Multi-objective Service Monitoring Rate Optimization using Memetic Algorithm

Pan He, Junhao Wen, Kaigui Wu and Peng Li
College of Computer Science, Chongqing University, Chongqing, China
Email: {hepan, jhwen, kaiguiwu, cheney}@cqu.edu.cn

Haijun Ren
School of Software Engineering, Chongqing University, Chongqing, China
Email: jhren@cqu.edu.cn

***Abstract*— In dynamic service-oriented environment, service monitoring could provide reliability improvement to service composition as well as cost increase. To reduce the overall cost brought by monitoring, existing literatures proposed to decrease the number of monitors through monitoring the most reliability-sensitive services. However, the optimal monitoring rate for those monitors was not taken into account at the same time. Aiming at choosing optimal monitoring rate for minimal number of monitors, this paper proposed to search appropriate monitoring rate to minimize multi kinds of resources cost by monitoring under reliability constraints. Firstly, two multi-objective optimization problems were presented with the reliability and cost models of service composition under monitoring analyzed through Markov chain. Then a multi-objective memetic algorithm (MOMA) was used to search the near-optimal solutions of monitoring rate for services. This algorithm employed nondominated sorting strategy as the global search method and used random walk with direction exploitation method as local search operator. Experimental studies results showed that multi-objective approach for service monitoring rate optimization could provide solutions with a variety of trade-offs between the system reliability and cost comparing with existing greedy sensitivity-based method. Comparison with other multi-objective evolutionary algorithms showed that, in terms of both the coverage rate and hypervolume indicator, MOMA searched more effectively than several state-of-art algorithms including NSGA II, PHC-NSGA-II and HaD-MOEA.**

***Index Terms*—service monitoring; Markov chain; software reliability; memetic algorithm; multi-objective optimization**

I. Introduction

Although the development techniques for service-oriented architecture have been extensively investigated, how to improve the reliability of service composition has not been comprehensively studied. In traditional software systems, testing time increase and redundant components configuration are two common ways to improve system reliability [1]. In dynamic service-oriented environments, it is difficult to test all services in a composition in advance, so monitoring is often used to improve service reliability [2]. Like traditional reliability assurance methods, monitoring mechanism also increases system cost, so it may be advantageous to achieve a balance between reliability improvement and cost increase. Since a service-oriented system typically comprises of several services, a natural question is how to deploy monitors on them and how to optimize the monitoring rates.

Our previous work [3] employed a greedy method to select the most reliability-sensitive services in a composition to monitor but it couldn't guarantee the minimal monitoring rate for these monitors at the same time. In this paper, we treat the problem of "finding the minimal number of monitors with minimal monitoring rate" as multi-objective optimization problems of "minimizing the cost under reliability constraints" and a memetic algorithm is used to search the near-optimal solutions. In the memetic algorithm, nondominated sorting genetic algorithm II (NSGA II) [4] is introduced in the global search process and iterative random walk strategy with direction exploitation [5] is employed as local search operator. Empirical studies results showed that in multi-objective monitoring rate optimization, our method outperformed the sensitivity-based method and several evolutionary algorithms including NSGA II, PHC-NSGA-II[6] and HaD-MOEA[7], which are state-of-art approaches for multi-objective reliability allocation.

The whole paper is organized as follows: Section II summarized related work on optimal reliability allocation and multi-objective memetic algorithms. Section III analyzed reliability model of service composition under monitoring and presented the problem formulation of monitoring rate optimization. Section IV introduced our memetic algorithm to find minimal monitors with minimal monitoring rate. Section V listed experimental studies results followed by a conclusion.

II. Related Work

Optimal reliability allocation has been researched for decades [1] but existing researches often deal with single-objective reliability allocation problems. Moreover, few approaches on monitoring resources allocation have been proposed. Reference [3] presented a greedy sensitivity-

Corresponding author: Pan He, Email: hepan@cqu.edu.cn

doi:10.4304/jsw.7.5.990-997

based method to select most sensitive services to monitor according to monitoring mechanisms. However, it didn't consider the trade-off between the number of monitors and the monitoring rate. Zai Wang[7] proposed a harmonic distance based multi-objective evolutionary algorithm(HaD-MOEA). It outperformed NSGA II but aimed at multi-objective testing time allocation. In a word, there is no existing work on efficient multi-objective algorithms for monitoring rate optimization, so we plan to build a memetic algorithm to solve this problem.

As an emerging area of evolutionary algorithm, memetic algorithm combines global search strategies with local search heuristics [8] and thus searches more efficiently than conventional genetic algorithms [9]. The success of memetic algorithm has been demonstrated on a variety of single objective optimization problems. Zai Wang [10] proposed a memetic algorithm to solve single objective redundancy allocation in multi-level system. Hongfeng Wang [11] presented a particle swarm optimization based memetic algorithm for dynamic optimization problems. T. Warren Liao [5] employed random walk with direction exploitation method as the local search operator in the single objective memetic algorithm. The application of memetic algorithm in multi-objective optimization hasn't drawn much attention. Slim Bechikh [6] introduced a novel multi-objective memetic algorithm (PHC-NSGA-II) for continuous optimization, which was a result of hybridization of the NSGA-II algorithm with polynomial mutation as local search procedure, and the efficiency of this local method could be improved.

III. PROBLEM DESCRIPTION

In this section, we first analyze the reliability and cost models for service composition under monitoring and then present the monitoring rate optimization model.

