Priority-based Selection of Individuals in Memetic Algorithms for Distributed Data-intensive Web Service Compositions

Soheila Sadeghiram, *Member, IEEE,* Hui Ma, *Member, IEEE,* and Gang Chen, *Member, IEEE*

***Abstract*—In distributed computing, *Web Service Composition (WSC)* leads to the effective reuse of existing services and produces added value. WSC must fulfil functional requirements and optimise *Quality of Service (QoS)* attributes, simultaneously. *Memetic Algorithms (MAs)* are promising for automatically composing numerous Web services to satisfy the above requirements. *Data-intensive Web services* focus on providing and updating data with a significant volume of data operation and exchange. However, current composition approaches have ignored the impact of data communication and the distribution of services, which significantly affect the performance when applied to the challenging *Distributed Data-intensive Web Service Composition (DDWSC)* problem. Although recent approaches have revealed the usefulness of local search, they have completely overlooked the question of preferring appropriate composition solutions for local search. To address this research issue, we propose a priority-based selection method for the local search that can be consistently integrated with any MA for DDWSC. This enables us to develop state-of-the-art algorithms for DDWSC by explicitly considering the problem-specific, population and solution-related information for choosing a solution. Extensive experimental evaluation using benchmark datasets shows that our proposed method significantly outperforms several recently proposed methods.**

***Index Terms*—Web services, distributed computing, memetic algorithms, priority-based selection, data-intensive Web service composition.**

I. Introduction

Web services are fundamental elements of *Service Oriented Architecture (SOA)*, allowing it to facilitate and expedite the development of distributed software applications [1]. *Web service composition (WSC)* accomplishes a specific task by combining numerous inter-operating, distributed and reusable services [2]. Service compositions must fulfil functional requirements and optimise *Quality of Service (QoS)* attributes, simultaneously. Consequently, distributed *Data-intensive Web service composition (DWSC)* has introduced new research challenges. Given the huge number of services available, manually designing service compositions to achieve optimal QoS is crucial. However, the automated WSC problem is NP-hard [3], [4]. As a result, it is very difficult (or impossible) to determine optimal solutions to large-scale DDWSC problems within a limited time frame [5], [6].

Many studies have been conducted on the *fully-automated* service composition [6], [7], [8], [9], [10], [11], [12], [13], [14], [15]. Fully-automated service composition meets different functional and non-functional requirements by establishing various service-oriented workflows. However, most of the studies have been designed for a centralised service repository [8], [10], [11], [12], [13], [15], [16], [17], [18], and therefore, ignore the impact of distance and data transfer between services on the performance of composite services. Accordingly, those studies are rendered ineffective in generating high-quality composite services when applied to DDWSC [7]. In fact, research has shown that the application of existing composition approaches on DDWSC does not often produce high-quality composite services [7], [19]. The quality of a composite service is affected by the location, QoS, and the quantity of data transferred among component services. Therefore, DDWSC poses new challenges.

Evolutionary Computing (EC) techniques allow for the scalable creation of compositions that efficiently meet both the functional and QoS requirements [20]. *Genetic Algorithm (GA)* [21] is a representative EC technique. With GA, a composite service is typically indirectly represented as a service sequence, which is particularly beneficial because it allows GA operators to be applied without restrictions. A *decoding process*, which will be explained in detail in Section IV-A, ensures the functional correctness of the composite service.

Recently, researchers have enhanced EC algorithms by local search techniques to improve EC effectiveness [22]. This seamless combination gives rise to *Memetic Algorithms (MAs)* [23]. In previous work [6], [7], [24], we have empirically demonstrated that MAs can significantly outperform GAs in solving DDWSC problems. Examples include *distance-guided MA* [25] and *cluster-guided MA* [7] (see Subsection III-E).

Evaluating the fitness of any new solution obtained through local search incurs non-negligible cost and time. Accordingly, local search cannot be applied to all individuals in the population. This limitation leads to serious problems in locating suitable individuals for local search. Many approaches rely merely on performing conventional fitness-based tournament selection. Two solutions are chosen at random and then compared to each other based on their fitness values. The winner is the solution with a better fitness value [26], [27], [28]. Fitness-based selection methods are expected to result in faster convergence but may easily be trapped by inferior local optima [22]. They do not enable local search to improve solutions from different search regions. Therefore, MAs can shift generations towards poor local optima, leading to premature convergence [22].

Adaptive local search variations have been successful for many problems [26], [27], [28], [29], [30], [31]. However, these methods ignore the importance of the effective selection of individuals and its impact on the performance of the search process. For example, selecting individuals with a similar structure may reduce the population diversity and thus increase the difficulty of escaping from local optima. Furthermore, by randomly selecting individual solutions, local search resources

S. Sadeghiram, H. Ma and G. Chen are with the School of Engineering and Computer science, Victoria University of Wellington, New Zealand.

E-mail: {Soheila.Sadeghiram, Hui.Ma, Aaron.Chen} @ ecs.vuw.ac.nz

are unnecessarily wasted on hard-to-improve solutions. On the other hand, some individuals have a higher potential for improvement through local search. For example, a solution that involves a high communication cost can be improved significantly by replacing services in the composition, which cause high inter-service communication costs, with more efficient alternatives. Additionally, the performance of local search depends not only on the fitness of the selected solution but also on its contribution towards evolving the whole population. For example, choosing an individual with a different structure than the best solution in the population (e.g., having communication links with different cost distributions) can help maintain population diversity and thus help avoid local optima. Therefore, it is highly desirable to develop a generalised selection method which selects high-priority solutions for local search through fitness, population, and solution-related measures.

Motivated by this understanding, our proposed method develops a new measure called $SelectionPriority$ (see Equation (9) in Section V) for selecting individuals for local search. This measure establishes the adaptive use of local search during the global search process, which can be widely applied to many MAs. For example, this adaptive local search can be integrated with cluster-guided [7] and distance-guided MAs [6], [25] (see Subsection IV-B) to significantly improve their performance.

Fig. 1 shows our proposed method in the context of the overall architecture of DDWSC. A user sends a service request to the composition agent. At the design stage, the agent is responsible for finding a suitable Web service composition that fulfils the request. The Web service composition agent accesses the distributed service repository to retrieve functional features, QoS, location and data information. Different service providers have created Web services and registered them in the repository. The Web service composition agent employs proposed adaptive MAs in this paper to generate a high-performance composite service.

At the execution stage, the execution engine can execute the recommended composite service and send the outputs to the user. Note that parallel data processing can be performed internally within a data-intensive Web service using a big data architecture such as Hadoop (implementing and executing the composite solution is outside the scope of the paper).

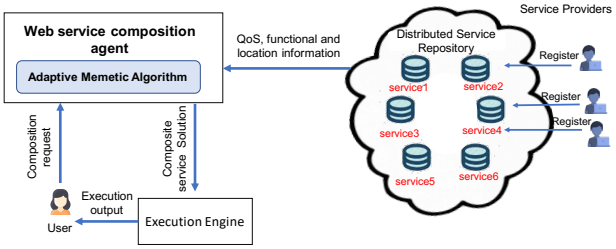


Fig. 1: Adaptive MAs for DDWSC.

The primary contributions of this paper are:

1) Two new measures, *novelty* and *improvability*, are proposed to prioritise solutions in a population. Specifically, the *novelty* is measured by comparing statistical values over all communication links in a solution with other archived solutions. The *improvability* is measured by the maximum possible improvement in fitness (the execution time and cost of a composite service) achievable by removing the *bottleneck link* in the solution.
2) To enhances the most widely used fitness-driven selection methods for local search, a novel priority-based selection method (i.e., $SelectionPriority$) is proposed. In $SelectionPriority$, we jointly consider the solution, population and problem-related information to select appropriate solutions, adaptively. To this aim, we use the *novelty*, *improvability* and fitness measures (fitness is obtained through the weighted sum of the execution cost and time of the composite service). The priority of a solution depends not only on its fitness but also on its similarity to other solutions. Therefore, we adaptively adjust the probability of selecting any solution based on the population. No existing work has ever studied the use of similar measures to guide the local search. Furthermore, this selection method is widely applicable to many memetic algorithms (MAs).
3) Cutting-edge adaptive MAs for DDWSC (i.e., adaptive cluster-guided, adaptive distance-guided and adaptive Combined cluster-distance-guided MAs (Com-C-D)) are developed by integrating recent MAs for DDWSC with our priority-based selection method. Empirical experiments show that our proposed MAs significantly outperform existing state-of-the-art methods for DDWSC problems, including the abstraction refinement method [18], cluster-guided MA [7] and distance-guided MA [6], as well as their combined version. Additionally, empirical evidence strongly demonstrates that both novelty and improvability measures play irreplaceable roles in $SelectionPriority$ and must be used jointly to achieve high performance. We show that integrating with $SelectionPriority$ helps MAs further enhance their effectiveness and outperform state-of-the-art algorithms.

The rest of this paper is organised as follows: in Section II, we review the existing research on WSC. Subsequently, a background including terminology, problem formulation, representation and decoding process used in fully-automated composition, and the cluster-guided and distance-guided MAs are introduced in Section III. An overview of the proposed method is provided in Section IV. Subsection IV-A discusses the different baseline algorithms and local search techniques. Priority-based selection method and $SelectionPriority$ measure are discussed in Section V. Sections VI and VII present experiment evaluations and discussions, respectively. Finally, we conclude the paper in Section VIII.

II. Related Work

EC-based methods have been widely used to find composite solutions that meet users' requirements [9], [11], [12], [13], [32], [33]. A majority of those methods aim to solve semi-automated WSC [19], [32], [33], [34], [35], [36], [37]. As opposed to fully-automated methods, these methods work on a predefined workflow which embodies a set of abstract tasks and their data dependency. Semi-automated methods reduce the composition problem to the problem of selecting and binding individual Web services to each task in the workflow rather than constructing different workflows from scratch. The disadvantage of these methods is that such workflows must be supplied by the service requester or during design.

Several distinct approaches have been employed to address fully-automated WSC [8], [9], [12], [15], [18], [38]. GraphPlan [39], a well-known AI planning algorithm, has been widely used for WSC [15], [40], [41], [42]. GraphPlan contains two stages: a forward expand stage that constructs a planning graph, and a backward search stage that retrieves a solution. GraphPlan may be combined with other techniques, such

as fuzzy techniques for ranking services [43], and skyline operators for filtering services [44] to reduce the number of services to be considered for the composition. Skyline is an operator frequently used for finding individuals that are not dominated by other individuals [45], [46]. However, since skyline only considers the QoS of individual services, it might filter services which are located very close to other services in DDWSC. Therefore, it fails to produce high-quality solutions because of high communication time and cost. Fuzzy methods also require very careful definitions of rules and parameters. The above work does not consider the distribution of services.

A graph search-based method, called abstraction refinement, is proposed in [18] that requires grouping services according to some functional relationships. Afterwards, the composition is performed using a representative service from each group. The algorithm can create full composite solutions according to the dependency graphs [47]. However, this approach assumes that all services are at the same geographical location.

A group of promising approaches for WSC have employed sequence representations [14], [48], [49], [50], [51], [52]. The sequence representation does not show the final service composition solutions directly. Sequences must be decoded to produce an executable workflow of component services. In comparison to the direct representation, the sequence representation allows the evolutionary process to concentrate on optimising the quality metrics, while the stringent requirement on functional correctness is ensured through the separate decoding process [53]. Further, a previous paper [54] has compared the performance of sequence representation and DAG representation on a very similar problem (fully-automated Web service composition). That work has demonstrated that the sequence representation combined with the decoding process can be more efficient and effective than the DAG representation.

Distributed WSC has been studied previously [16], where the skyline operator creates the initial population of GA for the semi-automated WSC problem in a geo-distributed cloud environment [16]. Their model includes the QoS of Web services and the network delay. That method achieves better results when the network environment is considered during the formulation.

MAs have been designed to address the fully-automated WSC [6], [7], [24], [25]. Among them, [6], [7], [25] deal with DDWSC, which will be used as baselines to compare with our method. A hybrid approach combining Genetic Programming (GP) [55] and Tabu search [56] is proposed in [24]; however, it is an approach designed for a centralised service repository without considering the distribution of services. Therefore, we will not compare [24] with our methods.

