An Adaptive Memetic Agorithm based on Multiobjecitve Optimization for Software Next Release Problem

Xin Cheng, Yuanyuan Huang, Xinye Cai* and Ou Wei
Computer Science and Technology
Nanjing University of Aeronautics and Astronautics
Nanjing,Jiangsu, 210016 P. R. China
chengxin0223@126.com,hyy_nust@sohu.com,xinye@nuaa.edu.cn and owei@nuaa.edu.cn

ABSTRACT

This paper proposes an adaptive multiobjective memetic algorithm to address the software next release problem. The proposed approach was tested and compared with single objective optimization approaches as well as multiobjective evolutionary approaches on real test instances mined from bug repository. Interestingly, results show multiobjective approach outperforms single objective approach in general and The proposed approach has the best performance.

Categories and Subject Descriptors

D.2.0 [**Software Engineering**]: General—*Standards*

Keywords

The Next Release Problem, Multiobjective Memetic Algorithm, Local Search, Adaptive Mechanism.

1. INTRODUCTION

In software industry, software companies usually develop and maintain large and complex software systems that have been sold to a range of diverse customers. One common problem that the companies face is to decide what requirements should be implemented in the next release of the software. The above problem is called next release problem(NRP). In this paper, we propose a memetic algorithm based on multiobjective optimization(MOMA) to address the next release problems. In our proposed MOMA, simulated annealing is integrated as the local search engine into decomposition based multiobjective optimization framework (MOEA/D). Furthermore, we adopt the concept of utility as an adaptive mechanism to determine the frequency of local search and the selection of solutions for local search in our proposed MOMA.

*Xinye Cai is the corresponding author

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage, and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).Copyright is held by the author/owner(s).
GECCO'14, July 12–16, 2014, Vancouver, BC, Canada.
ACM 978-1-4503-2881-4/14/07.
http://dx.doi.org/10.1145/2598394.2598452.

2. THE SINGLE AND BIOBJECTIVE NEXT RELEASE PROBLEM

Assumes that an software system is associated with several *customers* whose requirements need to be considered in the next release. The set S of the customers is denoted by $s_i(1 \leq i \leq m)$). The set R of all the *requirements* that need to be considered is denoted by $r_j(1 \leq j \leq n)$). Implementation of each requirement need to be allocated with certain number of resources, A cost vector C is represented by $c_j(1 \leq j \leq n)$), where c_i indicates that cost of the requirement $r_i \in R$. A request $q_{ij} \in Q$ represent whether a customer s_i request a requirement r_i, q_{ij}=1 denotes that s_i requests r_i or q_{ij}=0 denotes not. Moreover, we assume that customers can gain different profits, when their requests are satisfied. We represent these profits as W=$w_i(1 \leq i \leq m)$), where w_i denotes the profit when the requests of customer s_i are satisfied in the next release.

The goal of the *Single objective NRP* is to find an optimal solution $X^\star$, to maximize $Profit(X) = \sum_{(i,1)\in X} w_i$ gained by customers, which is the sum of profits gained by the selected customers, subject to $Cost(X) \leq b$, where b is a predefined budget constraint and $cost(X) = \sum_{r_k \in R(X)} c_k$, where $R(X) = \bigcup_{(i,1)\in X, q_{ij}=1} \{r_j\}$ is the requirements for X. Similarly, For *the multiobjective NRP*, The total profit for a solution X is defined by $W(X) = \sum_{(i,1)\in X} s_i$,which is the sum of profits gained by the selected customers. The required $Cost$ for implementing the requirements of a solution X is $Cost(X) = \sum_{r_k \in R(X)} c_k$, where $R(X) = \bigcup_{(i,1)\in X, q_{ij}=1} \{r_j\}$ is the requirements for X.

3. THE PROPOSED ALGORITHM

For convenience, our proposed approach is named as multiobjective memetic algorithm with simulated annealing as the local search(MOMA-SA). At each generation, the proposed framework is presented in **Algorithm 1**. It works as follows:

Step 1 Initialization: Initialize P and A.

Step 2 Local Search: Perform local search(SA) to P and generate a set of new solutions L.

Step 3 Global Search: Perform global search to generate a set of N solutions Y from P.

Step 4 Population Update: Use Y and L to update P and A.

Step 5 Stopping Condition: If a preset stopping condition is met, output A. Otherwise, go to Step 2.

4. EXPERIMENTAL DISCUSSIONS

In this section, we conduct experiments to validate our proposed MOMA-SA on real NRP instances. The detailed descriptions of the real instances from the open source software projects of Eclipse and Gnome. For convenience, they are denoted as nrp-e and nrp-g, respectively.