A. Reliability and Cost Models

1) Reliability model

Monitoring process checks the status of in-use service periodically and replaces unavailable ones in background. Since service running and monitoring process are both continuous process, we first build Continuous Time Markov Chain (CTMC)[12] model to analyze the reliability of single service under monitoring, as shown by Fig.1. In this model, state(1,0) and state(0,0) represent the normal execution and failure state of service. State(1,1) and state(0,1) represent that the service is in the monitoring state. $1/\lambda$ and $1/\mu$ represent mean time to failure and mean time to restore the service respectively. We assume that 1) $\lambda(\mu)$ of new service and the replaced service are the same because they are often selected according to certain QoS rules; 2) new changed services are always in the working state. λ_m represents the monitoring rate and $1/\mu_0$ denotes the mean time taken to wait for the response of the service. $1/\mu_m$ denotes the mean time used to wait for the response of services, select and change service. To simplify the calculation process, we consider the steady state of this model. Let $\pi_{i,j}$ denotes the steady probability of the service pool in state (i,j).

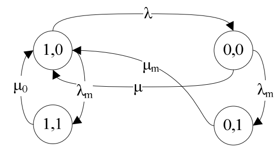

Figure 1. CTMC model for single service under monitoring.

According to the rule of "Rate of flow in=Rate of flow out for each state" [12], we can get (1) from Fig. 1.

$$\begin{cases} \pi_{1,0}(\lambda+\lambda_m)=\pi_{0,1}\mu_m+\pi_{1,1}\mu_0+\pi_{0,0}\mu \\ \pi_{0,0}(\lambda_m+\mu)=\pi_{1,0}\lambda \\ \pi_{0,1}\mu_m=\pi_{0,0}\lambda_m,\ \text{or}\ \pi_{1,1}\mu_0=\pi_{1,0}\lambda_m \\ \pi_{1,1}+\pi_{1,0}+\pi_{0,1}+\pi_{0,0}=1 \end{cases} \tag{1}$$

The reliability of this service could be perceived as the probability that the service in working states (1, i): $R=\pi_{1,0}+\pi_{1,1}$, shown by (2). All the parameters in R could be determined except for the monitoring rate λ_m , so we could solve (2) to get $R=R(\lambda_m)$.

$$R=(1+\frac{\lambda_m}{\mu_0})\Big/(1+\frac{\lambda_m}{\mu_0}+\frac{\lambda}{\lambda_m\cdot p+\mu}+\frac{\lambda_m}{\mu_m}\cdot\frac{\lambda}{\lambda_m+\mu}) \tag{2}$$

Then, we analyze the reliability of service composition under monitoring. Although there are some differences between the service composition and traditional modular software, it could be transformed to DTMC [14,15]. After building the DTMC model on the composition, we could get the number of invocations to each service V_i. The overall reliability model of service composition with monitored services could be calculated from each service using hierarchy approach of architecture-based software reliability model[13], as shown by (3). R_c denotes the reliability of the service composition, R_i denotes individual services' reliability under monitoring, m denotes the number of services in the composition and λ_{mi} denotes monitoring rate for the ith service ($1\le i\le m$).

$$R_c=\prod_{i=1}^{m}R_i^{V_i}=\prod_{i=1}^{m}R_i(\lambda_{m_i})^{V_i} \tag{3}$$

2) Cost model

There are generally two kinds of resources cost by monitoring. On one hand, additional resources are needed to store information about backup services, deploy the monitor and set the timer. Let S_i denotes this kind of resource cost by the ith monitor and S_c denotes the overall cost resources. S_c could be calculated from S_i, as shown by (4), where k_i denotes whether the ith monitor is deployed. If monitor is deployed on the ith service, $k_i=1$, else $k_i=0$. On the other hand, additional network bandwidth will be taken to transmit monitoring query and response messages. Let L_c denotes the bandwidth cost in a unit time and L_i denotes the length of query and response message of the ith monitor. L_c is determined by the monitoring rate λ_{mi}, as shown by (5). If monitor is deployed on the ith service, $\lambda_{mi}>0$, else $\lambda_{mi}=0$.

$$S_c=\sum_{i=1}^{m}S_ik_i,\ k_i=\{0,1\} \tag{4}$$

$$L_c = \sum_{i=1}^{m} L_i \lambda_{m_i}, \ \lambda_{m_i} \geq 0 \tag{5}$$

B. Service Monitoring Rate Optimization Problem

To minimize overall monitoring cost S_c and L_c, we aim at choosing appropriate k and λ_{mi} under reliability constraints R_0. To simplify the problem, we assume that S_i and L_i for each monitor is the same and denote them by S_0 and L_0. Let k denotes the number of monitors set in the composition. The optimal problem is shown in (6) and (7), depending on whether we need to maximize the reliability. Equation (6) is a bi-objective optimization problem and (7) is a tri-objective optimization problem.

$$\begin{aligned} &\text{Min } S_c = \sum_{i=1}^{m} S_i k_i = S_0 \sum_{i=1}^{m} k_i = S_0 k \quad k_i = \{0,1\} \\ &\text{Min } L_c = \sum_{i=1}^{m} L_i \lambda_{m_i} = L_0 \sum_{i=1}^{m} \lambda_{m_i} \quad \lambda_{m_i} \geq 0 \\ &\text{s.t. } R_c = \prod_{i=1}^{m} R_i(\lambda_{m_i})^{V_i} \geq R_0 \end{aligned} \tag{6}$$

$$\begin{aligned} &\text{Min } S_c = \sum_{i=1}^{m} S_i k_i = S_0 \sum_{i=1}^{m} k_i = S_0 k \quad k_i = \{0,1\} \\ &\text{Min } L_c = \sum_{i=1}^{m} L_i \lambda_{m_i} = L_0 \sum_{i=1}^{m} \lambda_{m_i} \quad \lambda_{m_i} \geq 0 \\ &\text{Max } R_c = \prod_{i=1}^{m} R_i(\lambda_{m_i})^{V_i} \\ &\text{s.t. } R_c = \prod_{i=1}^{m} R_i(\lambda_{m_i})^{V_i} \geq R_0 \end{aligned} \tag{7}$$

IV. Multi-objective Memetic Algorithm

In this section, we proposed a multi-objective memetic algorithm (MOMA) to solve the above two problems.

