Available online at www.sciencedirect.com

ScienceDirect

Procedia Manufacturing 00 (2019) 000–000

Procedia MANUFACTURING

www.elsevier.com/locate/procedia

47th SME North American Manufacturing Research Conference, NAMRC 47, Pennsylvania, USA

Optimal composition of tasks in cloud manufacturing platform: a novel hybrid GWO-GA approach

Hamed Bouzary[a], F. Frank Chen[a,*], Mohammad Shahin[a]

[a]*The University of Texas at San Antonio, 1 UTSA circle, San Antonio, TX, 78249 USA*

* Corresponding author. Tel.: +1-210-458-5382; fax: +1-210-458-6504. *E-mail address:* ff.chen@utsa.edu

Abstract

Emerging cloud manufacturing paradigm aims towards providing highly integrated solutions via enabling cooperation between distributed manufacturing resources and capabilities. Implementation of this groundbreaking idea, however, is facing serious challenges springing from the currently centralized industrial structures. In order to help it make its way out, researchers have to address a number of pivotal issues in this regard. Service composition and optimal selection (SCOS), which tackles the problem of optimally selecting and combining available resources into a composite service is one of them. To deal with this NP-hard problem, we developed a novel hybrid algorithm based on the recently-introduced grey wolf optimizer (GWO) in which evolutionary operators are also embedded into the hunting mechanism of the basic algorithm. This approach not only makes it possible to adapt an algorithm with continuous structure such as GWO to solve a combinatorial problem such as SCOS, but also empowers it with providing higher exploration through crossover and mutation operators. Experiments conducted clearly proves the superior performance of the proposed algorithm over existing discrete variations of GWO and genetic algorithm, especially in large-scale SCOS problems.

Keywords: service composition, cloud manufacturing (CMfg), QoS (quality of service), grey wolf optimizer (GWO)

1. Introduction

Today's rapidly changing and highly customized market is metamorphosing the traditional product-oriented manufacturing into a service-oriented structure. The service-oriented manufacturing makes it possible for customers to involve in the whole product life cycle and in the manufacturers' side, empowers them by providing more effective collaboration both within their facilities and across different partners. Cloud manufacturing (CMfg) has been proposed as the panacea to implement this groundbreaking idea in recent years. Nonetheless, to fully achieve this goal, there remains significant issues with regard to enabling technologies [3-7], trust evaluation [8], task and resource description [9,10], scheduling [11-13] and service composition, among which the last one, is the key to offering composite services through the above-mentioned collaboration among partners. This has to be done by selecting the optimal resource for each sub-task out of the existing candidates at the resources pool and composing them to achieve the highest quality of service (QoS). The focus of this research is to solve this problem through implementing a hybridized variation of recently developed grey wolf optimizer (GWO) in which crossover and mutation operators are also embedded in the structure of the basic algorithm in an attempt to achieve better exploration power.

There has been a growing interest in service composition and optimal selection (SCOS) problem in the context of CMfg, targeting different aspects of it including, among other things, developing and applying more advanced

algorithms, consideration of transportation in the modeling, trust evaluation and correlation awareness. In terms of implemented algorithms, the list covers a wide spectrum of metaheuristics from various variations of improved and/or hybridized GA (genetic algorithm) [14] and PSO (particle swarm optimization) [15] to AC (ant colony) [16] and IWO (invasive weed optimization) [17].

Grey wolf optimizer (GWO) is a recently introduced swarm intelligence algorithm mimicking the social hierarchy found in the wolf packs and their haunting behavior. The algorithm gained acknowledgment from researchers in various fields and has been applied to solve different problems such as parametric optimization of abrasive water-jet machining processes [18], optimization of production time in the multi-pass milling process [19], optimal control of thermal systems [20], economic load dispatch problem of electric power systems [21] , feature selection [22] and parameter selection in surface waves [23]. Generally, this algorithm has proved to produce competitive or superior results compared to other well-established algorithms such as PSO and DE (differential evolution), especially in terms of exploitation. However, this exploitative mechanism, which can be tracked down to its searching mechanism (called as “hunting”), makes the GWO algorithm prone to stagnation in local solutions. To overcome this premature convergence and achieve a better balance between exploration and exploitation, we propose incorporating mutation and crossover operators into the GWO, which eventually combines the merits of GWA and GA both.

The remainder of this paper is organized as follows. First, the SCOS problem is briefly explained in Section 2. The GWO algorithm is presented in Section 3 and then the hybridized GWO-GA algorithm along with its embedded operators to solve the SCOS problem are discussed in Section 4. Simulation experiments and results analysis are provided in Section 5. Finally, Section 6 concludes the paper.

