Applying Memetic Algorithm in Multi-Objective Resource Allocation among Competing Projects

Angela H.L. Chen
Department of Industrial Engineering and Management, Yuan Ze University, Taoyuan, Taiwan
Department of Finance, Nanya Institute of Technology, Taoyuan, Taiwan
Email: achen@nanya.edu.tw, angela@saturn.yzu.edu.tw

Chiuh-Cheng Chyu
Department of Industrial Engineering and Management, Yuan Ze University, Taoyuan, Taiwan
Email: iehshsu@saturn.yzu.edu.tw

***Abstract*—This paper presents a memetic algorithm for solving project resource allocation problems, where the resource requirements of each project concern numbers of monetary units and never exceeds the amount of capital available. Our objectives are to obtain the best overall result for which returns are maximized and costs are minimized. Such problem, considered as a multi-objective optimization, is too complex to be solved by exact methods. In the proposed MA, the population generated by the crossover or the mutation operator is further improved by a local search method. The approximated Pareto front is updated using all new solutions generated. The performance of MA is demonstrated via an instance, consisting of six projects, and 120 units of capitals, and compared with the reference Pareto front found by executing the exhaustive method. Based on both solution quality and CPU time, the results of such comparison have proved our MA to be an effective optimal method in the multi-objective resource allocation problems (MORAP).**

***Index Terms*—multi-objective optimization, resource allocation, memetic algorithm**

I. Introduction

The funding of any specific project is tied to the organizational capital and is important in transforming goals into both plans and actions. When an organization is involved in several projects competing for the same monetary resource, it has to evaluate alternative ways of distributing such resource across different projects. Thus, every organization is faced with several monetary allocation decisions permeating various levels of management. Typical challenges faced by executives in charge of allocating funding can be summarized as follows: How should a limited amount of resources get distributed among a large number of potential projects? How should benefits be determined by multiple and often conflicting objectives? What should allocation?

Academically, most studies have focused on the scheduling rather than on the resource allocation optimization. Moreover, in most cases, single objective problems are often discussed; however, most real-world optimization problems are multi-objective in nature. In this paper we examine decisions on monetary allocation in a multi-project environment. The firm executive has to decide on the amount of capital that should be invested in each of the projects to optimize its revenue (*i.e.* profit) and minimize its expenditure (*i.e.* cost).

Like previously mentioned, the conventional resource allocation problem only considers the single objective function. However, in most real-world optimization scenarios, the objectives are incommensurable and conflicting. In such cases, there is no single optimal solution but rather a set of alternative solutions. For this reason, some works approach this multi-objective decision problem by aggregating the objectives into a single one. Other works concentrate on the approximation of the Pareto optimal set. No matter what, this type of problems has two goals: one is to find a set of solutions as close as possible to the Pareto-optimal front; the other is to find a set of solutions as diverse as possible. While the first goal is mandatory, the second one is entirely problem specific.

There exist various methods to solve the multi-objective decision problem. Linear programming algorithms [1,2] have been applied to the resource allocation problem (RAP) with its objectives represented by a linear function of discrete and continuous decision variables. Dynamic programming algorithms [3–5] have also been applied to solve the RAP but with different problem formulations. For instance, Basso and Peccati [4] proposed an efficient pruning procedure implemented in a dynamic programming algorithm. In addition, branch and bound algorithms [6,7] have all been applied to solve the RAP sub-problems. Nevertheless, when the problem sizes become large, generating the Pareto optimal set can be computationally expensive and mathematical programming techniques usually do not guarantee to deliver exact solutions with reasonable time.

As for the previously mentioned reason, researchers have sought an alternative to overcome this obstacle. They apply various meta-heuristic algorithms since this approach has been known to obtain feasible and near-optimal solutions within reasonable computational expense at large problem sizes. Publications on Genetic

doi:10.4304/jsw.5.8.802-809

algorithm (GA) [8–11], ant colony optimization (ACO) [12–14], and particle swarm optimization (PSO) [15–17] have shown their effectiveness in solving well-known NP-hard problems, including project planning and activity scheduling under resource constraints. Dai *et al.* [7] applied a genetic algorithm to the single objective resource allocation problem (SORAP); however, Osman *et al.* [9], Lin and Gen [10,11] used for solving multi-objective resource allocation problem (MORAP). Yin and Wang [16] proposed a particle swarm optimization approach to the nonlinear type of problem. Inspired by previous studies, this paper intends to present one of effective metaheuristics – memetic algorithm (MA), for solving monetary allocation in project management.

