Lecture Notes in Management Science (2014) Vol. **6**: 4–14
6th International Conference on Applied Operational Research, Proceedings

ISSN 2008-0050 (Print), ISSN 1927-0097 (Online)

Approximated dynamic programming algorithms with variable neighbourhood search for reformed dynamic quadratic assignment problems

Pongchanun Luangpaiboon [1], Sirirat Muenvanichakul [2] and Peerayuth Charnsethikul [2]

[1] Industrial Statistics and Operational Research Unit (ISO-RU), Dept of Industrial Engineering, Faculty of Engineering, Thammasat University, Pathumthani, Thailand
lpongch@engr.tu.ac.th

[2] Dept of Industrial Engineering, Faculty of Engineering, Kasersart University, Bangkok, Thailand
sirirat.m@ku.ac.th

Abstract. When determining the optimal solution of the dynamic quadratic assignment problem (DQAP) it is extremely difficult since it is the NP hard problem. The reformed dynamic quadratic assignment problem (RDQAP) has been reformulated and applied in two alternatives of linearised and logic-based models after proving the model equivalence. This study follows the former and introduces the hybridisation of the conventional dynamic programming algorithm with the meta-heuristics of bee colony optimisation (ADPA-I) and simulated annealing (ADPA-II) algorithms called the approximated dynamic programming algorithms (ADPA). In order to improve quality of solutions the variable neighbourhood search is also included with given initial solutions from the ADPA. In the context of ADPA, the searching procedures are incomplete as in the original DPA. For each period, a set of best solutions provided by metaheuristics is determined as the initial solution set. The number of all possible solutions is the product of initial solution set of all periods. Numerical results explain the superior quality of the ADPA-I obtained solutions when compared.

Keywords: dynamic quadratic assignment problem; dynamic programming; bee colony optimisation; simulated annealing; variable neighbourhood search

Introduction

The dynamic quadratic assignment problem (DQAP) is an extended version of the conventional quadratic assignment problem (QAP) proposed by Koopmans and Beckman (Lacksonen and Enscore, 1993). Because the business conditions such as new manufacturing orders, new product lines and technological advances are constantly changing, the need of a dynamic nature is supported. There are a series of data in static problem with its own from-to flow matrix for given finite discrete time periods. They depend on the nature of the business, a period can be given in terms of months, quarters, years, etc. An additional rearrangement cost term in the objective function ties the static problems together whenever an area contains different department in consecutive time period or moving a specific department from its location. Rearrangement cost also include all the costs involved in a facilities project, including planning, removal, disposal, construction, installation, communication and utilities costs. If rearrangement costs are performed on off-shifts or weekends, wage premiums must be used to determine costs. One must also consider either the lost production costs or overtime costs to make up production on off-shifts. DQAP is a mathematical model for a specific problem and its aim is to search for the optimal location assignment among a set of facilities over a discrete time periods. During this time, many of the parameters of the problem such as demands and distribution costs are likely to be changeable. The objective is to minimise the total of flow and rearrangement costs over all discrete time periods. The application of this problem is necessary not only for the design of new facilities, but for the redesign of existing facilities due to introduction of new products, the installation of the new equipment or process, or realisation of an increase or decrease in throughput volume as well.

The DQAP model is powerful with a wide range of potential real world applications. Applications of the DQAP include the assignment of warehouses or indivisible operations to a number of geographical sites, facility layout, the layout of indicators and control panel, partitioning, assigning storage space on computer disc storage devices, sequencing work through a production facility and so on. Both metaheuristic and optimisation algorithms have been applied to determine the optimal alternatives. However, the potential of these applications depends on the existence of computationally feasible and efficient solution procedures. They are often measured by the functional dependence of execution time on each problem size. Since the DQAP is NP-hard problem that is difficult to approach the optimum. Mathematically, the DQAP was later formulated as a reformed DQAP with two alternatives of linearised and logic-based models. This study follows the former and its mathematical model is followed.

$$\text{MIN } Z = \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{N}\sum_{l=1}^{N}\sum_{t=1}^{T} C_{ijklt} Y_{ijklt} + \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{l=1}^{N}\sum_{t=1}^{T-1} R_{ijlt} M_{ijl(t+1)} \quad (1)$$

S.T.

