CIRRELT

Centre interuniversitaire de recherche
sur les réseaux d'entreprise, la logistique et le transport

Interuniversity Research Centre
on Enterprise Networks, Logistics and Transportation

Iterated Great Deluge for the Dynamic Facility Layout Problem

Nabil Nahas
Mustapha Nourelfath
Daoud Aït-Kadi

May 2010

CIRRELT-2010-20

Bureaux de Montréal :
Université de Montréal
C.P. 6128, succ. Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

Bureaux de Québec :
Université Laval
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

Iterated Great Deluge for the Dynamic Facility Layout Problem

Nabil Nahas*, Mustapha Nourelfath, Daoud Aït-Kadi

Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT) and Mechanical Engineering Department, Pavillon Adrien-Pouliot, Université Laval, Québec, Canada G1K 7P4

Abstract This paper introduces an iterated great deluge (IGD) heuristic for the dynamic facility layout problem (DFLP). This problem involves the arrangement of manufacturing facilities over time to minimize the sum of the material handling and rearrangement costs. The IGD heuristic combines a great deluge algorithm with a perturbation operator that helps escape from local optima. Our implantation of the IGD heuristic relies on two main steps: (i) we first generate a local optimum solution by running an extended great deluge (EGD) algorithm for N_1 iterations; (ii) the second step consists in a loop that perturbs the best current solution, and restarts an improvement by running the EGD algorithm for $N_2 < N_1$ iterations. Numerical results for 48 test problems from previous research are reported and compared. The solutions found by our approach are very competitive and a great set of the obtained results are better than or are in part with the well-known best solutions. New solutions (i.e., solutions better than the best solutions existing until now in the DFLP literature) are obtained in 17 problems. The newly developed IGD heuristic is discussed in the context of the general iterated local search paradigm.

Keywords. Dynamic facility layout problem, iterated local search, extended great deluge.

---

* Corresponding author: nahas.nabil@gmail.com

Dépôt légal – Bibliothèque et Archives nationales du Québec,
Bibliothèque et Archives Canada, 2010

1. Introduction

In the static facility layout problem (SFLP), the objective is to determine an optimal arrangement of space consuming activities or departments in a facility. The objective is to minimize the total cost of transferring materials between these departments. It is well known that the SFLP can be modeled as a quadratic assignment problem (QAP) by making space availability and requirements as discrete (Koopmans and Beckman, 1957). Exact methods for solving the QAP model of the discrete SFLP include branch and bound (Lawler, 1963) and cutting plane algorithms (Bazara and Sherali, 1980; Burkard and Bonniger, 1983). In practice these approaches can solve only moderately sized problem instances, because they require high computational time for large size problem instances. The QAP is known to be NP-hard (Shani and Gonzalez, 1976). While some NP-hard combinatorial optimization problems can be solved exactly for relatively large instances, QAP instances of sizes larger than 30 are considered intractable (Anstreicher et al., 2002). Hence, the use of heuristic methods for solving large SFLP instances is currently the only practicable solution in an industrial context. Although not ensuring that the solution found is the best one, heuristic algorithms give good results in an acceptable computation time. The most notable existing heuristic techniques for the discrete SFLP include H63, HC63-66 and CRAFT (Nugent et al., 1968), simulated annealing (SA) (Burkard and Rendl, 1984; Wilhelm and Ward, 1987; Connolly, 1990; Herargu and Alfa, 1992), genetic algorithms (GA) (Chan and Tansri, 1994; Tate and Smith, 1995), tabu search (Skorin-Kapov, 1990; Taillard, 1991; Battiti and Tecchiolli, 1994; Sondergeld and Voß, 1996), hybrid genetic-tabu search (Fleurent and Ferland, 1994), the scatter search (Cung et al., 1997), and ant colony optimization (ACO) (Bland, 1999a; Bland, 1999b; Gambardella et al., 1999; Maniezzo and Colorni, 1999; Solimanpur et al., 2003; Montreuil et al., 2004; McKendall and Shang, 2006; Nourelfath et al., 2007). A recent survey of meta-heuristic solution methods the SFLP modelled as a QAP can be found in Nehi and Gelareh (2007).

In the SFLP, it is assumed that all the activities are constant. However, in today's volatile markets, the business conditions are changing. So, the similar changes are

imposed on the facility projects and the flows of materials between activities can change during a planning horizon. In this case, the facility layout problem becomes dynamic. The dynamic facility layout problem (DFLP) involves the design of facility layouts based on a multi-period planning horizon. The objective of the DFLP is to obtain layouts for each period in the planning horizon such that the sum of the rearrangement and material handling flow costs is minimized (Driscol and Sawyer, 1985; Rosenblatt, 1986; Lacksonen and Enscore, 1993). The DFLP is more recent than the SFLP. As the DFLP extends the SPLP by assuming that the material handling flows can change over time, it might in turn necessitate layout rearrangement during the planning horizon. It is however important to note that the DFLP is not just a series of SFLP (McKendall et al., 2006), and more sophisticated approaches are needed to solve it. The DFLP was first introduced in details by Rosenblatt (1986) who proposed two solution methods both based on dynamic programming. Like the SFLP, the DFLP is computationally intractable and only small problems can be solved to optimality in a reasonable computation time. Therefore, most existing solution approaches are based on heuristics. In (Lacksonen and Enscore, 1993), the DFLP is modeled as a modified quadratic assignment problem (QAP) and the authors modified and adapted five existing algorithms to solve it. They conclude that a cutting plane algorithm found the best solutions. Urban (1993) proposed a heuristic algorithm based on the CRAFT procedure. Conway and Venkataraman (1994) solved the DFLP by using a GA. A good survey about the DFLP has been published in Balakrishnan and Cheng (1998). More recently, Balakrishnan and Cheng (2000) have also applied an improved genetic algorithm version. Baykasoglu and Gindy (2001) proposed an algorithm based on the simulated annealing approach for the DFLP, while in reference (Balakrishnan et al., 2003), the authors developed a hybrid genetic algorithm. In reference (Erel et al., 2003), the authors proposed also a heuristic scheme. More recently, McKendall et al. (2006) developed two SA heuristics. The first one is a direct adaptation of SA, and the second heuristic is the same as the first one but with a look-ahead/look-back strategy. A hybrid ant system and tabu search heuristics have also been proposed in McKendall and Shang (2006) and McKendall (2008), respectively. More recently, Balakrishnan and Cheng (2009) investigated whether using a rolling horizon planning would make a difference in the relative effectiveness of some heuristics for the DFLP,

and the effect of forecast error on this effectiveness. An extended discrete particle swarm optimization algorithm was proposed for the DFLP by Rezazadeh et al. (2009).

In this article, a modified QAP formulation of the discrete DFLP is adopted and the problem is solved by developing an iterated great deluge (IGD) heuristic approach. This heuristic can be seen as a special case of the general iterated local search (ILS) paradigm described by (Lourenço et al., 2002), where many example of successful applications are reported. In Misevicius (2004) and Misevicius et al. (2006), iterative tabu search has been proposed in as an improvement of standard tabu search. More recently, (Cordeau et al., 2008) introduced for the daily car sequencing problem, an iterated tabu search that incorporates a classical tabu search as a subcomponent of ILS. The local search used in our ILS heuristic is the extended great deluge (EGD) algorithm introduced in (Burke et al., 2004). The EGD is an extension of the great deluge (GD) algorithm proposed in (Dueck, 1993) as a variant of simulated annealing. The GD algorithm and its extension have the advantage to require the tuning of only one input parameter that can represent the search time.

There are many papers in previous literature dealing with concepts similar to the ideas that support our proposed heuristic. In fact, even if the concept behind our IGD heuristic is currently known in the literature as ILS paradigm (Lourenço et al., 2002), to the best of our current knowledge, the early origins of this concept go back to 1986 (Baum, 1986), where the author proposes a hill climbing attachment called *iterated descent* and shows that it is useful in conjunction with any local search algorithm, including neural net algorithms. Various modifications of this basic idea have been independently developed since that time. ILS is one of these modifications. Earlier papers containing the idea of ILS are (Martin et al., 1991; Stützle, 1998; Stützle, 1999; Lourenço et al., 2001). Another variant corresponds to the "ruin and recreate" principle presented in Schrimpf et al. (2000), where solutions are partly, but significantly, ruined and rebuilt or recreated afterwards. The authors show that by performing this kind of change frequently, it is possible to escape from local optima. As the design problem under study is of strategic nature, its financial impact is very high. This importance of the DFLP has been pointed by many authors.

The remainder of this paper is organized as follows. In section 2, the mathematical formulation of the problem is given. In section 3, the proposed iterated great deluge heuristic is presented. In section 4, the proposed approach is tested on 48 instances from previous literature. Conclusions are drawn in section 5.

2. Problem formulation

The mathematical formulation of the discrete representation of the DFLP (modified QAP) is presented in this section. This kind of formulation is used for example in McKendall et al. (2006). It was introduced in Balakrishnan et al. (1992) under a form constrained by a budget limitation on the total amount of funds used for layout rearrangement.