The term *Memetic Algorithms*, recognized as a hybrid genetic algorithm, first appeared in the computing literatures [18,19], has differentiated itself from the genetic algorithm classification. Unlike conventional genetic algorithms which emulate biological evolution, memetic algorithms imitate cultural evolution. Their conceptual difference is such that GA forbids individuals to choose, modify and improve their own genes in its natural process whereas MA allows individuals to intentionally acquire, modify, and improve their memes [19,20]. In other words, though GA is capable of finding good regions in the search space, the exploitation of these good regions needs more attention in which GA is not designed for. Alternatively, MA has the local search procedure employed every time some new solutions are generated and emphasizes on local optimal solutions. Our literature survey shows that memetic algorithms are very effective and efficient in many hard combinatorial optimization problems [20–23], but has not yet applied in multi-objective resource allocation problems.

The remaining paper is organized in such a way that the mathematical formulation on the multi-objective resource allocation problem (MORAP) is introduced in Section 2. Then the proposed methodological framework for the MORAP is described in Section 3. In Section 4 an illustrative application along with the experiments of corresponding parameter calibration is discussed. Finally, the concluding remarks are given in Section 5.

II. Problem Formulation

Since multi-objective optimization problems involve multiple, often conflicting objectives which are to be minimized or maximized, our resource allocation problem is formulated with multiple objectives in the following scenarios. For each of J independent projects, certain monetary amount x_{ik}, i=1, 2,…, J (called project capital) has to be assigned to project i to maximizing the overall return but minimize the total cost in this multi-objective optimization model. Furthermore, upon implementing allocation strategies in project i, p_{ik} is assumed to be the expected profit and c_{ik} denotes the cost of project i given monetary amount k respectively.

$$\max \sum_{i=1}^{J}\sum_{k=0}^{R_i} p_{ik} k_{ik} \ , \tag{1}$$

$$\min \sum_{i=1}^{J}\sum_{k=0}^{R_i} c_{ik} x_{ik} \ , \tag{2}$$

s.t.

$$\sum_{i=1}^{J}\sum_{k=0}^{R_i} j x_{ik} \le \sum_{i=1}^{J} R_i \ , \tag{3}$$

$$\sum_{k=0}^{R_i} x_{ik} = 1 \text{ for each } i = 1,2,...,J \ , \tag{4}$$

$$x_{ik} = \begin{cases} 1, & \text{if } k \text{ amount of money are assigned to project } i \\ 0, & \text{otherwise} \end{cases} . \tag{5}$$

The objective function (1) represents the expected profit associated with any possible capital allocation among projects. The objective function (2) justifies the total execution cost incurred by the allocation decision. The feasible allocations are specified by constraints (3)-(5). Constraints (3) insure that the final resource allocated to the projects do not exceed the upper bound of the available capital. Constraints (4) and (5) assure that exactly one allocation strategy is implemented for each project.

III. Methodolody

In this section, we propose a memetic algorithm to perform a monetary resource allocation (*i.e.* capital allocation) among competing projects. To find the best solution, our general framework makes use of the Pareto optimality concept. The study aims to obtain a set of non-dominated and sufficiently diversified solutions to cover the entire Pareto frontier. For that, our MA begins with **InitializePop** generating initial population which is then improved by **LocalSearch**. Starting from **Repeat** command, the search procedure goes through three main steps, i.e. **Selection**, **Crossover** +**LocalSearch**, and **Mutation** + **LocalSearch**, and completes when the stopping criterion is satisfied. That is, the termination occurs when the maximum number of generations is reached. The MA framework is shown below and the details are discussed accordingly.

InitializePop
LocalSearch
UpdateParetoFront
Repeat
  Selection
  Crossover
  LocalSearch
  UpdateParetoFront
  Mutation
  LocalSearch
  UpdateParetoFront
Until(the stopping criterion is satisfied)

A. Initial Population Generation

For the initial population generation, a pre-specified number of solutions (*PS*) are randomly generated. To accommodate the constraint (3) in Section 2, where the number of assigned monetary resource allocated to the projects do not exceed the upper bound of the available capital, a dummy task having zero cost and zero profit is created in the encoding. Then the solution encoding for a six-project example is shown in Figure 1 below. When the available amount of capital is said to be 15 monetary units, such amount will then be randomly assigned to each project including the dummy one. The process continues until all monetary units have been assigned. As a result, the final solution in Figure 1 consists of three monetary units for the first project, four for the second project, two for the third project, and so on.