To effectively use local search, MAs have been proposed that adaptively adjust parameters. Examples of adaptively adjusted parameters are the probability and neighbourhood size of local search [26], [29], [31], [57], [58], [59], [60], [61], local search depth (i.e., how many neighbours should be examined during the local search) [60], the step size [57] and the probability of accepting worse moves [26]. In [58], authors have combined the local search and the concept of search space partitioning used by cellular memetic algorithms (CMAs). The local search has been carried out adaptively on diverse individuals. However, a key limitation of this work in defining the diversity of solutions is that it does not consider the structural and behavioural characteristics of solutions. This limitation might lead to the *deception* problem [62].

When it comes to selecting previously obtained solutions for local search in MA, most of the existing approaches only choose either the fittest individuals, or through a conventional fitness-driven tournament selection [17], or uniformly at random. For example, some methods adaptively change the tournament size of the selection operator [63], [64], but they do not address which individual to be adaptively selected.

In particular, most of the above approaches fail to consider essential problem-specific information, such as the location information of services. Therefore, in this paper, such information is heavily utilised to develop effective fully-automated MAs for DDWSC.

In this paper, we adaptively select initial individuals (parents) for local search based on the priority-based measure. This measure is defined by the problem-specific information (i.e., novelty and improvability) to discover promising solutions and direct the search toward global optima. We primarily consider two baseline algorithms: cluster-guided [7] and distance-guided [6] MAs to design adaptive MAs. We then compare our adaptive MAs with the *abstraction refinement* method in [18]. Finally, to demonstrate the applicability of our proposed selection method to many MAs, we apply our priority-based selection method to the combination of cluster-guided and distance-guided MAs.

To the best of our knowledge, this is the first fully-automated approach on DDWSC that effectively uses problem-specific information to select individuals for local search. In particular, we propose a new priority-based selection method for local search.

III. Background

In this section, we present a motivating example of the DDWSC in Subsection III-A. We then discuss the fundamental concepts and terminology of DDWSC in Subsection III-B and the objective function in Subsection III-C. Afterwards, the representation of individuals and the decoding are explained in Subsection III-D. Finally, two existing works for DDWSC (i.e., cluster-guided and distance-guided MAs) are introduced in Subsection III-E.

A. Motivating Example

Several examples of DDWSC use cases have been reported in different fields, such as the disaster management system [65], [66], education [67] and government [68]. Further, data-intensive Web services and SOA are used to build the microarray experimental system, which is one of the hottest topics in the genome expression field [69], [70]. Gene expression has been researched for many years where related microarray experiments have been conducted all over the world [71] (e.g., National Cancer Institute[1]). Consequently, a vast amount of microarray data sets are produced by many institutions. Institutions create data-intensive services to share data sets for accessing and analysing purposes. Those services can be reused to generate value-added services (i.e., via service composition [72]).

The *Gene Expression Analysis Services (GEAS)* [73] is an example of a Web service repository which provides several publicly available RESTful Web services. Examples of data-intensive services are microarray data process services, data normalisation services, deferentially expressed genes identification services and functional analysis services [73].

For a given service request, e.g., gene prediction, existing services, in particular, data-intensive Web services can be composed to perform gene prediction. Gene prediction requires

[1] https://ccr.cancer.gov/

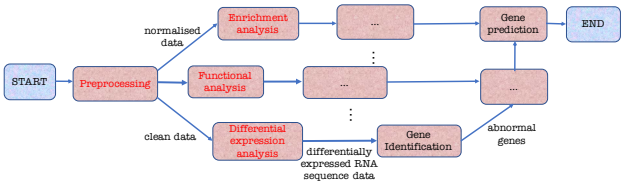


Fig. 2: An example of a Web service composition solution (data-intensive Web services are shown in red).

data analysis using multiple Web services (Fig. 2). However, with the number of available services in the repository, manually finding such a composite service is impractical. To find a composite service that best meets the functional and QoS requirement, it is necessary to explore the huge number of options of selections and compositions. Additionally, service locations should be considered. Specifically, some Web services exchange large volumes of data over a communication network such as the Internet.

B. Problem Formulation and Terminology

Building on the key concepts introduced in [7], this subsection presents definitions for understanding DDWSC problems.

Definition 1. A *service repository* is denoted as $\mathcal{R}$ and consists of a finite collection of Web services S_i, $i \in \{1, ..., n\}$, with n as the number of services in the repository.

Definition 2. A *data-intensive Web service* is a tuple $S = \langle I(S), O(S), D(S), QoS(S), h\rangle$, where $I(S)$ is a set of inputs required for the execution of S, and $O(S)$ is a set of outputs produced by S. $D(S) = \langle d_1, ..., d_m\rangle$ is a set of data items required by S. $QoS(S) = \langle Q^1(S), Q^2(S), ..., Q^n(S)\rangle$ is an associated tuple of quality attributes describing the non-functional properties of S.

In this paper, for each Web service, we suppose $QoS(S) = \langle T_S, C_S\rangle$, indicating the total time and cost required for executing service S, two frequently used quality attributes. h denotes the server which is hosting S.

Definition 3. A *server* is denoted by $h = \langle Lat, Long\rangle$ and its location is identified by the longitude and latitude of the server location. Services in a repository for composition are distributed on servers in a distributed environment. Any two services on the same server have the same location.

Definition 4. A *service composition task* (or a *service request*) is a tuple $\mathcal{T} = \langle I(\mathcal{T}), O(\mathcal{T})\rangle$, where $I(\mathcal{T})$ is a set of inputs provided by a user, and $O(\mathcal{T})$ is a set of task outputs expected by the user to be produced by the composite service.

Definition 5. A *composite service*, $\mathcal{CS}$, is essentially a workflow of component services, indicating the sequence of their execution and the necessary communication between them, to achieve the desired outputs. Two special services can be used to describe the overall composition's input and output: $Start$ with $I(Start) = \varnothing$ and $O(Start) = I(\mathcal{T})$, and End, with $I(End) = O(\mathcal{T})$ and $O(End) = \varnothing$. For a given task $\mathcal{T}$, we aim to find a composition ($\mathcal{CS}$) that takes $I(\mathcal{T})$ and produces $O(\mathcal{T})$. A composition is *functionally correct*, or *feasible*, if (1) the input(s) of each of the component services is (are) fulfilled by the outputs of its proceeding services or by $I(\mathcal{T})$, and (2) the output of the composition satisfies $O(\mathcal{T})$. A direct edge which connects two services S_1 and S_2 in a DAG indicates that there may be a communication link and data dependency between them in the network.

Definition 6. A *communication link* is a link joining two servers, and defined as a tuple $l = \langle h_{head}, h_{tail}, T_L(l), C_L(l)\rangle$, where h_{head} is the source server of l and h_{tail} is the destination server of l. T_L and C_L are the communication time and cost of the communication link. The set of all communication links in a composite service is denoted by $\mathcal{CL}$.

Fig. 3 is an example of a $\mathcal{CS}$, which contains two communication links and three Web services. For simplicity, we only show the associated components of one Web service and one communication link. The overall cost and execution time of a component service S include the cost and time for its corresponding datasets and the cost and time for the service execution. Correspondingly, T_S is the overall service processing time. T_S includes the service execution time, data transfer time from the data centre to the service and the data centre access latency. C_S includes service cost (imposed by the service provider) and data provision cost (determined by the data provider). Bandwidth properties are included in calculating communication attributes. Further details on the calculation of the time and cost can be found in [7].

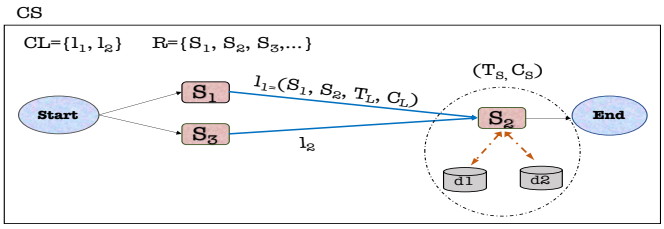


Fig. 3: A composite service and components.

C. Objective Function

Considering a set of distributed services, DDWSC aims to find a service composition with minimal cost and response time (including the cost and time of transferring and accessing data).

Web services in a composition are organised with a range of structures that control their interactions [74]. In this paper, parallel and sequence structures are considered.

The overall execution cost of $\mathcal{CS}$ is obtained by summing up the cost of all component services in the composition (i.e., C_S) and associated cost for edges (communication links) (i.e., C_L) as shown in Equation (1).

$$C_{total}(\mathcal{CS}) = \sum_{p \in P} C_S(p) + \sum_{l \in \mathcal{CL}} C_L(l) \quad (1)$$

where P is the set of all component services and $\mathcal{CL}$ is the set of all communication links in $\mathcal{CS}$.

Suppose that a composition $\mathcal{CS}$ has n paths in its workflow from $Start$ to End. The execution time of path i is denoted by t_i, which includes the time required by each component service on the path and communication links (Equation (2)). T_{total} is the overall response time of $\mathcal{CS}$ and is calculated as the time of the most time-consuming path in the composition, as shown in Equation (3):

$$t_i = \sum_{p \in i} T_S(p) + \sum_{l \in i} T_L(l) \quad (2)$$

where p is a component service on path i and l refers to a communication link on path i.

$$T_{total}(\mathcal{CS}) = \max_{i \in \{1...n\}} \{t_i\} \quad (3)$$

Finally, the quality of $\mathcal{CS}$ is obtained through Equation (4).

$$F(\mathcal{CS}) = \omega \hat{T}_{total} + (1 - \omega)\hat{C}_{total} \quad (4)$$

where $\hat{T}_{total}$ and $\hat{C}_{total}$ are normalised values of T_{total} and C_{total}, respectively. ω, $(0 \leq \omega \leq 1)$, is determined by users who provide the service composition task. Related works [12], [17] in the literature share the same requirement for the users to decide the suitable weights. The normalisation upper bounds for cost and time are obtained by adding the cost and time of all possible links between every pair of services and cost and time of services in the repository, respectively.

The DDWSC objective is to minimise F in Equation (4) by producing a suitable composite service, constructed from the services in repository $\mathcal{R}$.

As shown formerly, the DDWSC problems formulated in this section explicitly considers both the distribution of data and services over distant locations. We must carefully manage the cost and delay caused by massive data communication among component services, which is the primary focus of this paper while building high-quality composite services.

D. Representation of Individuals and the Decoding Process

In this paper, we employ a sequence of services [53] to indirectly represent a composite service in the GA population (see Section II). The sequence representation is used for searching (EC operators are applied on sequences). We consider the DAG representation too, which is used for evaluation. A *decoding process* [53] converts a sequence to the corresponding workflow (DAG) that ensures the functional correctness.

The decoding process is a graph-building algorithm that converts a sequence to a DAG (feasible workflows) that must fulfil the functional requirement of a given task. We use the backward decoding process proposed in [54], where the solution is built from End to $Start$. This decoding process employs the *layer* information of services. The layer information specifies the minimum depth of predecessors of a service between the service and the start node. The layer information is obtained by a simple discovery algorithm [75]. The decoding process is illustrated in Example 1 and Fig. 4.