Our MOMA is built as follows: 1) the **simulated binary crossover operator** and **polynomial mutation operator** are employed as genetic operator to generate new invididuals; 2) **nondominated sorting** strategy [4] is used to sort combined populations and to select needed solution; 3) **random walk with direction exploitation** method [5] is introduced to refine local best solutions. The pseudo code of MOMA's framework is shown below.

```
Begin
Set offspring population Q0=Ø, and generation counter t=0. Initialize population P0 with N individuals, P0={p1,p2,…,pN}.
While t<tmax(the maximum generation number)
   Use genetic operator to generate Qt from Pt.
   Combine parent and offspring population Rt=Pt∪ Qt.
   Calculate objective values and constraints values of individuals in Rt.
   Calculate nondominated fonts and crowding distance for elements in Rt,
   Sort Rt in descending order using nondominated fonts and crowding distance and select the first individual.
   Apply local search operator to the selected one.
   Include the feasible solutions obtained in the local search into Rt and calculate corresponding nondominated fonts and crowding distance.
   Sort Rt again in descending order using nondominated fonts and crowding distance and fill Pt+1 with first N individuals.
   Set t=t+1.
Output Pt
End
```

A. Genetic Operators

Each individual p_j($1 \leq j \leq N$) in the population P_t is an array of monitoring rate value for web services: $p_j=\{\lambda_{m1}, \lambda_{m2},\ldots,\lambda_{mm}\}_j$. The simulated binary crossover (SBX) operator and polynomial mutation [16] operator are employed as genetic operator in this algorithm. In each generation, SBX operator generates new individuals as (8) and (9), where $\lambda'_{mi,k}$ denotes the monitoring rate of the ith service monitor in the kth child and $\lambda_{mi,k}$ denotes the ith monitoring rate of the randomly selected parent ($1 \leq k \leq 2$). β is generated from (10), where μ is a random number between (0, 1) and η is the predefined distribution index for crossover.

$$\lambda'_{m_{i,1}} = 0.5[(1-\beta)]\lambda_{m_{i,1}} + (1+\beta)\lambda_{m_{i,2}}] \tag{8}$$

$$\lambda'_{m_{i,2}} = 0.5[(1+\beta)]\lambda_{m_{i,1}} + (1-\beta)\lambda_{m_{i,2}}] \tag{9}$$

$$\beta = \begin{cases} (2\mu)^{1/(\eta+1)}, & \text{if } 0 \leq \mu < 0.5 \\ 1/[2(1-\mu)]^{1/(\eta+1)}, & \text{if } 0.5 \leq \mu < 1 \end{cases} \tag{10}$$

Polynomial mutation operator could be presented by (11), where λ'_{mi} and λ_{mi} denote the ith monitoring rate of the child and parent. r is a random number between (0, 1) and η_m is the predefined mutation distribution index.

$$\lambda'_{m_i} = \begin{cases} \lambda_{m_i} + (2r)^{1/(\eta_m+1)} - 1, & \text{if } 0 \leq r < 0.5 \\ \lambda_{m_i} + 1 - [2(1-r)]^{1/(\eta_m+1)}, & \text{if } 0.5 \leq r < 1 \end{cases} \tag{11}$$

B. Nondominated Sorting and Selection Strategy

The pseudo code of sorting and selection strategy is shown below and the details of fast_non_dominated_sort() and crowding_distance_assignement() could refer to [4]. Nondominated sorting algorithm is used twice in MOMA, before local search and for global selection process.

```
Fast_non_dominated_sort_and_select()
Begin
   Fast_non_dominated_sort(Rt) to get nondominated fonts F of Rt, F={F1,F2,…}.
   If (before local search)
      crowding_distance_assignement(F1), calculate crowding distance in F1(the first nondominated font).
      Sort F1 in descending order using crowding distance.
      Local_best_solution=F1[1](the first solution in the first nondominated font).
   Else (for global search)
      Pt+1=Ø and i=1.
```

While $|P_{t+1}|+|F_i|\leq N$
crowding_distance_assignement(F_i), calculate crowding distance in F_i.
Include ith nondominated front in the parent population, $P_{t+1}=P_{t+1}\cup F_i$.
Set i=i+1.
End
Sort F_i in descending order using crowding distance.
$P_{t+1}=P_{t+1}\cup F_i[1:(N-|P_{t+1}|)]$, choose the first $(N-|P_{t+1}|)$ elements of F_i and combine them with P_{t+1}.
End
End

For bi-objective problem, number of monitors k (no. of positive λ_{mi} in p_j) and sum of monitoring rate $\Sigma\lambda_{mi}$ are used as the objective value in nondominated sorting algorithms and crowding distance calculation. For tri-objective problem, one more objective value, the reliability $R=R(\lambda_{mj})$ is added.

C. Local Search Strategy

After the local best solution is found through non-dominated sorting, the local search strategy takes use of random walk with direction exploitation method to search its neighbors and include feasible solutions. In the search process, a random array is generated and adjusted to local best solution. The new individual is then checked whether to be a nondominated solution. If so, it will be included into acceptable solutions, otherwise, the random array will be reduced by half and the process will continue until it reaches the maximal number of iterations. The pseudo code of local search method is shown below and the algorithm will return a set of nondominated solutions.

Local_search_procedure(Input local best individual x)
Begin
i=1 and Q=Ø.
While $i<i_{max}$(predefined maximal search number)
Randomly generate vector $\lambda=\{\lambda_1,\lambda_2,\ldots\}$.
Set t=1, x_t=x and ratio=1.
While $t<t_{max}$(predefined maximal iteration number)
$x_t=x_t$+ratio*λ.
If x_t constraint_dominate x
Include x_t into Q and break;
Else
ratio=ratio/2.
End
Set t=t+1.
End
Set i=i+1.
End
Output Q. (Include Q into combined population R_t)
End

V. Experimental Studies

In this section, our MOMA is compared with the sensitivity-based method and other multi-objective algorithms including original NSGA II [4], PHC-NSGA-II[6] and HaD-MOEA[7]. NSGA II is a most widely used algorithm in multi-objective optimization. PHC-NSGA-II is a new memetic algorithm built upon NSGA II and HaD-MOEA is a state-of-art evolutionary algorithm for multi-objective testing time allocation. The experimental studies were designed to consist of two parts. First, the MOMA was compared with other approaches using a single group of parameters with different reliability constraint values. Second, several groups of system parameters with a certain reliability constraint value were used for comparison.