Notations

N number of departments

T number of periods

A_{tijl} cost of shifting department i from location j to l in period t

C_{tijkl} cost of material flow between department i located at location j and k located at l in period t

Decision variables

$$X_{tij} = \begin{cases} 1 & \text{if department } i \text{ is assigned to location } j \text{ in period } t, \\ 0 & \text{otherwise.} \end{cases}$$

$$Y_{tijl} = \begin{cases} 1 & \text{if department } i \text{ is shifted from location } j \text{ to } l \text{ at the beginning of period } t, \\ 0 & \text{otherwise.} \end{cases}$$

The DFLP can be formulated as follows:

$$\text{Minimize} \qquad Z = \sum_{t=2}^{T}\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{l=1}^{N} A_{tijl}Y_{tijl} + \sum_{t=1}^{T}\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{N}\sum_{l=1}^{N} C_{tijkl}X_{tij}X_{tkl} \tag{1}$$

$$\text{Subject to} \qquad \sum_{j=1}^{N} X_{tij} = 1 \qquad i = 1,...,N;\ t = 1,...,T \tag{2}$$

$$\sum_{i=1}^{N} X_{tij} = 1 \qquad j = 1,...,N;\ t = 1,...,T \tag{3}$$

$$Y_{tijl} = X_{(t-1)ij}X_{til} \qquad i, j, l = 1,...,N;\ t = 2,...,T \tag{4}$$

$$X_{tij} = \{0,1\} \qquad i, j = 1,...,N;\ t = 1,...,T \tag{5}$$

$$Y_{tijl} = \{0,1\} \qquad i, j, l = 1,...,N;\ t = 2,...,T \tag{6}$$

The objective function (1) minimizes the sum of the cost of layout arrangement and the cost of material handling flow between departments during the planning horizon. Constraint set (2) ensures that each location is assigned to exactly one department in every period. Constraint set (3) requires every location to have a department assigned to it at each period. Constraint set (4) adds the rearrangement costs to the material handling flow cost if a department is shifted between locations in consecutive periods. Constraints (5) and (6) give the restrictions on the decision variables.

In each period, the number of possible layout combinations is $N!$. Thus, the total number of possible evaluations for the DFLP is given by $N!^{T}$. For example, in a problem with only N=10 departments and T=5 periods, there are more than 6.29×10^{32} possible configurations. Therefore, heuristics have to be developed to obtain solve efficiently the DFLP.

3. Iterated great deluge heuristic

3.1. The general algorithm

The proposed iterated great deluge (IGD) consists of two main steps. The objective of the first step is to find a local optimum solution $\hat{s}$ by running an extended great deluge (EGD) algorithm during N_1 iterations (this algorithm is called EGD1). The second step is a loop that allows the search process to alternate between diversification and intensification as follows. The starting solution for the process of this step is the solution $\hat{s}$ found in the first step. In each cycle, a diversification step is first performed by perturbing $\hat{s}$ to obtain a new solution s'. Intensification is then performed around s' by applying the EGD algorithm during a number of iterations $N_2 < N_1$, to produce a new solution $\tilde{s}$ (this algorithm is called EGD2). If $\tilde{s}$ is better than $\hat{s}$, then $\tilde{s}$ replaces $\hat{s}$ and the next perturbation is performed from that solution. Otherwise, the search returns to the previous solution $\hat{s}$. The perturbation is aimed at escaping from local optima and exploring other parts of the search space. The pseudo-code in Fig. 1 summarizes the steps of our proposed IGD.

1. $s_0 \leftarrow$ initial solution;

2. Apply EGD1 on s_0 to obtain a local optimum solution $\hat{s}$;

3. While the termination criterion is not satisfied
   (a) Apply a perturbation on $\hat{s}$ to obtain s',
   (b) Apply EGD2 on s' to obtain a local optimum solution $\tilde{s}$,
   (c) If $\tilde{s}$ is better than $\hat{s}$, $\hat{s} \leftarrow \tilde{s}$;

4. Return the best solution found s^*.

Fig. 1 Iterated great deluge algorithm

In our IGD implementation for the unconstrained DFLP considered in this paper, the initial solution for EGD1 is randomly generated. The algorithms EGD1, EGD2, the perturbation mechanism and the termination criterion will be specified later (in

subsections 3.3 and 3.4). Before this, we first recall the main characteristics of the basic great deluge algorithm and its extension.

3.2. The great deluge algorithm and its extension

The great deluge (GD) algorithm is a local search technique introduced in (Dueck, 1993). This algorithm resembles in its structure the simulated annealing (SA) method. Like other local search methods, SA and GD iteratively repeat the replacement of a current solution by a new one, until some stopping condition has been satisfied. The new solution is selected from a neighborhood. Like SA, GD may accept worse candidate solutions (than the current one) during its run. The essential difference between SA and GD consists of the different acceptance rules. While SA accepts worse solutions only with certain probabilities, GD accepts a worse solution if its cost is less than or equal (for minimization problems) to some given upper limit L. In (Dueck, 1993), this upper limit was called a "water-level" and the algorithm was presented for maximization. At the beginning, the value of L is equal to the cost of the initial solution. At each iteration, it is monotonically decreased by the decay rate ΔL whose value is the only input parameter for this technique. The GD algorithm has the advantage that it depends only on this single parameter. In (Dueck, 1993), the rate ΔL is called the "rain speed". If ΔL is high, the algorithm is very fast and produces poor quality results. If ΔL is chosen to be very small, the algorithm is run for a long computation time in order to produce high quality results (Dueck, 1993).

In Burke et al. (2004), the basic GD algorithm is extended by integrating the acceptance of all better moves (the hill-climbing rule). Fig. 2 shows the pseudo-code of the extended great deluge algorithm (EGD) as presented in Burke et al. (2004).

The EGD algorithm has been successfully applied to the exam timetabling problem in (Burke *et al*., 2004), to the buffer allocation problem in (Nahas *et al.*, 2006) and to the redundancy allocation problem in (Nahas *et al.*, 2007a). It has also been coupled with ACO to solve efficiently the discrete SFLP in (Nourelfath *et al.*, 2007). In the next subsection, the EGD algorithm is applied to the DFLP as a local search within our IGD.

1. Set the initial solution s;

2. Calculate initial cost function solution $f(s)$;

3. Initial level $L = f(s)$;

4. Specify input parameter ΔL;

5. While the termination criterion is not satisfied
   (a) Define neighborhood $N(s)$,
   (b) Randomly select the candidate solution $s^* \in N(s)$,
   (c) Calculate $f(s^*)$,
   (d) If ($f(s^*) \leq f(s)$) or ($f(s^*) \leq L$) then accept s^*,
   (e) Lower the level $L = L - \Delta L$;

6. Return the best solution found s^*.

Fig. 2 The extended great deluge algorithm (Burke et al., 2004)

3.3 Elements of algorithms EGD1 and EGD2

3.3.1 Particularities

Unlike the EGD of Burke et al. (2004) given by Fig. 1, in our implementation the parameter L is decreased only if the neighbor solution is accepted. This may allow for more exploration of the search space. In fact, the parameter L will be decreased less often than in Burke et al. (2004) and a higher L may lead to a longer computation time. Preliminary tests have shown that this turned to be more effective for the DFLP instances studied in this paper.

The difference between the algorithms EGD1 and EGD2 lies only in the number of iterations each algorithm is run: $N_2 < N_1$. This means that the parameter ΔL used in EGD1 is smaller than that of EGD2. Therefore, the execution time of EGD1 is larger that that of EGD2.

3.3.2. Solution neighbourhood

A solution or configuration s is represented by a matrix where each row represents a period and each column corresponds to a location. At each iteration, the local transformations (or neighbourhood moves), that can be applied to the current solution s, define a set of neighbouring solutions as: $N(s)$ = {new configuration obtained by applying a single move to the current solution s}. The following swapping procedure (Baykasoglu and Gindy, 2001) is used to define the neighbourhood $N(s)$:

1. Select a period (row) randomly.
2. Select two different locations (columns) randomly from the period selected and swap the facilities in these locations.

The swapping procedure described above always generates feasible configurations and it is easily implementable.

3.3.3. Termination criteria

The algorithms EGD1 and EGD2 terminate if the maximum numbers of iterations N_1 and N_2 (respectively) are reached. The termination criterion of the main IGD loop is the number of perturbations specified by the user.

3.4. Perturbation mechanism

If the perturbation is very small, it might not allow the system to escape from the basin of attraction of the local optimum just found. Furthermore, if the perturbation is too strong, the algorithm would be similar to a random restart local search. The mechanism used to perturb the current solution $\hat{s}$ and obtain a new solution s' in step 3(a) of the iterated great deluge heuristic is the same as in the main algorithm: selecting randomly a period and two corresponding locations, and swapping the facilities in these locations.

4. Computational experiments

The proposed IGD algorithm was coded in Matlab, and all experiments were performed on a 2 GHz Dual core processor.

4.1. Test problems

The set of test problems, used to evaluate the performance of the proposed IGD approach when applied to the DFLP, were obtained from the first author of Balakrishnan and Cheng (2000). This data set consists of 48 test problems that contain problems with 6, 15, and 30 departments for 5 and 10 periods.