Example 1. *A sequence with five component services and the input(s) and output(s) of those services are shown in a table (next to the sequence) in Fig. 4. A task is given with $\{a, b\}$ as inputs and $\{g\}$ as the output. The sequence is checked from left to right until the task is satisfied. The decoding process starts with the output of the service task, $End(g, \varnothing)$. First, the input of End should be fulfilled. Therefore, the decoding process checks the sequence to see if any of the services provide the input of End. Since service S_3 outputs $\{g\}$, which is required by End, we connect it to End. Afterwards, the decoding process continues to scan the sequence to look for services that can satisfy $I(S_3)$=$\{e, f\}$. Since $O(S_1) = \{e\}$, and $O(S_4) = \{f\}$, the union of the two services S_1 and S_4 satisfies the input of service S_3, we separately connect both S_1 and S_4 to service S_3 as the predecessor services of S_3. Subsequently, the decoding process looks for service(s) whose outputs satisfy the inputs of service S_1 and S_4. Since service S_5 outputs $\{c, d\}$, which satisfy the inputs required by both service S_1 and S_4, service S_5 is connected separately to both S_1 and S_4 as their predecessor. The process continues until service $Start$ is reached.*

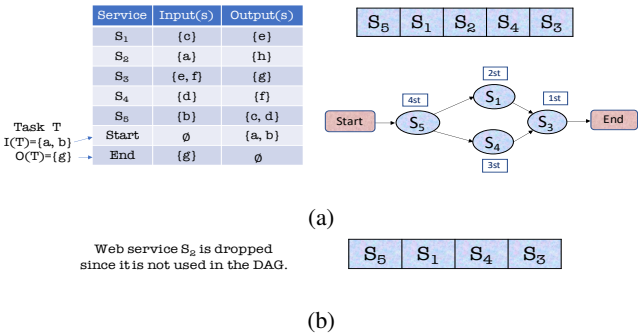


Fig. 4: (a) Example of the backward decoding process (the sequence is traversed as many times as possible), (b) the sequence after being decoded.

The backward decoding process in the above example produces a workflow with both sequential and parallel constructs (e.g., S_1 and S_4 are parallel in Fig. 4). If a functionally correct DAG exists for the corresponding sequence, the decoding is able to build the DAG successfully. Otherwise, the decoding stops after a predefined number of DAG building steps, the sequence is identified as "infeasible" and would be discarded.

In principle, running a reverse depth-first search (DFS) on a DAG (starting from the end node) performs a traversal on the DAG, which produces the sequence. Therefore, every possible composite service for the given service composition request can be obtained by using the indirect representation and the decoding process. This means that our MAs can evolve a service sequence that can be converted to a composite service in the DAG form with optimal performance.

The sequence after decoding is presented in Fig. 4 (b). Redundant services, which have not been used in the DAG, are removed from the sequence. Additionally, crossover, mutation and local search operators may produce sequences with duplicated services. Due to this reason, GA operators can generate offspring sequences with different sizes [17]. Redundant services will be removed from the sequences after the decoding process.

E. Cluster-Guided and Distance-Guided MAs for DDWSC

We mainly consider two baseline algorithms on DDWSC: cluster-guided MA [7] and distance-guided MA [25], which have considered the location of services. This factor hugely affects communication cost and time.

MAs integrate GA with local search techniques to effectively search for composite solutions for DDWSC problems. GAs are population-based global search techniques that start with an initial set of random solutions (i.e., individuals). A fitness measure (see Subsection III-C) is then used to evaluate the individuals. Two popular genetic operators, crossover and mutation, are then utilised to evolve new solutions from parent solutions selected by the conventional tournament selection method [25]. Newly generated solutions are evaluated. This evolutionary search process is iterated until a predefined number of generations is reached. The best solution produced by the final generation is identified as the composition solution.

Cluster-guided MA groups services according to their geographical location. Compositions using GA are initially made in each cluster so that the services in distant locations are

avoided. Afterwards, genetic operators are used to interchange information between clusters and produce new solutions from multiple clusters. Therefore, the cluster-guided MA uses the service location information only during the GA initialisation.

Distance-guided MA adopts the service location information explicitly for enhancing current composite services and to create new services [25]. In particular, problem-specific distance-based crossover and local search operators are used to generate better solutions.

Additionally, cluster-guided and distance-guided MAs employ different local search operators. The former swaps a random service with all other services in the sequence whereas the latter finds alternative services from the repository (i.e., services that offer replaceable functions). Details are explained in Subsection IV-B.

IV. Adaptive MAs for DDWSC With Priority-based Selection for Local Search

In this section, we present an overview of our adaptive MAs and two specific local search techniques to be used in MAs.

A. Adaptive Memetic Algorithms for DDWSC

In this section, we propose adaptive MAs for DDWSC. We develop the $SelectionPriority$ measure to decide the chance of each individual for the local search.

The pseudocode is shown in Algorithm 1. At line 3, the decoding process is performed (see Subsection III-D), after which solutions are evaluated by the fitness function (Section III-C). Solutions with the best fitness value (elite solutions) are copied to the next generation (line 4). Crossover and mutation operators are applied to solutions chosen by a tournament selection (lines 5-8). Each individual has a $SelectionPriority$, which is calculated at line 9 (the $SelectionPriority$ will be explained in detail in Subsection V). The $SelectionPriority$ is then used to select individuals for the local search (line 10). Finally, a local search operator is applied to the selected individual (see Subsection IV-B), and the above steps are repeated until the predefined number of generations is reached.

Algorithm 1: Adaptive MA for DDWSC

Input : $Composition\ Task\ (\mathcal{T})$, $Service\ Repository\ (\mathcal{R})$
Output: $A\ Service\ Composition\ Solution$
1: Generate N permutations of services in $\mathcal{R}$, as the initial population;
2: **while** *the number of iterations not reached* **do**
3: Decode sequences into composite solutions and evaluate them;
4: Find e elite solutions and copy them to the next generation;
5: Select solutions using a fitness-based tournament selection;
6: Apply crossover operator (with the probability of p_c) to generate N offspring solutions;
7: Apply mutation operator (with the probability of p_c) to the offspring solutions;
8: Calculate $SelectionPriority$ of all offspring solutions;
9: Select individuals using the priority-based tournament selection (which employs $SelectionPriority$);
10: Apply priority-based local search (Algorithm 2);
11: **end**
12: **return** $Individual\ With\ Best\ Fitness$;

B. Local Search

Algorithm 2 shows the local search procedure. A composite service (current solution) is chosen for the local search according to its $SelectionPriority$. A critical and difficult part in designing local search for MAs lays in the definition of the neighbourhood structure [30] (line 1 of the algorithm), i.e., how to obtain neighbour solutions from a previously evolved composite service. In this paper, our focus is to explore the effect of selection mechanism for local search on MAs, subject to different neighbourhood structures. To this aim, we consider two different local search operators employed in distance-guided [25] and cluster-guided [7] MAs.

In the cluster-guided MA, a fixed swap point in the sequence with length n is randomly chosen for local search. Afterwards, the neighbourhood is generated by going through the sequence, each time swapping the current service with the fixed point. This fixed swap point local search creates $n-1$ neighbours, causing substantial computational time.

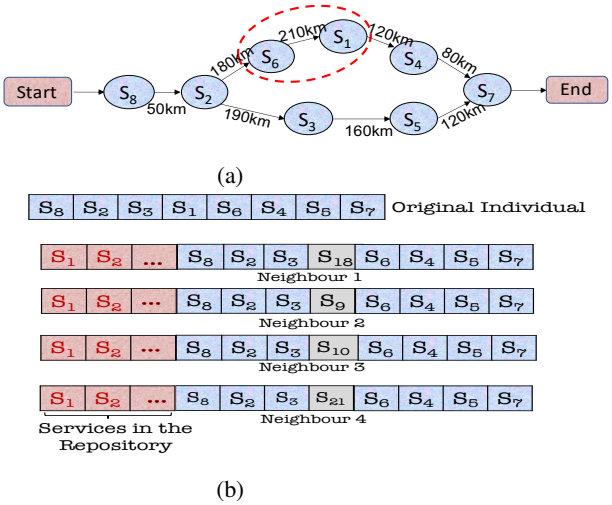
The local search in the distance-guided MA eliminates long communication links (i.e., *bottlenecks*). A *bottleneck* represents the longest link between any two adjacent services in the composite service. Since the distance has a decisive impact on communication time and cost, a *bottleneck* often dominates the overall communication time and cost of a composite service. The *bottleneck* is determined by comparing the length of communication links between any two directly connected services in the DAG.

Algorithm 2: Priority-based local search for DDWSC

Input : $Current\ Solution$, $N_l(NeighbourhoodSize)$, $Service\ Repository$
Output: $Neighbouring\ Solution$
1: Generate N_l neighbours for $Current\ Solution$, using one of the local search techniques in Subsection IV-B;
2: Evaluate all neighbours using the $SelectionPriority$ measure in Equation (9);
3: **if** $Best\ Neighbour$*'s fitness is better than* $Current\ Solution$*'s fitness* **then**
4: $Current\ Solution$=$Best\ Neighbour$;
5: **end**
6: **return** $Current\ Solution$;

Example 2. *Examples of a DAG and the distance-guided local search are illustrated in Fig. 5 (a) and Fig. 5 (b), respectively. The bottleneck link, L, is highlighted with the dotted circle, starting from S_6 and ending at S_1. Distance-guided local search creates neighbours through replacing the whole part enclosed within the red dotted circle (i.e., L, S_6 and S_1) by another path that performs the same task but without using long communication links. This is likely to enhance the fitness of a solution, F, as given in Equation (4).*

Recall that the backward decoding process generates the DAG from End to $Start$. In this example (Fig. 5 (b)), service S_7 is added to the composite service first. Next, service S_4 and service S_5 are connected. Afterwards, service S_3 and service S_1 are added. However, to eliminate the bottleneck link, L, service S_1 must be avoided. Therefore, the algorithm considers alternative services of S_1 that can also satisfy the inputs of service S_4. Suppose that there are four services in the repository, i.e., S_9, S_{10}, S_{18} and S_{21}, which can replace service S_1. Upon creating neighbours, these four services are placed in later positions than service S_1 in the neighbouring sequences, as shown in Fig. 5 (b). Therefore, the decoding process connects one of the four services to service S_4 (the neighbourhood size is limited to the number of alternative services in the repository). Moreover, a prefix, including randomly selected services from the repository, is attached to the beginning of these service sequences. This is performed to encourage the diversity of sequences.



(a)

(b)

Fig. 5: (a) Solution example and a bottleneck link, (b) original sequence and the neighbourhoods.

V. Priority-based Selection Method for Local Search

Through adaptively selecting individuals for local search, we direct the evolutionary search towards regions of the search space in MAs, where more interesting compromise solutions can be found. Some important factors need to be taken into account to achieve this goal: 1) solutions that increase diversity, 2) solutions that lead to better fitness, and 3) solutions that can be easily improved during local search.

One way to approach the desirable diversity is to employ a diversity-based method. For example, the method might select solutions that are far apart from each other. Inspired by the *novelty search* [62], we take solutions *novelty* into account. The *novelty* represents the degree to which a solution is dissimilar to existing solutions. The definition of *novelty* varies from one problem to another and remains an important yet difficult task to be determined. Although *novelty search* only prioritises solutions that are behaviorally different than the rest of the population, we employ this characteristic along with the objective-based measure. The aim is to purposefully drive search to the most ambitious objectives for identifying which solutions or individuals to choose for the neighbourhood search. To enable the explicit use of domain-knowledge, we define our *novelty* measure based on the communication links of composite services.

Further, we go beyond *novelty* and consider another method of biased selection for local search. We define the *improvability* to determine the likelihood that a local search will improve the solution. This requires us to incorporate knowledge regarding when a problem-specific local search operator is most effective. For example, this can be implemented by including the length of communication links in the composite service. This information can be used as the estimate of future effectiveness, along with the *novelty*. Therefore, less crowded regions in the search space can be adaptively cultivated and emphasised by controlling the local search based on the fitness function and communication links. *novelty* and *improvability* are explained in the following subsections.

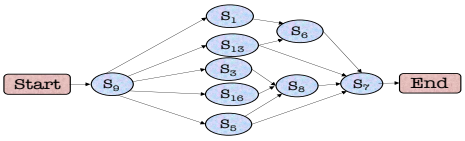
A. Novelty of Individuals

To direct the search toward any unexplored part of the solution space and maintain a high solution diversity in the GA population, we introduce a new diversity measure. A solution is considered for the local search if it is more novel than the effective solutions achieved so far. For this reason, an archive of previously discovered good solutions (regarding fitness values) must be constantly maintained. The archive is updated at each generation to include the fittest solutions (the number of solutions depends on the archive size). For each solution, the archive stores the statistics of communication links (e.g., how often they appear in the DAG). A vector, called $freq$, is generated to store the frequency of communication links ($freq$ is calculated for any new composite service $\mathcal{CS}$). The same communication link may appear multiple times in one composite service.

