Computer Systems Science & Engineering
DOI: 10.32604/csse.2023.031142
Article

Tech Science Press

Neighborhood Search Based Improved Bat Algorithm for Web Service Composition

Fadl Dahan[1,2,*]

[1]Department of Information System, College of Computer Engineering and Sciences, Prince Sattam bin Abdulaziz University, Al-Kharj, 11942, Saudi Arabia
[2]Department of Computer Sciences, Faculty of Computing and Information Technology Alturbah, Taiz University, Taiz, 9674, Yemen
*Corresponding Author: Fadl Dahan. Email: F.Naji@psau.edu.sa
Received: 11 April 2022; Accepted: 26 May 2022

Abstract: Web services are provided as reusable software components in the services-oriented architecture. More complicated composite services can be combined from these components to satisfy the user requirements represented as a workflow with specified Quality of Service (QoS) limitations. The workflow consists of tasks where many services can be considered for each task. Searching for optimal services combination and optimizing the overall QoS limitations is a Non-deterministic Polynomial (NP)-hard problem. This work focuses on the Web Service Composition (WSC) problem and proposes a new service composition algorithm based on the micro-bats behavior while hunting the prey. The proposed algorithm determines the optimal combination of the web services to satisfy the complex user needs. It also addresses the Bat Algorithm (BA) shortcomings, such as the tradeoff among exploration and exploitation searching mechanisms, local optima, and convergence rate. The proposed enhancement includes a developed cooperative and adaptive population initialization mechanism. An elitist mechanism is utilized to address the BA convergence rate. The tradeoff between exploration and exploitation is handled through a neighborhood search mechanism. Several benchmark datasets are selected to evaluate the proposed bat algorithm's performance. The simulation results are estimated using the average fitness value, the standard deviation of the fitness value, and an average of the execution time and compared with four bat-inspired algorithms. It is observed from the simulation results that introduced enhancement obtains significant results.

Keywords: Cloud computing; web service composition; bat algorithm; service-oriented architecture

1 Introduction

Service-Oriented Architecture is a software architecture where compassable functional units, self–contained, and stateless services are utilized to build distributed systems with loosely-coupled styles [1].

The user needs are represented using a business process where these services deal with these processes and may be combined and interacted with other services to deal with the business process [2]. However, the users' needs and the services created and published by various organizations are overgrowing. A composite service can be integrated to address this issue, called web service composition (WSC) [3]. The WSC can be initiated through the users' needs as a workflow submission; each need is represented as a task in the workflow along with quality of service (QoS) requirements such as availability, cost, response time, and so on [4].

In the WSC illustrated in Fig. 1, the workflow has a sequence of tasks n, and each task has a set of candidate web services m. Each web service is represented as WS_{ij} where i refers to the task serial number and j refers to the candidate web service to carry out the ith task. The solution of the WSC is a concrete path of web services as vector $WS_{1,X}$, $WS_{2,X'}$,…, $WS_{n,X''}$ where the vector dimension equals the number of tasks.