Algorithm 1:MOMA-SA

Input:

1. Multiobjective COPs;
2. a stopping criterion;
3. N: the number of subproblems; the population size of P and A;
4. a uniform spread of N weight vectors: $\boldsymbol{\lambda^1}, \ldots, \boldsymbol{\lambda^N}$;
5. the size of the neighborhood of each subproblem, denoted as T;

Output: A set of non-dominated solutions A;

Step 1: Initialization:

a) Decompose the original multiobjective COP into N subproblems associated with $\boldsymbol{\lambda^1}, \ldots, \boldsymbol{\lambda^N}$.

b) Generate an initial population $P = \{x^1, \ldots, x^N\}$ randomly.

c) Assign $A = P$.

d) Compute the Euclidean distance between any two weight vectors and obtain T closest weight vectors to each weight vector. For each $i = 1, \ldots, N$, set $B(i) = \{i_1, \ldots, i_T\}$, where $\lambda^{i_1}, \ldots, \lambda^{i_T}$ are the T closest weight vectors to λ^i.

Step 2: Local search

a) Select N solutions from P.

b) Perform local search simulated annealing(SA) to the selected solutions to generate $N * J$ solutions.

Step 3: Global search

For each i $\in P$, do:

a) Randomly select two indexes k and l from $B(i)$.

b) Apply one point crossover and bit-wise mutation operators to x_k and x_l to generate y_i for subproblem i.

Step 4: Population update

a) For each i $\in P$, do:

b) If y_i is generated from subproblem i, for each index $k \in B(i)$, if $g^{ws}(y_i|\lambda^k) \leq g^{ws}(x_k|\lambda^k)$, then set $x_k = y_i$.

c) Set $j = 1$.

d) If L_j is generated from subproblem i, for each index $k \in B(i)$, if $g^{ws}(L_j|\lambda^k) \leq g^{ws}(x_k|\lambda^k)$, then set $x_k = L_j$.

e) Set $j \rightarrow j + 1$. If $j \leq J * N$, go back to Step 4d.

f) Merge Y with A and new solution set L in Local search;to obtain $Z = A \cup Y \cup L$; sort the merged population Z with fast non-dominated sorting and crowding distance approach of NSGA-II and the best N solutions form the new A.

Step 5: Termination

a) If stopping criteria are satisfied, terminate the algorithm and output A. Otherwise, go to **Step 2**.

MOMA-SA using two select mechanisms for local search in **Algorithm 1**, namely, *global selection* and *adaptive selection based on utility* are termed gMOMA-SA and uMOMA-SA, respectively. We adopt hypervolume indicator (I_H) when comparing the performance of various multiobjective approaches. Table 1 demonstrate the mean and standard deviation of hypervolume (I_H) for four compared algorithms. It can be observed that uMOMA/SA always has the best performance compared with other approaches (include gMOMA/SA) in all NRP instances except nrp-g4.

Table 1: Mean and standard deviation of I_H obtained by various algorithms on MONRP instances.

—		NSGAII	MOEA/D	*gMOMA* -SA	*uMOMA* -SA	Wilcoxon test
nrp-e1	mean	1.1617e+08	1.4268e+08	1.4635e+08	**1.4642e+08**	0.0385
	std	1.2917e+06	6.6765e+05	1.6063e+05	1.2599e+005	
nrp-e2	mean	1.2142e+08	1.4758e+08	1.5158e+08	**1.5168e+08**	**0.0337**
	std	1.3331e+06	8.2047e+06	1.4228e+05	1.1180e+05	
nrp-e3	mean	8.8672e+07	1.0849e+08	1.1025e+08	**1.1028e+08**	**0.0639**
	std	1.2610e+06	3.2479e+05	4.7894e+04	4.1890e+04	
nrp-e4	mean	7.4862e+07	8.9218e+07	9.1123e+07	**9.1140e+07**	0.1478
	std	1.0352e+06	2.1358e+05	5.0771e+04	6.3316e+04	
nrp-g1	mean	9.4570e+07	1.1548e+08	1.1744e+08	**1.1748e+08**	**1.1590e-04**
	std	1.1625e+06	4.0086e+05	3.1842e+04	2.5012e+04	
nrp-g2	mean	7.3694e+07	8.7304e+07	8.7826e+07	**8.7826e+07**	0.7972
	std	7.3188e+05	5.5541e+04	1.5201e+04	1.7036e+04	
nrp-g3	mean	9.5143e+07	1.1637e+08	1.1737e+08	**1.1738e+07**	0.6554
	std	1.1466e+06	2.1827e+05	2.1360e+04	1.7755e+04	
nrp-g4	mean	5.9457e+07	7.0168e+07	**7.0691e+07**	7.0690e+07	0.4570
	std	6.8295e+05	1.1097e+05	3.5762e+04	1.4700e+04	