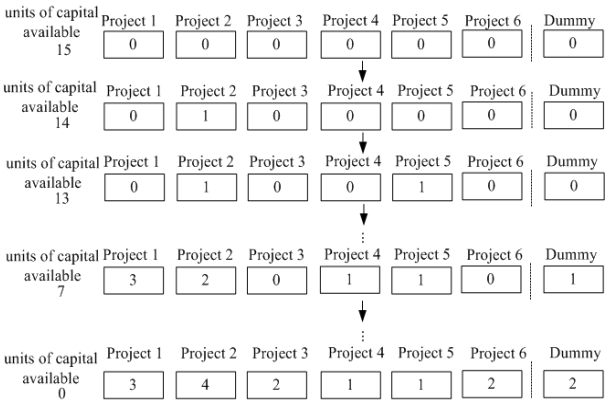

Figure 2. An example of the initial solution encoding and generation process.

B. Local Search

Recognized as an evolutionary-based search augmented by hybridizing with one or more phases of local search procedures, our local search is designed as follows: First, the algorithm randomly generates a number d from the set of {1, 2,…, d_{max}} where d represents the units of capital transferred from one project to another and d_{max} denotes a pre-specified upper bound for the capital change. Next, a complete local search will be conducted to the chosen solutions. Since the local search is embedded with initial population generation, crossover, and mutation, the chosen solutions can be either the entire initial population, the offspring generated in crossover operation, or the mutated solutions by the mutation operation, respectively. Once the neighboring solutions are constructed, all of them will be used to update the approximated Pareto front, *i.e.* the non-dominated set obtained by MA. Figure 2 demonstrates the neighboring solutions of a six-project example when the value of d is set to two. Looking at neighboring solution 1, two monetary units get transferred from project 1 to project 2.

Similarly, for neighboring solution 2, we have two monetary units transferred from project 1 to project 3. Note that in the local search mechanism, the amount of monetary unit in the dummy project can also be modified; in the case of neighboring solution *m*, two monetary units get transferred from project 3 to the dummy. In doing so, we can obtain more flexibility of increasing or decreasing monetary resources among competing projects. However, subtracting or adding d monetary units may cause the amount of project capitals to be either less than zero or greater than the upper bound limit. When this situation does occur, our local search procedure then skips these particular neighboring solutions and precedes the followings.

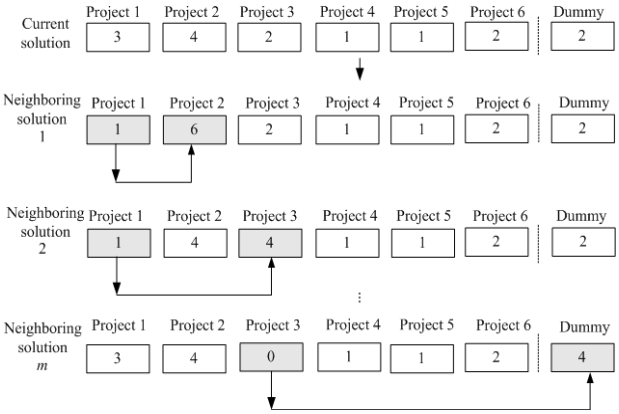

Figure 1. An example of neighboring solutions in local search.

C. Selection and Crossover Operations

The first operation in the search procedure begins with the selection, choosing two parents for the consecutive crossover operation. Since the study investigates on multi-objective resource allocation problem, the conventional biased roulette wheel cannot be applied here though this technique is most commonly applied in single objective GA approach. So instinctively at each selection step, we randomly select two non-dominated solutions from the current approximated Pareto front as parents.

After the selection of parent solutions, the uniform crossover operation is performed next and repeated for *PS*/2 times until *PS* parents are collected. Figure 3 illustrates the uniform crossover and the process description is as follows: First, assuming the total available capital are 15 monetary units and six projects are under consideration, the procedure picks the value of assigned capital units of project i from one of the parents with equal probability. Note that, the dummy projects for both parents are not considered when the crossover operation is performed; however, once the proto-offspring are constructed, the amounts of unassigned capitals are determined for the dummy project.

The following is an example in offspring 1. The capital of projects 1, 2 and 4 are chosen from parent 1 while the amount of capital allocation in projects 3, 5 and 6 are from parent 2 (as shown by the shaded blocks). The amount of capital units (*i.e.* 7) for the dummy project is

determined by subtracting the sum of all six project's capitals (*i.e.* 1 + 3 + 0 + 1 + 1 + 2 = 8) from the total available capital. In a similar manner another offspring can be generated. When the total number of capitals assigned exceeds the available monetary amount, we say such solutions are infeasible, and the crossover operation will be repeated until enough offspring are generated.