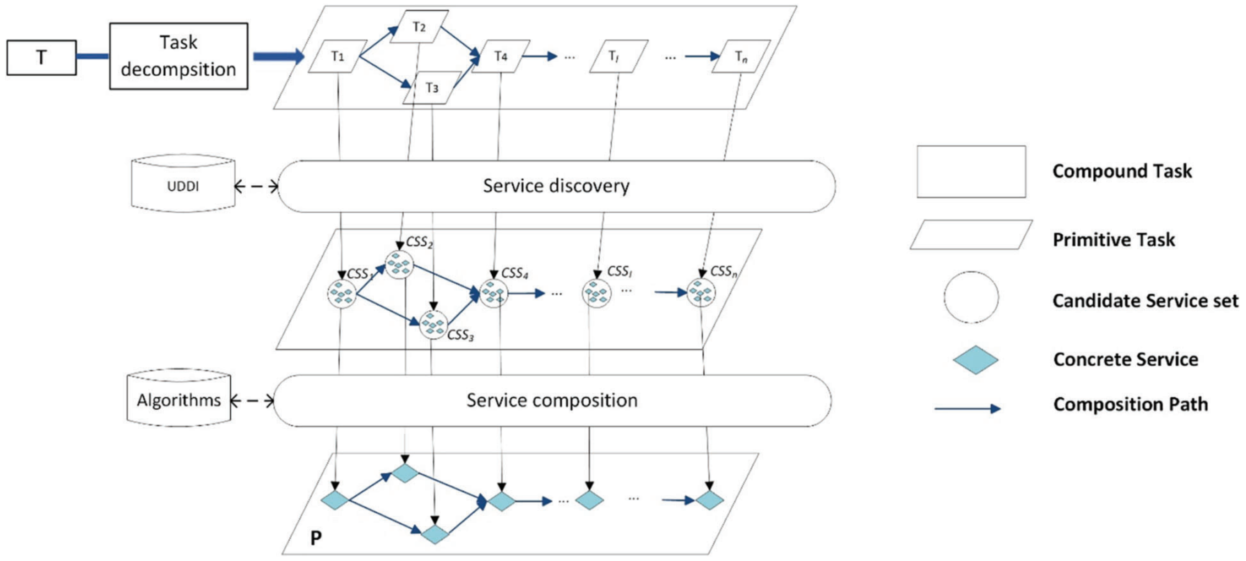

Figure 1: The WSC problem representation

Each web service in the cloud is associated with non-functional constraints and functionality. Therefore, these constraints represent the preference criteria when selecting web services. The non-functional constraints could be QoS, semantic functional, energy consumption, etc. In this work, four QoS constraints are considered as follows:

- Cost (C): represents the fee for using the service, and it is measured in any currency.
- Response Time (RT): represents the time to send and receive the service request and response. It is measured in milliseconds.
- Availability (A): represents the successful call to the total calls of services, measured in percentage.
- Reliability (R): represents the ratio of the number of errors to the total number of messages of services. It is measured in percentage.

For optimizing the WSC problem, the QoS constraints values are aggregated into a single value to estimate the solution quality based on their objective value. The C and RT objective values are minimized, while the A and R objective values are maximized. The aggregation formulas of QoS constraints are illustrated in Tab. 1.

Table 1: The QoS values aggregation formulas

QoS criteria	Aggregation formula
Cost (C)	$\sum_{i=1}^{n} C(ws_{ij})$
Response time (RT)	$\sum_{i=1}^{n} RT(ws_{ij})$
Availability (A)	$\prod_{i=1}^{n} A(ws_{ij})$
Reliability (R)	$\prod_{i=1}^{n} R(ws_{ij})$

Recently, nature-inspired algorithms have been widely adopted in service composition to search for optimal service composition. These algorithms are meta-heuristics that inspired the insects' swarm intelligence behavior, such as ant colony optimization [5], particle swarm optimization [6], and artificial bee colony [7]. The difference between these algorithms is the local and global search mechanisms, and the balance of these mechanisms is required for getting an optimal solution [8]. In this work, an existing nature-inspired algorithm was considered, i.e., the bat algorithm (BA), for solving to solve WSC problem. In the research community, the BA has become more popular optimize searching for an optimal solution to various optimization problems [9]. BA was introduced by Yang et al. [9] that was inspired by the microbats' echolocation behavior, especially prey obstacle avoidance and detection. The microbats use the echolocation feature for food (prey) detection and obstacle avoidance. For food detection, the bat emits a short pulse to produce an echo where the bat can determine the prey size and shape. The BA performance has several issues associated with global searches, local optima, convergence rate, tradeoff factor among local, and population initialization [8].

In this work, the contribution to addressing the BA shortcomings is given as:

- *Population initialization:* The nature-inspired algorithms' performance is significantly impacted by the population initialization [10,11]. The bad initialization can lead to premature convergence for future iterations [8]. Therefore, a cooperative and adaptive population initialization mechanism is introduced in this work.
- *Convergence rate:* The nature-inspired algorithms search for an optimal solution through a cooperative behavior of exploration and exploitation search mechanism. The convergence rate can be affected if a lack of collective behavior is found. An elitist mechanism is utilized to address the BA convergence rate by maintaining the cooperative behavior of the exploration and exploitation search mechanism.
- *Local optima:* An efficient neighborhood search mechanism is proved to achieve a better tradeoff between exploration and exploitation of nature-inspired algorithms [12]. In addition, it is an effective tool to overcome the local optima [8]. Therefore, an updated variant of the neighborhood search mechanism proposed in our previous works [12,13] is proposed.

2 Related Works

The recent research reported on BA-inspired algorithms for the WSC problem is summarized in this section. To the best of our knowledge, few works are developed to address the WSC problem using BA, discussed below.