A. Experimental Design

A sample service composition, containing six types of basic workflow patterns, taken from [17] is used as the example in this section, shown by Fig. 2. The transition probability information is listed in [17] and we could obtain V_i=[1,1,3.03,1,1,0.7,0.2,0.1,1,1,1], $R_c=R_1R_2R_3^{3.03}R_4R_5R_6^{0.7}R_7^{0.2}R_8^{0.1}R_9R_{10}R_{11}R_{12}$. 10 groups of failure rate, repair rate are simulated for 12 services in it. The data is randomly chosen from the sample data collected from real-world services through Membrane, an open source web service registry and monitoring tool [18].

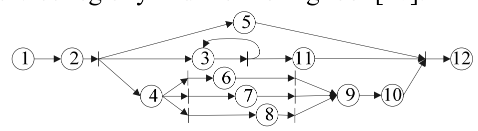

Figure 2. Sample workflow of service composition.

Assuming each service waits at most 10 seconds for response [19] and 10 seconds for retrying, the value of $\mu_0=1/t_0=3$. Generally service selection process takes about 0.5~2 seconds according to different algorithms [20]. We don't take the service replacement time into consideration since it is often much less than the time for service selection. As a result, the value of $\mu_s=1/t_s\in(30,120)$ and we choose $\mu_s=60$ in the following experiments. $t_m=1/\mu_m=t_0+t_s$. Let S_0 and L_0 to be unit 1 and omit them in the objective values calculation. In MOMA, the population size is set to 100, the maximum generation is 200, the crossover rate is set to 0.9, and the mutation rate is set to 0.1. The initial value of monitoring rate of each service λ_{mi} ($1\leq i\leq m$) is randomly generated from 0 to 1/min. The minimal monitoring interval is 1 minute and the maximal interval is ∞, which implies that no monitor is set. If the monitoring rate value after genetic operations exceeds the range of [0, 1], it will be replaced by 0 or 1 respectively.

B. Comparison using Different Reliability Constraint

In this experiment, a single group of service failure rate and repair rate was randomly chosen from the dataset and the reliability threshold value was set from 0.90 to 0.99. Fig. 3 shows the comparison of a single group of solutions obtained by different algorithms for bi-objective problem under different constraint value, where dR_0 denotes the minimal reliability improvement for sensitivity-based method. Sensitivity-based method only returns a single group of solution and it is dominated by the Pareto sets obtained by MOMA, which provides more kinds of solutions. Under the same number of monitors, the monitoring rate obtained by MOMA is generally smaller than those obtained by other solutions. Fig. 4 illustrates a set of nondominated solutions obtained by multi-objective algorithms on both the bi-objective problem and the tri-objective problem. According to

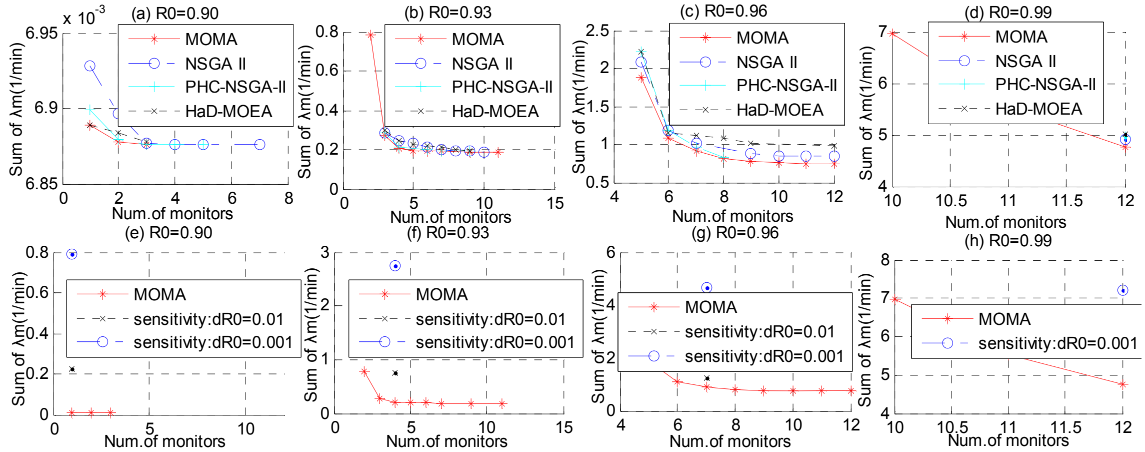

Figure 3. Comparison of solutions obtained by different algorithms for bi-objective problem using different reliability constraint (one case study).