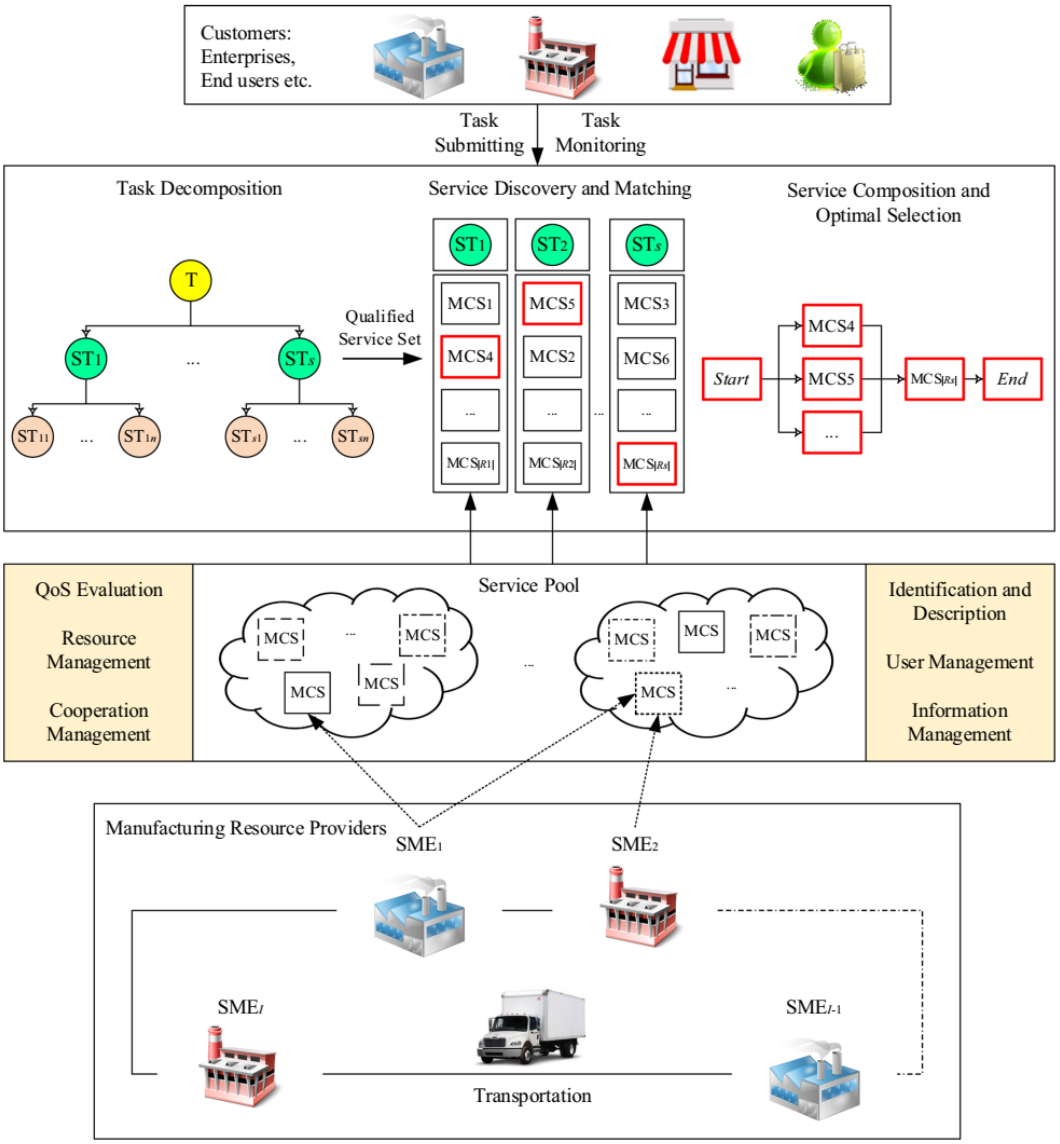

Figure 1: SCOS in CMfg Platform [1,2]

2. Description of SCOS problem

As one can see from Figure 1, the process of matching demands and resources should be studied from the very beginning after tasks' submission. Generally speaking, this process can be broken down into three steps as depicted in Figure 2. (1) Task decomposition: First of all, the submitted task which can be considered as a complex manufacturing project, should be decomposed into n subtasks denoted by $ST_1, ST_2, ..., ST_n$ according to certain logic rules. (2) Service discovery: Then, for subtask ST_i and based on functional requirements, a number of services will be retrieved by matching algorithms. These algorithms are mostly based on semantics similarity between subtasks and resources' descriptions. After this process, each subtask can be associated with a number of candidate manufacturing services ($MS_1, MS_2, ..., MS_m$). (3) Service composition and optimal selection: Finally, it becomes the time for service composition and optimal selection which can be defined as selecting one candidate service from each corresponding candidate set to compose the service while ensuring the overall QoS is optimal. Both aggregation models and normalization formulas are given in Bouzary and Chen [24] in a detailed manner.