Example 3. *Fig. 6 shows a composite service, $\mathcal{CS}$, and the corresponding $freq$ vector. Suppose that four servers, i.e., h_1, h_2, h_3 and h_4, are used to host all the nine services contained in $\mathcal{CS}$. The frequency of each communication link is shown in Fig. 4 (b). For example, the communication link between h_2 and h_3 has been used three times: (S_9, S_{13}), (S_{13}, S_6) and (S_{13}, S_7). The $freq$ vector of $\mathcal{CS}$ is compared to the $freq$ vector of all archived composite services one at a time, based on their Euclidean distance, which is one of the simplest and most commonly used measures of similarity [76].*

Subsequently, the average Euclidean distance of $\mathcal{CS}$ to its k nearest neighbours in the archive is calculated as $novelty$, as shown in Equation (5):

Server	Service
h_1	S_1, S_3, S_5
h_2	S_6, S_7, S_8, S_9
h_3	S_{13}
h_4	S_{16}



(a)

Servers	h_1, h_1	h_1, h_2	h_1, h_3	h_1, h_4	h_2, h_2	h_2, h_3	h_2, h_4	h_3, h_3	h_3, h_4	h_4, h_4
Links	-	$(S_1, S_6)(S_9, S_1)$ $(S_9, S_3)(S_9, S_8)$ $(S_9, S_5)(S_9, S_8)$ (S_5, S_7)	-	-	(S_6, S_7) (S_8, S_7)	(S_9, S_{13}) (S_{13}, S_6) (S_{13}, S_7)	(S_9, S_{16}) (S_{16}, S_8)	-	-	-
Frequency	0	7	0	0	2	3	2	0	0	0

(b)

Fig. 6: (a) A composite service, *CS*, (right) and the location information of services (left), (b) *freq* vector for *CS*.

$$Novelty(\mathcal{CS}) = (1/k)\sum_{i=1}^{k} \mathcal{D}(freq(\mathcal{CS}), freq(\mathcal{CS}_i)) \quad (5)$$

where $\mathcal{CS}_i$ is an archived solution evolved previously. $\mathcal{CS}_i$ is the i_{th} nearest neighbour of $\mathcal{CS}$.

A solution with a higher *novelty* has a higher priority for the neighbourhood search. The main rationale behind utilising this distance metric is to generate new genotypes and put more effort into exploiting the neighbourhood of novel solutions to avoid premature convergence. Additionally, this novelty measure facilitates incorporating problem-specific knowledge in the selection method.

B. Improvability of Individual Solutions

The second challenge in selecting individuals for local search is how to find solutions that can be easily improved.

Because fitness evaluation is computationally expensive, it is not worth to perform a local search if a better neighbouring solution cannot be found through the time-consuming neighbourhood search. In this section, we introduce a measure for the priority-based selection method. We utilise domain-knowledge to find individuals that have the potential to be enhanced during the local search, through the $improvability$ measure.

It is desirable to apply local search on solutions with diverse communication links and high average communication distance. In other words, when the solution includes links with significantly different lengths, the links are diversified, and services are farther from each other. This is also to be taken into account for individual selection.

To consider the length diversity of communication links in a composite service, standard deviation and mean of the lengths are calculated for each composite solutions. Afterwards, one z_value for the mean and another for the standard deviation are calculated. z_value is a numerical measurement used in statistics to determine a value's relationship to the average or standard deviation of the population. The z_value of deviations is obtained using Equation (6):

$$z\_value(\sigma) = \frac{\sigma_{\mathcal{CS}} - \mu_\sigma}{\sigma_\sigma} \tag{6}$$

where $z\_value(\sigma)$ is the z_value of the deviation of the current solution to determine how different its deviation is from the deviation of the population. $\sigma_{\mathcal{CS}}$ represents the deviation of the current solution, σ_σ and μ_σ are the mean and standard deviation of all solution's deviations in the population.

Similarly, to determine the difference of the current solution's mean value and the population's mean value, we calculate the z_value for the mean, as in Equation (7):

$$z\_value(\mu) = \frac{\mu_{\mathcal{CS}} - \mu_\mu}{\sigma_\mu} \tag{7}$$

where $z\_value(\mu)$ is the z_value for the mean of the current solution's communication links length ($\mu_{\mathcal{CS}}$ is the mean of its communication links length), μ_μ and σ_μ are the standard deviations and the mean over all solution's mean in the population. Note that this measure takes the diversity in the length of all links into account. The calculation of z_value is exemplified in Example 4.

$CS1$	
Communication link	Length
l_1	0.5
l_2	0.8
l_3	0.1
l_4	0.2
μ_{CS}=0.4	
σ_{CS}=0.3162	

$CS2$	
Communication link	Length
l_1	0.3
l_2	0.9
l_3	0.7
l_4	0.6
l_5	0.8
μ_{CS}=0.66	
σ_{CS}=0.23	

$CS3$	
Communication link	Length
l_1	0.7
l_2	0.5
l_3	0.4
l_4	0.5
l_5	0.3
μ_{CS}=0.48	
σ_{CS}=0.148	

Fig. 7: Normalised length of communication links for three composite services, with averages and standard deviations.

Example 4. *Consider a population of three composite services, as shown in Fig. 7. The links and their lengths are shown in the associated table for each composite service. The last two columns demonstrate the average and standard deviation of the link's length for each composite service. According to the value in Fig. 7, we can obtain means and standard deviations as follows:* $\mu_\mu = 0.513$ *(the mean of 0.4, 0.66 and 0.48),* $\sigma_\mu = 0.133$ *(the standard deviation of 0.4, 0.66 and 0.48). Similarly,* $\mu_\sigma = 0.23$ *(the mean of 0.3162, 0.23 and 0.148),* $\sigma_\sigma = 0.084$ *(the standard deviation of 0.3162, 0.23 and 0.148). Therefore, the* z_value *for* $\mathcal{CS}2$*, for example, is calculated as shown below:*

$$z\_value(\mu) = \frac{0.66 - 0.513}{0.133} = 1.1$$

$$z\_value(\sigma) = \frac{0.23 - 0.23}{0.084} = 0$$

Finally, the $improvability$ of $\mathcal{CS}$ is calculated as the sum of two z_values:

$$Imp(\mathcal{CS}) = z\_value(\mu) + z\_value(\sigma) \tag{8}$$

A higher value of $Imp(\mathcal{CS})$ represents a higher average length and deviation of communication links than the population.

C. Individual Selection Criteria for Local Search

Finally, a selection priority of $\mathcal{CS}$ for the local search is obtained using Equation (9):

$$\begin{aligned} SelectionPriority(\mathcal{CS}) =& w_1\widehat{Novelty}(\mathcal{CS}) + \\ & w_2\widehat{Imp}(\mathcal{CS}) + w_3 F(\mathcal{CS}) \end{aligned} \tag{9}$$

where $F(\mathcal{CS})$ is defined in Equation (4) and $\widehat{Novelty}$ and $\widehat{Imp}$ are normalised value of $Novelty(\mathcal{CS})$ and $Imp(\mathcal{CS})$, respectively. w_1, w_2 and w_3 are integer weight parameters in $\{0, 1\}$, where $w_1 + w_2 + w_3 \neq 0$. These parameters are changed during the experiments to show the contribution of each measure to the performance of MAs. If a weight parameter is zero, the corresponding criterion does not contribute to the selection priority.

VI. Empirical Evaluation

In this section, we answer the following research questions (RQs) through a series of experiments: RQ1) Can the proposed MAs in this paper, including adaptive cluster-guided, adaptive distance-guided, and adaptive *Combined Cluster-Distance-guided (Com-C-D)* MAs, outperform state-of-the-art MAs for DDWSC? RQ2) Which of the measures, including novelty, improvability and fitness, play a more important role in MAs? RQ3) Does improving the fitness value of solutions found by our MAs lead to the improvement in both response time and cost? RQ4) What is the computation time of all competing algorithms? These research questions are addressed in Subsections VI-E, VI-F, VI-G, and VI-H, respectively.

A. Competing Methods

Four competing algorithms are employed in this section: 1) distance-guided MA (which relies on the baseline local search), [6]; 2) cluster-guided MA [7]; 3) *Combined Cluster-Distance-guided MA* ((named *Com-C-D*)) that combines the cluster-guided and the distance-guided approaches. In Com-C-D, the cluster-guided approach helps GA form a high quality initial population while the distance-guided approach improves the fitness of evolved solutions by using distance-guided crossover and local search operators; and 4) abstraction refinement method [18], which is an existing state-of-the-art method (introduced in Section II). We make a few

TABLE I: Common parameter settings

Parameter	Value	Methods
Generations	100	All MAs
Elitism size (e)	2	All MAs
Population Size	30	All MAs
Fitness-based Selection (mutation and crossover)	Tournament (size=2)	All MAs
Population Size	30	All MAs
Crossover Rate (p_c)	0.95	All MAs
Local Search Rate	0.05	All MAs
ω (weight)	0.5	All MAs
Mutation Rate (p_m)	0.05	All MAs
$SelectionPriority$-based Selection (local search)	Tournament (size=2)	Adaptive MAs
k (Neighbourhood size)	3	Adaptive MAs
Archive Size	10	Adaptive MAs
Fitness-based Selection (for local search)	Tournament (size=2)	Non-adaptive MAs

modifications to the abstraction method to make it suitable for DDWSC. Abstraction refinement is chosen because it is one of the recently proposed methods for fully-automated WSC.

We also integrate the proposed priority-based selection method with the above three MAs, which results in adaptive distance-guided MA, adaptive cluster-guided MA and adaptive Com-C-D MA.

B. Datasets

A set of experiments is carried out using WSC-2008 [77] and WSC-2009 [78] benchmark datasets. WSC-2008 contains eight service repositories of varying sizes, while WSC-2009 provides five repositories with up to 15211 services in a repository. The first column of Table IV indicates the name and size of each repository. A taxonomy of concepts is predefined to help the service composition system determine which inputs and outputs are compatible. One associated composition task per repository of services are given in advance [77], [78]. These datasets were chosen because they are the largest benchmarks that have been broadly exploited in the WSC literature [17], [24], [25]. However, the original WSC datasets did not include QoS information for any service. We further obtain QoS settings from the QWS dataset [79]. Additionally, WSC datasets do not contain the location information of the servers hosting Web services [7]. The distance between two Web services is estimated using the same method adopted in [7] based on the location information in the WS-Dream open dataset [80]. This dataset is used to obtain communication attributes as well.

The data size is considered the same for all methods, as this paper focuses on communication between services. Our future work will analyse the impact of different data sizes.

C. Experimental Settings

Table I presents all parameter settings including both shared parameters among all competing algorithms as well as parameters that are unique to adaptive algorithms. The chosen parameter settings for all methods were based on popular settings discussed in the literature [8], [10], [17], [81]. A conventional fitness-driven tournament selection chooses candidate solutions for the mutation and crossover operators. The tournament size is set as two, according to existing works on WSC [7], [17]. Two elite solutions are copied to the next generation unchanged. They remain eligible for being selected as parents to evolve new solutions for the next generation through evolutionary operators.

Service composition users can set the weight (w in Equation (5)) to specify the relative importance of total time and cost. Following other WSC research works [8], [10], [12], [17], we assume that users set weight preferences to $w = 0.5$, to put equal importance on the time and cost in the fitness function (note that the value of w can be chosen freely by users).

Additionally, as shown in Table I, all adaptive MAs use an archive size of 10 and neighbourhood size of three for calculating *novelty*.

We have experimentally validated the abstraction refinement method for WSC, introduced in [18], using the test dataset WSC-2009. Our WSC test problem was the same as the DDWSC except that we did not consider the communication and data attributes in the objective function. After evaluating the abstraction refinement method for WSC (where we assumed all services were at the same location) we adopted it for the DDWSC.