Table 2: Comparison on single vs. multiobjective algorithms

instance			MSSA	GA	BMA	NSGA-II	MOEA/D	*gMOMA −SA*	*uMOMA −SA*
nrp-e1-0.3	Bound:3945	Best	5723	6662	7572	7135	7836	7773	**7856**
		Average	5656.5	6553.4	7528.2	7063.6	7774.6	7715.8	7771.3
nrp-e1-0.5	Bound:6575	Best	6575	9801	10664	10563	10949	11000	**11032**
		Average	8550	9756.3	10589.2	10491.4	10881.3	10877.6	10893.3
nrp-e2-0.3	Bound:4722	Best	5321	6275	7169	6699	7357	7290	**7360**
		Average	5281.0	6219.6	7109.9	6472.6	7280.8	7231.5	7278.7
nrp-e2-0.5	Bound:7871	Best	7932	9203	10098	9812	**10286**	10263	10278
		Average	7869.6	9172.9	9999.8	9773.6	10234.8	10163.1	10205.3
nrp-e3-0.3	Bound:4778	Best	5031	5795	6461	8558	8858	8850	**8852**
		Average	4906.4	5693.1	6413.0	8504.3	7800.6	8768.6	8803.9
nrp-e3-0.5	Bound:7964	Best	7436	8491	9175	10401	11922	**11922**	11921
		Average	7340.5	8391.1	9100.1	10183.2	11538.8	11864.5	11876.4
nrp-e4-0.3	Bound:5099	Best	4332	5065	5692	7251	7477	7450	**7488**
		Average	4267.9	5023.8	5636.2	7209.8	6961.5	7255.4	7362.4
nrp-e4-0.5	Bound:8499	Best	6459	7487	8043	9488	**10176**	10161	10175
		Average	6391.1	7418.9	7968.0	9090.2	9866.6	10098.7	10095.4
nrp-g1-0.3	Bound:4140	Best	4806	5494	5938	6036	6277	**6278**	6262
		Average	4723.0	5437.0	5911.3	5933.7	6159.0	6155.3	6173.8
nrp-g1-0.5	Bound:6900	Best	7339	8223	8714	8825	9123	9100	**9134**
		Average	7250.3	8151.7	8660.0	8792.3	9003	9019.4	9050.4
nrp-g2-0.3	Bound:3677	Best	3583	4256	**4526**	4361	4472	4480	4477
		Average	3549.9	4195.5	4486.2	4309.6	4262.4	4452.3	4447.0
nrp-g2-0.5	Bound:6129	Best	5433	6219	**6502**	6322	6398	6411	6413
		Average	5359.8	6138.4	6470.2	6283.3	6238.3	6384.2	6385.4
nrp-g3-0.3	Bound:4258	Best	4663	5351	5802	6372	6581	6560	**6593**
		Average	4593.1	5296.6	5736.5	6317.6	6373.9	6492.3	6525.1
nrp-g3-0.5	Bound:7097	Best	7032	7903	8714	9008	9375	9355	**9370**
		Average	6948.1	7849.8	8326.8	8952.1	9272.2	9301.4	9328.4
nrp-g4-0.3	Bound:3120	Best	3386	3951	**4190**	4029	4109	4121	4117
		Average	3313.9	3909.9	4159.0	3998.1	3703.8	4094.0	4095.8
nrp-g4-0.5	Bound:5350	Best	5041	5751	6030	5978	6054	6052	**6056**
		Average	4991.6	5721.3	5986.5	5942.5	5994.3	6040.2	6040.7

We compare various single objective approaches include multi-start simulated annealing(MSSA), genetic algorithm (GA) and the Backbone-based Multilevel Algorithm (BMA); while multiobjective approaches include NSGA-II, MOEAD, gMOMA-SA(without adaptive mechanism) and uMOMA-SA(with adaptive mechanism based on utility). Table2 shows the best and average of the best solution, obtained by 7 above-mentioned approaches.

Each NRP instance may have two different cost ratio. For convenience, the nrp-g1 instance with cost ratio 0.3 is denoted as nrp-g1-0.3. The budget bounds corresponding to the cost ratio are presented in Table2. The results are presented as the best solutions in terms of customers profits within budget bounds found by various approaches. It can also be seen that our proposed uMOMA-SA is able to achieve the best solutions in most NRP instances.