Boussalia et al. [14] proposed an extension for the BA algorithm. The proposed algorithm had three different features. The first feature was an adaptation for BA to work with the WSC problem. Next, a constructive heuristic method was used to initialize the population. Finally, a transformation for the real solutions to binary vector was proposed, and heuristic reparation to verify and correct the transformation of real solutions. Podili et al. [15] introduced an evaluation of BA and hybrid BA against genetic algorithm and particle swarm optimization. Boussalia et al. [16] proposed two multi-objective algorithms based on cuckoo search and BA. For the cuckoo search inspired, quantum computing integrates with cuckoo search where the measurement operator, the quantum mutation, and the qubit representation from quantum computing were added to the core of the cuckoo search to improve the cuckoo search performance. For the BA-inspired, an adaptation for BA was adapted to work with the WSC problem. Xu et al. [17] introduced a bat algorithm with a Fuzzy operator to address the WSC problem in the cloud. The Fuzzy operator is used for coding and encoding the obtained solution to redefine the velocity. Whereas virtual bats are created based on each bat, a crossover operator is between them. Xu et al. [18] proposed a self-adaptive bat algorithm with three flight behaviors. Two behaviors are introduced to improve the exploration and exploitation based on the local and global resetting mechanisms. The third one is the investigation of the differences between the flying and resetting mechanisms.

Recently, various researchers applied other swarm intelligence algorithms in the face of finding the best possible web services path. El Allali et al. [19] introduce a framework to address the WSC problem based on the ant colony optimization and the mobile agents. A hybrid algorithm ant colony optimization and the genetic algorithm is introduced in [20]. Li et al. [21] improved the convergence speed of the artificial bee colony algorithm using the genetic algorithm. Seghir [22] proposed an improved artificial bee colony algorithm based on the fuzzy ranking method to improve the artificial bee colony algorithm diversity. Teng et al. [23] proposed enhancing the whale optimization algorithm based on the logarithmic convergence factor and aggregation potential energy. The authors in [6] applied the mutation, nonlinear convergence factor, and chaos initialization to improve the whale optimization algorithm performance. The improved eagle algorithm is introduced in [24]. Dogani et al. [25] introduced a hybrid particle swarm optimization and genetic algorithm where the genetic algorithm is used to enhance the exploration and exploitation of particle swarm optimization.

The aforementioned bat-inspired algorithms have a limitation of guaranteeing the best web services path because of the stochastic behavior of the nature-inspired algorithms. In addition, the No-Lunch-Free theorem (NLF) [26] states that the optimizers are unable to find good enough to address all optimization problems. Therefore, the algorithms mentioned above cannot efficiently solve large-scale datasets and suffer from degraded performance. This discussion motivates the work of developing new optimization algorithms to address the WSC that introduces better evaluation factors for optimization algorithms which are the efficient performance and execution time [27,28].

3 Bat Algorithm (BA)

BA is inspired by the microbats' echolocation behavior, especially prey obstacle avoidance and detection [9]. The microbats determine the size and shape of the prey by emitting a short pulse to get the nearby objects' echoes. The BA simulates the prey detection of the microbats as follows:

First: initialize the pulse emission rate (r_i^t) and bat loudness (A_i^t) using Eqs. (1) and (2)

$$A_i^{t+1} = \propto (A_i^t) \tag{1}$$

$$r_i^{t+1} = [1 - \exp(-\alpha)] \tag{2}$$

where t is a timestamp, α represents a user-specified variable between [0–1].

Second: The bats' velocity (v_i^t) and frequency (f_i^t) are computed using Eqs. (3) and (4)

$$f_i^t = f_{min}^t + (f_{max}^t - f_{min}^t) * rand() \tag{3}$$

where, f^t_{min}, f^t_{max} are the minimum and maximum frequency. rand () is a random variable between [0–1].

$$v^t_i = v^{t-1}_i + (x^t_i - x_*)f^t_i \tag{4}$$

where, v^{t-1}_i is the previous velocity, x^t_i, x_* are the current and best bat position, respectively.

