentrations may vary as the ants traverse the network.

According to the transition rule, an ant located at city r, will choose its next city s, according to the equation:

$$s = \begin{cases} \arg\max_{u \in J_{k}(r)} \{[\tau(r,u)]^{\mu}[\eta(r,u)]^{\beta}\} & \text{if } q < q_0 \\ \text{Dttermined by Equation (5)} & \text{otherwise} \end{cases} \tag{1.4}$$

Where,
$J_{k(r)}$ is the set of cities that remain to be visited by ant k positioned in city r and $\eta(r, u) = 1/\delta(r, u)$.
$[\mu, \beta]$ are parameters that set the relative importance of pheromone vs. distance.
q_0 is a constant parameter, defined by the range $(0 < q_0 < 1)$, which is selected to estab-

lish the exploitation vs. exploration. As q_0 decreases the ant chooses more according to pheromone concentrations (more exploitation). As q_0 increases the ant chooses more according to probability (more exploration). Exploration would tend to keep the ants out of local minimums. Thus, q_0 may be related to risk tolerance; higher values suggest more risk tolerance. Thus, exploration (high values of q_0) would be appropriate at the beginning of a search and exploitation at the end of a search.
$q =$ is a random number defined by the range $(0 < q < 1)$. Higher values of q improve the richness of the search paths chosen by the ants. If q is less than q_0, the ant simply chooses the best path according to pheromone and η. If q is greater than or equal to q_0, the ant will choose its next city according to the probability equation:

$$p_k(r,s) = \begin{cases} \frac{[\tau(r,s)]^{\mu}[\eta(r,s)]^{\beta}}{\sum_{u \in J_k(r)} [\tau(r,u)]^{\mu}[\eta(r,u)]^{\beta}} & \text{if } s \in J_k(r) \\ 0 & \text{otherwise} \end{cases} \tag{1.5}$$

Where $P_k(r,s)$ is the chance of city s to be chosen by ant k positioned in city r. Since the pheromone on the edge is multiplied by η, which depends on δ, better odds are given to shorter edges with more pheromone.

After each ant relocation step, the pheromone on all edges is updated according to a local pheromone-updating rule given by:

$$\tau(r,s) = ((1-\rho) \times \tau(r,s) + \rho \times \Delta\tau(r,s)) \tag{1.6}$$

Where ρ is a parameter defined in the range $(0 < \rho < 1)$, describes the local evaporation of pheromone, and $\Delta\tau(r,s)$ is the sum of all pheromone left by ants that used this edge in their last step. Normally ρ is fixed during the path search. An ant using edge (r,s), normally leaves $1/\delta(r,s)$ pheromone on the edge.

Once all ants have completed a tour, a global pheromone-updating rule is applied. The global updating rule is described by the equation:

$$\tau(r,s) = ((1-\alpha) \times \tau(r,s) + \alpha \times \Delta\tau(r,s) \tag{1.7}$$

Where α is the global evaporation parameter defined by the range $(0 < \alpha < 1)$.

$$\Delta\tau(r,s) = \begin{cases} \frac{1}{\text{length of global best tour}} & \text{If } (r,s) \text{ belongs to this global best tour} \\ 0 & \text{otherwise} \end{cases} \tag{1.8}$$

Equation (7) is only applied to the global best solution. Normally, α is not varied during the search. In addition to this, according to a boolean parameter, each ant may leave a global trail, adding 1/(length of their tour) to each edge belonging to its last tour.

1.3 Complexity

One important factor in considering ACS over other algorithms is its efficiency compared to those other algorithms. It is assumed that in each step, each ant has to decide which way to go. It is further assumed that n is the number of cities that may be visited by m ants for k cycles. From every city, an ant can go to any city it hasn't visited yet. This means it has to decide from among n cities. Before deciding, it has to calculate the probability of each path. This part requires n calculations. Thus for all ants combined, one step requires $m \times n$ calculations. To complete 1 cycle, one ant must go through all n cities, which means it has to complete n steps. This requires $m \times n \times n$ calculations for each cycle. Therefore the running time is $O(mn^2k)$, which is much better for big graphs than the recursive solution - $O(n!)^3$. It is obvious that there are also a constant number of calculations each ant has to perform in each step.

1.4 Demonstration of local and global optimization techniques

Local minimum $f^\star = f(x^\star)$, local minimizer $x^\star$. Smallest function value in some feasible neighbourhood. For $x^\star \in \Omega$ there exists a $\delta > 0$ such that $f^\star \leq f(x)$ for all $x \in \{x \in \Omega : |x - x^\star| \leq \delta\}$.

Global minimum $f^\star = f(x^\star)$, global minimizer $x^\star$. Smallest function value over all feasible points, for all $x \in \Omega$, $f^\star \leq f(x)$.

There can be many local minima which are not global minima. n the context combinatorial problems, global optimization is NP-hard. Special properties (eg. convexity) of feasible region Ω and objective function f imply that any local solution is a global solution.

1.5 Solving QAP using ACS

Let us consider the following real world example:

1.5.1 Example 1: QAP of order 8

The spaces available are concentrated in the three following buildings:

1. **Tower**: a building on four identical floors, each has one unit, numbered from 1 to 4 (one per floor).
2. **Building A:** a three-floor construction near to the TOWER building, with direct pedestrian connections at the level of the first two floors (as well as the outside passage) and with one unit per floor, numbered from 5 to 6.
3. **Building B:** a construction with several floors, of which the first two are available for the company in question, detached from the previous buildings and connected to them by footpaths. Here also one unit is available on each usable floor, numbered from 7 to 8. The whole is shown in Figure.