4.2. Parameter settings

The user-specified parameters of our IGD algorithm are: the rate ΔL, N_1, N_2 and the number of perturbations. The values of N_1 and N_2 depend on the size of the problem instance. Obviously, when solving a problem with higher T and N, more computational effort is needed for EGD1 and EGD2, and the numbers of iterations N_1 and N_2 should be larger. Note also that the value of ΔL in EGD2 is higher than for EGD1, and N_2 is then higher than N_1.

Preliminary tests were used to set the values of these parameters. For each set of 8 problems (*i.e.* for each pair of values of N and T), we consider the data of the first instance (*i.e.* P01, P09, P17, P25, P33 and P41) to calibrate the parameters. Once the values of the parameters are set for these data, they are used for the others variations of the problem instances. This means for example that after tuning the parameters for P01 or P41 we show that the procedure is able to "extrapolate" and work well on problems (P02 to P08, or P42 to P48) that it has not "seen" when the input parameters are being calibrated. In this way, we avoid any parameter over-fitting. The parameters' values are:

- for EGD1: $\Delta L = 0.5$ and $N_1 = 35\,000 \times N \times T$;
- for EGD2: $\Delta L = 10$ and $N_2 = 1\,000 \times N \times T$;
- the number of perturbations is 50.

4.3. Comparing the best solutions of IGD and existing methods for the DFLP

The performance of the IGD heuristic is compared with the best-known heuristics for the DFLP from previous literature and that perform well on this data set. Tables 1-3 present

the best and the average results obtained by the IGD heuristic, as well as the best results obtained by the following approaches:

- the simulated annealing heuristics (SA I and SA II) in McKendall et al. (2006);
- the dynamic programming approach (DP) in Erel et al. (2003);
- the simulated annealing heuristic (SA_EG) presented in Baykasoglu and Gindy (2001) and used in Erel et al. (2003). Note that: (*i*) the results presented in Baykasoglu and Gindy (2001) have been corrected in Baykasoglu and Gindy (2004) and are not very competitive; (*ii*) the simulated annealing heuristic of Baykasoglu and Gindy (2001) when used in Erel et al. (2003) provides more competitive results. Therefore, the results given under the SA_EG column are those found by using the heuristic developed in Erel et al. (2003). The details about these results are presented in the following website: http://www.bilkent.edu.tr/~erel/dynlayout.html;
- the hybrid genetic algorithm (GA) in Balakrishnan et al. (2003); and
- the hybrid ant systems (HAS) in McKendall and Shang (2006).

Using different random seeds for each problem instance, ten runs of our IGD algorithm were performed and the best solution is selected. In the last columns of Tables 1-3, the percent deviation the best solution obtained from the proposed heuristics is below the best solution obtained from the SA I, SA II, GA, HAS, DP, or SA_EG heuristics is given, for each test problem, under "% Dev". The best objective function value obtained for each test problem is indicated by the bold numbers. Tables 1-3 show that:

(1) New best solutions are found for 17 test problems (P22, P23, P25, P28, P32, P33, P35-P38, and P40-P46). These solutions are better than the best solutions existing until now in the DFLP literature. In Appendix, we present for each problem instance, the new cost value and the corresponding allocation of departments in each period.

(2) In 19 of the 48 test cases (P01- P18 and P24), the solutions found by our algorithm are as good as those found by the existing approaches.

(3) In 12 of the 48 test problems, the results obtained by the proposed algorithm are very close to those found by the existing algorithms.

Therefore, we can conclude that the proposed IGD heuristic performed better than all of the other heuristics for this data set in term of the best obtained solutions. While these differences are not extreme, and there are many other factors to consider in choosing the best system design, it is advantageous to employ a search method that performs well over different problem sizes and parameters. Moreover, any improvement that can be attained while adhering to the design constraints is of some benefit, even if the percentage of cost improvement realized is relatively small. In fact, because the total cost can be very high, even a small percentage can represent a high cost reduction.

The standard deviation over ten runs of our IGD algorithm was verified to be very low for all the test instances. The average objective function values given in Table 3 for each instance are indeed very close to the best solutions. This implies that the proposed method is robust. The low standard deviation of IGD can be interpreted as a sign of insensitivity to the initial solution and the random number seeds.

Table 1. Comparison results for problems with 6 departments

Problem size		Pb #	SA I	SA II	DP	SA_EG	GA	HAS	IGD		% Dev
N	*T*		Best	Best	Best	Best	Best	Best	Best	Average	
		P01	**106419**	**106419**	**106419**	**106419**	**106419**	**106419**	**106419**	106419	**0**
		P02	**104834**	**104834**	**104834**	**104834**	**104834**	**104834**	**104834**	104834	**0**
		P03	**104320**	**104320**	**104320**	**104320**	**104320**	**104320**	**104320**	104320	**0**
		P04	**106399**	**106399**	106399	**106399**	106515	**106399**	**106399**	106399	**0**
6	5	P05	**105628**	**105628**	**105628**	**105628**	**105628**	**105628**	**105628**	105628	**0**
		P06	**103985**	**103985**	**103985**	**103985**	**104053**	**103985**	**103985**	103985	**0**
		P07	**106439**	**106439**	106447	**106439**	**106439**	**106439**	**106439**	106439	**0**
		P08	**103771**	**103771**	**103771**	**103771**	**103771**	**103771**	**103771**	103771	**0**
		P09	**214313**	**214313**	**214313**	**214313**	**214313**	**214313**	**214313**	214313	**0**
		P10	**212134**	**212134**	**212134**	**212134**	**212134**	**212134**	**212134**	212796	**0**
		P11	**207987**	**207987**	**207987**	**207987**	**207987**	**207987**	**207987**	208365	**0**
		P12	**212530**	212741	212741	212747	212741	**212530**	**212530**	213135	**0**
6	10	P13	**210906**	**210906**	211022	211072	210944	**210906**	**210906**	211322	**0**
		P14	**209932**	**209932**	**209932**	**209932**	209932	**209932**	**209932**	210138	**0**
		P15	**214252**	**214252**	**214252**	214438	215452	**214252**	**214252**	214741	**0**
		P16	**212588**	**212588**	**212588**	**212588**	**212588**	**212588**	**212588**	212760	**0**

Table 2. Comparison results for problems with 15 departments

Problem size		Pb #	SA I	SA II	DP	SA_EG	GA	HAS	IGD		% Dev
N	*T*		Best	Best	Best	Best	Best	Best	Best	Average	
		P17	**480453**	480496	482123	481738	484090	**480453**	**480453**	480934	**0**
		P18	**484761**	**484761**	485702	485167	485352	**484761**	**484761**	484764	**0**
		P19	489058	488748	491310	**487886**	489898	488748	489058	489711	+0.240
		P20	**484405**	484414	486851	485862	484625	484446	484446	485396	+0.008
15	5	P21	487882	487911	491178	489,304	489885	**487722**	487822	488217	+0.002
		P22	487162	487147	489847	488,452	488640	486685	**486493**	487601	**-0.039**
		P23	487232	486779	489155	487576	489378	486853	**486268**	487038	**-0.120**
		P24	491034	**490812**	493577	493030	500779	491016	**490812**	491540	**0**
		P25	980087	979468	983070	982298	987887	980351	**978848**	980982	**- 0.063**
		P26	979369	**978065**	983826	982714	980638	978271	978304	979805	+0.024
		P27	983912	982396	988635	988465	985886	**978027**	981172	984218	+0.321
		P28	974416	972797	976456	976456	976025	974694	**971759**	973731	**- 0.106**
15	10	P29	**977188**	978067	982893	982191	982778	979196	977234	978464	+0.004
		P30	970633	**967617**	974436	973199	973912	971548	968067	970258	+0.046
		P31	979198	979114	982790	**977410**	982872	980752	978930	980545	+0.155
		P32	984927	983672	988584	988304	987789	985707	**982888**	984939	**-0.079**

Table 3. Comparison results for problems with 30 departments

Problem size		Pb #	SA I	SA II	DP	SA_EG	GA	HAS	IGD		% Dev
N	*T*		Best	Best	Best	Best	Best	Best	Best	Average	
		P33	576039	576741	579741	583081	578689	576886	**575386**	576269	**- 0.113**
		P34	568316	**568095**	570906	573965	572232	570349	569045	570337	+0.167
		P35	573739	574036	577402	577787	578527	576053	**572104**	575083	**- 0.284**
		P36	567911	566248	569596	572139	572057	566777	**564398**	566799	**- 0.326**
30	5	P37	559277	558460	561078	563503	559777	558353	**555555**	557251	**- 0.501**
		P38	566077	566597	567154	570905	566792	566762	**564124**	566779	**- 0.345**
		P39	567131	568204	568196	571499	567873	**567131**	567775	569397	+0.113
		P40	576014	573755	575273	581614	575720	575280	**572802**	574815	**- 0.166**
		P41	1164359	1163222	1171178	1174815	1169474	1166164	**1157887**	1160571	**- 0.458**
		P42	1162665	1161521	1169138	1173015	1168878	1168878	**1158243**	1161562	**- 0.282**
		P43	1157693	1156918	1165525	1166295	1166366	1166366	**1155319**	1157820	**- 0.138**
		P44	1149048	1145918	1152684	1154196	1154192	1148202	**1140395**	1145043	**- 0.481**
30	10	P45	1126432	1127136	1128136	1140116	1133561	1128855	**1123385**	1126524	**- 0.270**
		P46	1145445	1145146	1143824	1158227	1145000	1141344	**1140723**	1143428	**- 0.054**
		P47	1148083	**1140744**	1142494	1157505	1145927	1140773	1145098	1147655	+0.381
		P48	1166672	**1161437**	1167163	1177565	1168657	1166157	1162700	1166020	+0.108