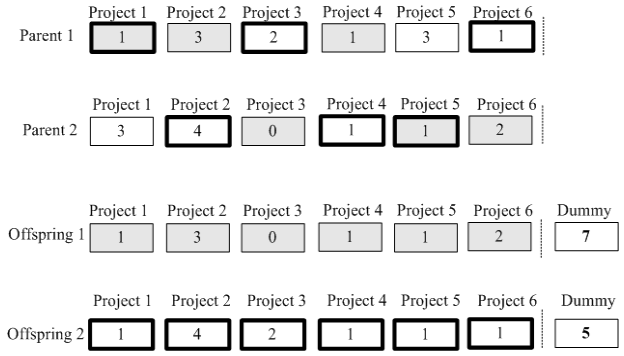

Figure 3. An example of the uniform crossover operation.

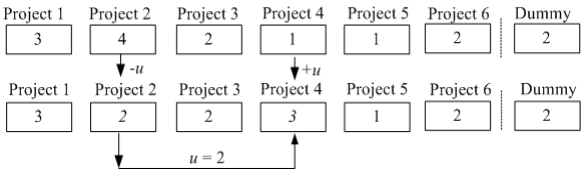

Figure 4. An example of mutation operation.

D. Mutation Operation

The last search step of the memetic algorithm is the mutation operation, applied to all non-dominated solutions in the current approximated Pareto front. The study applies such operation to prevent the solutions from trapping at the local optima. To do so, we first randomly select two projects, say projects 2 and 4 as shown in Figure 4; as well, we randomly determine a number of monetary units u allowable for capital exchange. The amount u must be within a range of $\{1, 2,\ldots, u_{max}\}$. In this case, say 2 monetary units. Thereafter, 2 monetary units will be taken away from project 2 and be reassigned to project 4 (as illustrated in Figure 4). The dummy project can also be selected for more alternatives.

IV. Result and Discussion

This section will demonstrate the performance of MA by testing on an instance which consists of six projects, and 120 units of capitals. Therefore, the upper bound of units (R_i) for each project is set to 20. The details of expected profit (p_{ij}) and expected cost (c_{ij}) for the instance are summarized in Tables 1 and 2 respectively. Our MA is coded on Borland C++ Builder 6.0, and run on a PC with Intel® Core™ Quad CPU Q6600 @ 2.40 GHz and 2.39 GHz and memory is 2.0 GB. To collect data for statistical analysis, MA is run five times using different random number seeds.

A. Performance Measurements

The performance of MA is evaluated mainly based on two measurements: the Accuracy ratio (AR) [24] and the $D1_R$ value [25]. For the determination of these measures, a reference Pareto front denoted by X_{ref} must first be obtained. This is achieved by an exhaustive method, which enumerates all feasible solutions and then finds the true non-dominated solutions for the front. Then, the numerator of the AR is calculated as in equation (6) where X_{MA} denotes the approximated Pareto front obtained by the MA, a_i a binary variable accounts for the number of non-dominated solutions falls in both X_{MA} and X_{ref}, and N_{MA} represents the number of non-dominated solutions in the set of X_{MA}. The denominator of the accuracy ratio is defined as the number of non-dominated solutions in the reference Pareto front denoted by $|X_{\text{ref}}|$. It is obvious that the accuracy ratio is the higher the better.

$$\text{AR}=\frac{\sum_{i=1}^{N_{\text{MA}}} a_i}{|X_{\text{ref}}|} \quad \text{where } a_i=\begin{cases}1 & \text{if a solution in } X_{\text{MA}} \in X_{\text{ref}} \\ 0 & \text{otherwise}\end{cases}. \tag{6}$$

The second measure, $D1_R$, proposed by Ishibuchi *et al.* [25] indicates the average minimum distance from each point in the reference Pareto front X_{ref} to the approximated Pareto front X_{MA}. In equations (7) and (8), x^* denotes a solution in X_{ref} and x represents a solution in X_{MA}. In addition, $f_c(.)$ indicates the cost objective function value of a solution, and $f_p(.)$ is the profit objective function value of a solution. Therefore, the Euclidean distance (d_{x^*x}) between the x^* and x in the objective function space can be calculated consequently. Since this measure is used to evaluate both the diversity and convergence of a multi-objective algorithm [25], the smaller value of the $D1_R$ measure, the better.

$$D1_R(X_{\text{MA}})=\frac{1}{|X_{\text{ref}}|}\sum_{x^*\in X_{\text{ref}}}\min\left\{d_{x^*x}\mid x\in X_{\text{MA}}\right\}. \tag{7}$$

$$d_{x^*x}=\sqrt{\left(f_p(x^*)-f_p(x)\right)^2+\left(f_c(x^*)-f_c(x)\right)^2}. \tag{8}$$

B. Parameter Calibration

The analysis on our computational results starts with the parameter calibration. Several parameters in the proposed MA need to be tuned up to get the best results, such as the u_{max} value in the mutation operator (that controls the range of mutation), the d_{max} value in the local search (that controls the size of neighborhood), the *PS* value in the initial population (that represents the initial population size and number of offspring in the crossover operator), and the maximum number of generations for the stopping criterion. To conduct the experiments, the default parameter values are set to u_{max} = 3, d_{max} = 10, *PS*

= 20, and the maximum number of generations = 50 unless the parameters are specified.

