Optimization of Location Allocation of Web Service using non-dominated sorting algorithm(NSGA-II)

Boxiong Tan tanboxi@ecs.vuw.ac.nz
Department of Science and Engineering, Victoria of Wellington, Wellington, Zip, New Zealand

Hui Ma hui.ma@ecs.vuw.ac.nz
Department of Science and Engineering, Victoria of Wellington, Wellington, Zip, New Zealand

Abstract

The abstract goes here. It should be about 200 words and give the reader a summary of the main contributions of the paper. Remember that readers may decide to read or not to read your paper based on what is in the abstract. The abstract never contains references.

Keywords

multi-objective, NSGA-II, genetic algorithms, evolutionary programming

1 Introduction

Web Services are considered as self-contained, self-describing, modular applications that can be published, located, and invoked across the Web Ran (2003). In recent years, web services technology is becoming increasingly popular because the convenience, low cost and capacity to be composed into high-level business processes Aboolian et al. (2009). Because of modularization and open interfaces, Web services facilitate the development of highly customizable and adaptable applications to meet business demand. Furthermore, Web services offer a convenient registration, search and discovery system. e.g. Universal Description, Discovery and Integration (UDDI).

With the ever increasing number of functional similar web services being available on the Internet, the web service providers (WSPs) are trying to improve the quality of service (QoS) to become competitive in the market. QoS also known as non-functional requirements to web services, is the degree to which a service meets specified requirements or user needs Zhou and Niemela (2006), such as response time, security and availability. Among numerous QoS measurements, service response time is a critical factor for many real-time services, e.g. traffic service or finance service. Service response time has two components: transmission time (variable with message size) and network latency Johansson (2000). Study Johansson (2000); Jamin et al. (2001) has shown that network latency is a significant component of service response delay. Ignoring network latency will underestimate response time by more than 80 percent. Since network latency is related to network topology as well as physical distance Huffaker et al. (2002). The network latency could also vary with the network topology changes. The only way to reduce the network latency is move the service to a location where has smaller network latency to the user center. Hence, the WSPs need to consider which physical location to deploy their services so that it could minimize the cost as well as ensure the QoS.

The Web service location-allocation problem is essentially a multi-objective optimization problem Caramia (2008). Because of the confliction between service quality and deployment cost. Ideally, WSP could deploy their services to each user center in order to provide the best quality. That is, the more services deployed, the better the quality and the higher cost. This problem is considered as an NP-hard due to the fact that the combinatorial explosion of the search space Vanrompay et al. (2008).

Very few researches Aboolian et al. (2009); Sun and Koehler (2006) study this problem. Both studies try to solve the problem by integer linear programming techniques. However, integer programming techniques do not scale well, so that no satisfactory results can be obtained for large-scale datasets.

Evolutionary algorithms (EAs) have been used in solving multi objective optimization problems in recent years. EAs are ideal for solving multi objective optimization problems Desai et al. (2012), since EA works with a population of solutions, a simple EA can be extended to maintain a diverse set of solutions. With an emphasis for moving toward the true Pareto-optimal region, an EA can be used to find multiple Pareto-optimal solutions in one single simulation run Kanagarajan et al. (2008).

Hai Huang et al. (2014) proposed an enhanced genetic algorithm-based approach which make use of the integer scalarization technique to solve the multi-objective problem. The genetic algorithm (GA) is an EA that uses genetic operators to obtain optimal solutions without any assumptions about the search space. This algorithm solve the scalability problem in the dataset, however the integer scalarization technique Caramia (2008) has some disadvantages:

1. The decision maker needs to choose an appropriate weights for the objectives to retrieve a satisfactorily solution.
2. The algorithm does not produce an uniform spread of points on the Pareto curve. That is, all points are grouped in certain parts of the Pareto front.
3. Non-convex parts of the Pareto set cannot be reached by minimizing convex combinations of the object functions.