Third: The bats' positions are updated using Eq. (5)

$$x^{t+1}_i = x_{new} + v^t_i \tag{5}$$

where x_{new} denotes the new bats' positions that are computed using Eq. (6)

$$x_{new} = x^t_i + randi[-1,\ 1]A^t_i \tag{6}$$

4 Improved Bat Algorithm (IBA) for WSC

The BA proposed improvements are presented in this section and shown in Fig. 2. These improvements are (1) a proposed population initialization method, (2) achieving a better convergence rate using an elitist method (3) achieving a better tradeoff between search mechanisms to avoid local optima using the neighborhood method.

4.1 Initialization Cooperative Method

The best initialization of the bats affects the search performance in future iterations, and the most used initialization approach is the random approach. To improve the performance of the proposed algorithm, an adaptive and cooperative initialization method is proposed based on the divide and conquer paradigm. In the proposed initialization method, the workflow is divided into sub workflows based on the number of tasks where each sub-workflow can be solved individually. Then these sub workflows solutions will be combined to obtain the final solution.

The Sub Workflows (SW) are obtained adaptively based on the size of the tasks in the workflow, as mentioned in Eq. (7).

$$\text{SW} = \left(\frac{n}{P}\right) \tag{7}$$

where n is the number of tasks, and P is the population size.

The population size is duplicated according to the number of the sub workflows where each population searches for a solution in the sub workflows sequentially. The sub-population sizes are obtained as mentioned in Eq. (8).

$$\begin{cases} P_1 = 1 \ \ to\ P \\ P_2 = P + 1 \ \ to\ P * 2 \\ P_3 = P * 2 + 1 \ \ to\ P * 3 \\ . \\ . \\ P_n = P * (n-1) + 1 \ \ to\ P * \beta \end{cases} \tag{8}$$

where β is the number of population partitions.

Each subpopulation's solution is computed using Eq. (9) to initialize the main bat population. Then the probability of each bat in the subpopulation among all bats is calculated using Eq. (10). The combination of the sub workflows to initialize the main population is done regarding the probability of each solution based on the roulette wheel approach.

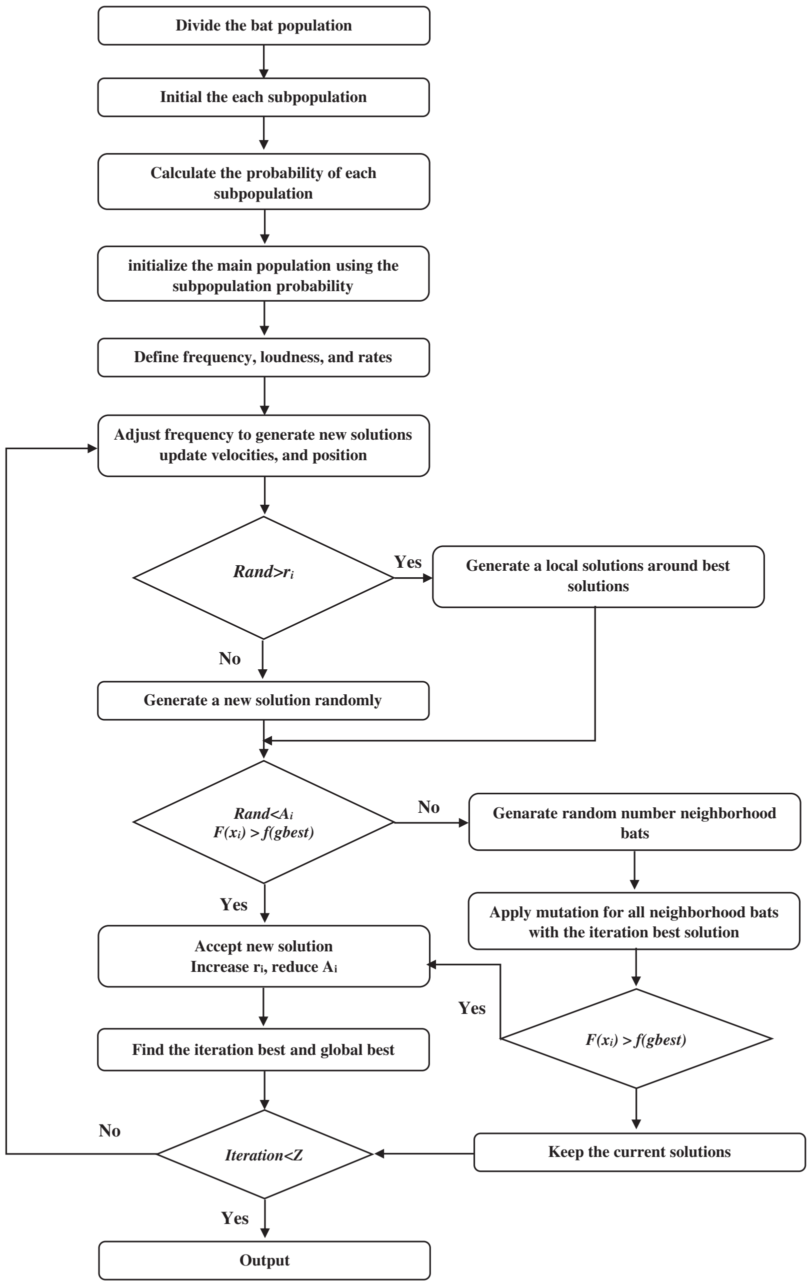
Divide the bat population
Initial the each subpopulation
Calculate the probability of each subpopulation
initialize the main population using the subpopulation probability
Define frequency, loudness, and rates
Adjust frequency to generate new solutions update velocities, and position
Rand>rᵢ
Yes
Generate a local solutions around best solutions
No
Generate a new solution randomly
Rand<Aᵢ F(xᵢ) > f(gbest)
No
Genarate random number neighborhood bats
Yes
Apply mutation for all neighborhood bats with the iteration best solution
Accept new solution Increase rᵢ, reduce Aᵢ
Yes
F(xᵢ) > f(gbest)
Find the iteration best and global best
No
Iteration<Z
Keep the current solutions
Yes
Output