The first experiment investigates the effect of mutation range, where we set u_{max} to be 1, 3, 5, and 7. The relationship between the logarithmic value of $D1_R$ and the mutation range is shown in Figure 5. Our observation indicates when the u_{max} value increases from one to three, the performance improves while the improvement slows down when the u_{max} value becomes five. Moreover, the improvement stagnates in the u_{max} values of five and seven. Figure 6 summarizes the box plot of another measure – the accuracy ratio. A similar conclusion can also be drawn because the accuracy ratio reaches the top when the mutation range is five or seven.

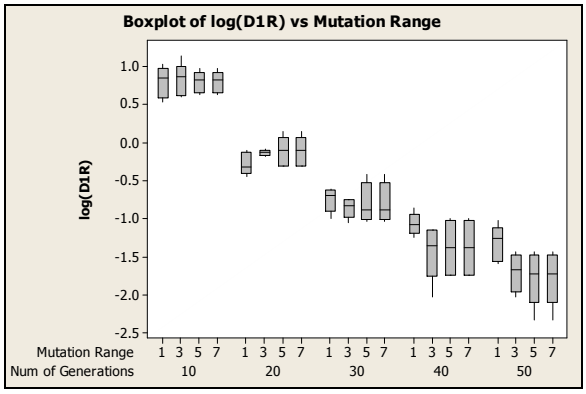

Figure 8. Box plot of mutation range versus log($D1_R$) over different generations..

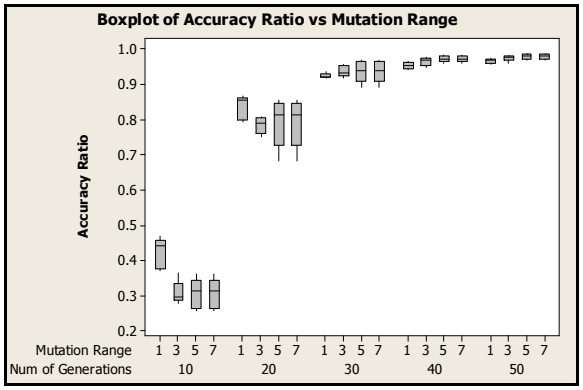

Figure 9. Box plot of mutation range versus the accuracy ratio over different generations.

Our investigation continues with the range of local search, *i.e.* the upper bound for unit change. Three values of d_{max} (*i.e.* 5, 10 and 15) are tested respectively. The relationship of log($D1_R$) value and d_{max} over different generations is summarized in Figure 7. Our finding has indicated that the best performance is found when d_{max} = 10 with lower variance and lower log($D1_R$) value which represents a more consistent performance. Comparison on the accuracy ratios shown in Figure 8, the performance of d_{max} = 10 is slightly better than the case of d_{max} = 15, and outperforms the setting of d_{max} = 5.

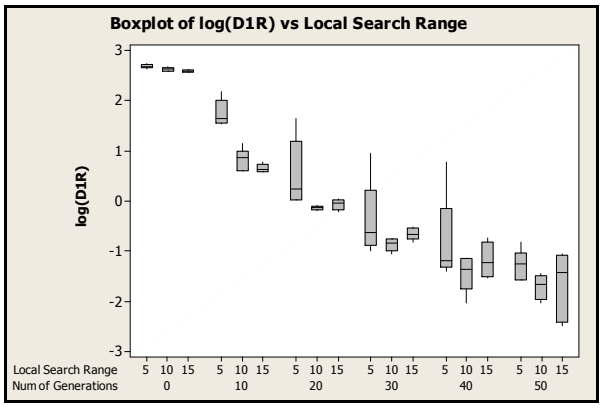

Figure 6. Box plot of local search range versus log($D1_R$) over different generations.

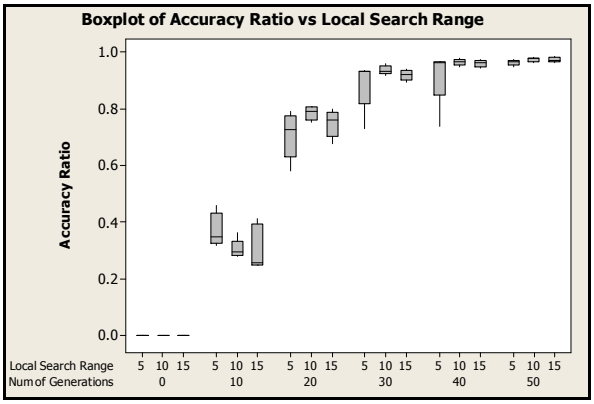

Figure 5. Box plot of local search range versus the accuracy ratio over different generations.