D. Implementation

We implemented all competing algorithms in Java 8. Experiments were conducted on a desktop computer with 8 GB RAM and an Intel Core i7-4790 and 3.6GHz processor running Linux Arch 4.9.6. We used the implementation of GA provided by the ECJ toolkit (Java evolutionary computation)[1]. Github links for the full implementation of all proposed algorithms have been provided [2].

E. Effectiveness of the Priority-based Selection Method for MA (RQ1)

RQ1 is to investigate the effectiveness of the new priority-based selection method.

1) Measurements for RQ1: We compare the performance of different algorithms based on the fitness value obtained through the weighted sum in Equation (4), following the common practice in the literature [17], [18].

2) RQ1 Results: RQ1 Evaluation results are shown in Table II. First, a pairwise one-sample statistical t-test was performed between the abstraction refinement method and each of the MAs (six pairs) at 0.05 significance level ($p\text{-}value = 0.05$) for all tests), where adaptive MAs outperformed the abstraction refinement method on all tasks. For example, for task 08-1, adaptive distance-guided MA performed significantly better than the abstraction refinement method. However, there was no significant difference between the cluster-guided MA and the abstraction refinement method on task 08-1. The abstraction refinement and the cluster-guided methods focus on grouping Web services according to only one aspect of their quality. The former groups Web services based on their functional characteristics and then selects the one with the best QoS, while the latter clusters Web services considering locations of Web services exclusively and prefers those Web services located close to other Web service used for the composition.

Second, a pairwise one-sample statistical t-test was conducted between each algorithm and the corresponding adaptive version at a significance level of 0.05 ($p\text{-}value = 0.05$) for all tests. Our tests confirmed that each MA is significantly outperformed by the corresponding adaptive MA.

Finally, a one-way ANOVA ($p\text{-}value = 0.05$) was performed between six MAs, i.e., cluster-guided MA, distance-guided MA Com-C-D MA and their corresponding adaptive MAs, followed by Tukey's post-hoc analysis. Results showed that the adaptive Com-C-D (obtained by applying our selection method to Com-C-D) significantly outperforms other MAs on more than 90 % of the tasks (as highlighted in bold in Table II).

[1] https://cs.gmu.edu/eclab/projects/ecj/

[2] https://github.com/soheilasadeghi

Results confirm that our proposed priority-based selection method can be applied to many existing MAs for the DDWSC and improve their performance. Further, our MAs can outperform state-of-the-art MAs [7], [25].

F. Examination of Different Selection Criteria (RQ2)

To answer RQ2 we examined the influence of different selection criteria. In this subsection, we examine how MAs can benefit from considering improvability and novelty. Therefore, we assess their effectiveness on MAs, separately.

1) Measurements for RQ2: We considered the following combinations for the priority-based selection for local search: fitness with novelty, fitness with improvability, and, fitness with novelty and improvability (Note that except for the selection of solutions for local search, fitness value is used for the overall evaluation of solutions and the fitness of the final solution is then reported after running each algorithm). We conducted experiments with five separate weight configurations for Equation (9) (i.e., C1 to C5). Configuration C1 indicates a non-adaptive algorithm, which employs only fitness during the selection ($w_1 = w_2 = 0$ and $w_3 = 1$). Configuration C2 means that the contribution of novelty in the priority-based selection is zero ($w_1 = 0$ and $w_2 = w_3 = 1$), while $w_2 = 0$ and $w_1 = w_3 = 1$ in C3 shows that only novelty and fitness affect the selection process. Configuration C4 is for applying only novelty ($w_1 = 1$ and $w_2 = w_3 = 0$). Finally, in C5, all three criteria are applied ($w_1 = w_2 = w_3 = 1$).

2) RQ2 Results: Results of the priority-based selection method combined with distance-guided and cluster-guided MAs are shown in Table III and Table IV, respectively. A pairwise statistical t-test with $p\text{-}value = 0.05$ has been performed on the fitness values. Methods that significantly outperform other methods are highlighted. However, on some tasks (e.g., 08-2 and 08-3), more than one methods are highlighted, indicating that those methods significantly outperform non-highlighted ones but do not significantly outperform any other highlighted methods. Second columns of tables indicate the original algorithm that relies purely on the fitness value for selecting individuals for the local search (configuration C1). The rest columns in both tables present results obtained by using one criterion along with the fitness value in priority-based selection. According to those tables, for most of the tasks, the priority-based selection method produces better results compared to the original algorithm. For example, for task 08-1, the fitness of the best solution found by the original algorithm is 0.4. However, when applying priority-based selection (in columns 3-6) we obtain solutions with better fitness. This applies for tasks 08-4, 08-5, 08-6, 08-7, 09-1 and 09-5, as well. The method presented in the last column (configuration C5), which uses all measures for priority-based selection, outperforms all other methods for task 08-1. Overall, configurations C2, C3, C4 and C5 outperformed the original algorithms (configuration C1) on 92%, 62%, 54%, and 100% of the tasks, respectively.

For configuration C2, we can witness a slight performance improvement, i.e., when the improvability is utilised along with the fitness value in selecting solutions for local search. For example, on tasks 08-1, 08-2 and 08-5 and all WSC09 tasks, the final fitness value of the solution is lower (better) than the final fitness value for the original configuration (C1). However, for the cluster-guided MA, this improvement is not significant (C2 has not produced better results than C1). In other words, for the cluster-guided MA, C2 outperforms C1 only on a small number of tasks. For example, on task 08-2, the solution's fitness with C1 is 0.44 and with C2 is 0.43 which is marginally better, while in task 08-3 the value is the same (0.53) for both C1 and C2.

We obtain higher quality results using both novelty criterion and the fitness value (i.e., configuration C3). We subsequently considered the individual effect of novelty in fifth columns (i.e., C4) of tables. For about half of the tasks (e.g., 08-2, 08-8, 09-2 and 09-4), we obtained solutions with similar quality to the original method in column two. However, marginally superior solutions were achieved (supported by 08-1, 08-3, 08-4, etc.) on other tasks. On task 08-8 for the cluster-guided MA, the novelty was able to find even fitter solutions compared to the combined use of novelty and fitness. Comparing the configurations C2 and C3 in Table III and Table IV, we find out that novelty (C3) plays a more important role than improvability (C2) in enhancing the performance of MAs.

Finally, in association with the last columns of Table III and Table IV (i.e., using configuration C5), an overall improvement on all tasks for distance-guided MA and on all tasks except 08-2 for the cluster-guided MA can be verified.

G. Examination of Response Time and Cost (RQ3)

In this subsection, we address the third research question and to show the practical significance of the improvement made by our MAs.

1) Measurements for RQ3: The weighted sum approach, which has been used in previous subsections, provides the user with a single final solution. Therefore, unlike in multi-objective Pareto-based approaches, the user is not required to select one solution out of a possibly large number of compromising solutions in the Pareto front. However, to prove the practical significance of our method, we separately explore time and cost improvements. Therefore, we individually examined the response time and cost besides fitness values.

2) RQ3 Results: The response time and cost achieved by the evolved composite services are shown in Table V, for distance-guided and Com-C-D MAs. These two methods are chosen because they have outperformed other baselines, as shown in Table II. A one-way ANOVA with $p\text{-}value = 0.05$ was performed followed by a Tukey's post-hoc analysis. According to the table, the time and cost savings offered by the proposed MAs are significant. The average response time and cost improvements for all benchmark problems are 13% and 3%, respectively. For Com-C-D (columns 4-5), the average time and cost are 5% and 10%, respectively. For example, comparing columns two and three of Table V for task 08-1 verifies that both response time and cost have been reduced significantly along with improvement in fitness (i.e., 16% for the response time and 26% for the cost). The response time reduces from 32251.9 ms to 27001.6 ms, and the cost reduces from \$159.3 to \$103.9. As another example, we compare Com-C-D and adaptive Com-C-D MAs on task 08-8, where the reduction in cost and time are 19% and 6% respectively. Additionally, the case where one objective degrades in favour of the other has occasionally appeared. For example, for task 08-3, columns two and three have the same fitness value; however, the cost degrades in favour of the response time. Results confirm that the improvement in the fitness value is usually a consequence of improvements in both objectives.

H. Examination of Computation Time (RQ4)

1) Measurement for RQ4: To answer RQ4, we compared all algorithms in term of their execution time measured in seconds.

TABLE II: Comparison of the abstraction refinement for DDWSC, (adaptive) distance-guided, (adaptive) cluster-guided, and (adaptive) Com-C-D MAs. (Note: the lower the fitness, the better). All three factors have been considered in the selection priority for adaptive MAs. Significantly better values are highlighted in bold for each task, with the significance level of 0.05.

Task	abstraction refinement [18]	cluster-guided MA [7]	distance-guided MA [6]	adaptive cluster-guided MA	adaptive distance-guided MA	Com-C-D MA	adaptive Com-C-D MA
WSC08-1	0.46	0.46 ± 0.04	0.4 ± 0.07	0.43 ± 0.22	0.37 ± 0.04	0.37 ± 0.03	$\mathbf{0.36 \pm 0.09}$
WSC08-2	0.43	0.44 ± 0.02	0.41 ± 0.03	0.41 ± 0.04	0.37 ± 0.01	0.37 ± 0.15	0.37 ± 0.09
WSC08-3	0.51	0.53 ± 0.03	0.44 ± 0.011	0.46 ± 0.13	0.42 ± 0.03	0.44 ± 0.02	$\mathbf{0.42 \pm 0.05}$
WSC08-4	0.51	0.5 ± 0.09	0.39 ± 0.03	0.46 ± 0.03	0.36 ± 0.12	0.36 ± 0.1	$\mathbf{0.35 \pm 0.02}$
WSC08-5	0.52	0.53 ± 0.041	0.45 ± 0.12	0.49 ± 0.09	0.41 ± 0.04	0.44 ± 0.09	$\mathbf{0.4 \pm 0.02}$
WSC08-6	0.56	0.56 ± 0.08	0.47 ± 0.01	0.51 ± 0.12	0.44 ± 0.04	0.43 ± 0.08	$\mathbf{0.42 \pm 0.13}$
WSC08-7	0.55	0.55 ± 0.27	0.52 ± 0.02	0.51 ± 0.23	0.49 ± 0.01	0.47 ± 0.06	$\mathbf{0.46 \pm 0.07}$
WSC08-8	0.48	0.49 ± 0.04	0.46 ± 0.09	0.45 ± 0.14	0.41 ± 0.07	0.43 ± 0.11	$\mathbf{0.4 \pm 0.09}$
WSC09-1	0.54	0.54 ± 0.23	0.51 ± 0.02	0.5 ± 0.02	0.48 ± 0.16	0.47 ± 0.12	$\mathbf{0.47 \pm 0.02}$
WSC09-2	0.52	0.51 ± 0.07	0.45 ± 0.22	0.48 ± 0.12	0.41 ± 0.02	0.41 ± 0.07	$\mathbf{0.4 \pm 0.05}$
WSC09-3	0.53	0.54 ± 0.11	0.48 ± 0.1	0.5 ± 0.06	0.45 ± 0.01	0.42 ± 0.13	$\mathbf{0.41 \pm 0.08}$
WSC09-4	0.51	0.5 ± 0.05	0.48 ± 0.02	0.46 ± 0.01	0.43 ± 0.01	0.45 ± 0.14	$\mathbf{0.43 \pm 0.08}$
WSC09-5	0.48	0.46 ± 0.04	0.46 ± 0.06	0.45 ± 0.02	0.42 ± 0.03	0.43 ± 0.09	$\mathbf{0.42 \pm 0.1}$

TABLE III: Mean and standard deviation of fitness values per 30 runs for distance-guided MA with different selection parameters. Significantly better values are highlighted in bold for each task (Note: the lower the fitness, the better).