The mathematical model of the SCOS problem is given as follows:

$$Max(Qos) = Max\sum w_k \times Norm(Q_k) \quad (1)$$

st.

$$agg(Q_k) \geq Q_{k0} \quad \text{if } Q_k \text{ is a positive attribute} \quad (2)$$

$$agg(Q_k) \leq Q_{k0} \quad \text{if } Q_k \text{ is a negative attribute} \quad (3)$$

$$\sum_{k=1}^{r} w_k = 1 \quad (4)$$

Where w_k is the weight of the k th attribute, $Norm(Q_k)$ is the normalized value for the k th QoS attribute, r is the number of attributes, Q_{k0} is the lower bound value of QoS for the k th attribute and $w_k \in [0,1]$.

3. Grey wolf optimizer

Grey wolf optimizer (GWO) is a recently-developed swarm intelligence algorithm which was introduced by Mirjalai, et al in 2014 [25]. Since then, it has gained acknowledgment and has been repeatedly implemented to solve various optimization problems. It mimics the leadership hierarchy and hunting mechanism of grey wolf packs in nature. The behavior of wolves is simulated through three main steps including hunting (searching for prey), encircling prey, and attacking prey. The pack is divided into four categories. The fittest solution is considered as the alpha (α). Second best and third best solutions are categorized as beta (β) and delta wolves (δ), respectively. The rest of the population are assumed to be omega (ω). Search for prey is led only by α , β and δ wolves and ω wolves have to follow these three [25]. The mathematical interpretation of encircling prey, hunting and attacking prey as the three pillars of GWO algorithm follows immediately.

3.1.1. Encircling prey

Grey wolves tend to encircle prey before the hunt. This behaviour can be mathematically simulated through equations (5-8).

$$\vec{D} = \left|\vec{C}.\vec{X}_p(t) - \vec{X}(t)\right| \quad (5)$$

$$\vec{X}(t+1) = \vec{X}_p(t) - \vec{A}.\vec{D} \quad (6)$$

$$\vec{A} = 2\vec{a}.\vec{r}_1 - \vec{a} \quad (7)$$

$$\vec{C} = 2.\vec{r}_2 \quad (8)$$

where t represents the current iteration, $\vec{X}_p$ indicates the position of the prey, and $\vec{X}$ represents the position vector of a wolf. $\vec{A}$ and $\vec{C}$ are coefficient vectors which make it possible for wolves to reach different locations around the prey. Essentially, D equals to the distance of wolf from the prey. Components of $\vec{a}$ linearly decrease from 2 to 0 over the course of iterations and r_1, r_2 are random vectors in [0,1] [25].

3.1.2. Hunting

Despite the fact that hunting is usually guided by the alpha, beta and delta, in an abstract search space, the location of the optimum (prey) is unknown. In order to mathematically simulate the hunting process, it is assumed that α , β and δ have better knowledge regarding the possible location of prey. Therefore, position of three best individuals found so far are saved and position of every other wolf will be updated according to these three solutions. This can be achieved using formulas 9-11 [25].

$$\vec{D}_\alpha = \left|\vec{C}_1.\vec{X}_\alpha - \vec{X}\right|, \vec{D}_\beta = \left|\vec{C}_1.\vec{X}_\beta - \vec{X}\right|, \vec{D}_\delta = \left|\vec{C}_1.\vec{X}_\delta - \vec{X}\right| \quad (9)$$

$$\vec{X}_1 = \vec{X}_\alpha - \vec{A}_1.\vec{D}_\alpha, \vec{X}_2 = \vec{X}_\beta - \vec{A}_2.\vec{D}_\beta, \vec{X}_3 = \vec{X}_\delta - \vec{A}_3.\vec{D}_\delta \quad (10)$$

$$\vec{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \quad (11)$$

Through equations (12-14) below, alpha, beta, and delta estimate the position of the prey, and other wolves update their positions randomly about the prey [25].

3.1.3. Attacking prey and searching for prey

The process of approaching the prey is simulated through decreasing value of $\vec{a}$. In this way, while random values of $\vec{A}$ change between -1 and 1, the next position of a wolf can be somewhere between its current position and the position of the prey. On the other hand, any random value of $\vec{A} > 1$ or $\vec{A} < -1$, leads to an opposite behaviour called "searching for prey" which provides better global search and helps to avoid getting stuck in local optimal solutions.

Through the above-mentioned operators, GWO guides its search agents to gradually update their positions based on the position of best three agents and eventually obtain a near-optimal solution. A more detailed explanation on this algorithm can be found in [25].