4.4. Computational effort of IGD

As shown in Table 4, the computational time of the proposed algorithm depends of the problem size. From the last column of this table, we conclude that:

- For problems P01-P08 (N = 6 and T = 5), the average running times are less than one minute.
- For problems P09-P16 (N = 6 and T = 10), the highest average running time is less than 2 minutes.
- For problems P17-P24 (N = 15 and T = 5), the highest average running time is less than 7 minutes. The average run time for this set of 8 problems is 6.13 minutes.
- For problems P25-P32 (N = 15 and T = 10), the highest average running time is less than 15 minutes. The average run time for this set of 8 problems is 12.62 minutes.
- For problems P33-P40 (N = 30 and T = 5), the highest average running time is about 20 minutes. The average run time for this set of 8 problems is 18.84 minutes.
- For problems P41-P48 (N = 30 and T = 10), the highest average running time is less than 50 minutes. The average run time for this set of 8 problems is 42.2 minutes.

For the existing approaches, the considered problems were solved using different computer, programming language, operating systems, etc. Therefore, it is very difficult to compare computation times. However, the following results are given for reference. Problems P41-P48 were solved in an average of 7.8 min using SA I and SA II heuristics (on a Pentium IV 2.4 GHz PC), about 16.7 min using GA (on DEC Alpha machines), an average of 45 min using HAS (on a Pentium IV 2.4 GHz PC), an average of 18.5 h using SA_EG (on an Ultra Enterprise server operating under Solaris 7 at 250 MHz), and between 30 min and 2 h on the average for the best solutions using DP (on an Ultra Enterprise server operating under Solaris 7 at 250 MHz).

Table 4. Average running times and best objective values of our algorithm with and without step 2

Problem size		Pb #	Without step 2	With step 2	Total computational time (mn)
N	*T*				
6	5	P01	106419	-	< 1
		P02	104834		< 1
		P03	104320		< 1
		P04	106399		< 1
		P05	105628		< 1
		P06	103985		< 1
		P07	106439		< 1
		P08	103771		< 1
6	10	P09	214313	-	< 1
		P10	212134		1.5
		P11	207987		1.2
		P12	212530		1.9
		P13	210906		< 1
		P14	209932		1.1
		P15	214252		< 1
		P16	212588		< 1
15	5	P17	480453	480453	5.58
		P18	484761	484761	5.81
		P19	489059	489058	5.54
		P20	485168	484446	6.91
		P21	487896	487822	6.70
		P22	486493	486493	6.36
		P23	486592	486268	6.53
		P24	491080	490812	5.65
15	10	P25	979847	978848	12.45
		P26	978304	978304	13.46
		P27	981280	981172	14.70
		P28	971779	971759	10.32
		P29	977542	977234	10.14
		P30	968362	968067	13.40
		P31	979469	978930	13.64
		P32	983486	982888	12.82
30	5	P33	576308	575386	17.04
		P34	569550	569045	19.48
		P35	573674	572104	18.30
		P36	565413	564398	19.41
		P37	560332	555555	19.98
		P38	569653	564124	17.54
		P39	567957	567775	18.92
		P40	576548	572802	20.08
30	10	P41	1162255	1157887	39.17
		P42	1161524	1158243	38.13
		P43	1158523	1155319	40.65
		P44	1143898	1140395	45.00
		P45	1131529	1123385	45.64
		P46	1149436	1140723	49.47
		P47	1153831	1145098	43.44
		P48	1171392	1162700	36.08

4.5. Discussion

The new best solutions presented above were first published in our conference paper (Nahas et al., 2007b). In this short paper, the solution method, even slightly different from the present one, contains also the idea of perturbing a good solution and iteratively

running a local search. It was named “a two phase EGD”. In that time, we were not aware about the existence of the ILS methods. In this subsection we will illustrate the impact of the second step, and the behaviour of the IGD heuristic as a special case of the ILS paradigm.

4.5.1. Impact of the second step

Table 4 presents also the best objective values, among ten trials, with and without step 2. From this table, we remark that:

- For problems P01-P16 (N = 6), the optimal solutions are reached by using only step 1 (i.e., the algorithm EGD1).
- For problems P17-P48 (N = 15 and N = 30), the results obtained by step 1 are all improved when running step 2.

Fig. 3 sketches the cost reduction for each problem instance when running step 2. This figure shows for example that the costs of the obtained solutions are decreased by 0.7% for test instances P37 and P40.

4.5.2. IGD behaviour illustration

As an example, Figs. 4-6 depict convergence curves for the problem instance P40. While Fig. 4 shows how the first step converges to a good solution, Fig. 5 illustrates how the second step reaches the best obtained solution. In Fig. 6, for another IGD run, we draw the solutions obtained by the 50 perturbations used in step 2. We observe from this curve how a solution is perturbed and improved. That is, the desirable IGD steps that occur when solving the DFLP are as follows: the minimum cost of the solution $\hat{s}$ is perturbed, then EGD2 is applied and a new local minimum is reached. This is the heart of the ILS paradigm, of ITS and of our proposed IGD as well.