Evolutionary multi objective optimization (EMO) methodologies on the other hand, successfully avoid the above mentioned problems and demonstrated their usefulness in find a well-distributed set of near Pareto optimal solutions Aboolian et al. (2009). Non-dominated sorting GA (NSGA-II) Deb et al. (2002), Strength Pareto Evolutionary Algorithm 2 (SPEA-2) Deb et al. (2005) have become standard approaches. Some schemes based on particle swarm optimization approaches Elhossini et al. (2010); Huang et al. (2006) are also important. Among numerous EA approaches, NSGA-II is one of the most widely used methods for generating the Pareto frontier. NSGA-II implements elitism and uses a phenotype crowd comparison operator that keeps diversity without specifying any additional parameters Deb et al. (2006). In our approach, we apply a modified version of NSGA-II since the web service location-allocation is a discrete problem.

In this paper we consider the problem faced by a WSP who has existing facilities but wishes to use the collected data to re-allocate their services in order to maximum their profit. The WSP must decide on facility locations from a finite set of possible locations. In order to make the decision, the WSP must first analyze the data which were collected from current services. The collected data should includes the records of invocation from each unique IP address. Therefore, based on these data, the WSP could summarize several customer demand concentrated at n discrete nodes Aboolian et al. (2009), namely user centers. We assume the WSP has already done this step and list of user centers and candidate service deployment locations are given. In addition to decide which location to re-allocate the services, a dataset which contains the network latency between demand user center and candidate location are critical. The WSP could collect the data or use existed dataset Zheng et al. (2010); Zhang et al. (2011). Then, the service provider could use the algorithm which proposed by this paper, to select an optimal plan based on their funds. The algorithm will produce a near optimal solution which indicate the services deployment locations with a minimum cost and best service quality. The main objectives are:

- To model the web service location-allocation problem so that it can be tackled with NSGA-II
- To develop a modified NSGA-II approach for the web service location-allocation problem
- To evaluate our approach by comparing it to a GA approach which use integer scalarization technique.

2 Problem Description

2.1 Model formulation

The problem is to determine which facility locations that could maximus WSPs profit as well as ensure low network latency. Let $S = \{1, 2, ..., s\}$ be the set of services. We assume that the demand for service is concentrated at i demand nodes $I = \{1, 2, ..., i\}$. Let $J = \{1, 2, ..., j\}$ be the set of j candidate facility locations. To model the service location-allocation problem we use four matrices: service network latency matrix L, service location matrix A, service invocation frequency matrix F and cost matrix C.

The server network latency matrix $L = [l_{ij}]$, is used to record network latency from user centers to candidate locations, where l_{ij} is a real number denotes the network latency from user center i to candidate location j. These data could be retrieved from implementing a network latency experiment or using existed datasets Zheng et al. (2010); Zhang et al. (2011).

$$L = \begin{array}{c} \\ i_1 \\ i_2 \\ i_3 \end{array} \begin{array}{c} \begin{array}{cccc} j_1 & j_2 & j_3 & j_4 \end{array} \\ \begin{bmatrix} 5.09 & 2.37 & 4.01 & 3.9 \\ 0.8 & 2.9 & 3.2 & 1.2 \\ 2.74 & 1.2 & 5.3 & 0.95 \end{bmatrix} \end{array}$$

The service location matrix $A = [y_{sj}]$ represents the actual service location-allocation, where y_{sj} is a binary value (i.e., 1 or 0) shows whether a service s is deployed in candidate location j or not. We use the service location matrix A as the representation of chromosome that evolve itself during the progress.

$$A = \begin{array}{c} \\ s_1 \\ s_2 \\ s_3 \end{array} \begin{array}{c} \begin{array}{cccc} j_1 & j_2 & j_3 & j_4 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix} \end{array}$$

The service invocation frequency matrix $F = [f_{is}]$, is used to record services invocation frequency from user centers, which f_{is} is an integer that indicate the number of invocation in a period of time from user center i to service s. e.g. 120 invocations per day from user center i_1 to s_1.