Task	$w_1 = w_2 = 0, w_3 = 1$	$w_1 = 0, w_2 = w_3 = 1$	$w_2 = 0, w_1 = w_3 = 1$	$w_1 = 1, w_2 = w_3 = 0$	$w_1 = w_2 = w_3 = 1$
WSC08-1	0.4 ± 0.07	0.39 ± 0.12	0.38 ± 0.02	0.39 ± 0.08	$\mathbf{0.37 \pm 0.04}$
WSC08-2	0.41 ± 0.03	0.4 ± 0.02	$\mathbf{0.38 \pm 0.04}$	0.41 ± 0.08	$\mathbf{0.37 \pm 0.1}$
WSC08-3	0.44 ± 0.11	0.44 ± 0.02	$\mathbf{0.42 \pm 0.02}$	0.43 ± 0.03	$\mathbf{0.42 \pm 0.03}$
WSC08-4	0.39 ± 0.03	0.38 ± 0.01	$\mathbf{0.36 \pm 0.01}$	0.37 ± 0.09	$\mathbf{0.36 \pm 0.12}$
WSC08-5	0.45 ± 0.12	0.43 ± 0.16	$\mathbf{0.41 \pm 0.04}$	0.44 ± 0.13	$\mathbf{0.41 \pm 0.04}$
WSC08-6	0.47 ± 0.01	0.46 ± 0.22	0.45 ± 0.15	0.45 ± 0.04	$\mathbf{0.44 \pm 0.04}$
WSC08-7	0.46 ± 0.04	0.45 ± 0.23	$\mathbf{0.43 \pm 0.02}$	0.44 ± 0.05	$\mathbf{0.42 \pm 0.01}$
WSC08-8	0.46 ± 0.09	0.45 ± 0.05	0.43 ± 0.08	0.46 ± 0.02	$\mathbf{0.41 \pm 0.07}$
WSC09-1	0.51 ± 0.02	0.5 ± 0.04	$\mathbf{0.49 \pm 0.02}$	0.5 ± 0.05	$\mathbf{0.48 \pm 0.16}$
WSC09-2	0.45 ± 0.22	0.44 ± 0.04	0.42 ± 0.16	0.45 ± 0.01	$\mathbf{0.41 \pm 0.02}$
WSC09-3	0.48 ± 0.1	0.47 ± 0.06	0.46 ± 0.02	0.48 ± 0.001	$\mathbf{0.45 \pm 0.01}$
WSC09-4	0.48 ± 0.02	0.47 ± 0.09	0.47 ± 0.21	0.48 ± 0.21	$\mathbf{0.43 \pm 0.01}$
WSC09-5	0.46 ± 0.06	0.45 ± 0.03	$\mathbf{0.43 \pm 0.06}$	0.45 ± 0.22	$\mathbf{0.43 \pm 0.03}$

TABLE IV: Mean fitness values and standard deviations per 30 runs for the cluster-guided MA with different selection parameters. Significantly better values are shown in bold. The significance level is 0.05 (Note: the lower the fitness, the better).

Task (size)	$w_1 = w_2 = 0, w_3 = 1$	$w_1 = 0, w_2 = w_3 = 1$	$w_2 = 0, w_1 = w_3 = 1$	$w_1 = 1, w_2 = w_3 = 0$	$w_1 = w_2 = w_3 = 1$
WSC08-1 (158)	0.46 ± 0.04	0.46 ± 0.12	$\mathbf{0.43 \pm 0.02}$	0.46 ± 0.03	$\mathbf{0.43 \pm 0.22}$
WSC08-2 (558)	0.44 ± 0.02	0.43 ± 0.02	$\mathbf{0.41 \pm 0.04}$	0.42 ± 0.03	0.41 ± 0.04
WSC08-3 (604)	0.53 ± 0.03	0.53 ± 0.02	0.47 ± 0.15	0.5 ± 0.17	$\mathbf{0.46 \pm 0.13}$
WSC08-4 (1041)	0.5 ± 0.09	0.49 ± 0.01	0.48 ± 0.01	0.49 ± 0.02	$\mathbf{0.46 \pm 0.03}$
WSC08-5 (1090)	0.53 ± 0.041	0.52 ± 0.16	0.5 ± 0.04	0.52 ± 0.03	$\mathbf{0.49 \pm 0.09}$
WSC08-6 (2198)	0.56 ± 0.08	0.56 ± 0.22	$\mathbf{0.51 \pm 0.15}$	0.55 ± 0.03	$\mathbf{0.51 \pm 0.12}$
WSC08-7 (4113)	0.55 ± 0.236	0.54 ± 0.11	$\mathbf{0.51 \pm 0.02}$	0.53 ± 0.03	$\mathbf{0.51 \pm 0.23}$
WSC08-8 (8119)	0.49 ± 0.04	0.48 ± 0.05	$\mathbf{0.46 \pm 0.08}$	$\mathbf{0.46 \pm 0.03}$	$\mathbf{0.45 \pm 0.14}$
WSC09-1 (572)	0.54 ± 0.23	0.53 ± 0.22	$\mathbf{0.5 \pm 0.05}$	0.54 ± 0.03	$\mathbf{0.5 \pm 0.02}$
WSC09-2 (4129)	0.51 ± 0.07	0.51 ± 0.14	$\mathbf{0.48 \pm 0.01}$	0.5 ± 0.08	$\mathbf{0.48 \pm 0.12}$
WSC09-3 (8138)	0.54 ± 0.11	0.53 ± 0.06	$\mathbf{0.5 \pm 0.001}$	0.52 ± 0.19	$\mathbf{0.5 \pm 0.06}$
WSC09-4 (8301)	0.5 ± 0.05	0.5 ± 0.09	$\mathbf{0.47 \pm 0.21}$	0.49 ± 0.13	$\mathbf{0.46 \pm 0.1}$
WSC09-5 (15211)	0.46 ± 0.04	0.46 ± 0.03	0.44 ± 0.06	0.45 ± 0.07	$\mathbf{0.42 \pm 0.02}$

TABLE V: Mean and standard deviation of cost and response time per 30 runs for distance-guided, Com-C-D, adaptive distance-guided and adaptive Com-C-D MAs. The corresponding fitness value can be found in Table II. Significantly better values are shown in bold. The significance level is 0.05.

Task	distance-guided MA		adaptive distance-guided MA		Com-C-D MA		adaptive Com-C-D MA	
	Time	Cost	Time	Cost	Time	Cost	Time	Cost
WSC08-1	32251.9 $\pm$1247	159.3 $\pm$ 21	27001.6 $\pm$599	103.9 $\pm$ 9	26169.14 $\pm$799	110.1 $\pm$ 11	**26001.1 $\pm$809**	**101.6 $\pm$ 15**
WSC08-2	24701.5 $\pm$395	180.1 $\pm$ 14	18515.8 $\pm$365	99.6 $\pm$ 8.5	18951.98 $\pm$501	101.1 $\pm$ 13	**17999.6 $\pm$998**	101.9 $\pm$ 13
WSC08-3	75661.65 $\pm$1010	219.1 $\pm$ 15	72093.1 $\pm$587	196.45 $\pm$ 16	75913.59 $\pm$599	214.09 $\pm$ 17	72893.9 $\pm$580	**192.9 $\pm$ 15**
WSC08-4	17121.54 $\pm$109	126.8 $\pm$ 11	12709.12 $\pm$114	99.9 $\pm$ 6	12569.45 $\pm$101	101.1 $\pm$ 7	**11061.9$\pm$75**	**95.4 $\pm$ 5**
WSC08-5	34121.18 $\pm$312	255.45 $\pm$ 33	30911.19 $\pm$379	209.9 $\pm$ 16	32660.9 $\pm$481	234.33 $\pm$ 15	**29966.1 $\pm$111**	**204.1 $\pm$ 20**
WSC08-6	31213.19 $\pm$387	388.35 $\pm$ 15	27429.25 $\pm$263	300.14 $\pm$ 15	26901.18 $\pm$362	299.9 $\pm$ 16	**25422.5 $\pm$299**	**281.95 $\pm$ 16**
WSC08-7	34301.8 $\pm$299	301.23 $\pm$ 59	29496.1 $\pm$150	199.09 $\pm$ 19	27859.2 $\pm$343	192.1 $\pm$ 25	**27099.8 $\pm$198**	**185.64 $\pm$ 45**
WSC08-8	58022.74 $\pm$454	313.92 $\pm$ 33	50126.9 $\pm$329	239.13 $\pm$ 20	53319.1 $\pm$341	260.3 $\pm$ 40	**49995.15 $\pm$409**	**211.9 $\pm$ 31**
WSC09-1	24713.7 $\pm$913	112.93 $\pm$ 11	20195.4 $\pm$629	92.1 $\pm$ 5.3	19610.15 $\pm$659	90.01 $\pm$ 6	**19002.11 $\pm$491**	90.5 $\pm$ 10
WSC09-2	48876.15 $\pm$6875	304.13 $\pm$ 36	44381.19 $\pm$360	214.9 $\pm$ 21	43819.9 $\pm$399	218.17 $\pm$ 23	42706.9 $\pm$6699	211.57 $\pm$ 37
WSC09-3	31760.56 $\pm$115	293.43 $\pm$ 39	27114.5 $\pm$69	204.19 $\pm$ 22	25719.5 $\pm$77	202.11 $\pm$ 15	**24013.13 $\pm$119**	**199.08 $\pm$ 34**
WSC09-4	74340.9 $\pm$2314	699.49 $\pm$ 125	66439.5 $\pm$1309	491.13 $\pm$ 78	71117.1 $\pm$1599	596.31 $\pm$ 132	65042.2 $\pm$1211	506.74 $\pm$ 108
WSC09-5	66505.9 $\pm$990	491.89 $\pm$ 68	61650.92 $\pm$681	389.89 $\pm$ 29	60070.32 $\pm$401	406.35 $\pm$ 28	**59777.92$\pm$250**	**362.9 $\pm$ 13**

2) RQ4 Results: Table VI demonstrates that the abstraction refinement [18] is the fastest algorithm. For example, it requires only 0.33 seconds to provide solutions for 08-2.

TABLE VI: Mean and standard deviation of execution time (seconds) for each algorithm over 30 runs.

Task	abstraction refinement [18]	cluster-guided MA [7]	distance-guided MA [6]	adaptive cluster-guided MA	adaptive distance-guided MA	Com-C-D MA	adaptive Com-C-D MA
WSC08-1	0.4	3.7 ± 0.14	0.43 ± 0.03	4.2 ± 0.54	1.19 ± 0.15	4.6 ± 0.03	4.9 ± 0.04
WSC08-2	0.33	3.6 ± 0.12	0.4 ± 0.03	4.11 ± 0.5	1.2 ± 0.04	4.36 ± 0.6	5.02 ± 0.08
WSC08-3	1.23	7.13 ± 0.83	1.28 ± 0.02	7.49 ± 1.01	2.1 ± 0.14	8.08 ± 1.03	8.37 ± 1.07
WSC08-4	1.01	5.5 ± 0.09	0.74 ± 0.02	6.01 ± 0.14	2.46 ± 1.03	6.64 ± 0.22	6.87 ± 0.3
WSC08-5	2.12	7.63 ± 2.1	1.02 ± 0.07	7.94 ± 1.04	1.9 ± 0.39	8.75 ± 0.9	8.94 ± 1.02
WSC08-6	4.66	11.56 ± 2.08	4.45 ± 0.17	11.94 ± 2.1	5.91 ± 1.12	12.86 ± 3.3	12.95 ± 3.4
WSC08-7	3.28	7.15 ± 1.21	2.42 ± 0.19	8.52 ± 0.8	3.51 ± 0.09	9.94 ± 1.2	10.23 ± 1.19
WSC08-8	4.98	9.49 ± 1.04	5.18 ± 0.12	10.63 ± 0.07	6.95 ± 1.16	13.24 ± 1.3	13.86 ± 1.4
WSC09-1	0.97	4.98 ± 1.02	0.44 ± 0.02	6.02 ± 0.2	0.98 ± 0.18	6.43 ± 0.3	7.03 ± 0.32
WSC09-2	3.98	6.95 ± 0.7	4.05 ± 0.02	7.08 ± 0.12	4.39 ± 0.09	7.58 ± 0.19	7.78 ± 0.18
WSC09-3	4.45	8.54 ± 1.1	4.08 ± 0.03	5.98 ± 1.06	5.05 ± 0.6	6.71 ± 2.1	7.02 ± 1.9
WSC09-4	3.01	12.5 ± 3.11	4.16 ± 0.09	14.8 ± 0.73	4.95 ± 0.04	15.9 ± 1.4	16.65 ± 1.7
WSC09-5	2.99	13.68 ± 4.2	4.15 ± 0.12	15.02 ± 1.08	5.92 ± 1.0	15.73 ± 1.36	16.13 ± 1.47