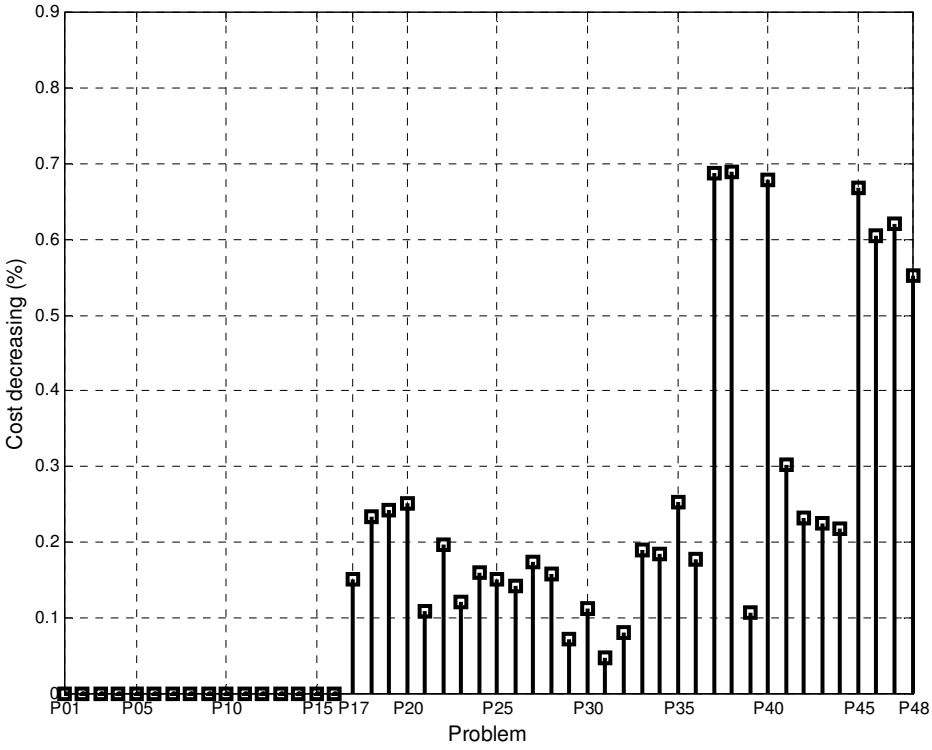

Fig. 3 The cost reduction obtained when running step 2

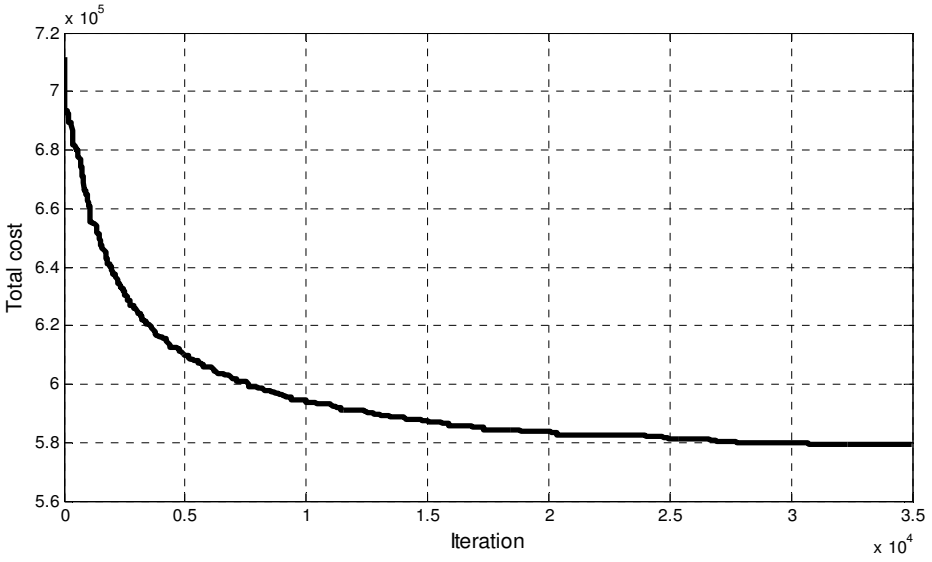

Fig. 4 Convergence curve for P40: step 1

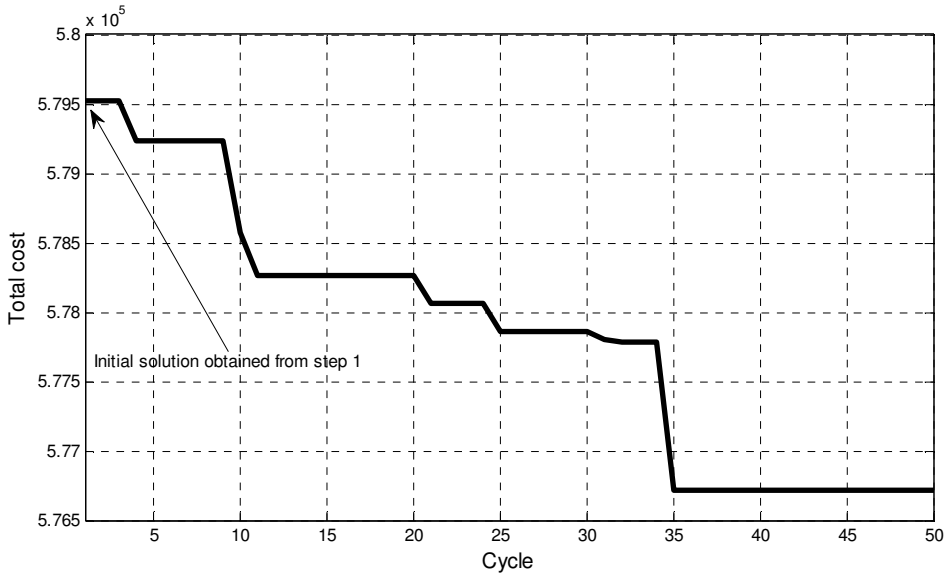

Fig. 5 Convergence curve for P40: step 2

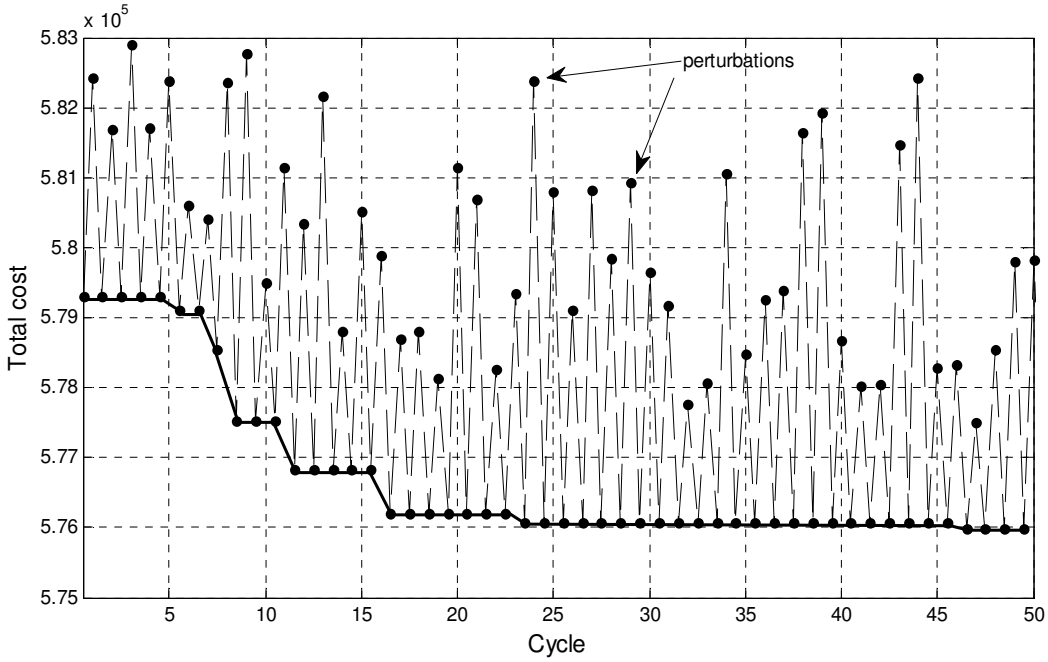

Fig. 6 Illustration of perturbations and improvements in step 2

5. Conclusion

Inspired by the paradigm of iterated local search described in Lourenço et al. (2002), an improvement of the great deluge algorithm is proposed. We call this approach an *iterated great deluge* (IGD) and we apply it to the discrete DFLP formulated as a modified QAP. The proposed IGD performed well for the data set used in Balakrishnan and Cheng (2000). New best solutions are found for 17 problem instances. This IGD is composed of two main steps that balance adequately between intensification and diversification. During the first step, an EGD algorithm is applied to find a local optimum solution. In the second step, this solution is perturbed and another EGD procedure is applied to improve the solution just "ruined"; hopefully, the new improved solution is better than the solutions obtained in the previous iterations. By repeating the second step many times, one tries to find high quality solutions for the DFLP. Inheriting the characteristics of the great deluge algorithm and its extension, our IGD requires the tuning of only few parameters. Future work will concern the application of this approach to other difficult combinatorial optimization problems.

References

Anstreicher, K.M., Brixius, N.W., Goux, J.-P., and Linderoth, J. (2002). Solving large quadratic assignment problems on computational grids. *J. Math. Programm.*, **91**(3), 563–588.

Balakrishnan, J. and Cheng, C.H. (1998). Dynamic layout algorithms: a state-of-the-art survey. Omega, *International Journal of Management Science*, **26**, 507–521.

Balakrishnan, J. and Cheng, C.H. (2000). Genetic search and the dynamic layout problem. *Computers & Operations Research*, **27**(6), 587–593.

Balakrishnan, J. and Cheng, C.H. (2009). The dynamic plant layout problem: Incorporating rolling horizons and forecast uncertainty. Omega, *International Journal of Management Science*, **37**, 165–177.

Balakrishnan, J., Cheng, C.H., Conway, DG., and Lau, C.M. (2003). A hybrid genetic algorithm for the dynamic plant layout problem. *International Journal of Production Economics*, **86**, 107–120.

Balakrishnan, J., Jacobs, R.F., and Venkataramanan, M.A. (1992). Solutions for the constrained dynamic facility layout problem. *European Journal of Operational Research*, **57**, 280–286.

Battiti, R. and Techiolli, G. (1994). The reactive tabu search. *ORSA J. Comput.*, **6**(2), 126–140.

Baum, E.B. (1986). Towards Practical 'Neural' Computation for Combinatorial Optimization Problems. *In J.S. Denker (ed.) Neural networks for computing, American Institute of Physics, New York*, 53–58.

Baykasoglu, A. and Gindy, N.N.Z. (2001). A simulated annealing algorithm for dynamic facility layout problem. *Computers & Operations Research*, **28**(14), 1403–1426.

Baykasoglu, A. and Gindy, N.N.Z. (2004). Erratum to "A simulated annealing algorithm for dynamic facility layout problem [Computers & Operations Research 28 (2001) 1367–460]". *Computers & Operations Research*, **31**(2), 313–315.

Bazara, M.S. and Sherali, M.D. (1980). Benders' partitioning schema applied to a new formulation of the quadratic assignment problem. *Naval Res. Logis. Quart.*, **27**(1), 29–41.

Burkard, R.E. and Bonniger, T. (1983). A heuristic for quadratic Boolean programs with applications to quadratic assignment problems. *Eur. J. Operat. Res.*, **13**(4), 374–386.

Burkard, R.E. and Rendl, F. (1984). A thermodynamically motivated simulation procedure for combinatorial optimization problems. *Eur. J. Operat. Res.*, 1984, **17**(2), 169–174.

Bland, J.A. (1999a). Space-planning by ant colony optimization. *Int. J. Comput. Appl. Technol.*, **12**(6), 320–328.

Bland, J.A. (1999b). Layout of facilities using an ant system approach. *Eng. Optimiz.*, **32**(1), 101–115.

Burke, E.K., Bykov, Y., Newall, J.-P., and Petrovic, S. (2004). A time-predefined local search approach to exam timetabling problems. *IIE Trans.*, **36**(6), 509–528.

Chan, K.C. and Tansri, H. (1994). A study of genetic crossover operations on the facilities layout problem. *Comput. Indust. Eng.*, **26**(3), 537–550.