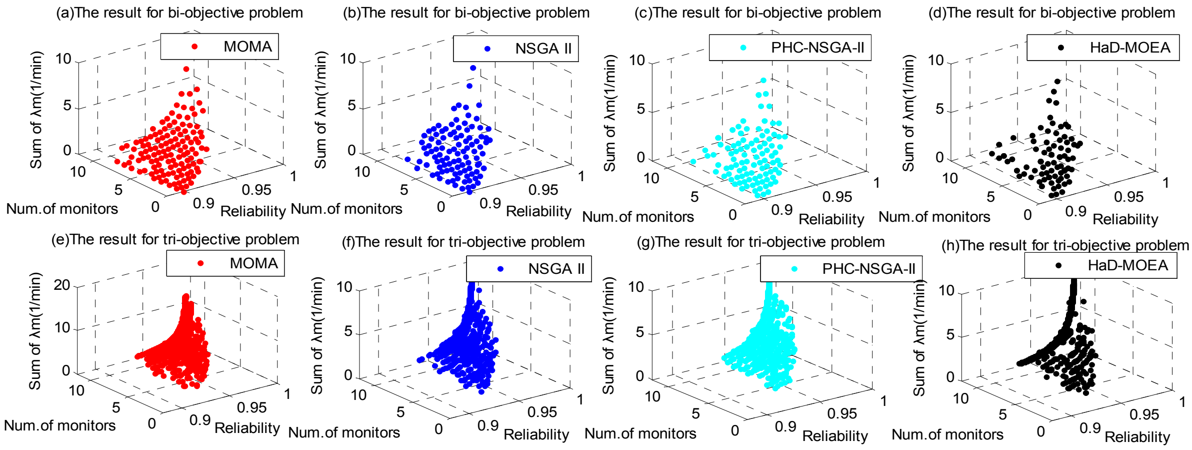

Figure 4. Nondominated sets obtained by multi-objective algorithms for bi-objective and tri-objective problems using different reliability constraint (one case study).

Fig.4, the solutions obtained by MOMA spread better in the objective space than those obtained by other algorithms. This result suggests that MOMA might be advantageous to other methods. However, the figures are just a simple illustration, and to compare MOMA and other multi-objective evolutionary algorithms in detail, we use two measures: 1) the coverage fraction [21] and 2) hypervolume indicator [22]. The coverage fraction stands for the fraction of nondominated solutions obtained by one algorithm, which are covered by the nondominated solutions of another algorithm. It indicates a direct comparison of two nondominated sets. The hypervolume indicator is a set measure reflecting the volume enclosed by a set of solutions. It guarantees that any approximation set that achieves the maximally possible quality value contains all the Pareto-optimal objective vectors.

Experiment for each reliability constraint value was carried out for 10 times and the average coverage fraction of nondominate sets obtained by four algorithms was shown in Table I. The result obtained by MOMA that is significantly better than the other is emphasized in **boldface**. Again, MOMA covers greater faction of the Pareto sets achieved by other algorithms, which implies that the solution space of MOMA is bigger than the others. Table II presents the mean values of hypervolume indicator obtained by four algorithms in 10 independent runs for each reliability constraint value. Except for few test cases, the hypervolume indicator of MOMA is higher that other algorithms.

C. Comparion Using Different Service Reliability

In this experiment, different groups of service failure rate and repair rate from the dataset were used and reliability constraint value was set to 0.95. Fig. 5 shows the comparison of a single group of solutions obtained by different algorithms for bi-objective problem using different groups of parameters and Fig. 6 illustrates the nondominated solutions obtained on both the bi-objective problem and the tri-objective problem. Similarly, sensitivity-based method only returns single solution for each experiment and the solution is dominated by the Pareto sets of MOMA. From Fig.6, it is difficult to determine which algorithm dominates another, so we will compare the coverage fraction and hypervolume indicator of solutions obtained by different algorithms.

Each experiment (for different failure rate and repair rate) was carried out for 10 times and the average coverage fraction of nondominate sets obtained by four algorithms was shown in Table III. Again, MOMA covers greater faction of the Pareto sets achieved by the other algorithms. Table IV presents the average values of hypervolume indicator obtained by the four algorithms in

TABLE I.
THE FRACTION OF NONDOMINATED SOLUTIONS COVERED BY OTHER NONDOMINATED POINTS (THE LAST COLUMN COMPRISES THE MEAN VALUE FOR EACH ROW).

Coverage (A cover B)												
bi-objective problem												
algorithm		Reliability constraint										
A	B	0.90	0.91	0.92	0.93	0.94	0.95	0.96	0.97	0.98	0.99	mean
NSGA II	MOMA	52.0%	60.1%	36.8%	25.5%	18.9%	4.9%	10.9%	1.7%	0.0%	0.0%	21.1%
MOMA	NSGA II	47.5%	35.2%	**59.3%**	**71.2%**	**75.5%**	**94.2%**	**86.6%**	**95.0%**	**100.0%**	**100.0%**	**76.5%**
PHC-NSGA-II	MOMA	64.7%	57.6%	62.0%	35.2%	11.1%	4.4%	3.8%	3.7%	0.0%	0.0%	24.2%
MOMA	PHC-NSGA-II	17.3%	41.3%	34.3%	**59.9%**	**80.1%**	**98.3%**	**86.1%**	**86.8%**	**100.0%**	**100.0%**	**70.4%**
HaD-MOEA	MOMA	43.3%	27.3%	12.3%	17.1%	17.4%	13.9%	8.2%	3.3%	0.0%	0.0%	14.3%
MOMA	HaD-MOEA	**62.3%**	**64.8%**	**67.6%**	**43.1%**	**45.8%**	**56.1%**	**64.6%**	**96.6%**	**100.0%**	**100.0%**	70.1%
tri-objective problem												
NSGAII	MOMA	6.3%	17.8%	12.7%	7.6%	12.9%	12.2%	14.2%	9.9%	12.6%	2.9%	10.9%
MOMA	NSGAII	91.5%	77.5%	73.1%	87.3%	81.4%	85.3%	80.8%	81.1%	72.2%	95.1%	82.5%
PHC-NSGA-II	MOMA	20.1%	20.5%	21.9%	22.8%	21.8%	17.4%	12.8%	7.0%	4.4%	0.3%	14.9%
MOMA	PHC-NSGA-II	**23.1%**	**23.6%**	**27.3%**	22.0%	**26.2%**	**34.5%**	**44.5%**	**52.2%**	**55.2%**	**91.4%**	**40.0%**
HaD-MOEA	MOMA	11.0%	16.6%	2.5%	12.6%	11.6%	12.1%	21.3%	8.2%	21.4%	4.8%	12.2%
MOMA	HaD-MOEA	**66.9%**	**59.4%**	**94.0%**	**55.9%**	**57.0%**	**52.2%**	**53.6%**	**47.1%**	**49.2%**	**61.0%**	**59.6%**

TABLE II.
VALUES OF HYPERVOLUME INDICATOR OF OBTAINED RESULTS FROM DIFFERENT ALGORITHMS.