$$Y_{ijklt} \geq X_{ijt} + X_{klt} - 1 \qquad ; i = 1,2,\ldots,N; j = 1,2,\ldots,N; t = 1,2,\ldots,T \quad (2)$$

$$M_{ijl(t+1)} \geq X_{ijt} + X_{il(t+1)} - 1 \qquad ; i = 1,2,\ldots,N; j = 1,2,\ldots,N; t = 1,2,\ldots,T \quad (3)$$

$$\sum_{i=1}^{N} X_{ijt} = 1 \qquad ; j = 1,2,\ldots,N; t = 1,2,\ldots,T \quad (4)$$

$$\sum_{j=1}^{N} X_{ijt} = 1 \qquad ; i = 1,2,\ldots,N; t = 1,2,\ldots,T \quad (5)$$

$$Y_{ijklt}, M_{ijl(t+1)} \geq 0 \qquad ; \forall_{i,j,k,l,t} \quad (6)$$

$$X_{ijt} \in \{0,1\} \qquad ; \forall_{i,j,t} \quad (7)$$

where,

N = the number of facilities or locations in the t time period

T = the number of discrete time periods

C_{ijklt} = the cost of assigning the i facility to the j location and the k facility to the l location at the t time period, and $C_{ijklt} = f_{ikt} d_{jlt}$

f_{ikt} = the workflow cost from the i facility to the k facility in the t time period

d_{jlt} = the distance from the j location to the l location at the t time period

R_{ijlt} = the rearranging cost when the i facility located on the j location at the t time period is moved to the l location at the (t+1) time period

X_{ijt} = the decision variable of 1or 0 if the i facility is assigned to the l location at the t time period or 0 otherwise

$$Y_{ijklt} = X_{ijt} X_{klt} \tag{8}$$

$$M_{ijl(t+1)} = X_{ijt} X_{il(t+1)} \tag{9}$$

The objective of this paper is to investigate how the choice of the approximated dynamic programming algorithms based on the bee colony optimisation and simulated annealing algorithms perform on the reformed DQAP when parameter levels are optimised. The numerical comparisons are limited to some selected problem sizes in the literatures. This paper is organised as follows. Section 2 describes the approximated dynamic programming algorithms. Sections 3 and 4 illustrate computational results and analyses for comparing the performance of the proposed methods and conclusions, respectively.

Approximated Dynamic Programming Algorithms (ADPA)

In the fundamental paper on the dynamic facility layout problem, Rosenblatt proposed a dynamic programming to develop an optimal solution. With T time periods or stages the maximal number of solutions or states is $(N!)^T$; where N is the number of departments or locations. When considering all number of process layout combinations it is extremely time consuming. This problem is then simplified by applying only possible combinations in each period. Obviously, the recursive formulation is developed and the total cost for each of the layouts considered in the horizon is established by the following recursive relationship:

$$L_{tq}^* = \text{Min } \{L_{t-1,r}^* + R_{rq}\} + F_{tq}; \tag{10}$$

where,

L_{tq}^* = minimal cost to reach the q layout at the t time period

r = process layouts (states) for each time period

R_{rq} = rearrangement cost from the r layout to the q layout

F_{tq} = flow cost of the q layout in the t time period

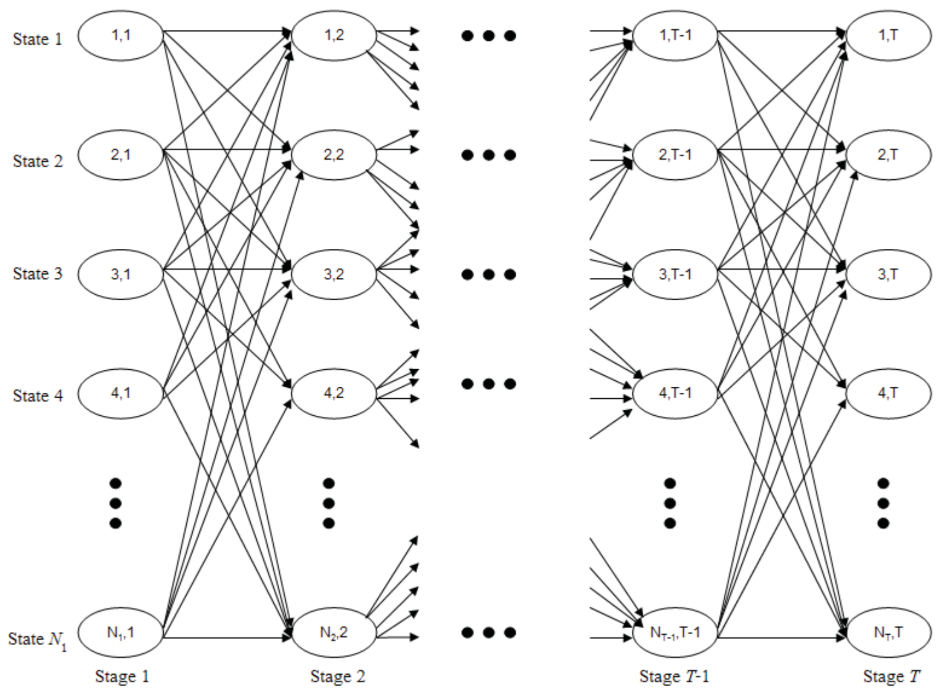


Fig. 1. Approximated dynamic programming framework (Muenvanichakul and Charnsethikul, 2010).

With Rosenblatt's dynamic programming algorithm, the discrete time period and the particular layout arrangement correspond to the stage and the state, respectively. Therefore, there are N! states in each of the T stages. Restricting the state space of the algorithm was determined that any layout arrangement for a given period does not need to be considered. If the difference between the total cost of the arrangement and the cost of optimal static solution for that period is larger than the difference between the levels of the upper and lower bounds of the algorithm. Therefore, only the best static solutions for each period need to be considered. For small problems, this algorithm provides the optimal solutions but for very large problems, exploring all possible solutions in each period needs the capability software and hardware used to solve the problem. Let N' is the applicable number of the static assigning layouts in each period. The main concept of this ADPA is it will provide the optimal solution for the layouts included in its procedure (Figure 1). However, when $N' < N!$, it then cannot guarantee the optimum since there are not all the possible static layouts included in the procedure. Concerning the powerful selecting the best N' layouts in each period via metaheuristics, the best so far solutions should finally lead to better solutions (Dunker *et al*, 2005; Urban, 1993). There were some literatures to apply single method alone of metaheuristics such as genetic, tabu search and simulated annealing algorithms by the same set of initial layouts (Muenvanichakul and Charnsethikul, 2010).

ADPA with Bee Colony Optimisation (ADPA-I)

Bee colony optimisation algorithm (BCO) is integrated to the ADPA for selecting the best so far alternatives in each time period. BCO mimics the behaviour of the bees when finding the nectar. The searching strategy is operated through the bees which are divided into two classes of scout bees and employee bees (Karaboga and Basturk, 2007a and Karaboga and Basturk, 2007b). Nectar sources are assumed to be the solutions. Scout bees randomly search for the nectar in the range of possible solutions of their search space. When scout bees find the solutions, they fly back to their hive and communicate with other bees. Via the use of dance styles bees communicate to tell the amount of nectar and the direction of that nectar. Employee bees then carry the nectar in a nectar source and vary the amount of nectar and the distance. Algorithmic procedures of the BCO are given as follow. Firstly, the n number of scout bees is set to determine the initial solutions. Each solution from scout bees is evaluated and all solutions are sorted in descending order. The m best sites from best sites are chosen. These best sites are separated into two groups. The first group is the e best sites which are in the top and the other group is the m-e remaining sites. The BCO conducts searches in the *nep* and *nsp* neighbourhood sites of the e best bees and of the other selected bees, respectively (Karaboga and Basturk, 2008). For each site only the bee with the highest fitness will be selected to form the next m bee population. Randomly assign remaining bees to meet the n number of scout bees and repeat iterative processes until the stopping criterion is met.

ADPA with Simulated Annealing Algorithm (ADPA-II)

Simulated annealing algorithm (SA) is also integrated to the ADPA for selecting the best so far alternatives in each time period. It has been derived from an interesting analogy between problems in statistical mechanics and multivariate or combinatorial optimisation (Kirkpatrick *et al*., 1983). This algorithm is a set of rules for searching large solution spaces in a manner that mimics the annealing process of metals. The algorithm simulates the behaviour of an ensemble of atoms in equilibrium at a given finite temperature and its original framework can be traced to Metropolis and the team (Granville *et al*., 1994). This algorithm has been regularly used in global function optimisation and statistical applications. In case of maximisation the procedures of this algorithm start at a corresponding initial value of the objective function. The new objective value, y_1, will be then determined. The new solution will be unconditionally accepted if its objective value is improved and the process regularly continues. Otherwise the difference or size of increment in objective values is calculated and with an auxiliary experiment the new solution would be accepted with probability. A random number is generated from the uniform distribution on (0, 1) and is compared to reject or accept the new solution. This stochastic element is from Monte Carlo sampling. It occasionally allows the algorithm to accept the new solution to the problems, which deteriorate rather than improve the objective function value. However, Simulated Annealing includes a number of parameters and they have been claimed that affect the efficiency of the algorithm.