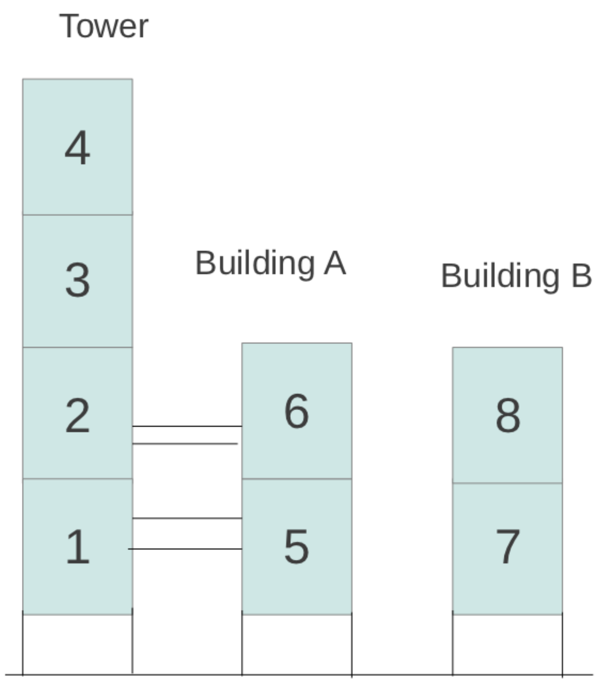


Figure 1.3: Position of the units in the three buildings available

The distance matrix is made of the times (in seconds) an employee need to move from location i to location h $(i, h = 1, 2, ..., 8)$.

For simplicity, consider the distances between the units on the various floors of each building are identical, although sometimes there are obligatory paths which may cause small differences. The distances are estimated on the basis of the conditions of normal activity of the offices themselves (waiting times for the service lifts and/or any use of alternative routes, walkways or stairs).

$$D = \begin{bmatrix} 0 & 1 & 2 & 3 & 1 & 2 & 3 & 4 \\ & 0 & 1 & 2 & 2 & 1 & 2 & 3 \\ & & 0 & 1 & 3 & 2 & 1 & 2 \\ & & & 0 & 4 & 3 & 2 & 1 \\ & & & & 0 & 1 & 2 & 3 \\ & & & & & 0 & 1 & 2 \\ & & & & & & 0 & 1 \\ & & & & & & & 0 \end{bmatrix}$$

Flow between activities is the number of personal contacts necessary on average in a week by the employees of various offices, weighted according to the qualification of the person involved (the employees were assigned weight 1 and the managers weight 2), thus trying to correlate the movements to the effective burden in terms of working costs. The matrix of the flows between the various activities was obtained by quali-quantitative indications obtained from all the managers of the various services.

$$F = \begin{bmatrix} 0 & & & & & & & \\ 5 & 0 & & & & & & \\ 2 & 3 & 0 & & & & & \\ 4 & 0 & 0 & 0 & & & & \\ 1 & 2 & 0 & 5 & 0 & & & \\ 0 & 2 & 0 & 2 & 10 & 0 & & \\ 0 & 2 & 0 & 2 & 0 & 5 & 0 & \\ 6 & 0 & 5 & 10 & 0 & 1 & 10 & 0 \end{bmatrix}$$

Please note that distance and flow matrices are symmetric therefore we need only upper and lower or lower and upper input for distance and flow matrices as input respectvely.

For simplicity we can combine distance and flow matrices together and form a new matrix distance-flow (DF) as follows:

$$DF = \begin{bmatrix} 0 & 1 & 2 & 3 & 1 & 2 & 3 & 4 \\ 5 & 0 & 1 & 2 & 2 & 1 & 2 & 3 \\ 2 & 3 & 0 & 1 & 3 & 2 & 1 & 2 \\ 4 & 0 & 0 & 0 & 4 & 3 & 2 & 1 \\ 1 & 2 & 0 & 5 & 0 & 1 & 2 & 3 \\ 0 & 2 & 0 & 2 & 10 & 0 & 1 & 2 \\ 0 & 2 & 0 & 2 & 0 & 5 & 0 & 1 \\ 6 & 0 & 5 & 10 & 0 & 1 & 10 & 0 \end{bmatrix}$$

In the distance-flow matrix upper half is represented as distance and lower half is represented as flow and $DF_{ii} = 0$, for $i = 1, 2, ..., n$ is obvious (no distance-flow from location i to i).

For ants = 5, $\mu = -1$ (weight of pheromone), $\beta = 1$ (weight of heuristic info), $\rho = 0.9$ (evaporation parameter), $Q = 10$ (Constatnt for pheromone update). These are the most effective parameter values has been found to get the best optimum value.

We get, the objective function of the permutation (8,7,4,6,5,1,3,2) corresponding to the real location of the offices in the units available 118.

Figure 4 shows a ACS solution for the QAP input matrix shown above, this is a sub optimal solution at 118 whereas the ideal solution is 107.

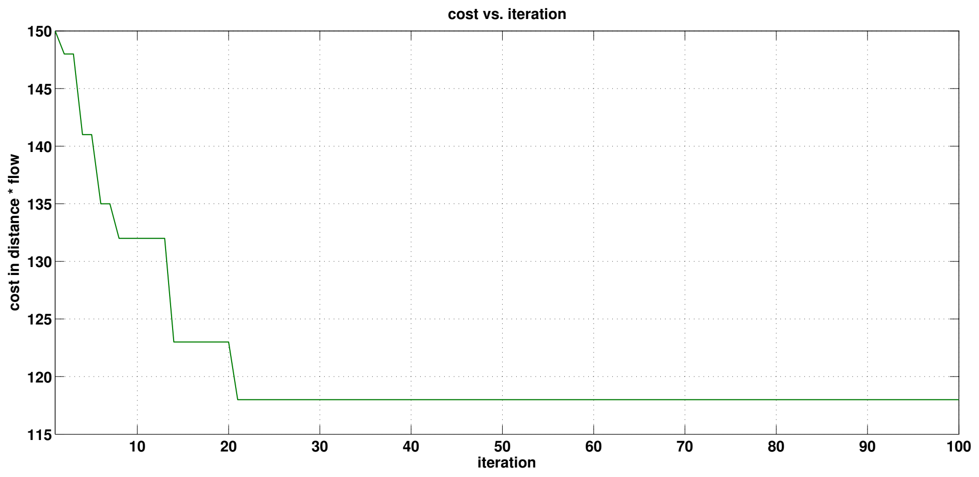


Figure 1.4: Typical Iteration to Solution

1.5.2 Demonstration of local and global optimization techniques of QAP of order 8

Here we give statistical design of global optimization experiments to obtain optimal combination of global optimization parameters that would be giving solution close to the global optimal solution.