Hypervolume Indicator								
Reliability constraint	bi-objective problem				tri-objective problem			
	NSGA II	PHC-NSGA-II	HaD-MOEA	MOMA	NSGA II	PHC-NSGA-II	HaD-MOEA	MOMA
0.90	0.3478	0.1238	0.4814	0.4300	0.9160	0.9160	0.9155	0.9152
0.91	0.4879	0.3811	0.4201	**0.4953**	0.9109	0.9123	0.9106	**0.9127**
0.92	0.4222	0.2931	0.4625	**0.4763**	0.8216	0.8202	0.8243	0.8222
0.93	0.6669	0.2657	0.5880	0.6171	0.8055	0.8088	0.7311	**0.8095**
0.94	0.5705	0.3965	0.3487	0.5247	0.7078	0.7056	0.7051	**0.7200**
0.95	0.2664	0.3213	0.2634	**0.4202**	0.6016	0.6037	0.6003	**0.6276**
0.96	0.1943	0.2944	0.1794	**0.3035**	0.4779	0.4973	0.4865	**0.5270**
0.97	0.2353	0.1840	0.1258	**0.2480**	0.3625	0.3799	0.3225	**0.4094**
0.98	0.0914	0.0281	0.0391	**0.1163**	0.2054	0.2033	0.2105	**0.2445**
0.99	0.0000	0.0000	0.0000	**0.0188**	0.0317	0.0355	0.0265	**0.0726**
Mean	0.3283	0.2288	0.2908	**0.3650**	0.5841	0.5883	0.5733	**0.6061**

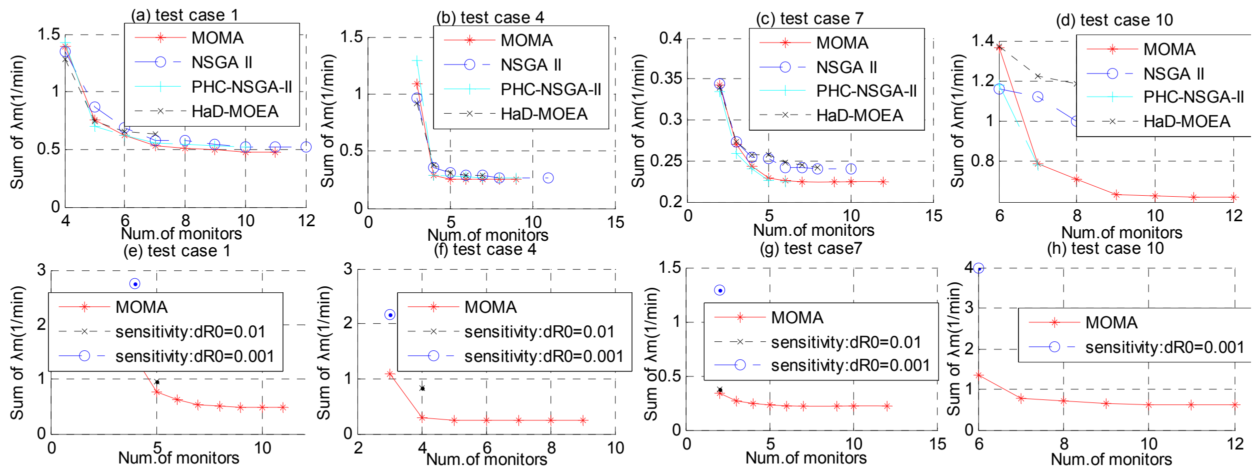

Figure 5. Comparison of solutions obtained by different algorithms for bi-objective problem using different groups of parameters (one case study).

10 independent runs for each group of parameter. For most test cases, the hypervolume indicator of MOMA outperforms other algorithms.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we propose to optimally deploy monitors and set monitoring rate in service composition to improve the reliability as well as to save unnecessary cost. To improve the optimization efficiency, we formulate this problem as two multi-objective optimization problems

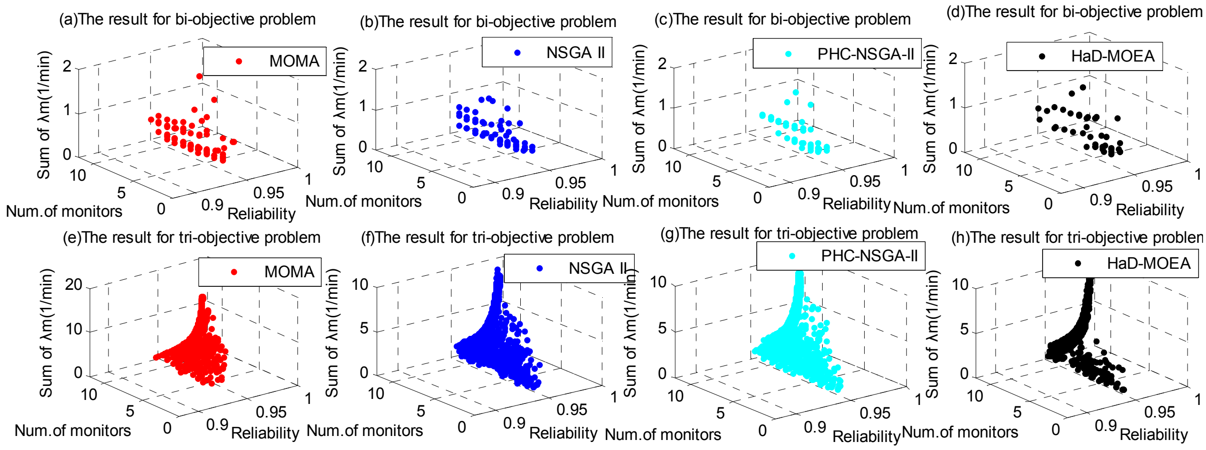

Figure 6. Nondominated sets obtained by multi-objective algorithms for bi-objective and tri-objective problem using different groups of parameters (one case study).