Computational Results and Analyses

In this work, for the computational procedures described above a computer simulation program was implemented in a C# language program. A computer with processor Intel core i5, CPU 2.3GHz, and Ram 4 GB was used for computational experiments. A comparison of the hybridisations of the ADPA-I and ADPA-II are stated to determine their advantages and disadvantages in the applications on the linearised DQAP. A set of initial layouts are provided by the metaheuristic algorithms of bee colony optimisation and simulated annealing. These will be used as possible alternatives of solution in each time period before applying the iterative procedures from the DPA itself. Additionally, searching the best so far (BSF) neighbourhoods is also applied to systematically exploit the idea of neighbourhood change within a local search method to approach a better solution. It proceeds by a descent method to a local minimum. In each time, a local search routine for optimisation repeatedly proceeds by performing a sequence of local changes of one or several solutions within the current neighbourhood in an initial solution (Mladenovic and Hansen, 1997). This searching procedure, without forbidden moves, escapes from the current solution to a new one if and only if an improvement has been achieved. That is, it will be used to enhance the current solution via its neighbourhood until no further improvements are found or a local optimum is reached (Figure 2). Considering the plant layout example with twenty facilities, the facilities are assumed to be equal size. In this paper, the preliminary study was conducted by applying the steepest descent algorithm (SDA) to determine the proper levels of BCO and SA parameters on the small size of the linearised DQAP with N and T of 20 and 2, respectively. Based on SDA, if P-value exceeds the 5% preset value of significance level (α), there is no effect of parameters. The significant parameters of the BCO (Figure 3) and SA are applied to both ADPA-I and ADPA-II. Additionally, the number of solutions for all iterations from the BCO (nxi) will be applied to the SA throughout.

There are six problem sizes of the reformed DQAP which consist of (N = 5, T = 5), (N = 5, T = 8), (N = 6, T = 5), (N = 6, T = 8), (N = 40, T = 3) and (N = 20, T = 5) when N represents the facilities or locations and T represents time periods. Experimental results in each run will show the effectiveness of the algorithms in terms of the mean, standard deviation (Stdev) and the best so far (BSF) of the total cost levels. There are fifty replicates in each case. When ADPA-I and ADPA-II with the preset number of states (N') excluding repeating layouts were applied to small problems of (N = 5, T = 5), (N = 5, T = 8), (N = 6, T = 5) and (N = 6, T = 8) it can be concluded that ADPA-II provided the slightly lower levels of total cost on average (Table 1). However, other performance measures of the sample standard deviation and BSF were not different, when compared. For solving selected problems of (N = 40, T = 3) and (N = 20, T = 5) by both ADPA-I and ADPA-II with the preset number of states (N') excluding repeating layouts, it is found that ADPA-I is capable to provide the better solutions than ADPA-II based on the lower levels of sample mean and standard deviation of total cost including the BSF (Table 2). However, there is no statistically significant (Figure 4). Additionally, when considering for the whole process the average execution time of the computational runs using ADPA-I was approximately 57.25 minutes whilst 43.29 minutes was averagely taken by ADPA-II. On the other hand, ADPA-II is able to obtain the best so far solutions (Table 2).