We also describe comparison between multistart local optimization and global optimization.

Figure 5 and 6 compare Monte Carlo trials with 8 (the same number of nodes) and 4 ants (agents) with 200 trials.

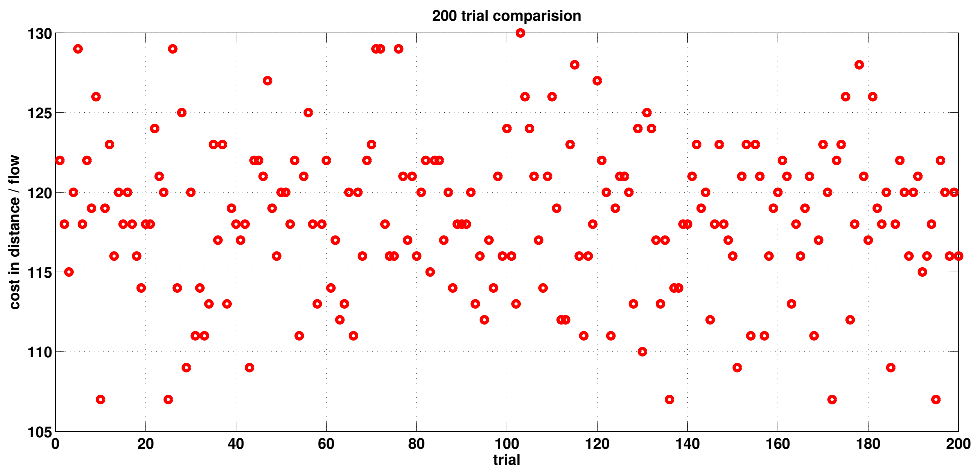


Figure 1.5: Comparison with the Number of Ants Equal to Number of Nodes (8)

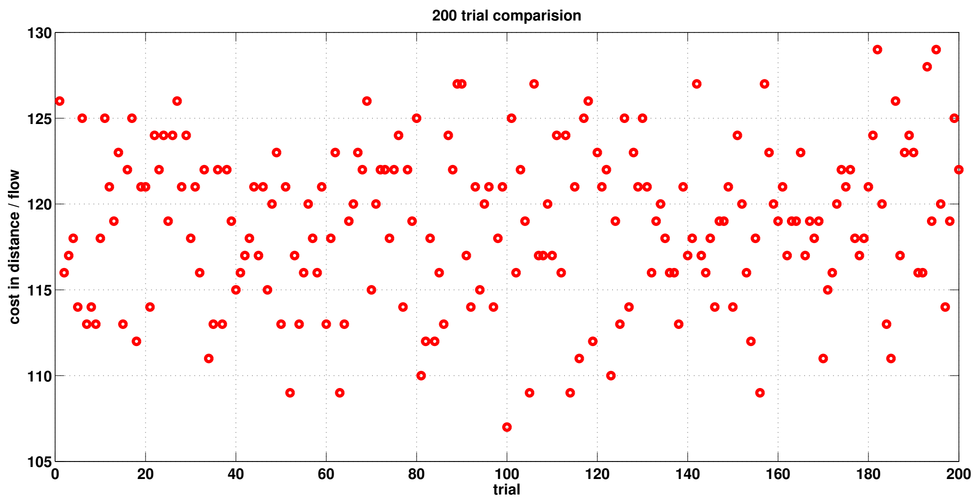


Figure 1.6: Comparison with Number of Ants Fixed at Four

1.5.3 Example 2: QAP of order 33

Let us extend the QAP constructed above with the example of Tower, Building A and Building B. Assuming all assumptions are same.
Figure 7 represents a QAP formulation of Tower, Building A and Building B with QAP of order 33 (problem length).

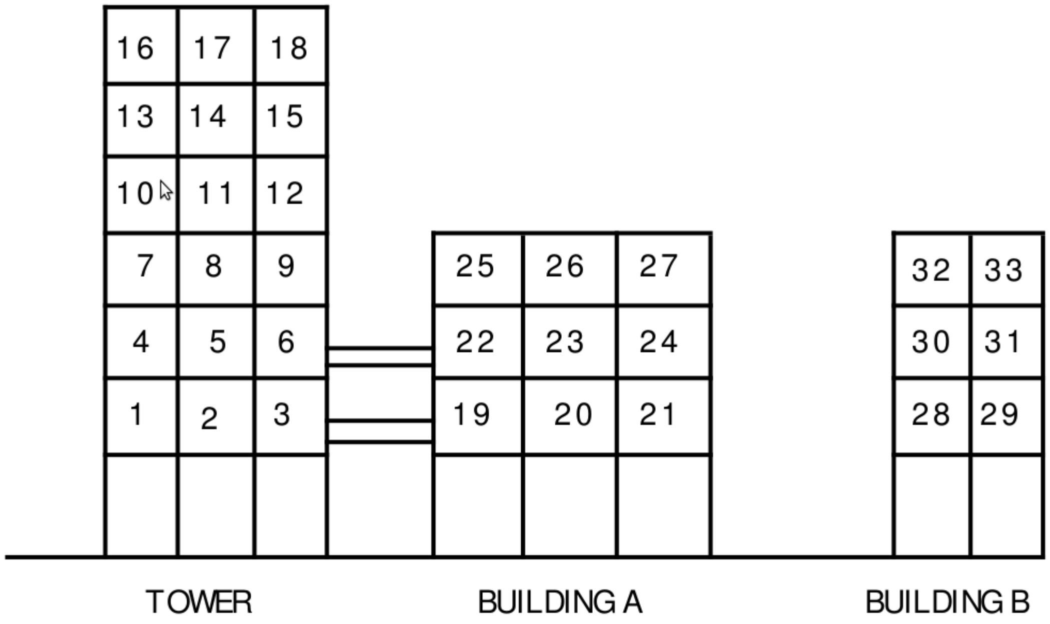


Figure 1.7: Position of the units in the three buildings available

The distance matrix is made of the times (in seconds) an employee need to move from location i to location h $(i, h = 1, ..., 33)$. The distance-flow matrix is given in apendix A.