4. Hybrid GWO-GA algorithm

As mentioned before, the standard GWO is designed to solve optimization problems with continuous solution space, since the updated position of wolves could take any possible value within the solution space. However, the SCOS problem is a typical combinatorial problem. Therefore, a discrete hybrid variation of grey wolf optimizer equipped with evolutionary operators, has been developed to pave the way for making use of this powerful algorithm in the context of SCOS problem, and any other combinatorial problem in general.

4.1. Representation of SCOS problem

The solution for the SCOS problem is a composition of cloud manufacturing candidates for a task. In this case, a wolf's position vector represents a feasible composition. For instance, a solution for a composition involving 5 subtasks and each subtask having 10 available candidate services, can be represented in the form of a string as shown below.

2	10	3	7	1

4.2. Embedded evolutionary operators

Solving a combinatorial optimization problem is very different from solving a problem with a continuous solution space. In a continuous search space, the search agents of GWO are able to update their position vectors according to the hunting process mimicked by equations (12-14). In a discrete space, however, the position of wolves cannot be updated likewise, since the position vectors can only take on discrete values. As a result, there have been some attempts to adapt the algorithm to solve problems with discrete solution spaces. For instance, Emary et al. [22] have developed a binary variation of GWO algorithm through employing a sigmoid function to solve the feature selection problem. However, in case of SCOS problem, we did not find the same approach effective enough, which urged us to develop another GWO-based approach by adding evolutionary operators into the structure of the standard algorithm. This approach can, to a great extent, improve the balance between exploration and exploitation, yet preserving the advantages of both GWO and GA algorithms. All the considered modifications intended to adapt GWO to the SCOS problem, as well as the added crossover and mutation operators are presented as follows.

4.2.1. Range checking

Updating the position of wolves during the hunting process using equations (9-11), does not guarantee that the value of every direction does not exceed the bounds of the problem. As a result, after attaining the discrete values of each direction through rounding toward zero the updated values of positions obtained from equation (9-11), an additional step is needed to be considered to return back the exceeding values into the boundaries of the problem. In the case of SCOS problem, upper bound denoted by ub, equals to the number of available candidates for each subtask, and lower bound, denoted by lb, is equal to 1. Equation (12) is applied to achieve this goal.

$$\vec{X}_i = \lfloor \vec{X}_i \rfloor.(\overline{u+l}) + ub.u + lb.l \tag{12}$$

Where $u = \lfloor \vec{X}_i \rfloor > ub$ and $l = \lfloor \vec{X}_i \rfloor < lb$.

4.2.2. Selection operator

Roulette wheel selection is a widely-used selection operator applied in several proposed genetic algorithms solving the service composition problem [26]. The basic idea behind this operator is this fact that the better the fitness of an individual, the larger the probability of its survival and mating. The same selection operator is applied here to select the parents that will go through the crossover and mutation operators.

To select an individual for crossover and mutation processes out of a population with size n (size of wolf pack in our case) we use the following steps:

Step 1. Generate a random real number (r) ∈ [0, 1].

Step 2. If r ≤ p_1, select the first wolf in the population w_1, otherwise; select the i^{th} wolf w_i (i=1,2,…, n) such that p_{i-1} < r ≤ p_i, where p_i is the cumulative probability for each wolf w_i in the current population (pack) which will be calculated according to equation (13).

$$\begin{cases} p_1 = \dfrac{f_i}{\sum_{j=1}^{n} f_j} \\ p_i = p_{i-1} + \dfrac{f_i}{\sum_{j=1}^{n} f_j} \quad if \quad 2 \le i \le n \end{cases} \tag{13}$$

Where f_i is the fitness of individual i in the population [27].

4.2.3. Crossover operator

Every selected pair of parents will go through the crossover operator and a pair of new individuals (children) will be produced by combining their genes. Here, we have employed a two-point crossover operator similar to some other research works on service composition [28]. During this operator, two crossover points are randomly chosen in the two parents. Then, the subsequent genes, between these two cutting points of each parent are exchanged from a chromosome to another one (see Figure 2 (a)).

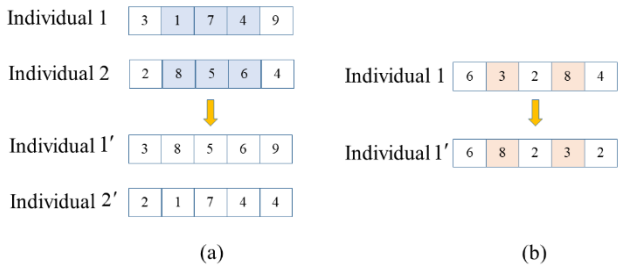

Figure 2: Evolutionary operators in genetic phase. a) two-point crossover, b) swap mutation

4.2.4. Mutation operator

Mutation operator is intended to provide exploration and to avoid local optimum stagnation through maintaining the

diversity of population which will be achieved by making a small and slow genetic frequency change in the generated chromosomes. Swap mutation is used here in which two genes in the chromosome of an individual will be randomly selected and then their values will be interchanged (see Figure 2(b)).