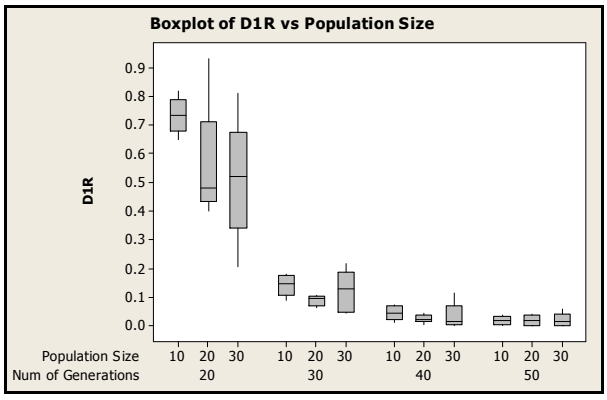

Figure 7. Box plot of population size versus $D1_R$ over different generations.

Next, we are at exploring the effect of population size (*PS*), and once again, three *PS* values (*i.e.* 10, 20 and 30) are carried out. Figure 9 indicates the relationship of population size and $D1_R$ over generations. Clearly, the performance of *PS* = 20 is the most consistent one among all three settings while the largest population size *PS* = 30 has the largest variance. As shown in Figure 10, the comparison on the accuracy ratio verifies the superiority of *PS* = 20 again.

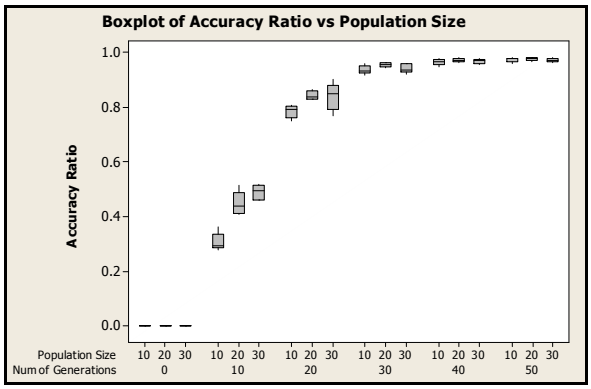

Figure 10. Box plot of population size versus the accuracy ratio over different generations.

C. Performance of the Memetic Algorithm

From the experiments above, the best parameter combination can be fixed as u_{max} = 5, d_{max} = 10 and PS = 20, so a convergence analysis is then conducted to determine the stopping criterion of MA. The average $\log(D1_R)$ value and the average accuracy ratio of MA over five runs at the increment of 10 generations are illustrated in Figure 9. As the number of generations increases, the $\log(D1_R)$ value drops to a very small value of 0.022 and the average ratio reaches 97.9% of accuracy at 50 generations.

The reference Pareto front is found by executing the exhaustive method, and a total number of 305 non-dominated solutions are identified. Note that the total number of feasible solutions in this instance is 8,576,612 and the CPU time needed for running the exhaustive method is 4,865 seconds. However, as for MA, the average number of evaluations is 283,667 at 50 generations, the CPU time is only 12.36 seconds, and average 298.6 non-dominated solutions over five runs can be found. Based on both solution quality and CPU time, the results of such comparison have verified the potential of our proposed MA in the multi-objective resource allocation problems.

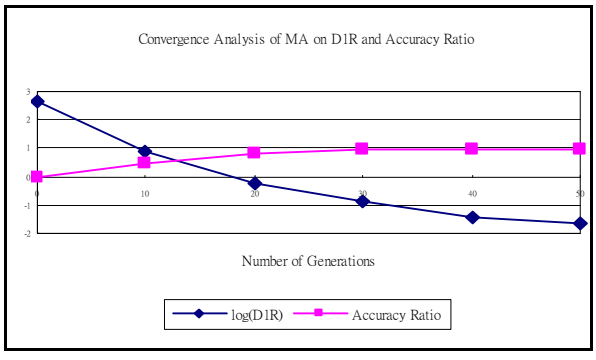

Figure 11. Convergence analysis of MA using the best parameter setting.

V. Conclusion

Inspired by one of real world scenario in which the funding of any specific project is tied to the organizational capital and is important in transforming goals into both plans and actions, our study investigates on monetary resource allocation among competing projects. We say the problem to be one application of resource allocation problems. Though the conventional approach to such resource allocation problem only considers the single objective function, a more applicable approach should be multi-objective. Thus, this study formulates a multi-objective optimization model, *i.e.* maximizing the overall return but minimizing the total cost. Under such model, the monetary requirements of each project never exceed the overall amount of monetary resource available. The approximation of the Pareto optimal set is considered and can be computationally expensive if mathematical programming techniques are applied. To overcome this limitation, our literature survey shows that memetic algorithms are very effective and efficient in many hard combinatorial optimization problems but has not yet applied in multi-objective resource allocation problems. Therefore, we propose a memetic algorithm to the MORAP.