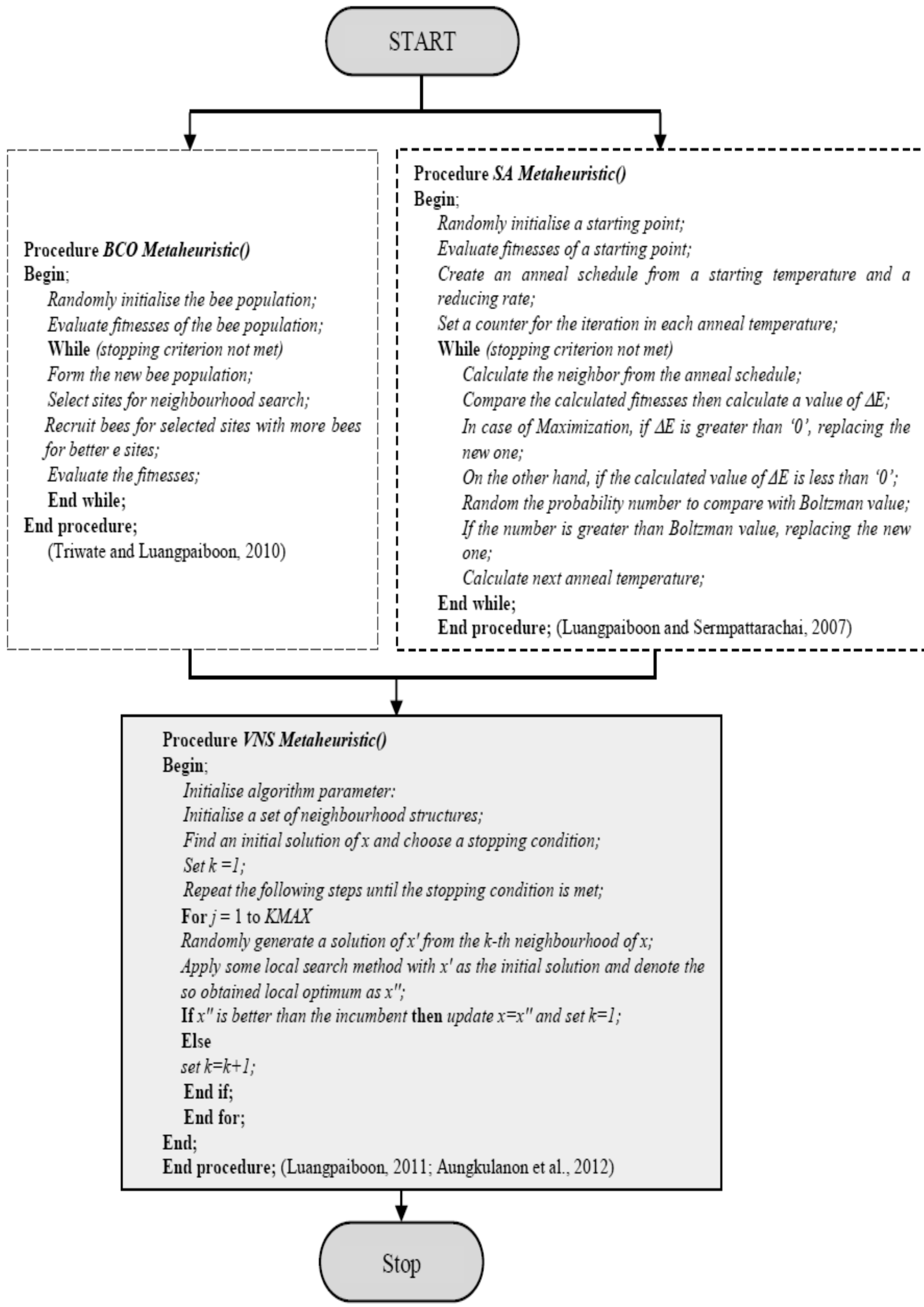


Fig. 2. The structure of ADPA-I and ADPA-II

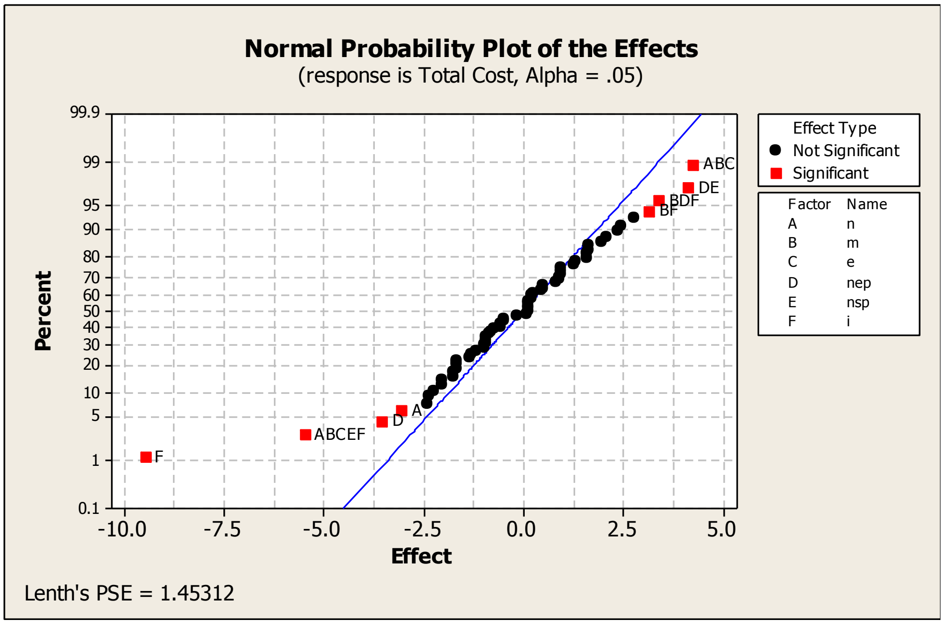


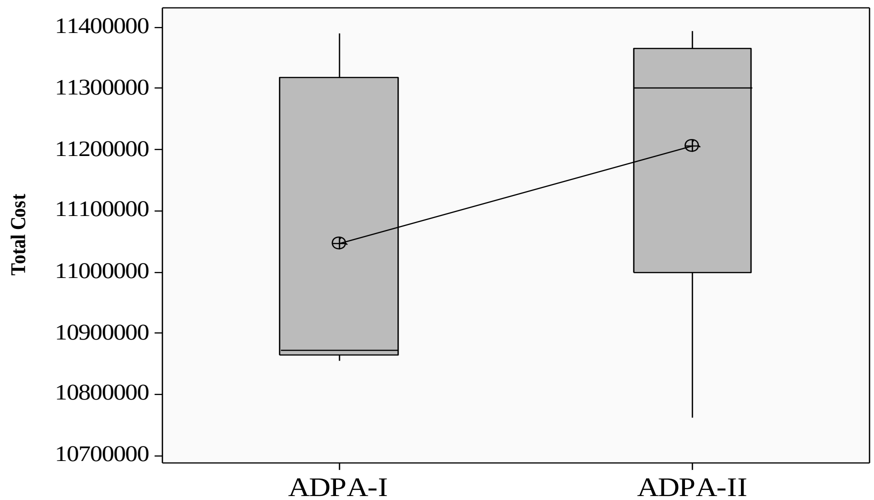
Fig. 3. Normal probability plot of effects on the BCO parameters.

Table 1. Numerical results of ADPA-I and ADPA-II on the small problems

Location	*Time Period*	*Total Cost*					
		ADPA-I			*ADPA-II*		
N	*T*	*Mean*	*Stdev*	*BSF*	*Mean*	*Stdev*	*BSF*
5	5	184,667	5,260	176,028	184,245	3,897	179,157
5	8	302,350	11,573	284,409	294,857	11,010	283,769
6	5	302,147	12,059	285,070	295,021	11,597	286,869
6	8	498,981	22,224	465,084	481,637	26,114	461,882

Table 2. Numerical results of ADPA-I and ADPA-II on the BIG problems

Location	*Time Period*	*Total Cost*					
		ADPA-I			*ADPA-II*		
N	*T*	*Mean*	*Stdev*	*BSF*	*Mean*	*Stdev*	*BSF*
20	5	4,428,337	104,025.9698	4,361,570	4,488,956	140,105.734	4,256,480
40	3	11,047,043	252,518.0338	10,854,672	11,205,815	255,079.8743	10,761,358



Two-sample T for ADPA-I vs ADPA-II

	N	*Mean*	*StDev*	*SE Mean*
ADPA-I	5	11047043	252518	112929
ADPA-II	5	11205815	255080	114075

Difference = mu (ADPA-I) - mu (ADPA-II)
Estimate for difference: -158772; 95% CI for difference: (-538338, 220794)
T-Test of difference = 0 (vs not =): T-Value = -0.99; P-Value = 0.356 DF = 7

Fig. 4. Statistical test and the Box-Whisker plot for the problem size of ($N = 40$, $T = 3$).