Now that crossover and mutation operators are applied, an overlapping population will emerge, where parents and offspring are merged, and the best individuals from this union will form the next iteration's population. In fact, this merged population will be sorted in descending order and the first best n individuals (equal to the size of the wolf pack) will be selected and replaced as the initial population for the next iteration. The whole process from the initialization up until this point, will be repeated until a stopping criteria is met. Exceeding a specific CPU time, reaching a predefined number of iterations, a certain number of iterations without any improvement in the fitness of the fittest solution and converging to a specific fitness value are among the most popular termination criteria for metaheuristic algorithms. In this study, we have used the maximum number of iterations as the stopping criterion for our proposed algorithm. A summary of the embedded procedures of the proposed hybrid GWO algorithm is organized step by step in Algorithm 1.

Algorithm1: Hybrid GWO algorithm for SCOS problem

(1) ***Initialize*** (t = 0)

step1: Randomly generate an initial population $X_i(i=1,2,...,n)$, where n denotes the size of the wolf pack

(2) ***Repeat*** (t = 1 to *maxiter*)

Step2: Initialize vectors $\vec{a},\vec{A},\vec{C}$ according to Equations (7-8)

Step3: Calculate the fitness values of each wolf

Step4: Record the position of first three best wolves as X_α, X_β and X_δ, respectively and update accordingly

Step5: Update the position of wolves through hunting process using Equations (9-11)

Step6: Discretize the elements of position vectors obtained in step5 ($\vec{X}(t+1)$) using the floor function

Step7: Perform range checking for values obtained in Step6 and map out of bound values (if any) into the allowed space according to Equation (12)

Step8: Select $P_c \times n$ number of parents for crossover operation and $P_m \times n$ number of parents for mutation operator according to Equation (13), where P_c and P_m are probability of crossover and mutation, respectively

Step9: Perform the two-point crossover on the selected parents

Step10: Perform the swap mutation operator on the selected individuals for mutation

Step11: Merge the populations created in Steps 9 and 10 with the original population obtained after Step7

Step12: Sort the Step11's population in descending order and only keep the first fittest n individuals. Step13: Replace this new population as the initial population for the next iteration

Step14: Memorize the best solution achieved so far. Assign $t+1 \to t$.

Step15: If stop criterion is not reached, go to Step2.

Step16: Output the best solution

(3) ***Output***

5. Experiments and results

In this section, we investigate the performance and efficiency of the proposed hybrid GWO algorithm for SCOS problem, and its potential for solving other combinatorial problems as well. To this end, this section conducts performance comparison with two other metaheuristics under the following testing scenarios as well as different user QoS preferences:

1) Scenario1, in which the number of subtasks N remains fixed as 10, while the number of available candidate services M in each CMCSS various in the range of 50 to 500 with an increment of 50.
2) Scenario2, where the number of candidate services M is fixed as 50 for all the cases, while the number of subtasks varies from 5 to 20 with an increment of 5.

In order to substantiate the superior performance of the proposed hybrid GWO algorithm, it is compared with genetic algorithm and also an existing discrete variation of GWO algorithm introduced by Emary et al. [22], in which the algorithm is implemented to solve the feature selection problem. In fact, Emary et al. developed a binary variation of the GWO algorithm using a logistic transfer function and here we have adapted the same idea to achieve a discrete algorithm using the following function:

$$\begin{cases} \text{Update the associated direction of the individual (equation (14))} & \text{if: } sigmoid\left(\frac{X_1+X_2+X_3}{3}\right) \geq rand \\ \text{Do not change the value of associated direction} & \text{if: } otherwise \end{cases} \quad (14)$$

Equation (18) is adapted from the equation (24) of [22] (other steps of the algorithm are implemented exactly according to Algorithm 3 from Page 374 of [22]). From now on, this algorithm is going to be denoted as DGWO in this paper. In addition, GA, as a well-established and widely used intelligent algorithm to solve the SCOS problem, is also implemented and compared with the proposed approach. Without loss of generality, all of the QoS values of available candidates are randomly generated with a uniform distribution within the range [0.7, 0.9]. Four QoS attributes of time, cost, reliability and availability are considered to

build the objective function, while their weights (w_1, w_2, w_3 and w_4, respectively) are all set to 0.25. In addition, considering the fact that every composite MCS path, including sequential, parallel, circular and selective connections, can eventually boil down into a certain combination of equational paths [29], only the sequential aggregation model is considered here without loss of generality. Having the stochastic nature of metaheuristic algorithms in mind, each experiment has been run 10 times and the average and standard deviation are reported to provide a fair comparison. The proposed and compared algorithms are coded in MATLAB R2013b on a computer with 2.3 GHz processor and 8 GB of RAM and a 64-bit Windows 10 operating system. As one can see from Table 1, grey wolf optimizer has the advantage of needing very few parameters to be tuned. The special parameters of these three algorithms are set as Table 1.