Connolly, D.T. (1990). An improved annealing scheme for the QAP. *Eur. J. Operat. Res.*, **46**(1), 93–100.

Conway, D.G. and Venkataramanan, M.A. (1994). Genetic search and the dynamic facility layout problem. *Computers & Operations Research*, **21**(8), 955–60.

Cordeau, J.-F., Laporte, G., and Pasin, F. (2008). Iterated tabu search for the car sequencing problem. *Eur. J. Operat. Res.*, **191**(3), 945–956.

Cung, V.D., Mautor, T., Michelon, P., and Tavares, A. (1997). A Scatter Search Based Approach for the Quadratic Assignment Problem. *IEEE ICEC'97 Conference*, 165–170.

Driscoll, J. and Sawyer, J.H.F. (1985). A computer model for investigating the relayout of batch production areas. *International Journal of Production Research*, **23**(4), 783–794.

Dueck, G. (1993). New Optimization Heuristics: The Great Deluge Algorithm and the Record-to-Record Travel. *Journal of Computational Physics*, **104**(1), 86–92.

Erel, E., Ghosh, J.B. and Simon, J.T. (2003). New heuristic for the dynamic layout problem. *Journal of the Operational Research Society*, 54(12), 1275–1282.

Fleurent, C. and Ferland, J. (1994) Genetic hybrids for the quadratic assignment problem. *DIMACS Ser. Math. Theoret. Comput. Sci.*, **16**(1), 190–206.

Gambardella, L.M., Taillard, E.D., and Dorigo, M. (1999). Ant colonies for the QAP. *Journal of the Operational Research Society*, **50**(2), 167–176.

Herargu, S.S. and Alfa, A.S. (1992). Experimental analysis of simulated annealing based algorithms for the layout problem. *Eur. J. Operat. Res.*, **57**(2), 190–202.

Koopmans, T.C. and Beckmann, M.J. (1957). Assignment problems and the location of economic activities. *Econometrica*, **25**(1), 53–76.

Lawler, E. (1963). The quadratic assignment problem. *Manag. Sci.*, **9**(4), 586–599.

Lacksonen, T.A and Enscore, E.E. (1993). Quadratic assignment algorithms for the dynamic layout problem. *International Journal of Production Research*, **31**(3), 503–517.

Lourenço H.R., Martin, O., and Stützle, T. (2002). Iterated local search. *In F. Glover and G. Kochenberger, editors*, *Handbook of Metaheuristics*, volume 57 of *International Series in Operations Research & Management Science*, Norwell, MA: Kluwer Academic Publishers, 321-353.

Lourenço H.R., Martin, O., and Stützle, T. (2002). A beginner's introduction to iterated local search. *Proceedings of MIC'2001 - 4th Meta-heuristics International Conference at Porto*, Portugal.

Maniezzo, V. and Colorni, A. (1999). The ant system applied to the quadratic assignment problem. *IEEE Transactions on Knowledge and Data Engineering*, **11**(5), 769-778.

Martin, O., Otto, S.W., and Felten, E.W. (1991). Large-step Markov chains for the travelling salesman problem. *Complex Systems*, **5**, 299-326.

McKendall Jr., A.R., Shang, J., and Kuppusamy, S. (2006). Simulated Annealing Heuristics for the Dynamic Facility Layout Problem. *Computers & Operations Research*, **33**(8), 2431-2444.

McKendall Jr., A.R. and Shang, J. (2006). Hybrid Ant Systems for the Dynamic Facility Layout Problem. *Computers & Operations Research*, **33**(3), 790-803.

McKendall Jr., A.R. (2008). Improved Tabu Search Heuristics for the Dynamic Space Allocation Problem. *Computers & Operations Research* , 35(10), 790-803.

Misevicius, A. (2004). Using iterated tabu search for the traveling salesman problem. *Information Technology and. Control*, **3**(32), 29–40.

Misevicius, A., Lenkevicius, A., and Rubliauskas, D. (2006). Iterated tabu search: an improvement to standard tabu search. *Information Technology and. Control*, **35**(3), 187–197.

Montreuil, B., Ouazzani, N., Brotherton, E., and Nourelfath, M. (2004). AntZone layout metaheuristic: coupling zone-based layout optimization, ant colony system and domain knowledge. In *Progress in Material Handling Research* (Material Handling Institute of America: USA), 301–331.

Nahas, N., Ait-Kadi, D., and Nourelfath, M. (2006). A new Approach for Buffer Allocation in Unreliable production Lines. *International Journal of Production Economics*, **103**(2), 873-881.

Nahas, N., Nourelfath, M. and Ait-Kadi, D. (2007a). Coupling ant colony and the degraded ceiling algorithm for the redundancy allocation problem of series–parallel systems. *Reliability Engineering and System Safety*, **92**(2), 211-222.

Nahas, N., Nourelfath, M., and Ait-Kadi, D. (2007b). A two-phase extended great deluge algorithm for the dynamic layout problem. *Proceedings of* the *International*

Conference on Industrial Engineering and Systems Management (IESM 2007), BEIJING – CHINA. 30 Mai – 2 Juin.

Nehi, H.M. and Gelareh, S. (2007). A survey of meta-heuristic solution methods for the quadratic assignment problem. *Applied Mathematical Sciences*, **1**(46), 2293-2312.

Nourelfath, M., Nahas, N., and Montreuil B. (2007). Coupling ant colony optimization and the extended great deluge algorithm for the discrete facility layout problem. *Engineering Optimization*, **39**(8), 953–968.

Nugent, C.E., Vollman, T.E., and Ruml, J. (1968). An experimental comparison of techniques for the assignment of techniques to locations. *Operat. Res.*, **16**(1), 150–173.

Rezazadeh, H., Ghazanfari, M., Saidi-Mehrabad, M., and Jafar Sadjadi S. (2009). An extended discrete particle swarm optimization algorithm for the dynamic facility layout problem. *Journal of Zhejiang University - Science A*, **10**(4), 520–529.

Rosenblatt, M.J. (1986). The dynamics of plant layout. *Management Science*, **32**(1), 76-86.

Schrimpf, G., Schneider, S., Stamm-Wilbrandt, H., and Dueck, G. (2000). Record breaking optimization results using the ruin and recreate principle. *Journal of Computational Physics*, **159**(2), 139-171.

Shani, S. and Gonzalez, T. (1976). P-complete approximations problems. *J. ACM*, 1976, **23**(3), 555–565.

Skorin-Kapov, J. (1990). Tabu search applied to the quadratic assignment problem. *ORSA J. Comput.*, **2**(1), 33–45.

Solimanpur, M., Vrat, P., and Shankar, R. (2003). Ant colony optimization to the inter-cell layout problem in cellular manufacturing. *Eur. J. Operat. Res.*, **157**(3), 592–606.

Sondergeld, L. and Voß, S. (1996). A star-shaped diversification approach in tabu search. In *Meta-Heuristics: Theory and Applications*, edited by I.H. Osman and J.P. Kelly, 489–502 (Kluwer: Dordnecht).

Stützle, T. (1998). Applying iterated local search to the permutation flow shop problem. *Technical Report AIDA-98-04*, FG Intellektik, TU Darmstadt, August.

Stützle, T. (1999). Local search algorithms for combinatorial problems - analysis, algorithms and new applications. *DISKI - Dissertationen zur Künstliken Intelligenz.* infix, Sankt Augustin, Germany.

Taillard, E. (1991). Robust Taboo Search for the Quadratic Assignment Problem. *Parallel Computing*, **17**, 443-455.

Tate, D.M. and Smith, A.E. (1995). A genetic approach to the quadratic assignment problem. *Computers and Operations Research*, **22**(1), 73-83.

Urban, T.L. (1993). A heuristic for the dynamic facility layout problem. *IIE Transactions*, **25**(4), 57-63.

Wilhelm, M.R. and Ward, T.L. (1987). Solving Quadratic Assignment Problems by Simulated Annealing. *IIE Transactions*, **19**(1), 107-119.

APPENDIX

Table A1. P22 new solution.

Problem P22															
Period 1	11	12	2	14	3	6	8	9	10	1	4	13	7	5	15
Period 2	15	12	11	14	3	6	8	9	10	1	4	13	7	5	2
Period 3	15	12	11	14	3	6	8	10	7	1	4	9	13	5	2
Period 4	1	12	7	8	3	6	14	11	15	10	4	9	13	5	2
Period 5	1	12	7	8	3	6	14	11	15	10	4	9	13	5	2
Total cost: 486493															

Table A2. P23 new solution.

Problem P23															
Period 1	13	12	4	2	15	1	7	9	8	6	5	3	10	11	14
Period 2	2	12	4	1	15	13	7	9	8	6	5	3	10	11	14
Period 3	3	12	4	1	5	13	7	9	8	6	15	2	10	11	14
Period 4	14	12	4	8	15	6	5	1	3	10	7	2	9	11	13
Period 5	14	12	4	8	9	6	5	2	3	10	7	15	1	11	13
Total cost: 486268															

Table A3. P25 new solution.