The implementation of MA involves three main steps – **Selection**, **Crossover** + **LocalSearch**, and **Mutation** + **LocalSearch**. Their details and illustrations have been shown in Section 3. The performance of our MA is based on an instance which consists of six projects, and 120 units of capitals, and the parameter calibration where several parameters in the proposed MA are tuned up for the best results. Here, the u_{max} value in the mutation operator is set to be 5, the d_{max} value in the local search is 10, the PS value in the initial population is 20, and the total number of generations is 50. Our computational results are evaluated by two measurements: the accuracy ratio [24] and the $D1_R$ value [25], whose detailed calculations have been explained in Section 4.

The study compares the proposed MA with the exhaustive method in terms of the Pareto fronts. As shown by the experiments earlier, the quality of non-dominated solutions obtained by MA is superior and the distance between the approximated Pareto front and the true Pareto front is small. The relative performance of MA demonstrated its potential to be a very effective and efficient optimal method in the multi-objective resource allocation problems.

References

[1] A.A. Zoltners, P. Sinha, "Integer programming models for sales resource allocation," *Manage. Sci.*, vol. 26, pp. 242-260, 1980.

[2] A.A. Stinnett, A.D. Paltiel, "Mathematical programming for the efficient allocation of health care resource," *J. Health Econ.*, vol. 15, pp. 641-653, 1996.

[3] K.K Lai., L. Li, "A dynamic approach to multiple-objective resource allocation problem," *Eur. J. Oper. Res.*, vol. 117, pp. 293-309, 1999.

[4] A. Basso, L.A. Peccati, Optimal resource allocation with minimum activation levels and fixed costs, *Eur. J. Oper. Res.*, vol. 131, pp. 536-549, 2001

[5] D. Morales, F. Almeida, F. Garcia, J.L. Roda, C. Rodriguez, Design of parallel algorithms for the single resource allocation problem, *Eur. J. Oper. Res.*, vol. 126, pp. 166-174, 2000.

[6] J.H. Holland, *Adaptation in Natural and Artificial Systems*, Ann Arbor, MI: Univ. Michigan Press, 1975.

[7] Y.S. Dai, M. Xie, K.L. Poh, B. Yang, "Optimal testing-resource allocation with genetic algorithm for modular software systems," *J. Syst. Software*, vol. 66, pp. 47-55, 2003.

[8] X. Yang, Z. Yang, Z. Shen, J. Li, "Node ordinal encoded genetic algorithm for the optimal allocation of water resources," *Prog. Nat. Sci.*, vol. 20, pp. 1019-1034, 2004.

[9] M.S. Osman, M.A. Abo-Sinna, A.A. Mousa, "An effective genetic algorithm approach to multi-objective resource allocation problems," *Appl. Math. Comput.*, vol. 163, pp. 755-768, 2005.

[10] C.M. Lin, M. Gen, "Multiobjective resource allocation problem by multistage decision-based hybrid genetic algorithm," *Appl. Math. Comput.*, vol. 187, pp. 574-583, 2007.

[11] C.M. Lin, M. Gen, "Multi-criteria human resource allocation for solving multistage combinatorial optimization problems using multiobjective hybrid genetic algorithm," *Expert Syst. Appl.*, vol. 34, pp. 2480-2490, 2008.

[12] M. Dorigo, Optimization, Learning, and natural algorithms, Ph.D. Thesis, Dip. Elettronica e Informazione, Politecnico di Milano, Italy, 1992.

[13] P.Y. Yin, J.Y. Wang, "Ant colony optimization for the nonlinear resource allocation problem," *Appl. Math. Comput.*, vol. 174, pp. 1438-1453, 2006.

[14] S.K. Chaharsooghi, A.H. Meimand Kermani, "An effective ant colony optimization algorithm for multi-objective resource allocation problem," *Appl. Math. Comput.*, vol. 200, pp. 167-177, 2008.

[15] J. Kennedy, R.C. Eberhart, "Particle swarm optimization," in: *Proceedings of the IEEE International Conference on Neural Networks*, vol. 5, pp. 1942-1948, 1995.

[16] P.Y. Yin, J.Y. Wang, "A particle swarm optimization approach to the nonlinear resource allocation problem," *Appl. Math. Comput.*, vol. 183, pp. 232-242, 2008.