We get, the objective function of the permutation (21, 11, 5, 30, 29, 4, 8, 25, 16, 24, 28, 17, 1, 26, 13, 32, 33, 22, 18, 7, 31, 3, 19, 6, 27, 20, 9, 2, 10, 14, 15, 23, 12,) corresponding to the real location of the offices in the units available 227696 m.sec.

Figure 8 shows a ACS solution for the QAP input matrix given in apendix,

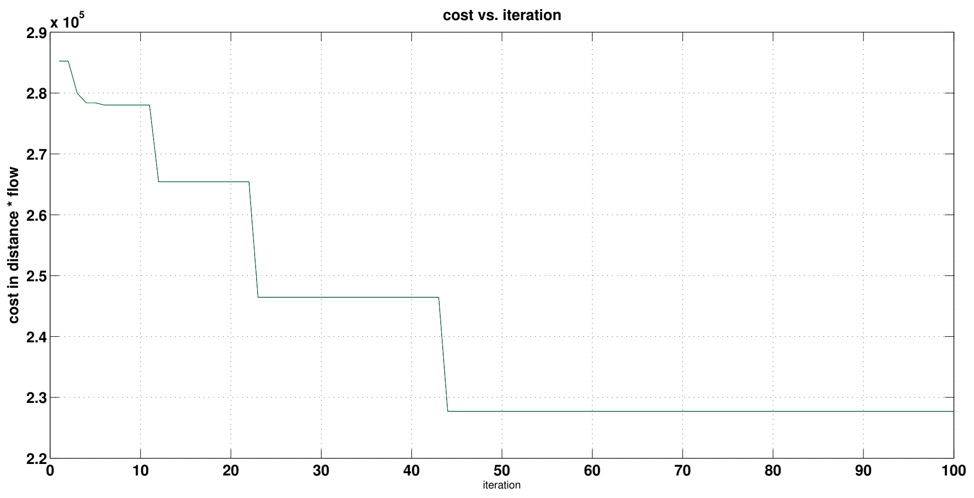


Figure 1.8: Typical Iteration to Solution

1.5.4 Demonstration of local and global optimization techniques of QAP of order 33

Here we give statistical design of global optimization experiments to obtain optimal combination of global optimization parameters that would be giving solution close to the global optimal solution.

We also describe comparison between multistart local optimization and global optimization.

Figure 9 and 10 compare Monte Carlo trials with 33 (the same number of nodes) and 10 ants (agents) with 200 trials.

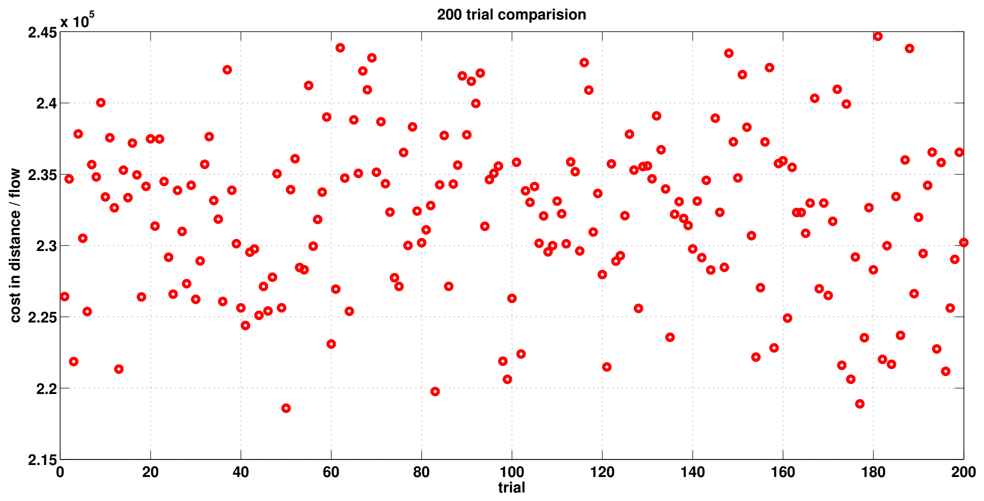


Figure 1.9: Comparison with the Number of Ants Equal to Number of Nodes (33)

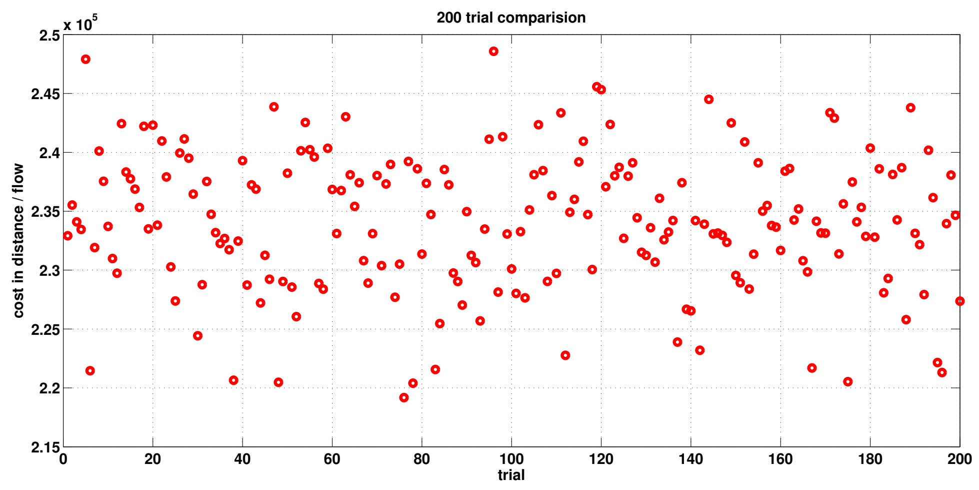


Figure 1.10: Comparison with Number of Ants Fixed at 10

1.6 Performance Analysis

Figure 11 and 12 represents CPU time tradeoff analysis of Time vs Maximum Assignment (When To Stop) and Problem Size vs Time.