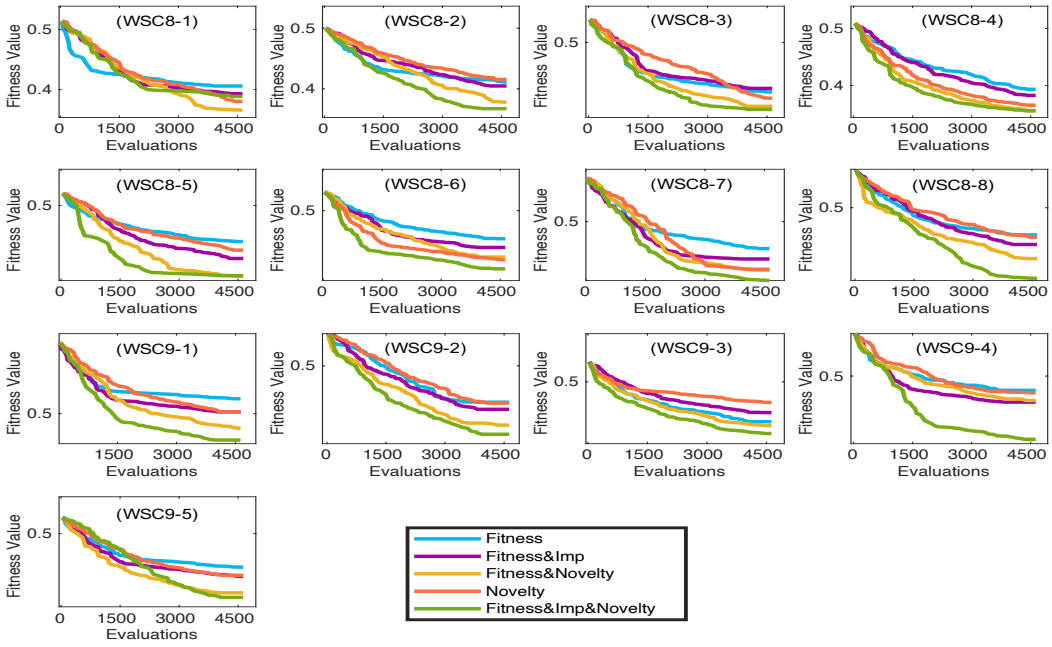


Fig. 8: Mean fitness over evolution numbers for all tasks for adaptive distance-guided MA (the lower the fitness, the better).

The cluster-guided MA is always slower than the distance-guided MA. For example, the cluster-guided MA spends 3.6 seconds while the distance-guided MA requires only 0.4 seconds on 08-2. Therefore, the combination of cluster-guided MA and distance-guided MA (i.e., Com-C-D MA) requires slightly more computation time than the cluster-guided MA and substantially more time than the distance-guided MA. All adaptive MAs take slightly more computation time than their non-adaptive version. Finally, adaptive cluster-guided and adaptive distance-guided MAs are faster than Com-C-D MA.

VII. Discussions

Preventing premature convergence of MAs to a local optimum is very important. One possible way of detecting the convergence is to observe novelty concerning the solutions in the archive. We may prevent the algorithm from getting stuck at a local optimum by performing local search around higher-priority solutions to introduce new individuals who maintain higher fitness than the current one. Solutions with lower fitness but higher novelty still have a chance to be improved by local search. While the high fitness solutions aid in the convergence of the GA, the low fitness solutions prevent MAs from getting stuck at a local optimum.

The fitness component of $SelectionPriority$ ensures that promising solutions still maintain a reasonable chance for local search. Moreover, the improvability of a solution is taken into account while selecting a solution for local search. Overall, MAs integrated with priority-based selection method outperform the other MAs with conventional fitness-based selection. Using the three components of $SelectionPriority$ together is superior compared to other combinations of them. Fig. 8 shows the convergence of each method. For example, for task 08-6, the blue plot (i.e., the baseline distance-guided MA) has converged to a local optimum and, therefore, failed to discover a satisfactory solution compared to other methods, which are all adaptive. The orange, green and mustard plots represent the influence of novelty achieving improvements in initial generations. Overall, the green plot (i.e., the combination of fitness, novelty and improvability) outperformed all other techniques.

VIII. Conclusions

In this paper, we proposed a priority-based selection method to select individuals for local search in MAs. Our proposed method uses two novel measures, i.e., the solution novelty and improvability, besides the fitness measure. Our experimental results demonstrate that the priority-based selection

method significantly enhances the performance of MAs for DDWSC. In particular, we discovered that both the novelty and improvability measures play indispensable roles in the SelectionPriority. They jointly enable our MAs to substantially outperform several state-of-the-art algorithms on a range of benchmark DDWSC problems.

ACKNOWLEDGMENT

This work is in part supported by the New Zealand Marsden Fund with the contract number (VUW1510), administrated by the Royal Society of New Zealand.

REFERENCES

[1] G. Alonso, F. Casati, H. Kuno, and V. Machiraju, "Web services," in *Web Services*. Springer, 2004, pp. 123–149.

[2] M. P. Papazoglou, P. Traverso, S. Dustdar, and F. Leymann, "Service-oriented computing: State of the art and research challenges," *Computer*, vol. 40, no. 11, pp. 38–45, 2007.

[3] H. Song, Y. Sun, Y. Yin, and S. Zheng, "Dynamic weaving of security aspects in service composition," in *IEEE International Workshop on Service-Oriented System Engineering*, 2006, pp. 189–196.

[4] A. Strunk, "QoS-aware service composition: A survey," in *IEEE European Conference on Web Services*, 2010, pp. 67–74.

[5] V. Gabrel, M. Manouvrier, and C. Murat, "Web services composition: complexity and models," *Discrete Applied Mathematics*, vol. 196, pp. 100–114, 2015.

[6] S. Sadeghiram, H. Ma, and G. Chen, "Composing distributed data-intensive Web services using a flexible memetic algorithm," in *IEEE Congress on Evolutionary Computation*, 2019, pp. 2832–2839.

[7] ——, "Cluster-guided genetic algorithm for distributed data-intensive Web service composition," *IEEE Congress on Evolutionary Computation*, pp. 1–7, 2018.

[8] S. Mistry, A. Bouguettaya, H. Dong, and A. K. Qin, "Metaheuristic optimization for long-term IaaS service composition," *IEEE Transactions on Services Computing*, vol. 11, no. 1, pp. 131–143, 2016.

[9] P. Rodriguez-Mier, M. Mucientes, M. Lama, and M. I. Couto, "Composition of Web services through Genetic Programming," *Evolutionary Intelligence*, vol. 3, no. 3-4, pp. 171–186, 2010.

[10] T. Weise, S. Bleul, D. Comes, and K. Geihs, "Different approaches to semantic Web service composition," in *IEEE International Conf. on Internet and Web Applications and Services*, 2008, pp. 90–96.

[11] M. Mucientes, M. Lama, and M. I. Couto, "A Genetic Programming-based algorithm for composing Web services," in *IEEE International Conf. on Intelligent Systems Design and Applications*, 2009, pp. 379–384.

[12] S. R. Boussalia and A. Chaoui, "Optimizing QoS-based Web services composition by using quantum inspired Cuckoo search algorithm," in *International Conf. on Mobile Web and Information Systems*. Springer, 2014, pp. 41–55.

[13] L. Aversano, M. Di Penta, and K. Taneja, "A Genetic Programming approach to support the design of service compositions," *International Journal of Computer Systems Science & Engineering*, vol. 21, no. 4, pp. 247–254, 2006.

[14] A. S. da Silva, H. Ma, Y. Mei, and M. Zhang, "A survey of evolutionary computation for Web service composition: A technical perspective," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 4, no. 4, pp. 538 – 554, 2020.

[15] Z. Wang, B. Cheng, W. Zhang, and J. Chen, "Q-graphplan: QoS-aware automatic service composition with the extended planning graph," *IEEE Access*, vol. 8, pp. 8314–8323, 2020.

[16] D. Wang, Y. Yang, and Z. Mi, "A genetic-based approach to Web service composition in geo-distributed cloud environment," *Computers & Electrical Engineering*, vol. 43, pp. 129–141, 2015.

[17] A. S. da Silva, Y. Mei, H. Ma, and M. Zhang, "Evolutionary computation for automatic Web service composition: an indirect representation approach," *Journal of Heuristics*, vol. 24, no. 3, pp. 425–456, 2018.

[18] S. Chattopadhyay and A. Banerjee, "QoS constrained large scale Web service composition using abstraction refinement," *IEEE Transactions on Services Computing*, vol. 13, no. 3, pp. 529–544, 2017.

[19] L. Wang, J. Shen, J. Luo, and F. Dong, "An improved Genetic Algorithm for cost-effective data-intensive service composition," in *IEEE International Conf. on Semantics, Knowledge and Grids*, 2013, pp. 105–112.

[20] L. Wang, J. Shen, and J. Yong, "A survey on bio-inspired algorithms for Web service composition," in *IEEE International Conf. on Computer Supported Cooperative Work in Design*, 2012, pp. 569–574.

[21] J. H. Holland, "Genetic algorithms," *Scientific American*, vol. 267, no. 1, pp. 66–73, 1992.

[22] M. W. S. Land and R. K. Belew, *Evolutionary algorithms with local search for combinatorial optimization*. university of California, San Diego, 1998.

[23] P. Moscato *et al.*, "On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms," *Caltech concurrent computation program, C3P Report*, vol. 826, 1989.

[24] Y. Yu, H. Ma, and M. Zhang, "A hybrid GP-Tabu approach to QoS-aware data-intensive Web service composition," in *Asia-Pacific Conf. on Simulated Evolution and Learning*. Springer, 2014, pp. 106–118.

[25] S. Sadeghiram, H. Ma, and G. Chen, "Composing distributed data-intensive web services using distance-guided memetic algorithm," in *International Conf. on Database and Expert Systems Applications*. Springer, 2019, pp. 411–422.

[26] N. Krasnogor and J. Smith, "A memetic algorithm with self-adaptive local search: TSP as a case study," in *Conf. on Genetic and Evolutionary Computation*. Morgan Kaufmann Publishers Inc., 2000, pp. 987–994.

[27] E. Özcan, J. H. Drake, C. Altıntaş, and S. Asta, "A self-adaptive multimeme memetic algorithm co-evolving utility scores to control genetic operators and their parameter settings," *Applied Soft Computing*, vol. 49, pp. 81–93, 2016.

[28] M. J. Dinneen and K. Wei, "On the analysis of a (1+1) adaptive memetic algorithm," in *IEEE Workshop on Memetic Computing*, 2013, pp. 24–31.

[29] T. Yao, X. Yao, S. Han, Y. Wang, D. Cao, and F. Wang, "Memetic algorithm with adaptive local search for capacitated arc routing problem," in *IEEE International conf. on Intelligent Transportation Systems*, 2017, pp. 836–841.

[30] J. Yan, M. Li, and J. Xu, "An adaptive strategy applied to memetic algorithms," *IAENG International Journal of Computer Science*, vol. 42, no. 2, pp. 29–36, 2015.

[31] V. Yannibelli and A. Amandi, "Project scheduling: A memetic algorithm with diversity-adaptive components that optimizes the effectiveness of human resources," *Polibits*, no. 52, pp. 93–103, 2015.

[32] M. Hosseini Shirvani, "Bi-objective Web service composition problem in multi-cloud environment: a bi-objective time-varying Particle Swarm Optimisation algorithm," *Journal of Experimental & Theoretical Artificial Intelligence*, pp. 1–24, 2020.