Table 1: parameter settings

Algorithm	Initial population	Crossover probability	Mutation probability	x-value of Sigmoid's midpoint (x_0)	Steepness of the logistic function (k)
HGWO	20	0.15	0.85	-	-
DGWO	20	-	-	0.5	10
GA	20	0.15	0.85	-	-

5.1. Performance of the proposed hybrid GWO algorithm (HGWO) compared with DGWO and GA

As our first evaluation, the aggregated fitness values of the best composed services are obtained according to scenario 1 and the average and standard deviation values are reported as each experiments has been repeated 10 times. As one can see from Figure 3(a), the average values for HGWO slightly decreases as the number of available candidates M increases from 50 to 150 and also from 150 to 300, which can generally be attributed to the fact that as the size of the solution space grows, the probability of finding an optimal solution drops. However, the results indicate a positive trend over the span of [150-200] M (number of candidates), which can mark the optimal size of SCOS problem to be solved by our proposed HGWO algorithm as M=200, when N (number of subtasks) is equal to 10. Moreover, in all of the scenario 1 cases , HGWO outperform GA and DGWO in terms of the obtained optimal QoS value and this superiority compared with DGWO, becomes more notable in the large-scale problems (average fitness value equal to 0.65 for HGWO, versus value of 0.41 for DGWO in case of M=300 and N=10). In addition, GA proves a more stable performance as the number of candidates increases and according to Figure 3(a), DGWO is the one that experiences the most significant deterioration in performance as we move towards large-scale problems.

According to Figure 3(b), the optimal QoS fitness values obtained by all three algorithms, decrease drastically as the number of subtasks increases. However, HGWO still outperforms the others with an average fitness of 0.61, followed by GA which has obtained an average fitness of 0.52 and DGWO with the worst performance among them with an average QoS equal to 0.48. Comparing Figures 3(a) and 3(b), we can notice that the performance of these algorithms are heavily influenced by an increment in the number of subtasks, while that does not change meaningfully as the number of available candidates goes up. This can be due to the fact that the size of the solution space associated with SCOS problem can be obtained through an exponentiation operation b^n, where the base b is equal to the number of available candidates M, and the exponent *n* would be equal to the number of subtasks N. As a result, any increase in the number of subtasks will lead to a bigger impact on the size of the solution space and subsequently, a higher reduction in the probability of finding an optimal solution in a certain number of iterations.

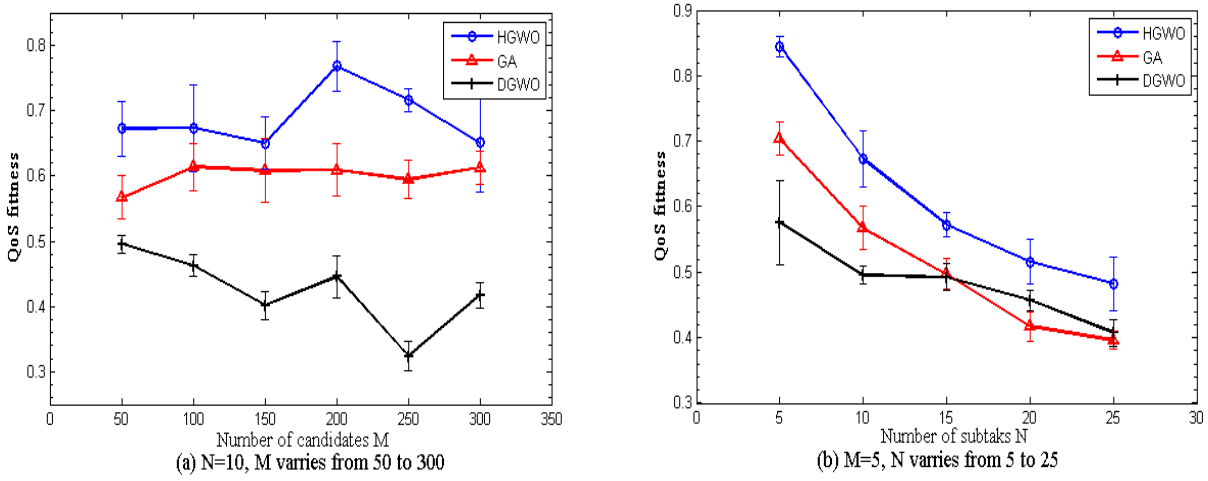

Figure 3: The proposed HGWO compared with DGWO and GA for different scales: a) N = 10, M varies from 50 to 300, b) M = 5, N varies from 5 to 25