Problem P25															
Period 1	4	5	15	6	7	3	12	13	11	9	2	14	8	10	1
Period 2	11	5	9	12	13	3	7	10	6	4	2	14	8	15	1
Period 3	11	5	9	12	13	1	7	10	6	4	2	14	8	15	3
Period 4	11	4	9	12	13	1	7	10	6	5	2	14	8	15	3
Period 5	11	4	3	12	13	7	5	2	1	10	8	14	6	15	9
Period 6	5	4	11	12	2	3	14	13	1	10	8	7	6	15	9
Period 7	5	4	11	12	2	3	14	13	9	10	8	7	6	15	1
Period 8	5	15	11	12	4	14	10	6	7	3	8	13	2	9	1
Period 9	7	15	11	4	12	14	1	3	5	10	8	6	2	9	13
Period 10	7	2	11	15	12	14	1	3	5	10	8	6	4	9	13
Total cost: 978848															

Table A4. P28 new solution.

Problem P28															
Period 1	15	10	4	7	11	8	5	3	1	12	14	13	2	9	6
Period 2	15	10	3	7	11	8	5	4	1	12	14	13	2	9	6
Period 3	7	10	3	15	11	8	5	4	1	12	14	13	2	9	6
Period 4	5	10	3	15	2	1	8	7	11	12	14	13	9	4	6
Period 5	5	11	13	15	2	1	10	7	8	12	14	4	9	3	6
Period 6	3	11	13	15	2	1	10	7	9	12	14	4	8	5	6
Period 7	6	3	13	15	2	1	12	14	9	10	7	4	11	5	8
Period 8	6	3	13	7	1	2	12	14	15	10	9	4	11	5	8
Period 9	6	3	13	7	1	2	12	11	15	10	9	4	14	5	8
Period 10	6	3	13	7	1	2	12	11	15	10	9	4	14	5	8
Total cost: 971759															

Table A5. P28 new solution.

Problem P32															
Period 1	14	1	13	5	4	2	10	7	9	11	6	8	3	12	15
Period 2	14	1	13	5	15	2	10	7	9	11	6	8	3	12	4
Period 3	5	1	13	9	15	2	8	7	10	11	6	12	3	14	4
Period 4	13	1	6	9	15	8	5	4	2	11	10	12	3	14	7
Period 5	13	12	6	10	15	8	2	4	1	11	9	5	3	14	7
Period 6	13	12	3	6	15	10	5	4	1	8	9	2	11	14	7
Period 7	13	12	4	6	15	10	5	3	1	8	9	2	11	14	7
Period 8	4	3	15	6	5	10	12	13	1	8	9	2	11	14	7
Period 9	4	3	1	6	11	13	9	7	8	2	15	10	5	14	12
Period 10	5	3	1	4	11	13	9	7	8	2	15	10	6	14	12
Total cost: 982888															

Table A6. P33 new solution.

Problem P33																													
Period 1	26	5	23	22	29	24	4	1	20	28	6	10	27	12	15	17	13	7	2	18	19	25	14	30	11	8	16	9	21	3
Period 2	20	5	17	19	15	13	4	1	21	28	6	10	27	24	29	23	26	7	2	18	22	25	14	30	11	8	16	9	12	3
Period 3	20	5	17	19	15	13	4	1	21	28	6	10	27	24	29	2	26	3	7	18	22	25	14	30	11	8	16	9	12	23
Period 4	20	5	17	21	1	2	4	8	19	28	11	10	27	24	15	13	26	3	7	18	22	25	14	30	29	6	16	9	12	23
Period 5	20	5	17	21	1	19	4	8	29	28	11	10	27	24	30	23	26	3	7	13	22	25	14	15	2	6	16	9	12	18
Total cost: 575386																													

Table A7. P35 new solution.

Problem P35																													
Period 1	6	21	2	16	27	5	13	26	15	12	22	8	1	17	19	14	10	29	28	9	18	20	23	11	30	3	7	4	25	24
Period 2	6	14	2	16	13	5	20	26	24	12	22	8	1	30	27	21	10	29	19	9	18	28	23	11	17	3	7	4	25	15
Period 3	16	14	21	30	13	12	20	26	24	1	22	28	23	5	6	2	10	29	19	9	7	8	3	11	17	18	27	4	25	15
Period 4	16	14	21	30	13	12	20	26	24	1	22	28	23	5	6	2	10	29	19	9	7	8	3	11	17	18	27	4	25	15
Period 5	16	14	21	30	13	12	20	26	24	1	22	6	2	5	23	28	10	29	19	9	7	8	3	11	17	18	27	4	25	15
Total cost: 572104																													

Table A8. P36 new solution.

Problem P36																													
Period 1	7	6	4	17	28	8	30	9	25	12	10	21	24	23	18	14	19	5	22	29	13	16	27	20	26	2	15	11	3	1
Period 2	7	6	4	17	28	8	30	9	25	24	10	18	16	23	22	21	19	5	12	29	13	26	27	20	14	2	15	11	3	1
Period 3	19	13	21	17	28	25	30	9	1	2	10	18	16	6	5	3	8	22	12	29	4	7	27	20	14	26	15	11	24	23
Period 4	19	13	21	17	28	25	30	9	1	2	10	18	16	6	5	3	8	22	12	29	4	7	27	20	14	26	15	11	24	23
Period 5	19	13	21	17	28	25	30	9	1	2	10	18	16	6	5	3	8	22	12	29	4	7	27	20	14	26	15	11	24	23
Total cost: 564398																													

Table A9. P37 new solution

Problem P37																													
Period 1	16	7	5	14	19	9	20	21	24	22	29	12	3	26	27	30	23	8	15	10	25	28	1	17	13	6	4	2	11	18
Period 2	16	7	5	14	19	9	20	21	24	22	29	12	3	26	27	30	23	8	15	10	25	28	1	17	13	6	4	2	11	18
Period 3	16	7	5	14	19	9	20	21	24	22	29	12	3	26	27	30	23	8	15	10	25	28	1	17	13	6	4	2	11	18
Period 4	16	25	24	14	19	28	20	5	7	4	2	12	22	26	3	1	6	29	15	10	8	9	23	17	13	30	27	21	11	18
Period 5	16	25	24	14	19	28	20	5	7	4	2	12	22	26	3	1	6	29	15	10	8	9	23	17	13	30	27	21	11	18
Total cost: 555555																													

Table A10. P38 new solution

Problem P38
Period 1 29 26 22 13 25 14 24 3 8 7 2 21 18 5 4 10 17 23 15 19 6 1 9 16 30 11 12 27 28 20
Period 2 29 26 22 13 25 14 24 3 8 7 2 21 18 5 4 10 17 23 15 19 6 1 9 16 30 11 12 27 28 20
Period 3 29 26 22 13 25 14 24 3 8 7 2 21 18 5 4 10 16 23 15 19 6 1 9 17 30 11 12 27 28 20
Period 4 8 6 22 26 25 3 24 29 12 15 2 21 4 5 13 14 16 23 10 19 17 18 9 1 30 7 11 27 28 20
Period 5 8 6 22 26 25 3 24 29 12 15 2 21 4 5 13 14 16 23 10 19 17 18 9 1 30 7 11 27 28 20
Total cost: 564124

Table A11. P40 new solution

Problem P40
Period 1 29 26 17 10 24 23 25 8 12 11 14 3 21 13 6 2 9 30 22 20 4 7 5 18 16 28 1 19 27 15
Period 2 29 26 17 10 24 23 25 8 12 11 14 3 21 13 6 2 9 30 22 20 4 7 5 18 16 28 1 19 27 15
Period 3 9 4 17 10 2 13 25 18 24 22 15 3 12 27 26 28 29 30 6 20 21 16 23 11 5 14 1 19 7 8
Period 4 9 4 17 10 2 13 26 18 24 25 15 3 12 27 19 28 29 30 6 20 21 16 23 11 5 14 1 22 7 8
Period 5 9 4 17 10 2 13 26 18 24 25 15 3 12 27 19 28 29 30 6 20 21 16 23 11 5 14 1 22 7 8
Total cost: 572802

Table A12. P41 new solution

Problem P41
Period 1 26 4 9 2 10 22 5 6 12 23 20 3 27 25 15 17 13 1 8 19 16 28 14 11 21 30 18 7 29 24
Period 2 12 4 9 2 10 20 5 6 22 23 17 3 27 25 29 21 24 1 8 19 26 28 14 11 13 30 18 7 15 16
Period 3 12 4 9 21 10 20 5 6 22 23 17 3 27 25 29 2 24 1 8 19 26 28 14 11 13 30 18 7 15 16
Period 4 7 4 9 21 10 20 5 18 22 15 17 3 27 25 29 13 24 1 6 19 26 28 14 11 2 30 16 12 8 23
Period 5 7 19 9 21 10 20 5 18 22 15 17 3 27 25 29 23 24 1 6 30 26 28 14 11 2 13 16 12 8 4
Period 6 7 29 9 21 10 23 5 18 22 15 17 3 27 25 19 20 24 1 6 30 26 28 14 11 2 13 16 12 8 4
Period 7 15 2 9 21 10 20 5 18 22 29 17 3 16 25 27 23 24 1 6 30 26 28 14 11 19 13 7 12 8 4
Period 8 15 2 9 21 10 20 5 18 22 29 17 3 16 25 27 23 24 1 6 30 26 28 14 11 19 13 7 12 8 4
Period 9 21 27 9 15 10 20 5 18 22 26 17 2 16 25 11 19 24 1 6 30 13 28 14 23 29 3 7 12 8 4
Period 10 21 27 9 15 10 20 5 18 22 26 17 2 16 25 23 29 24 1 6 30 13 28 14 7 11 3 19 12 8 4
Total cost: 1157887