[33] M. Yaghoubi and A. Maroosi, "Simulation and modeling of an improved multi-verse optimization algorithm for QoS-aware Web service composition with service level agreements in the cloud environments," *Simulation Modelling Practice and Theory*, vol. 103, no. 4, pp. 1–16, 2020.

[34] S. Zheng, D. Feng, and J. Yu, "The algorithm of Web services composition of group decision-making based on fuzzy numbers," in *International Conference on Computer Science and Application Engineering*. ACM, 2019, pp. 1–7.

[35] T. Mohsni, Z. Brahmi, and M. M. Gammoudi, "Data-intensive service composition in cloud computing: State-of-the-art," in *IEEE/ACS International Conf. of Computer Systems and Applications*. IEEE, 2016, pp. 1–7.

[36] A. Bousrih and Z. Brahmi, "Optimizing cost and response time for data-intensive services' composition based on ABC algorithm," in *IEEE International Conf. on Information and Communication Technology and Accessibility*, 2015, pp. 1–6.

[37] Y. Chen, J. Huang, C. Lin, and X. Shen, "Multi-objective service composition with QoS dependencies," *IEEE Transactions on Cloud Computing*, vol. 7, no. 2, pp. 537–552, 2016.

[38] Y.-Y. Fanjiang, Y. Syu, C.-H. Wu, J.-Y. Kuo, and S.-P. Ma, "Genetic algorithm for QoS-aware dynamic Web services composition," in *IEEE International Conf. on Machine Learning and Cybernetics*, vol. 6, 2010, pp. 3246–3251.

[39] A. L. Blum and M. L. Furst, "Fast planning through planning graph analysis," *Artificial intelligence*, vol. 90, no. 1-2, pp. 281–300, 1997.

[40] M. Chen and Y. Yan, "QoS-aware service composition over graphplan through graph reachability," in *IEEE International Conf. on Services Computing*, 2014, pp. 544–551.

[41] R. B. Lamine, R. B. Jemaa, and I. A. B. Amor, "Graph planning based composition for adaptable semantic Web services," *Procedia Computer Science*, vol. 112, pp. 358–368, 2017.

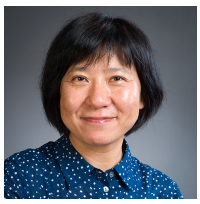
[42] Y. Yan, M. Chen, and Y. Yang, "Anytime QoS optimization over the plangraph for Web service composition," in *symposium on applied computing*. ACM, 2012, pp. 1968–1975.

[43] M. Zhu, G. Fan, J. Li, and H. Kuang, "An approach for QoS-aware service composition with graphplan and fuzzy logic," *Procedia Computer Science*, vol. 141, pp. 56–63, 2018.

[44] J. Li, Y. Yan, and D. Lemire, "Scaling up Web service composition with the skyline operator," in *IEEE International Conf. on Web Services*, 2016, pp. 147–154.

[45] S. Borzsony, D. Kossmann, and K. Stocker, "The skyline operator," in *IEEE International Conf. on Data Engineering*, 2001, pp. 421–430.
[46] K.-L. Tan, P.-K. Eng, B. C. Ooi *et al.*, "Efficient progressive skyline computation," in *VLDB*, vol. 1, 2001, pp. 301–310.
[47] Y.-M. Xia and Y.-B. Yang, "Web service composition integrating QoS optimization and redundancy removal," in *IEEE International Conference on Web Services*, 2013, pp. 203–210.
[48] S. Chattopadhyay and A. Banerjee, "QoS-aware automatic Web service composition with multiple objectives," *ACM Transactions on the Web (TWEB)*, vol. 14, no. 3, pp. 1–38, 2020.
[49] S. Sadeghiram, H. Ma, and G. Chen, "A user-preference driven lexicographic approach for multi-objective distributed Web service composition," in *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2020, pp. 791–797.
[50] ——, "A novel repair-based multi-objective algorithm for QoS-Constrained distributed data-intensive Web service composition," in *International Conference on Web Information Systems Engineering*. Springer, 2020, pp. 489–502.
[51] A. S. da Silva, Y. Mei, H. Ma, and M. Zhang, "Fragment-based genetic programming for fully automated multi-objective Web service composition," in *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, 2017, pp. 353–360.
[52] S. Sadeghiram, H. Ma, and G. Chen, "A distance-based Genetic algorithm for robust data-intensive Web service composition in dynamic bandwidth environment," in *2020 IEEE International Conference on Services Computing (SCC)*. IEEE, 2020, pp. 248–255.
[53] A. S. da Silva, Y. Mei, H. Ma, and M. Zhang, "Particle swarm optimisation with sequence-like indirect representation for Web service composition," in *European Conference on Evolutionary Computation in Combinatorial Optimization*. Springer, 2016, pp. 202–218.
[54] ——, "A memetic algorithm-based indirect approach to Web service composition," in *IEEE congress on Evolutionary Computation*, 2016.
[55] J. R. Koza *et al.*, *Genetic programming II*. MIT press Cambridge, 1994, vol. 17.
[56] F. Glover, "Tabu search—part I," *ORSA Journal on computing*, vol. 1, no. 3, pp. 190–206, 1989.
[57] P. Pecháč and M. Sága, "Controlling of local search methods' parameters in memetic algorithms using the principles of simulated annealing," *Procedia Engineering*, vol. 136, pp. 70–76, 2016.
[58] N. Q. Huy, O. Y. Soon, L. M. Hiot, and N. Krasnogor, "Adaptive cellular memetic algorithms," *Evolutionary Computation*, vol. 17, no. 2, pp. 231–256, 2009.
[59] M. J. Dinneen and K. Wei, "A (1+1) adaptive memetic algorithm for the maximum clique problem," in *IEEE congress on Evolutionary Computation*, 2013, pp. 1626–1634.
[60] C. Liu and B. Li, "Memetic algorithm with adaptive local search depth for large scale global optimization," in *IEEE congress on Evolutionary Computation*, 2014, pp. 82–88.
[61] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *science*, vol. 220, no. 4598, pp. 671–680, 1983.
[62] J. Lehman and K. O. Stanley, "Abandoning objectives: Evolution through the search for novelty alone," *Evolutionary computation*, vol. 19, no. 2, pp. 189–223, 2011.
[63] A. Eiben, M. C. Schut, and A. De Wilde, "Boosting Genetic Algorithms with self-adaptive selection," in *IEEE International Conf. on Evolutionary Computation*, 2006, pp. 477–482.
[64] P. Vajda, A. E. Eiben, and W. Hordijk, "Parameter control methods for selection operators in genetic algorithms," in *International Conf. on Parallel Problem Solving from Nature*. Springer, 2008, pp. 620–630.
[65] M. Razian, M. Fathian, H. Wu, A. Akbari, and R. Buyya, "SAIoT: Scalable anomaly-aware services composition in CloudIoT environments," *IEEE Internet of Things Journal*, 2020.
[66] Y. Wu, C. Yan, L. Liu, Z. Ding, and C. Jiang, "An adaptive multilevel indexing method for disaster service discovery," *IEEE Transactions on Computers*, vol. 64, no. 9, pp. 2447–2459, 2014.
[67] M. Deng, "Facilitating data-intensive education and research in earth science through geospatial Web services," Ph.D. dissertation, 2009.
[68] S. Deng, L. Huang, Y. Li, and J. Yin, "Deploying data-intensive service composition with a negative selection algorithm," *International Journal of Web Services Research (IJWSR)*, vol. 11, no. 1, pp. 76–93, 2014.
[69] A. Monaco, E. Pantaleo, N. Amoroso, L. Bellantuono, A. Lombardi, A. Tateo, S. Tangaro, and R. Bellotti, "Identifying potential gene biomarkers for Parkinson's disease through an information entropy based approach," *Physical Biology*, 2020.
[70] Y. Larriba, C. Rueda, M. A. Fernández, and S. D. Peddada, "Microarray data normalization and robust detection of Rhythmic features," in *Microarray Bioinformatics*. Springer, 2019, pp. 207–225.
[71] G. Agapito, "Computer tools to analyze microarray data," in *Microarray Bioinformatics*. Springer, 2019, pp. 267–282.
[72] D. Habich, S. Richly, S. Preissler, M. Grasselt, W. Lehner, and A. Maier, "BPEL DT—Data-aware extension for data-intensive service applications," in *Emerging Web Services Technology, Volume II*. Springer, 2008, pp. 111–128.
[73] G. D. Guardia, L. F. Pires, R. Z. Vêncio, K. C. Malmegrim, and C. R. de Farias, "A methodology for the development of RESTful semantic Web services for gene expression analysis," *PloS one*, vol. 10, no. 7, p. e0134011, 2015.
[74] P. Wohed, W. M. van der Aalst, M. Dumas, and A. H. Ter Hofstede, "Analysis of Web services composition languages: The case of BPEL4WS," in *International Conf. on Conceptual Modeling*. Springer, 2003, pp. 200–215.
[75] A. Wang, H. Ma, and M. Zhang, "Genetic Programming with greedy search for Web service composition," in *International Conference on Database and Expert Systems Applications*. Springer, 2013, pp. 9–17.
[76] Y. Xiang, Y. Zhou, X. Yang, and H. Huang, "A many-objective evolutionary algorithm with Pareto-adaptive reference points," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 1, pp. 99–113, 2019.
[77] A. Bansal, M. B. Blake, S. Kona, S. Bleul, T. Weise, and M. C. Jaeger, "WSC-08: continuing the Web services challenge," in *E-Commerce Technology and IEEE Conf. on Enterprise Computing*, 2008, pp. 351–354.
[78] S. Kona, A. Bansal, M. B. Blake, S. Bleul, and T. Weise, "WSC-2009: a quality of service-oriented Web services challenge," in *IEEE Commerce and Enterprise Computing*, 2009, pp. 487–490.
[79] E. Al-Masri and Q. H. Mahmoud, "Investigating Web services on the World Wide Web," in *International conf. on World Wide Web*. ACM, 2008, pp. 795–804.
[80] Z. Zheng, Y. Zhang, and M. R. Lyu, "Investigating QoS of real-world Web services," *IEEE Transactions on services computing*, vol. 7, no. 1, pp. 32–39, 2014.
[81] J. R. Koza, *Genetic programming: on the programming of computers by means of natural selection*. MIT press, 1992.

Soheila Sadeghiram received her B.E. degree from University of Isfahan and her M.S. from Shahid Beheshti University (SBU), Tehran. From January 2011 to August 2017 she worked as a full-time lecturer at UMA, Iran, She is currently a doctoral candidate at the School of Engineering and Computer Science, Victoria University of Wellington, Wellington, New Zealand.



Hui Ma Dr. Hui Ma received her B.E. degree from Tongji University and her B.S. (Hons.), M.S. and Ph.D degrees from Massey University. She is currently an Associate Professor in Software Engineering at Victoria University of Wellington. Her research interests include service computing, conceptual modelling, database systems, and resource allocation in clouds. Hui has more than 100 publications, including leading journals and conferences in databases, service computing, evolutionary computation, and conceptual modelling, such as IEEE TPDS, IEEE TCC, IEEE TSC, ER, GECCO, IEEE ICWS, IEEE SCC. She is a steering committee member of ER, and has served seven times as a PC chair for conferences such as ER, DEXA, and APCCM, and a local co-chair for Australian AI 2018 and CEC 2019.

Gang Chen Dr Gang Chen obtained his B.Eng degree from Beijing Institute of Technology in China and PhD degree from Nanyang Technological University (NTU) in Singapore respectively. He is currently a senior lecturer in the School of Engineering and Computer Science at Victoria University of Wellington. His research interests include service computing, evolutionary computation, reinforcement learning and multi-agent systems. He has more than 100 publications, including leading journals and conferences in machine learning, evolutionary computation, and distributed computing areas, such as IEEE TPDS, IEEE TEVC, JAAMAS, ACM TAAS, IEEE ICWS, IEEE SCC. He is serving as the PC member of many prestigious conferences including ICML, NeurIPS, IJCAI, and AAAI, and co-chair for Australian AI 2018 and CEC 2019.