[17] N. Aravindhu, S. Siva Sathya, "Swarm intelligence based genetic algorithm for multi-objective optimization," *Int. J. Recent Trends Eng.*, vol. 1, pp. 211-214, 2009.

[18] P. Moscato, *On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms*, Caltech Concurrent Computation Program, Report No. 826, 1989.

[19] P. Moscato, "Memetic algorithms: a short introduction," in D. Corne, F. Glover, and M. Dorigo (eds.), *New Ideas in Optimization*, McGraw-Hill, pp. 219-234, 1999.

[20] N Krasnogor., J. Smith, "A memetic algorithm with self-adaptive local search: TSP as a case study," in: *Proceedings of the Genetic and Evolutionary Computation Conference* (GECCO-2000), Morgan Kaufman, pp. 987-994, 2000.

[21] P. Merz, B. Freisleben, "Memetic algorithms for the traveling salesman problem," *Complex Syst.*, vol. 14, pp. 297-345, 2001.

[22] A. Yin, "A new neighborhood structure for the job shop scheduling problem," In *Proceedings of the 3rd International Conference on Machine Learning and Cybernetics*, August 26-29, Shanghi, vol. 4, pp. 2098-2101, 2004.

[23] W.S. Herroelen, E. Demeulemeester., P. Van Dommelen, "Project network models with discounted cash flows: A guided tour through recent developments," *Eur. J. Oper. Res.*, vol. 100, pp. 97-121, 1997.

[24] E. Zitzler, L. Thiele, M. Laumanns, C.M. Fonseca, V.G. Fonseca, "Performance assessment of multi-objective optimizers: an analysis and review," *IEEE T. Evolut. Comput.*, vol. 7, pp. 117-132, 2003.

[25] H. Ishibuchi, T. Yoshida, T. Murata, "Balance between genetic search and local search in memetic algorithms for multi-objective permutation flow shop scheduling," *IEEE T. Evolut. Comput.*, vol. 7, pp. 204-223, 2003.

Angela H.L. Chen is currently pursuing her Ph.D. degree in the Department of Industrial Engineering and Management at Yuan-Ze University, Taoyuan, Taiwan. She received her Master in Business Administration at University of Houston, Victoria, Houston, USA in 1996. She completed her bachelor's degree in Bio-Resource Engineering from University of British Columbia, Vancouver, B.C., Canada, in 1994. She has been a college instructor in the Department of Finance at Nanya Institute of Technology in Taiwan. Her research interests include resource allocation, project management, and portfolio management.

Chiuh-Cheng Chyu received his bachelor's degree in Mechanical Engineering from National Taiwan University in Taiwan in 1977. Then he completed his master degree in Managerial Science from Stevens Institute of Technology, NJ, USA, in 1982, and his Ph.D. degree in Industrial Engineering from University of California, Berkley, USA, in 1990. He has been an Associated Professor of Industrial Engineering and Management at Yuan Ze University in Taiwan since 1990. He has been an international keynote speaker and tutorial lecturer, and has consulted for industry and government. His areas of research include Bayesian decision analysis, network analysis, and operations research.

TABLE I.
EXPECTED PROFIT P_{IJ} OF INSTANCE

Projects	**Units of Capitals (*j*)**																				
(*i*)	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	22	32	48	56	74	87	96	109	112	120	133	146	147	154	173	189	208	216	226	240	251
2	36	50	65	76	77	83	102	120	123	130	146	157	174	194	212	221	226	240	254	269	283
3	37	53	51	62	75	81	83	99	104	114	119	121	128	130	144	153	159	178	179	181	186
4	32	42	61	76	78	93	96	111	118	129	141	151	167	158	171	162	154	172	188	197	200
5	46	53	67	87	93	111	116	132	141	153	168	188	189	199	194	198	192	199	208	223	230
6	50	70	84	90	91	97	107	111	105	120	118	129	131	132	141	146	153	156	157	162	166

TABLE II.
EXPECTED COST C_{ij} OF INSTANCE

Projects	**Units of Capitals (*j*)**																				
(*i*)	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	40	33	46	63	70	72	73	77	91	98	100	113	117	126	146	159	161	179	184	204	198
2	28	44	50	68	81	93	112	122	140	157	177	189	196	213	219	233	244	259	252	263	278
3	45	46	47	48	51	53	61	81	82	96	112	113	110	122	126	134	138	139	131	139	152
4	28	45	62	70	89	94	97	113	118	135	155	163	182	183	188	205	222	239	241	246	266
5	48	56	64	79	82	96	101	98	118	122	134	141	159	169	188	181	200	204	211	228	244
6	43	63	69	77	80	95	101	121	113	126	134	136	149	153	165	180	198	204	201	202	212