TABLE III.
THE FRACTION OF NONDOMINATED SOLUTIONS COVERED BY OTHER NONDOMINATED POINTS (THE LAST COLUMN COMPRISES THE MEAN VALUE FOR EACH ROW).

Coverage (A cover B)												
bi-objective problem												
algorithm		Test case no.										
A	B	1	2	3	4	5	6	7	8	9	10	mean
NSGA II	MOMA	6.3%	17.8%	12.7%	7.6%	12.9%	12.2%	14.2%	9.9%	12.6%	2.9%	10.9%
MOMA	NSGA II	**91.5%**	**77.5%**	**73.1%**	**87.3%**	**81.4%**	**85.3%**	**80.8%**	**81.1%**	**72.2%**	**95.1%**	**82.5%**
PHC-NSGA-II	MOMA	2.9%	21.2%	18.1%	5.0%	16.0%	10.7%	19.9%	6.0%	21.6%	2.5%	12.4%
MOMA	PHC-NSGA-II	**97.1%**	**72.5%**	**64.8%**	**91.7%**	**82.2%**	**87.7%**	**74.2%**	**86.7%**	**70.2%**	**98.0%**	**82.5%**
HaD-MOEA	MOMA	11.0%	16.6%	2.5%	12.6%	11.6%	12.1%	21.3%	8.2%	21.4%	4.8%	12.2%
MOMA	HaD-MOEA	**66.9%**	**59.4%**	**94.0%**	**55.9%**	**57.0%**	**52.2%**	**53.6%**	**47.1%**	**49.2%**	**61.0%**	**59.6%**
tri-objective problem												
NSGAII	MOMA	11.5%	20.4%	18.8%	9.9%	14.5%	14.0%	19.0%	6.4%	18.3%	8.1%	14.1%
MOMA	NSGAII	**45.3%**	**26.6%**	**31.0%**	**49.3%**	**38.3%**	**45.1%**	**28.3%**	**57.9%**	**36.0%**	**52.2%**	**41.0%**
PHC-NSGA-II	MOMA	16.3%	17.0%	22.2%	18.8%	21.3%	15.2%	19.6%	9.9%	17.4%	6.7%	16.4%
MOMA	PHC-NSGA-II	**32.1%**	**29.5%**	**30.2%**	**31.9%**	**27.5%**	**41.4%**	**30.0%**	**45.5%**	**34.9%**	**60.2%**	**36.3%**
HaD-MOEA	MOMA	14.9%	13.4%	16.3%	7.4%	13.4%	9.9%	12.3%	3.1%	12.4%	3.3%	10.6%
MOMA	HaD-MOEA	**29.7%**	**15.7%**	16.0%	**51.5%**	**25.8%**	**40.4%**	**15.7%**	**77.5%**	**26.4%**	**83.5%**	**38.2%**

TABLE IV.
VALUES OF HYPERVOLUME INDICATOR OF OBTAINED RESULTS FROM DIFFERENT ALGORITHMS.

Hypervolume Indicator								
Test case no.	bi-objective problem				tri-objective problem			
	NSGA II	PHC-NSGA-II	HaD-MOEA	MOMA	NSGA II	PHC-NSGA-II	HaD-MOEA	MOMA
1	0.4290	0.3510	0.3160	**0.4714**	0.6013	0.6027	0.6075	**0.6216**
2	0.2679	0.2257	0.2609	**0.4460**	0.7961	0.8018	0.8021	**0.8077**
3	0.3249	0.1797	0.3887	0.3451	0.7251	0.7315	0.7307	**0.7359**
4	0.5528	0.4499	0.5531	**0.6211**	0.7033	0.7137	0.7058	**0.7245**
5	0.3873	0.3310	0.3550	**0.5170**	0.7112	0.7212	0.7190	**0.7257**
6	0.2886	0.2550	0.4626	0.4330	0.6097	0.6241	0.6159	**0.6344**
7	0.3781	0.2520	0.2613	0.3437	0.7998	0.8035	0.8026	**0.8110**
8	0.4195	0.3815	0.3454	**0.4594**	0.5931	0.6052	0.5996	**0.6268**
9	0.5934	0.3460	0.4272	0.5871	0.7980	0.8078	0.8038	**0.8100**
10	0.2210	0.2555	0.1978	**0.3383**	0.4194	0.4274	0.4287	**0.4646**
Mean	0.3863	0.3027	0.3568	**0.4562**	0.6757	0.6839	0.6816	**0.6962**

and use memetic algorithm to search solutions. The memetic algorithm employs nondominate sorting strategy from NSGA II as the global search method and applies random walk with direct exploitation method as local search strategy. The experimental studies results showed that our MOMA outperformed both sensitivity-based method and other multi-objective evolutionary algorithms including NSGA II, PHC-NSGA-II and HaD-MOEA.

This paper presented a further step of work on monitoring resource allocation comparing with our pervious work [3]. But there is still a lot of work to do in the future. For example, we take use of Markov chain in this paper to get the reliability model of service composition under monitoring but we only consider the steady state of Markov chain. In Internet environment, states of services may change too quickly to achieve the steady sate. So, in the next step, we plan to consider real time processing of monitoring resources optimization and consider the usage of dynamic resources allocation.