Figure 2: IBA flowchart

$$argmax_{p\in P}(f) = \sum_{i=1}^{n} \frac{W_a A_i^p + R_i^p}{W_c C_i^p + W_{rt} RT_i^p} \tag{9}$$

where p represents a bat $1 \le p \le P$; W_x are the client preferences regarding QoS limitations that are adopted in this work, and x refers to the QoS limitations, which include availability weight (W_a), reliability weight (Wr), cost weight (W_c), and response time weight (W_{rt}).

$$Prop_p = \frac{f_p}{\sum_i^P f_i} \tag{10}$$

where f_p refers to the fitness value of bat p obtained from Eq. (9).

4.2 Elitist Strategy

In this work, the elitist strategy is inspired by [8], which contributes to enhancing the convergence speed rate of BA. The algorithm search pattern is the main factor of convergence speed, so the elitist strategy in this work is adapted and adjusted to accommodate the WSC problem. Accordingly, the best solution moves through future iterations.

In the elitist strategy, the iteration best solution (S_{Ibest}) can be defined as the best solution among all bats in each iteration (maximum value of iteration based fitness value f(t)). In contrast, the global best solution (S_{Gbest}) can be defined as the best solution among all bats in all iterations (maximum value of distance-based fitness value D(t)). The iteration best and global best solutions are obtained from the fitness values of the bats during a search for optimal solutions using Eqs. (11) and (12).

$$S_{Ibest} = \max(\text{iteration based fitness value}) \tag{11}$$

$$S_{Gbest} = \max(\text{distance fitness value}) \tag{12}$$

The S_{Ibest} and S_{Gbest} values are obtained using Eq. (9).

The S_{Ibest} and S_{Gbest} values are compared with the fitness values of the following iterations. If the next fitness values are better than the current values, the S_{Ibest} and S_{Gbest} are updated using Eqs. (13) and (14), respectively.

$$S_{Ibest} = \begin{cases} S_{Ibest}^{t-1} = S_{Ibest}^{t} & if\ f(t) \geq f(t-1) \\ S_{Ibest}^{t} = S_{Ibest}^{t-1} & if\ f(t) \leq f(t-1) \end{cases} \tag{13}$$

$$S_{Gbest} = \begin{cases} S_{Gbest}^{t-1} = S_{Gbest}^{t} & if\ D(t) \geq D(t-1) \\ S_{Gbest}^{t} = S_{Gbest}^{t-1} & if\ D(t) \leq D(t-1) \end{cases} \tag{14}$$

The formulas mentioned above require modifying the bats' basic formulas search, velocity, and frequency equations to attain a better tradeoff among searching mechanisms using Eqs. (15)–(18) [8].

$$f_i^t = \frac{\min(S_{Gbest}^t) + (\max(S_{Gbest}^t) - \min(S_{Gbest}^t))rand()}{\max(S_{Ibest}^t)} \tag{15}$$

$$v_i^t = v_i^{t-1} + (S_{Gbest}^t - S_{Ibest}^t)f_i^t \tag{16}$$

$$x_{new} = S_{Gbest}^t + randi[-1,\ 1] \tag{17}$$

$$x_i^{t+1} = \begin{cases} S_{Gbest}^t & if\ rand\ >\ r_i \\ x_{new} + v_i^t & otherwise \end{cases} \tag{18}$$

where i refer to ith bat at iteration t.