Conclusions

The reformed dynamic quadratic assignment problem that incorporates any changes of known future data into the linearised problem gains many benefit. They consist of the reduction of the indirect costs-flow and rearrangement costs over all discrete time periods and an improvement of closeness ratings and productivity when compared to the traditional or static problem. The existing algorithm of dynamic programming is approximated by the artificial intelligence algorithms including the neighbourhood search around the best so far solutions. An aim is to find the near optimal solutions within reasonable execution time. Those metaheuristics are the bee colony optimisation and simulated annealing algorithms. The first is the biologically-based inspiration with the initialisation of a population of solutions. The second is the physically-based inspiration with the initialisation of individual. However, the performance of both algorithms depends on their parameter levels and need to be determined and analysed before its implementation. All parameters are determined through the steepest descent algorithm based on the statistically significant regression analysis. Experimental results were analysed in terms of best solutions found so far, mean and standard deviation on both the total cost and execution time to converge to the optimum. Recommended level settings of parameters were applied for all selected problem sizes that can be used as a guideline for future applications. This is to promote ease of use of the metaheuristics in real life problems. From the numerical results on the small problems, the approximated dynamic programming algorithms based on the initial solutions from simulated annealing algorithm performed better on average. However, when the problem sizes were increased the ADPA-I based on the bee colony optimisation algorithm were superior, on average.

The best so far solutions from the ADPA-II approached the optimum closer when compared. There also were several hybrid techniques to develop a better solution by starting with the best solution from one method, and then uses it as an initial solution for others. However, the quality measures were not improved significantly. Finally, various experimental designs can be modified by using replicated orthogonal array or fractional factorial design for improving parameter levels that are related to quality of solutions in each problem instead of using only the recommended parameter levels from the sample problem throughout. It is also interesting to investigate the behaviour of other metaheuristic algorithms in these tested problems. Hence, the numerical results focused on the quality of total cost merely. Further research could consider more on the computation time in forms of the desirability function of dual responses. These requirements can be modified to make it better suited to practical applications.

Acknowledgments— The first author wishes to thank the Faculty of Engineering, Thammasat University for the financial support.

References

Aungkulanon, P, Phruksaphanrat, B and Luangpaiboon, P (2012). Harmony Search Algorithm with Various Evolutionary Elements for Fuzzy Aggregate Production Planning. Lecture Notes in Electrical Engineering 110: 189-201.

Dunker, T, Randons, G and Westkamper, E (2005). Combining Evolutionary Computation and Dynamic Programming for Solving a Dynamic Facility Layout Problem. European Journal of Operational Research 165: 55-69

Granville, V, Krivanek, M and Rasson, JP (1994). Simulated annealing: A proof of convergence. IEEE Transactions on Pattern Analysis and Machine Intelligence 16 (6): 652–656

Karaboga, D and Basturk, B (2007a). A Powerful And Efficient Algorithm For Numerical Function Optimisation: Artificial Bee Colony (ABC) Algorithm. Journal of Global Optimisation 39: 459-471

Karaboga, D and Basturk, B (2007b). Artificial Bee Colony (ABC) Optimisation Algorithm for Solving Constrained Optimisation Problems. LNCS: Advances in Soft Computing: Foundations of Fuzzy Logic and Soft Computing 4529/2007: 789–798.

Karaboga, D and Basturk, B (2008). On The Performance of Artificial Bee Colony (ABC) Algorithm. Applied Soft Computing 8(1): 687–697

Kirkpatrick, S, Gelatt, CD and Vecchi, MP (1983). Optimisation by Simulated Annealing. Science 220 (4598): 671–680

Lacksonen, TA and Enscore, EE (1993). Quadratic Assignment Algorithms for the Dynamic Layout Problem. International Journal of Production Research 31(3): 503-517

Luangpaiboon, P (2011). Noisy Response Surface Optimisations via Harmony Search Algorithm and its Hybridisations. Journal of Applied Operational Research 3(1): 46-57

Luangpaiboon, P and Sermpattarachai P (2007). Response Surface Optimisation via Steepest Ascent, Simulated Annealing and Ant Colony Optimisation Algorithms. Proceedings of the DSI Int'l Conference, Thailand

Mladenovic, N and Hansen, P (1997). Variable Neighbourhood Search. Computers & Operations Research 24: 1097-1100

Muenvanichakul, S. and Charnsethikul, P. (2010). Two reformulations for the Dynamic Quadratic Assignment Problem. Journal of Mathematics and Statistics 6 (4): 449-453

Triwate, P and Luangpaiboon, P (2010). Bees Algorithm for Dynamic Multi-zone Dispatching in Truck Load Trucking. Proceedings of the IEEE International Conference on Industrial Engineering and Engineering Management, Macao, China

Urban, TL (1993). A Heuristic for the Dynamic Facility Layout Problem. IIE Transaction 25: 57-63