Table A13. P42 new solution

Problem P42
Period 1 11 4 19 1 6 3 21 27 14 17 8 5 29 30 16 13 15 7 2 18 23 20 24 28 26 10 12 9 22 25
Period 2 11 4 19 1 6 3 13 27 29 28 8 20 5 30 25 21 26 7 2 18 23 22 24 15 16 10 12 9 14 17
Period 3 11 4 19 1 6 3 13 27 29 28 8 20 5 30 25 21 26 7 2 18 23 22 24 15 16 10 12 9 14 17
Period 4 11 4 19 1 6 3 13 27 29 28 8 20 5 30 25 21 26 7 2 18 23 22 24 15 16 10 12 9 14 17
Period 5 11 7 19 1 6 3 13 27 29 28 8 20 5 30 25 4 26 17 2 18 23 22 24 15 16 10 12 9 14 21
Period 6 11 7 19 1 6 3 4 27 29 28 16 20 5 17 25 14 26 23 2 18 24 22 13 15 30 10 12 9 8 21
Period 7 11 7 19 1 6 3 4 27 29 28 16 20 5 17 25 30 26 23 2 18 24 22 13 15 8 10 12 9 14 21
Period 8 23 7 22 1 6 3 4 27 29 28 16 20 5 17 25 11 26 24 2 19 15 18 13 30 8 10 12 9 14 21
Period 9 11 7 22 1 6 3 4 27 29 28 16 20 5 17 25 23 26 24 2 19 15 30 13 18 8 10 12 9 14 21
Period 10 11 7 22 1 6 3 4 27 29 28 16 25 5 17 19 20 26 24 2 14 15 30 13 18 8 10 12 9 23 21
Total cost: 1158243

Table A14. P43 new solution

Problem P43
Period 1 24 11 26 10 5 6 4 9 14 12 16 27 7 19 20 15 21 1 23 28 18 17 22 8 30 3 25 29 13 2
Period 2 17 20 26 10 5 19 4 9 30 2 16 14 7 24 6 27 21 1 12 28 18 23 22 8 15 3 25 29 13 11
Period 3 17 20 26 4 16 19 27 9 30 2 5 14 18 24 6 7 21 23 12 28 1 10 22 8 15 3 25 29 13 11
Period 4 17 20 26 4 16 19 27 9 30 2 5 14 18 24 6 7 21 23 12 28 1 10 22 8 15 3 25 29 13 11
Period 5 17 20 26 4 16 19 2 9 30 27 5 14 18 24 23 7 21 6 12 28 1 10 22 8 15 3 25 29 13 11
Period 6 9 20 11 4 16 19 2 1 30 27 17 14 18 24 23 22 21 3 12 28 25 26 15 8 6 10 13 29 7 5
Period 7 9 20 11 4 16 19 2 1 30 27 17 14 21 24 23 22 18 3 12 28 25 26 15 8 6 10 13 29 7 5
Period 8 9 20 11 4 16 30 2 22 12 27 17 6 21 24 14 19 18 3 5 28 13 26 15 8 23 10 25 29 7 1
Period 9 9 13 11 4 16 20 2 22 12 27 17 6 21 24 30 19 18 3 5 28 23 26 15 8 14 10 25 29 7 1
Period 10 9 13 11 4 16 30 2 22 15 27 20 6 21 24 17 19 18 3 5 28 12 26 14 8 23 10 25 29 7 1
Total cost: 1155319

Table A15. P44 new solution

Problem P44
Period 1 8 1 18 5 2 26 3 27 29 19 23 4 21 20 13 14 16 11 10 28 12 25 30 6 22 24 17 15 9 7
Period 2 8 1 18 5 2 10 3 27 29 4 23 12 19 26 21 22 16 11 13 28 24 25 30 6 20 14 17 15 9 7
Period 3 25 22 18 16 26 29 21 9 1 4 24 12 19 10 5 7 8 11 13 28 2 3 30 6 20 14 17 15 27 23
Period 4 25 22 18 16 26 29 21 9 1 4 6 12 24 10 5 7 8 11 13 28 2 3 30 19 20 14 17 15 27 23
Period 5 25 22 18 16 26 29 21 9 1 4 6 12 24 10 5 7 8 11 13 28 2 3 30 19 20 14 17 15 27 23
Period 6 25 22 18 16 26 29 21 9 1 4 6 12 24 10 5 7 8 11 13 28 2 3 30 19 20 14 17 15 27 23
Period 7 3 5 18 1 10 17 8 9 27 24 21 12 4 23 29 25 26 11 13 28 30 22 7 19 20 14 6 15 2 16
Period 8 3 5 18 1 10 17 8 9 27 24 21 12 4 23 29 25 26 11 13 28 30 22 7 19 20 14 6 15 2 16
Period 9 3 5 18 1 10 17 8 9 27 24 21 12 4 23 29 25 26 11 13 28 30 22 7 19 20 14 6 15 2 16
Period 10 3 5 18 1 10 24 8 9 19 20 13 22 4 23 16 12 26 11 25 28 30 14 7 29 21 17 6 15 2 27
Total cost: 1140395

Table A16. P45 new solution

Problem P45
Period 1 16 13 20 15 2 12 3 21 26 27 29 14 8 24 30 28 22 4 6 10 23 25 7 11 5 19 1 17 18 9
Period 2 16 13 20 15 2 12 3 21 26 27 29 14 8 24 30 28 22 4 6 10 23 25 7 11 5 19 1 17 18 9
Period 3 16 13 20 15 2 12 3 21 26 27 29 14 8 5 30 28 22 4 6 10 23 25 24 11 7 19 1 17 18 9
Period 4 13 22 20 15 23 12 25 26 7 1 2 29 28 5 3 9 6 19 27 10 8 4 24 11 21 14 30 17 18 16
Period 5 13 22 20 15 23 12 25 26 7 1 2 29 28 5 3 9 6 19 27 10 8 4 24 11 21 14 30 17 18 16
Period 6 13 22 20 15 23 24 25 26 7 1 2 29 28 5 3 9 6 19 27 10 8 4 12 11 21 14 30 17 18 16
Period 7 7 30 20 3 8 24 23 14 12 11 2 29 28 18 17 13 15 4 27 10 16 19 22 25 21 26 6 9 1 5
Period 8 7 30 20 3 8 24 23 14 12 11 2 29 28 18 17 13 15 4 27 10 16 19 22 25 21 26 6 9 1 5
Period 9 7 30 20 3 8 24 23 14 12 11 2 29 28 18 17 13 15 4 27 10 16 19 22 25 21 26 6 9 1 5
Period 10 7 30 20 3 8 10 23 14 12 11 2 29 28 18 17 13 15 4 27 24 16 19 22 25 21 26 6 9 1 5
Total cost: 1123385

Table A17. P46 new solution

Problem P46
Period 1 11 15 23 17 20 21 27 6 1 10 18 28 25 8 4 7 9 22 24 5 3 2 12 16 29 26 19 13 30 14
Period 2 11 15 23 17 20 21 27 6 1 10 18 28 25 8 4 7 9 22 24 5 3 2 12 16 29 26 19 13 30 14
Period 3 11 15 23 17 20 21 27 6 14 10 18 28 1 8 4 7 9 22 24 5 3 2 12 16 29 26 19 13 30 25
Period 4 3 2 23 5 20 21 27 4 14 16 18 28 1 15 17 12 11 22 24 29 19 13 7 10 8 26 9 6 30 25
Period 5 3 2 23 5 20 21 27 4 14 16 18 28 1 15 17 12 11 22 24 29 19 13 7 10 8 26 9 6 30 25
Period 6 3 7 23 5 2 21 27 4 14 16 18 28 1 15 17 12 11 22 24 29 19 13 20 10 8 26 9 6 30 25
Period 7 3 7 23 5 2 21 27 4 14 16 18 28 1 15 17 12 11 22 24 29 19 13 20 10 8 26 9 6 30 25
Period 8 3 7 11 12 2 4 14 30 23 22 18 13 1 25 28 26 27 5 17 29 24 21 20 10 8 15 9 6 19 16
Period 9 3 7 11 12 2 4 14 30 23 29 18 13 1 22 28 26 27 5 17 21 24 25 20 10 8 15 9 6 19 16
Period 10 3 7 11 12 2 4 14 30 23 29 18 13 1 22 28 26 27 5 17 21 24 25 20 10 8 15 9 6 19 16
Total cost: 1140723