$$F = \begin{array}{c} \\ i_1 \\ i_2 \\ i_3 \end{array} \begin{array}{c} \begin{array}{ccc} s_1 & s_2 & s_3 \end{array} \\ \begin{bmatrix} 120 & 35 & 56 \\ 14 & 67 & 24 \\ 85 & 25 & 74 \end{bmatrix} \end{array}$$

The cost matrix $C = [c_{sj}]$, is used to record the cost of deployment of services from candidate locations, which c_{sj} is an integer that indicate the cost of the deployment fee from a candidate location. e.g 130 $ to deploy s_1 from j_1.

$$C = \begin{array}{c} \\ s_1 \\ s_2 \\ s_3 \end{array} \begin{array}{c} \begin{array}{cccc} j_1 & j_2 & j_3 & j_4 \end{array} \\ \begin{bmatrix} 130 & 80 & 60 & 68 \\ 96 & 52 & 86 & 78 \\ 37 & 25 & 54 & 46 \end{bmatrix} \end{array}$$

Consider the following key modeling assumptions:

1. The new WSP decides where to locate his facilities regardless if there is existed functional similar services from other WSPs.
2. This choice is made only consider two factors: total network latency and total cost.
3. We assume a fixed customer allocation policy for WSPs. In practice, Web Services typically offer clients persistent and interactive services, which often span over multiple sessions. Therefore, a dynamic reallocation scheme is not practical as it may disrupt the continuity of the services.

2.2 Discreted NSGA-2 algorithm

NSGA-2 belong to the larger class of evolutionary algorithms (EAs), which generate approximate solutions to optimization and search problems by using techniques inspired by the principles of natural evolution: selection, crossover and mutation.

The steps involved in the solution of optimization problem using NSGA-II are summarized as follows.

- Population initialization
- Non-dominated sort
- Crowding distance
- Selection
- Genetic operators
  1. crossover
  2. mutation
  3. repair operators
- Recombination and selection

2.3 Chromosome Representation

In our problem, the chromosome is the service location matrix A that we mentioned in the previous section.

2.4 Objectives and Fitness Function

The objective functions of this entire problem are following:

- Minimize the total cost of services. n is the number of service, m is the number of candidate location.

$$CostFitness = \sum_{s \in S} \sum_{j \in J} C_{sj} \times A_{sj} \tag{1}$$

- Minimize the network total latency of the services.
  - For each chromosome, firstly, calculate the number of invocation for each service.
  $$Invocation_s = \sum_{i \in I} F_{is}$$
  - Calculate the total number of each services.
  $$ServiceNo_s = \sum_{j \in J} A_{sj}$$
  - Divide Invocations by Service in order to calculate the average invocation for each location.
  $$AverageInvocation_i = Invocation_s \div ServiceNo_s$$
  - Calculate the latency of each service by multiply latency matrix L by service allocation matrix A(chromosome).
  $$LocationLatency_i = \sum_{i \in I} L_{ij} \times A_{sj}$$
  - Calculate the total latency of the multiplication of AverageInvocationi and $LocationLatency_i$.

$$LatencyFitness = \sum_{i \in I} AverageInvocation_i \times LocationLatency_i \tag{2}$$

The population is initialized based on the problem range and constraints. In our problem, we have two constraints, the cost constraint and latency constraint. If the initialized chromosome does not satisfy the constraints, then repair operators will attempt to recover from possible constraint violations. The detail of repair operators will be discuss in section 2.5.2.

2.5 Constraints

The our approach, each chromosome have to satisfy two constraints to be a feasible solution. The first constraint controls the service number which is make sure that each service is deployed in at least one location.

$$\sum_{s \in S} A_{sj} \geq 1 \tag{3}$$

The second constraint is cost constraint which predefined the upper boundary of the total cost. An integer number CostLimitation is defined. This constraint will try to constraints

$$\sum_{s \in S} \sum_{j \in J} C_{sj} \times A_{sj} \leq CostLimitation \tag{4}$$

The constraint is implemented by repair operators which is introduced in the next section.