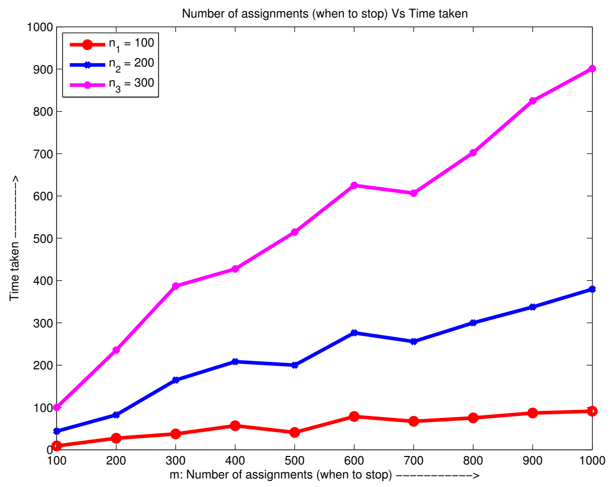


Figure 1.11: Time vs Maximum Assignment (When To Stop)

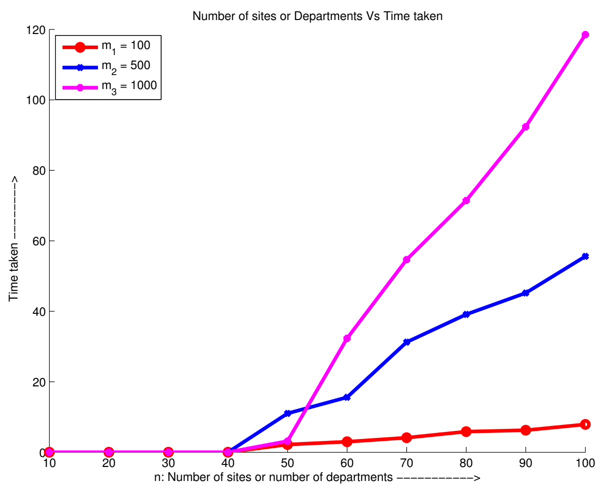


Figure 1.12: Problem Size vs Time

1.7 Applications of QAP

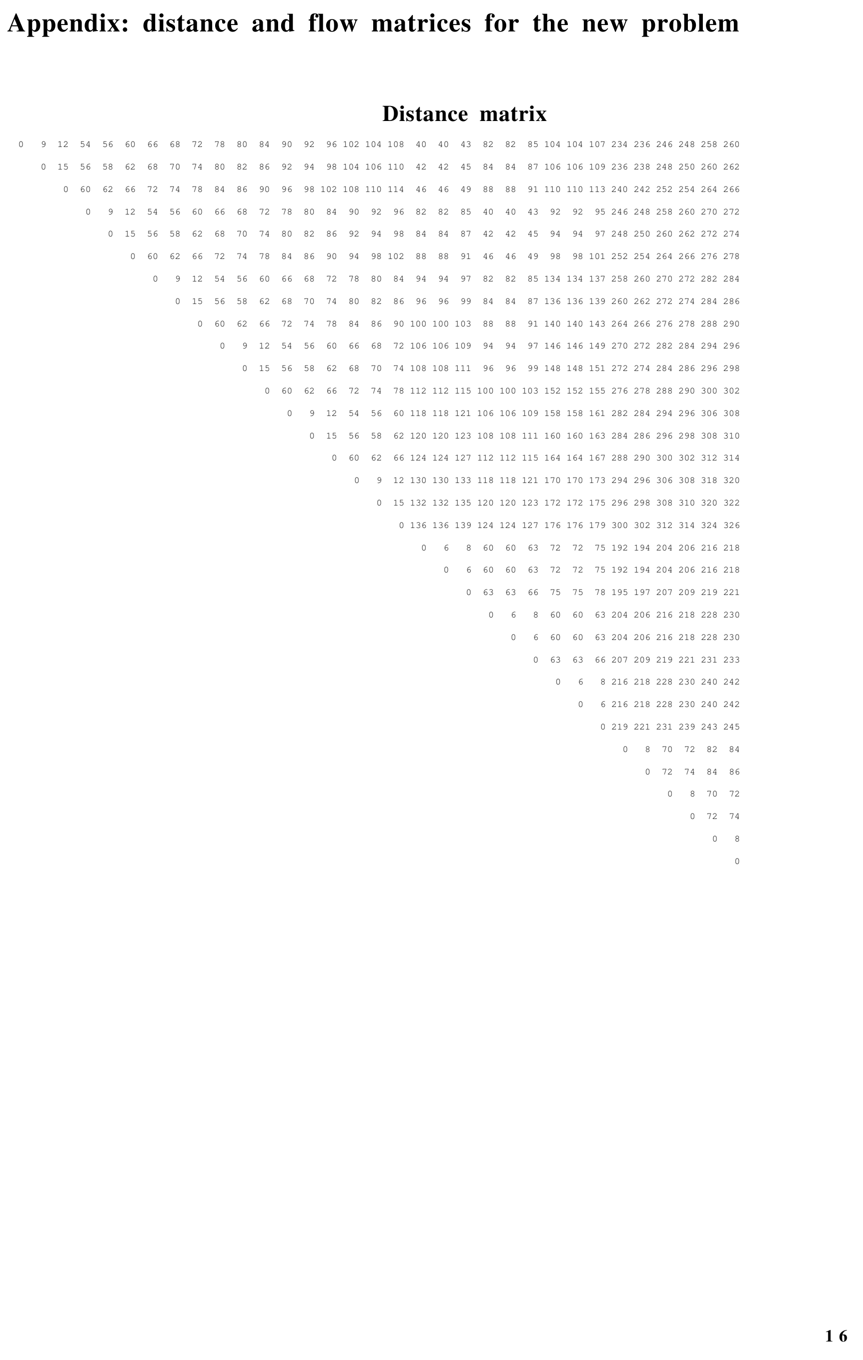
It is astonishing how many real-life applications can be modeled as QAPs. An early natural application in location theory was used in a campus planning model. The problem consists of planning the sites of n buildings on a campus, where b_{jl} is the distance from site j to site l and a_{ik} is the traffic intensity between building i and building k. The objective is to minimize the total weekly walking distance between the buildings.