ACKNOWLEDGMENT

The work reported in this paper has been supported by the Fundamental Research Funds for the Central Universities Project, under contact No.CDJXS11181161, the Fundamental Research Funds for Central Universities Project, under contact No.CDJRC10090002 and Major Program of the National Natural Science Foundation of China, under contact No.90818028.

REFERENCES

[1] W. Kuo and R. Wan, "Recent advances in optimal reliability allocation," *IEEE Trans. Systems, Man, and Cybernetics–Part A: Systems and Humans*, vol.37, pp.143-156, March 2007.
[2] B.R.Halima, E. Fki, K. Drira and M. Jmaiel, "A Large-scale monitoring and measurement campaign for web services-based applications," *Concurrency and Computation: Practice and Experience,* vol.22, pp. 1172-1177, July 2010.
[3] P. He, K.G. Wu, J.H. Wen, and J. Xu, "Monitoring resources allocation for service composition under different monitoring mechanisms", *in Proc. The Fifth International Conference on Complex, Intelligent, and Software Intensive Systems*, *Soul*, accepted, 2011.
[4] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evolutionary Computation*, vol. 6, pp. 182-197, April 2002.
[5] T.W. Liao, "Two hybrid differential evolution algorithms for engineering design optimization," *Applied Soft Computing*, vol.10, pp. 1188–1199, Sep. 2010.
[6] S. Bechikh, N. Belgasmi, L.B. Said, and K.Ghédira, "PHC-NSGA-II: a novel multi-objective memetic algorithm for continuous optimization," *in Proc. 20th IEEE International Conference on Tools with Artificial Intelligence,* Dayton, OH USA, Nov. 2008, pp.180-189.
[7] Z. Wang, K. Tang, and X. Yao, "Multi-objective approaches to optimal testing resource allocation in modular software systems,"*IEEE Trans. Reliability*, vol.59, pp.563-575, Sep. 2010.
[8] P. Moscato, *On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithm*, CalTech Concurrent Computation Program Report 826. Pasadena, CA: CalTech, 1989.
[9] N. Krasnogor, and J. Smith, "A tutorial for competent memetic algorithms: model, taxonomy, and design issues," *IEEE Trans. Evolutionary Computation*, vol. 9, pp. 474–488, Oct. 2005.
[10] Z. Wang, K. Tang, and X. Yao, "A memetic algorithm for multi-level redundancy allocation," I*EEE Trans. Reliability*, vol. 59, pp. 754-765, Dec. 2010.
[11] H.F. Wang, S.X. Yang, W.H. Ip and D.W. Wang, "A particle swarm optimization based memetic algorithm for dynamic optimization problems," *Nat. Comput.*, vol.9, pp. 703-725, Sep. 2010.
[12] K.S. Trivedi, *Probability and Statistics with Reliability, Queuing and Computer Science Applications*, 2nd ed., New York: John Wiley & Sons, 2001.
[13] S.S. Gokhale and K.S Trived, "Analytical models for architecture-based software reliability prediction: a unification framework," *IEEE Trans. Reliability*, vol.55, pp. 578-590, Dec. 2006.
[14] L.J. Wang, X.Y. Bai, L.Z. Zhou, and Y.N. Chen, "A hierarchical reliability model of service-based software system," *in Proc. IEEE 33rd International Computer Software and Applications Conference*, Seattle WA, July 2009, pp.199-208.
[15] Y.X. Ren, Q. Gu, J.X. Qi, and D.X. Chen, "Reliability prediction of web service composition based on DTMC," *in Proc. 3rd International Conference on Secure Software Integration and Reliability Improvement, Shanghai, China, July*, 2009, pp.369-375.
[16] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuoussearch space," in Complex Syst., April 1995, vol. 9, pp. 115–148.
[17] Y.N. Xia, H.P. Wang, and Y.L. Huang, "A stochastic model for workflow QoS evaluation," *Scientific Programming*, vol. 14, pp. 251-265, 2006.
[18] Directory of Public SOAP Web Services, http://www.service-repository.com, 2010.
[19] Z.B. Zheng, and M.R. Lyu, "WS-DREAM: a distributed reliability assessment mechanism for web services," *in Proc. IEEE International Conference on Dependable Systems and Networks with FTCS and DCC*, Anchorage, AK USA, June 2008, pp.392-397.
[20] M.L. Tang, and L.F Ai, "A hybrid genetic algorithm for the optimal constrained web service selection problem in web service composition," *in Proc. IEEE Congress on Evolutionary Computation*, Spain, July 2010, p.8.
[21] E. Zitzler, and L.Thiele, "Comparison of multiobjective evolutionary algorithms: empirical results", *Evolutionary Computation*, vol.8, pp. 173–195, 2000.
[22] A. Auger, J. Bader, D. Brockhoff and E. Zitzler, "Theory of the hypervolume indicator: optimal μ-distributions and the choice of the reference point." *in Proc.10th Conference on Foundations of Genetic Algorithms*, Orlando, FL, 2009, pp.87-102.

Pan He received the B.S. and M.S. degree in software engineering from the School of Software Engineering, Chongqing University, Chongqing ,China, in 2006 and 2008; and is currently working toward the Ph.D degree at the College of Computer Science, Chongqing University, Chongqing, China.

Junhao Wen received the B.S. and M.S degree in computer science and application from the College of Computer Science, Chongqing University, Chongqing, China, in 1991 and 1999; and the Ph.D. degree from the College of Computer Science, Chongqing University, Chongqing, China, in 2008.

Since 1991, he has been working in the College of Computer Science, Chongqing University, Chongqing, China. Since 2008, he has been a professor in the College of Computer Science, Chongqing University. From 2009, he has been working as a doctor supervisor. He is the co-author of more than 30 referred pulications. His major research interests include service-oriented computing, data mining and real-world applications.

Dr. Wen is the chairman of branch forum of CCFYOCSEF (CCF Young Computer Scientists & Engineers Forum) in Chongqing.