2.6 Genetic operators

Specific selection, mutation and crossover operators were implemented. It is important to note that mutation and crossover operators can produce solutions that might violate the constraints. Therefore, repair operators are needed to try to maintain feasible solutions. The original NSGA-II use a simulated binary crossover (SBX) Beyer and Deb (2001) and polynomial mutation Raghuwanshi and Kakde (2004) to cope with continuous problem. However, our problem is discretized, therefore we use the regular GA mutation and crossover.

2.6.1 Mutation

The mutation operator works as follows: Initially choose one random location from the chromosome. Then, reverse the bit, e.g. 0 to 1 or 1 to 0.

In this study, a chromosome in a population will be selected based on mutation possibility P_m. As shown in below, a random position will be selected and replaced by a reversed number.

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & \boxed{1} & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \xrightarrow{\text{Mutation}} \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & \boxed{0} & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Figure 1

2.6.2 Crossover

The crossover operator in this paper is the single point crossover. The crossover is controlled by crossover probability P_c. The crossover point is created randomly within the length of the chromosome. As in the example below, two parents crossover at a point, then two offspring were generated.

2.6.3 Repair operators

Since mutation and crossover are very likely to generate offspring that violate the constraints, therefore, repair operators are necessary. Normally, each constraint would has an unique repair operator in order to recover different types of violation.

After the process of mutation and crossover, the repair operators are examine each chromosome. If there is violation found in the chromosome, then it will try to repair it. In our problem, we have two repair operators: service number and cost.

Service number repair operator

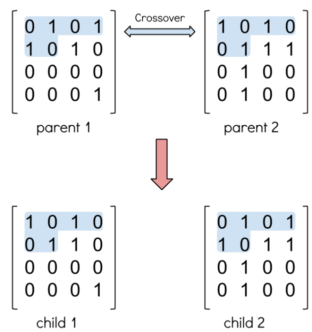

Figure 2

$$\sum_{s \in S} A_{sj} \geq 1$$

service number constraint

The operator will go through each row of the Location Allocation matrix A, if the number of each service is less than one, then randomly choose one location and reverse the selected bit. Although the randomness does not necessarily provide an optimal solution, the computation time is better than exhaustively compare all solutions.

Cost repair operator

$$\sum_{s \in S} \sum_{j \in J} C_{sj} \times A_{sj} \leq CostLimitation$$

cost constraint

If the cost exceed the predefined limitation. Then the operator would iteratively check if any service has been deployed in more than one location. After found one service has been deployed multiple locations, the operator will randomly select one of them and change to zero. That means, cancel the deployment of that service in this location. After doing that, re-examine the chromosome, if it is still exceed the limitation then repeat this process until there is no redundant services or it satisfies the constraint.

This algorithm will try its best to reduce the cost regardless of other factors. Although the modified chromosome may end up with higher network latency, however the cost constraint has higher priority than network latency.

Worth noting that this algorithm may not provide a strictly correct solution that satisfied the constraint, partially because the randomly chosen of canceling deployment. On the other hand, it may also indicate that the predefined upper boundary of cost limitation is too low.

References

Aboolian, R., Sun, Y., and Koehler, G. J. (2009). A locationallocation problem for a web services provider in a competitive market. *European Journal of Operational Research*, 194(1):64 – 77.

Beyer, H.-G. and Deb, K. (2001). On self-adaptive features in real-parameter evolutionary algorithms. *Evolutionary Computation, IEEE Transactions on*, 5(3):250–270.

Caramia, M. (2008). Multi-objective optimization. In *Multi-objective Management in Freight Logistics*, pages 11–36. Springer London.