Another early application in this area, where minimizing the number of connections in a backboard wiring problem, nowadays an outdated technology of historical interest only. Another way to use QAPs in hospital planning, related problem for forest parks.

In addition to facility location, QAPs appear in a variety of applications such as computer manufacturing, scheduling, process communications, and turbine balancing. In the field of ergonomics Pollatschek, Gershoni and QAPs can be applied to typewriter keyboard design. The problem is to arrange the keys on a keyboard so as to minimize the time needed to write texts. Let the set of integers $N = \{1, 2, ..., n\}$ denote the set of symbols to be arranged. Then a_{ik} denotes the frequency of the appearance of the ordered pair of symbols i and k. The entries of the distance matrix b_{jl} are the times needed to press the key in position l after pressing the key in position j . A permutation $\phi \in S_n$ describes an assignment of symbols to keys. An optimal solution $\phi^\star$ for the QAP minimizes the average time for writing a text. A similar application related to ergonomic design is the development of control boards in order to minimize eye fatigue.

The turbine runner problem. The blades of a turbine, which due to manufacturing have slightly different masses, should be welded on the turbine runner such that the center of gravity coincides with the axis of the runner. It has been shown that the minimization of the distance between the center of gravity and the axis of the runner is NP-hard, whereas the maximization can be obtained in polynomial time.

Apendix A



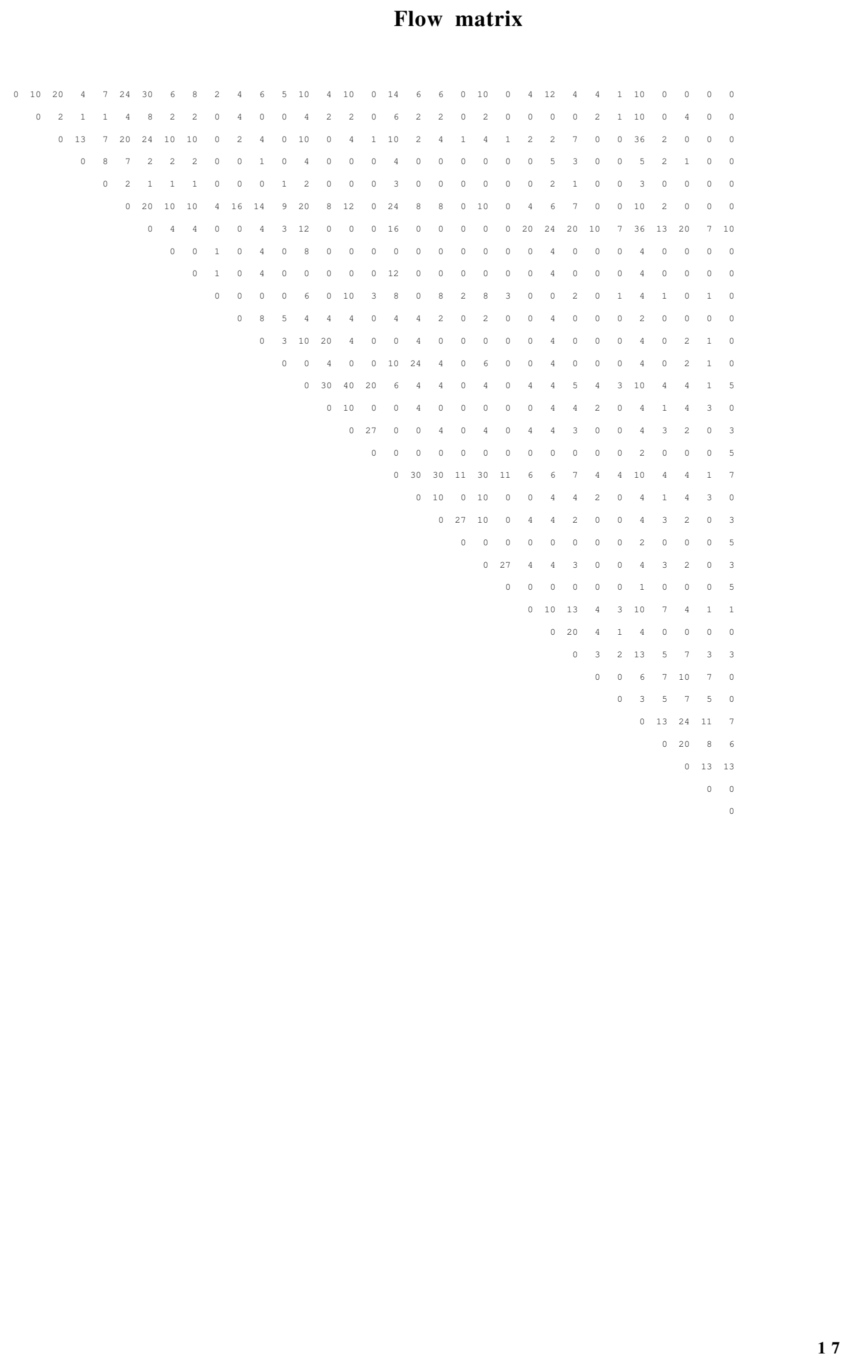
Appendix: distance and flow matrices for the new problem