5.2. Effect of QoS preferences

The corresponding weights of the four attributes comprising the QoS function were all set to 0.25 in the above-mentioned simulations. However, this is not always the case with regard to customer preferences. As a result, in order to investigate the performance of the proposed HGWO algorithm when dealing with different QoS preferences, 10 different random QoS vectors are considered as shown in Table 2. The experiments are conducted assuming a SCOS problem consisting of 10 subtasks (N=10) and 200 available candidate services for each subtasks (M=200. The results are presented in Table 3. As one can observe from Table 3, in 8 cases (out of total of 10 vectors), the optimal QoS value obtained by HGWO is higher than what is found by GA and DGWO. The average optimal fitness value obtained by HGWO equals to 0.6444, followed by 0.5672 for GA and 0.4471 for DGWO. In addition, as the weights associated with time and cost increase (or weights associated with reliability and availability decrease equivalently), the optimal QoS fitness value decreases drastically in all three algorithms. This happens in vector numbers 3, 4, 6 and 8 in which almost all three algorithms obtain their lowest QoS value. On the contrary, in vector numbers 1, 2, 7 and 10, with higher availability and reliability weights, the algorithms achieve relatively higher fitness values. This can be attributed to the fact that, in a sequential composition, availability and reliability QoS values of the collaborating services will be multiplied together, while the QoS values associated with time and cost will be added together (see Table 1 in [24]). As a result, an equal decrease in the QoS value of availability or reliability, will lead to a higher drop in the value of aggregated QoS, as opposed to the amount of drop resulting from an equal decrease in time and cost Qos. In summary, the results prove the superior performance of HGWO compared to GA and DGWO when facing different user QoS preferences.

Table 2: Different user QoS preference vectors

Vectors	W_1	W_2	W_3	W_4
1	0.18	0.16	0.33	0.33
2	0.11	0.28	0.24	0.37
3	0.29	0.32	0.11	0.28
4	0.46	0.11	0.08	0.35
5	0.34	0.12	0.22	0.32
6	0.35	0.38	0.22	0.05
7	0.1	0.17	0.56	0.17
8	0.35	0.1	0.4	0.15
9	0.13	0.16	0.4	0.31
10	0.15	0.36	0.25	0.24

Table 3: Comparison results of different user QoS references

Vectors	HGWO	GA	DGWO
1	**0.64417**	0.575928	0.459644
2	**0.78924**	0.575982	0.458107
3	0.54774	**0.557496**	0.478126
4	**0.55317**	0.552339	0.434594
5	**0.58366**	0.564813	0.441499
6	**0.61126**	0.606447	0.363826
7	**0.75292**	0.553266	0.501433
8	0.54141	**0.596376**	0.393712
9	**0.66592**	0.557757	0.47542
10	**0.7550**	0.532116	0.46469
Average	**0.644449**	0.567252	0.4471051
Superiority rate	**80%**	20%	0

6. Conclusions and recommended future work

Service composition and optimal selection (SCOS) is one of the pivotal issues in the realization of CMfg paradigm. Growing number of available services compounds the problem since a bigger solution space decreases the probability of searching that in polynomial time. As a result, an increasing number of papers have dedicated their efforts to develop novel suitable algorithms that can deal with the problem more efficiently. For addressing such SCOS problems with large solution spaces, a novel hybrid grey wolf optimizer algorithm with evolutionary operators has been proposed in this paper. Experiments conducted clearly proves the superior performance of the proposed algorithm over existing discrete variations of GWO and genetic algorithm, especially in large-scale SCOS problems. In the future work, we aim at extending the proposed algorithm with Pareto-based approaches to solve multi-objective SCOS problem, instead of using simple additive weighting method.

References

1. Akbaripour H, Houshmand M, van Woensel T, Mutlu N. Cloud manufacturing service selection optimization and scheduling with transportation considerations: mixed-integer programming models. The International Journal of Advanced Manufacturing Technology 2018; 95 (1):43-70.
2. Akbaripour H, Houshmand M. Service composition and optimal selection in cloud manufacturing: landscape analysis and optimization by a hybrid imperialist competitive and local search algorithm. Neural Computing and Applications 2018:1–22.
3. Ostasevicius V, Jurenas V, Markevicius V, Gaidys R, Zilys M, Cepenas M, Kizauskiene L. Self-powering wireless devices for cloud manufacturing applications. The International Journal of Advanced Manufacturing Technology 2016;83 (9):1937-1950.
4. Zhong RY, Lan S, Xu C, Dai Q, Huang GQ. Visualization of RFID-enabled shopfloor logistics Big Data in Cloud Manufacturing. The International Journal of Advanced Manufacturing Technology 2016;84 (1):5-16.
5. Zarreh A, Saygin C, Wan H, Lee Y, Bracho A. A game theory based cybersecurity assessment model for advanced manufacturing systems. Procedia Manufacturing 2018; 26:1255-1264.
6. Bracho A, Saygin C, Wan H, Lee Y, Zarreh A. A simulation-based platform for assessing the impact of cyber-threats on smart manufacturing systems. Procedia Manufacturing 2018;26:1116-1127.
7. Sahba, Amin, John J. Prevost. "Hypercube based clusters in Cloud Computing." In World Automation Congress (WAC), 2016, pp. 1-6. IEEE,