Deb, K., Mohan, M., and Mishra, S. (2005). Evaluating the ε-domination based multi-objective evolutionary algorithm for a quick computation of pareto-optimal solutions. *Evol. Comput.*, 13(4):501–525.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multi-objective genetic algorithm: Nsga-ii. *Evolutionary Computation, IEEE Transactions on*, 6(2):182–197.

Deb, K., Sundar, J., N, U. B. R., and Chaudhuri, S. (2006). Reference point based multi-objective optimization using evolutionary algorithms. In *International Journal of Computational Intelligence Research*, pages 635–642. Springer-Verlag.

Desai, S., Bahadure, S., Kazi, F., and Singh, N. (2012). Article: Multi-objective constrained optimization using discrete mechanics and nsga-ii approach. *International Journal of Computer Applications*, 57(20):14–20. Full text available.

Elhossini, A., Areibi, S., and Dony, R. (2010). Strength pareto particle swarm optimization and hybrid ea-pso for multi-objective optimization. *Evol. Comput.*, 18(1):127–156.

Huang, H., Ma, H., and Zhang, M. (2014). An enhanced genetic algorithm for web service location-allocation. In Decker, H., Lhotsk, L., Link, S., Spies, M., and Wagner, R., editors, *Database and Expert Systems Applications*, volume 8645 of *Lecture Notes in Computer Science*, pages 223–230. Springer International Publishing.

Huang, V. L., Suganthan, P. N., and Liang, J. J. (2006). Comprehensive learning particle swarm optimizer for solving multiobjective optimization problems: Research articles. *Int. J. Intell. Syst.*, 21(2):209–226.

Huffaker, B., Fomenkov, M., Plummer, D., Moore, D., and claffy, k. (2002). Distance Metrics in the Internet. In *IEEE International Telecommunications Symposium (ITS)*, pages 200–202, Brazil. IEEE.

Jamin, S., Jin, C., Kurc, A., Raz, D., and Shavitt, Y. (2001). Constrained mirror placement on the internet. In *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 1, pages 31–40 vol.1.

Johansson, J. M. (2000). On the impact of network latency on distributed systems design. *Inf. Technol. and Management*, 1(3):183–194.

Kanagarajan, D., Karthikeyan, R., Palanikumar, K., and Davim, J. (2008). Optimization of electrical discharge machining characteristics of wc/co composites using non-dominated sorting genetic algorithm (nsga-ii). *The International Journal of Advanced Manufacturing Technology*, 36(11-12):1124–1132.

Raghuwanshi, M. M. and Kakde, O. G. (2004). Survey on multiobjective evolutionary and real coded genetic algorithms. In *Proceedings of the 8th Asia Pacific symposium on intelligent and evolutionary systems*, pages 150–161.

Ran, S. (2003). A model for web services discovery with qos. *SIGecom Exch.*, 4(1):1–10.

Sun, Y. and Koehler, G. J. (2006). A location model for a web service intermediary. *Decis. Support Syst.*, 42(1):221–236.

Vanrompay, Y., Rigole, P., and Berbers, Y. (2008). Genetic algorithm-based optimization of service composition and deployment. In *Proceedings of the 3rd International Workshop on Services Integration in Pervasive Environments*, SIPE '08, pages 13–18, New York, NY, USA. ACM.

Zhang, Y., Zheng, Z., and Lyu, M. (2011). Exploring latent features for memory-based qos prediction in cloud computing. In *Reliable Distributed Systems (SRDS), 2011 30th IEEE Symposium on*, pages 1–10.

Zheng, Z., Zhang, Y., and Lyu, M. (2010). Distributed qos evaluation for real-world web services. In *Web Services (ICWS), 2010 IEEE International Conference on*, pages 83–90.

Zhou, J. and Niemela, E. (2006). Toward semantic qos aware web services: Issues, related studies and experience. In *Web Intelligence, 2006. WI 2006. IEEE/WIC/ACM International Conference on*, pages 553–557.