Distance matrix

0	9	12	54	56	60	66	68	72	78	80	84	90	92	96	102	104	108	40	40	43	82	82	85	104	104	107	234	236	246	248	258	260
	0	15	56	58	62	68	70	74	80	82	86	92	94	98	104	106	110	42	42	45	84	84	87	106	106	109	236	238	248	250	260	262
		0	60	62	66	72	74	78	84	86	90	96	98	102	108	110	114	46	46	49	88	88	91	110	110	113	240	242	252	254	264	266
			0	9	12	54	56	60	66	68	72	78	80	84	90	92	96	82	82	85	40	40	43	92	92	95	246	248	258	260	270	272
				0	15	56	58	62	68	70	74	80	82	86	92	94	98	84	84	87	42	42	45	94	94	97	248	250	260	262	272	274
					0	60	62	66	72	74	78	84	86	90	94	98	102	88	88	91	46	46	49	98	98	101	252	254	264	266	276	278
						0	9	12	54	56	60	66	68	72	78	80	84	94	94	97	82	82	85	134	134	137	258	260	270	272	282	284
							0	15	56	58	62	68	70	74	80	82	86	96	96	99	84	84	87	136	136	139	260	262	272	274	284	286
								0	60	62	66	72	74	78	84	86	90	100	100	103	88	88	91	140	140	143	264	266	276	278	288	290
									0	9	12	54	56	60	66	68	72	106	106	109	94	94	97	146	146	149	270	272	282	284	294	296
										0	15	56	58	62	68	70	74	108	108	111	96	96	99	148	148	151	272	274	284	286	296	298
											0	60	62	66	72	74	78	112	112	115	100	100	103	152	152	155	276	278	288	290	300	302
												0	9	12	54	56	60	118	118	121	106	106	109	158	158	161	282	284	294	296	306	308
													0	15	56	58	62	120	120	123	108	108	111	160	160	163	284	286	296	298	308	310
														0	60	62	66	124	124	127	112	112	115	164	164	167	288	290	300	302	312	314
															0	9	12	130	130	133	118	118	121	170	170	173	294	296	306	308	318	320
																0	15	132	132	135	120	120	123	172	172	175	296	298	308	310	320	322
																	0	136	136	139	124	124	127	176	176	179	300	302	312	314	324	326
																		0	6	8	60	60	63	72	72	75	192	194	204	206	216	218
																			0	6	60	60	63	72	72	75	192	194	204	206	216	218
																				0	63	63	66	75	75	78	195	197	207	209	219	221
																					0	6	8	60	60	63	204	206	216	218	228	230
																						0	6	60	60	63	204	206	216	218	228	230
																							0	63	63	66	207	209	219	221	231	233
																								0	6	8	216	218	228	230	240	242
																									0	6	216	218	228	230	240	242
																										0	219	221	231	239	243	245
																											0	8	70	72	82	84
																												0	72	74	84	86
																													0	8	70	72
																														0	72	74
																															0	8
																																0

16

Figure 1.13: Distance matrix of QAP of order 33



Flow matrix

0	10	20	4	7	24	30	6	8	2	4	6	5	10	4	10	0	14	6	6	0	10	0	4	12	4	4	1	10	0	0	0	0
	0	2	1	1	4	8	2	2	0	4	0	0	4	2	2	0	6	2	2	0	2	0	0	0	0	2	1	10	0	4	0	0
		0	13	7	20	24	10	10	0	2	4	0	10	0	4	1	10	2	4	1	4	1	2	2	7	0	0	36	2	0	0	0
			0	8	7	2	2	2	0	0	1	0	4	0	0	0	4	0	0	0	0	0	0	5	3	0	0	5	2	1	0	0
				0	2	1	1	1	0	0	0	1	2	0	0	0	3	0	0	0	0	0	0	2	1	0	0	3	0	0	0	0
					0	20	10	10	4	16	14	9	20	8	12	0	24	8	8	0	10	0	4	6	7	0	0	10	2	0	0	0
						0	4	4	0	0	4	3	12	0	0	0	16	0	0	0	0	0	20	24	20	10	7	36	13	20	7	10
							0	0	1	0	4	0	8	0	0	0	0	0	0	0	0	0	0	4	0	0	0	4	0	0	0	0
								0	1	0	4	0	0	0	0	0	12	0	0	0	0	0	0	4	0	0	0	4	0	0	0	0
									0	0	0	0	6	0	10	3	8	0	8	2	8	3	0	0	2	0	1	4	1	0	1	0
										0	8	5	4	4	4	0	4	4	2	0	2	0	0	4	0	0	0	2	0	0	0	0
											0	3	10	20	4	0	0	4	0	0	0	0	0	4	0	0	0	4	0	2	1	0
												0	0	4	0	0	10	24	4	0	6	0	0	4	0	0	0	4	0	2	1	0
													0	30	40	20	6	4	4	0	4	0	4	4	5	4	3	10	4	4	1	5
														0	10	0	0	4	0	0	0	0	0	4	4	2	0	4	1	4	3	0
															0	27	0	0	4	0	4	0	4	4	3	0	0	4	3	2	0	3
																0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	5
																	0	30	30	11	30	11	6	6	7	4	4	10	4	4	1	7
																		0	10	0	10	0	0	4	4	2	0	4	1	4	3	0
																			0	27	10	0	4	4	2	0	0	4	3	2	0	3
																				0	0	0	0	0	0	0	0	2	0	0	0	5
																					0	27	4	4	3	0	0	4	3	2	0	3
																						0	0	0	0	0	0	1	0	0	0	5
																							0	10	13	4	3	10	7	4	1	1
																								0	20	4	1	4	0	0	0	0
																									0	3	2	13	5	7	3	3
																										0	0	6	7	10	7	0
																											0	3	5	7	5	0
																												0	13	24	11	7
																													0	20	8	6
																														0	13	13
																															0	0
																																0