2016.
8. Kai Y, Ying C, Fei T. A trust evaluation model towards cloud manufacturing. International Journal of Advanced Manufacturing Technology 2016;84 (1-4):133-146.
9. Wang T, Guo S, Lee C-G. Manufacturing task semantic modeling and description in cloud manufacturing system. The International Journal of Advanced Manufacturing Technology 2014;71 (9):2017-2031
10. Sahba R, Ebadi N, Jamshidi M, Rad P. Automatic Text Summarization Using Customizable Fuzzy Features and Attention on the Context and Vocabulary. In: World Automation Congress (WAC), 2018. pp 1-5.
11. Liu Y, Wang L, Wang XV, Xu X, Zhang L.Scheduling in cloud manufacturing: state-of-the-art and research challenges. International Journal of Production Research 2018: 1-26.
12. Zhou L, Zhang L, Zhao C, Laili Y, Xu L. Diverse task scheduling for individualized requirements in cloud manufacturing. Enterprise Information Systems 2018;12 (3):300-318.
13. Krishnaiyer K, Chen FF, Bouzary H. Cloud Kanban Framework for Service Operations Management. Procedia Manufacturing 2018;17:531-538.
14. Li H-F, Zhao L, Zhang B-H, Li J-Q. Service Matching and Composition Considering Correlations among Cloud Services. In: Systems, Man, and Cybernetics (SMC), 2015 IEEE International Conference on, 2015. IEEE, pp 509-514
15. Zheng H, Feng Y, Tan J. A fuzzy QoS-aware resource service selection considering design preference in cloud manufacturing system. International Journal of Advanced Manufacturing Technology 2016;84 (1-4):371-379.
16. Wei X, Liu H. A cloud manufacturing resource allocation model based on ant colony optimization algorithm. Int J Grid Distributed Comput 2015; 8 (1):55-66
17. Bouzary H, Chen FF, Krishnaiyer K. A modified discrete invasive weed algorithm for optimal service composition in cloud manufacturing systems. Procedia Manufacturing 2018;17:403-410.
18. Chakraborty S, Mitra A. Parametric optimization of abrasive water-jet machining processes using grey wolf optimizer. Materials and Manufacturing Processes (2018);33 (13):1471-1482.
19. Khalilpourazari S, Khalilpourazary S. Optimization of production time in the multi-pass milling process via a Robust Grey Wolf Optimizer. Neural Computing and Applications 2018; 29 (12):1321-1336.
20. Sharma Y, Saikia LC. Automatic generation control of a multi-area ST – Thermal power system using Grey Wolf Optimizer algorithm based classical controllers. International Journal of Electrical Power & Energy Systems 2015;73:853-862.
21. Kamboj VK, Bath SK, Dhillon JS. Solution of non-convex economic load dispatch problem using Grey Wolf Optimizer. Neural Computing and Applications 2016;27 (5):1301-1316.
22. Emary E, Zawbaa HM, Hassanien AE. Binary grey wolf optimization approaches for feature selection. Neurocomputing 2016;172:371-381.
23. Song X, Tang L, Zhao S, Zhang X, Li L, Huang J, Cai W. Grey Wolf Optimizer for parameter estimation in surface waves. Soil Dynamics and Earthquake Engineering 2015;75:147-157.
24. Bouzary H, Chen FF. Service optimal selection and composition in cloud manufacturing: a comprehensive survey. The International Journal of Advanced Manufacturing Technology 2018;97(1):795-808.
25. Mirjalili S, Mirjalili SM, Lewis A. Grey Wolf Optimizer. Advances in Engineering Software 2014;69:46-61.
26. Wang D, Yang Y, Mi Z. A genetic-based approach to web service composition in geo-distributed cloud environment. Computers & Electrical Engineering 2015; 43:129-141.
27. Seghir F, Khababa A. A hybrid approach using genetic and fruit fly optimization algorithms for QoS-aware cloud service composition.Journal of Intelligent Manufacturing 2016:1-20.
28. Wu Q, Zhu Q, Zhou M. A correlation-driven optimal service selection approach for virtual enterprise establishment. Journal of Intelligent Manufacturing 2014;25 (6):1441-1453.
29. Tao F, Zhao D, Hu Y, Zhou Z. Resource Service Composition and Its Optimal-Selection Based on Particle Swarm Optimization in Manufacturing Grid System. IEEE Transactions on Industrial Informatics 2008;4 (4):315-327.