1 7

Figure 1.14: Flow Matrix of QAP of order 33

Apendix B

Pseudo Code of our implmentation

Let us see what kind of input and corresponding output we expect from the Ant algorithm we use to solve Quadratic assignment problem. Here we assume that the distance-flow (DF) matrix must be $n \times n$ secondly upper triangular must contain distances and lower tringular must contsain flows. Most importantly the diagonal entries must be NAN, This is obvios because assigning ith department to ith department does not make any sense.

```
**On input** (n: number of sites or department, m: number of assignments when to
stop, the distance-flow matrix: DF, where $DF_{ij}, i < j$ = distance, $DF_{ij}, i > j$ =
flow and $DF_{ii} = NaN$. )
**Result**: Cheapest cost, number of assignments, assignment to each department i.e.
         ith department to site $x^\star$
```

Algorithm 1: Input-Output objective for Ant algorithm

Next we describe and initialize the parameters for the Ant algorithm the we are implenting here to solve the quadratic assignment problem. For example number of ants, an optimal guessed solution, weight of pheromone, weight of heuristic information, evaporation parameter, constant for pheromone updating etc need to be initialized befor we begin with final algorithm.

```
Initialization:
Number of Ants, $ants = NumAnts$.
Number of assignments (when to stop), $MaxAssigns = m$
Optimal solution, $optimal = c^\star$
Weight of pheromone, $a = -1$
Weight of heuristic information, $b = 1$
Evaporation parameter, $\lambda = 0.9$
Constatnt for pheromone updating, $Q = 10$
Assignment of each Ant, $AM_{ants \times n}$, where $AM_{ij} = 1$
Minimum cost, $MinCost = -1$
Heuristic information - sum of distance between sites,
for $i$ = 1,2,...,n do
    $D_i = \sum_{j=1}^{i-1} DF_{ji} + \sum_{j=i+1}^{n} DF_{ij}$
end
```

Algorithm 2: Initialization for Ant algorithm

At each iteration $t^\star$ the Ant algorithm for quadratic assignment problem runs in two phases, phase 1 and phase 2. In phase 1, the algorithm finds pheromone and then algorithm enters in phase 2, which is very critical and most important part of this algorithm. In phase 2, algorithm assigns departments to sites by making assignments to each ant.

```
Start the Algorithm:
$assign = 1$.
repeat
    Stopping criterion: // Algorithm stopes when at iteration $t^\star$, it satisfy the following:
    $assign > MaxAssigns$ and ( $MinCost \leq optimal$ or $MinCost \neq -1$ )
    Phase I: Finding pheromone
    //At first loop of iteration, initialize pheromone
    Phase II: Assigning departments to sites
    //Making assignments to each Ant
        //Get random department order
        For each department index
            //Get sum of the preferences and the preferences for each site
            //Get probabilities of assigning the department to each free site
            //Get the site where the department will be assigned
            //Eliminate the selected site from the free sites
        //Get the cost of the Ant's assignment
until $assign \leq MaxAssigns$ and ( $MinCost > optimal$ or $MinCost = -1$ );
```

Algorithm 3: The ANT Algorithm for the Quadratic Assignment Problem

Next algorithm 4 and algorithm 5 contains step by step procedure done in phase 1 and in phase 2 respectvely at each iteration $t^\star$. Please note that at each iteration $t^\star$, the phase 1 and phase 2 is compalsory to run in order to get the optimal solution to QAP. Let at the end of iteration $t^\star$ the stopping crtiterion is satisfied and the algorithm terminates, then the algorithm guarantee the optimality of solution to given input instances for QAP.

```
Phase I: Finding pheromone
//At first loop of iteration, initialize pheromone
if $assign = 1$ then
    Set 1 as initial pheromone, $pher_{ij} = 1$, for $i, j = 1, 2, ..., n$
else
    $\delta_{ij} = \sum \frac{Q}{cost}$, $pher = \lambda \times pher + \delta$
end
```

Algorithm 4: Phase 1: The ANT Algorithm for the Quadratic Assignment Problem

```
Phase II: Assigning departments to sites
//Making assignments to each Ant
for $ant$ = $1,2,...,ants$ do
    Get random department order, $depts_{n\times 2}$, $depts_{i1} = i$ and sort it
    Keep available sites in a vector, $FreeSites_i = i$
    Preference for each site, $pref_i = 1$, for $i = 1, 2, ..., n$.
    Probabilities for each department, $prob_i = 1$, for $i = 1, 2, ..., n$
    for $DeptIndex = 1, 2, ..., n$ do
        $CurDept = depts_{DeptIndex}$
        //Get sum of the preferences and the preferences for each site
        for $SiteIndex = 1, 2, ...size(FreeSites, 2)$ do
            $pref_{SiteIndex} = (FreeSites_{SiteIndex})^a \times (\frac{1}{D_{FreeSites_{SiteIndex}}})^b$
            $PrefSum = PrefSum + pref_{SiteIndex}$
        end
        //Get probabilities of assigning the department to each free
        site
        $prob = FreeSites$, $prob_{i,2} = \frac{pref}{PrefSum}$, for $i = 1, 2, ..., n$
        //Get the site where the department will be assigned
        Sort rows of prob in ascending order
        $SelectedSite = prob_{size(prob,1)}$, $AM_{ant,CurDept} = SelectedSite$
        //Eliminate the selected site from the free sites
        Find $index$, where $FreeSites = SelectedSite$
        $prob_{1j} = NULL$, $FreeSites_{index} = NULL$, $pref_{index} = NULL$
    end
    //Get the cost of the Ant's assignment
    for $i$ = $1,2,...,n$ do
        for $j$=$1...i$-$1$ do
            $DeptFlow = DF_{ij}$, $site_1 = AM_{ant,i}$, $site_2 = AM_{ant,j}$
            if $site_1 < site_2$ then
                $SitesDistance = DF_{site_1,site_2}$
            else
                $SitesDistance = DF_{site_2,site_2}$
            end
            $costs_{ant} = costs_{ant} + DeptFlow \times SitesDistance$
        end
    end
    if $costs_{ant} < MinCost$ $or$ $MinCost = -1$ then
        $MinCost = costs_{ant}$, $ChAssign = AM_{ant}$
    end
    $assign = assign + 1$
end
```

Algorithm 5: Phase 2: ANT Algorithm for the Quadratic Assignment Problem