4.3 Neighborhood Search

An updated variant of the neighborhood search mechanism proposed in our previous works [12,13] is proposed. An adaptive neighborhood search mechanism has been proved to achieve a better tradeoff between exploration and exploitation of nature-inspired algorithms [12]. In addition, it is an effective tool to overcome the local optima [8]. Previously in [12,13], the neighborhood search was based on the neighborhood web services of the best web services selected so far. However, this process is time-consuming depending on the number of neighborhood web services. In this work, the neighborhood search is improved to be done using the neighborhood bats where:

First: Check the improvement status in the current iteration; if there is no improvement, then continue to the neighborhood search procedure; else, terminate the neighborhood search procedure.

Second: A Random Number (RN) of neighborhood bats is selected in a case where $1 \leq RN \leq P$.

Third: For each neighborhood bat, a mutation is applied where a randomly selected task in the workflow will be mutated with a task value from the iteration best solution.

5 Experimental Setup and Results

The simulation results of the IBA are represented in this section. Since there are no standard datasets for WSC [18], in this work, the synthetic dataset is generated using the tool introduced in [29] for simulation in five datasets. All datasets are a large scale where the number of tasks (n) is 100, 200, 300, 400, and 500, and the number of web services in each dataset equals 2*n. The idea behind these large-scale datasets is the growing number of providers and the expansion in the demands for web services over the cloud. Therefore, the larger-scale datasets are necessary to test the algorithms' performance [18] rather than small-scale datasets. The QoS constraints (C, A, RT, and R) values are generated arbitrarily from 1 to 1000 for each web service. All algorithms shown in this work are implemented using Java, and all experiments are conducted on a PC Intel(R) Core(TM) i7-7500U CPU @ 2.70 GHz and 8.0GB RAM of memory.

5.1 Experimental Setup

In this simulation, four bat-inspired algorithms are chosen for comparison purposes. These algorithms are the standard BA, Random Bat Algorithm (RBA) [18], Self-Adaptive Bat Algorithm (SABA) [18], and BA-WSC [14]. The control parameter of RBA, SABA, and BA-WSC are set as mentioned in their works, except for the population size (P) and maximum iterations (Z), where they are set to 100 and 500, respectively. The adaptive behavior of the IBA reduces the parameter setting where the neighborhood bats, sub workflows, and neighborhood bats are obtained automatically. For other parameters, the initial values are listed in Tab. 2.

Table 2: Control parameter settings of IBA

Parameter	*Value*
F_{min}	0
F_{max}	2
A	0.25
r	0.5

5.2 Improvement Strategies Verification

The proposed algorithm is composed of three different improvement strategies (Initialization Cooperative (IC), Elitist Strategy (ES), and Neighborhood Search (NS)). This section will assess the effectiveness of these three improvements on the BA incrementally. To test the effectiveness of all improvement strategies, the three instances of the proposed algorithm were created and compared to BA.

The three instances are IBA-IC (the proposed algorithm with the initialization cooperative method only), IBA-ES (the proposed algorithm with elitist strategy and initialization cooperative methods), and IBA (proposed algorithm with all improvement strategies). The experiments were done on all datasets with only one run and similar parameters mentioned above.

Fig. 3 illustrates the fitness value curve of the BA, IBA-IC, IBA-ES, and IBA on all datasets based on 500 maximum iterations. It is clearly noticed from the figure that the IC method increases the start search point and effects on enhancing the solution quality compared to the BA. These results support the abovementioned claim that the nature-inspired algorithms' performance is significantly impacted by the population initialization [10,11]. However, the random searching behaviors may contribute to the searching results during the execution, so we can see from the result on dataset 400 × 800 that the IBA-IC suffers from local optima. As a result, the BA outperforms it.

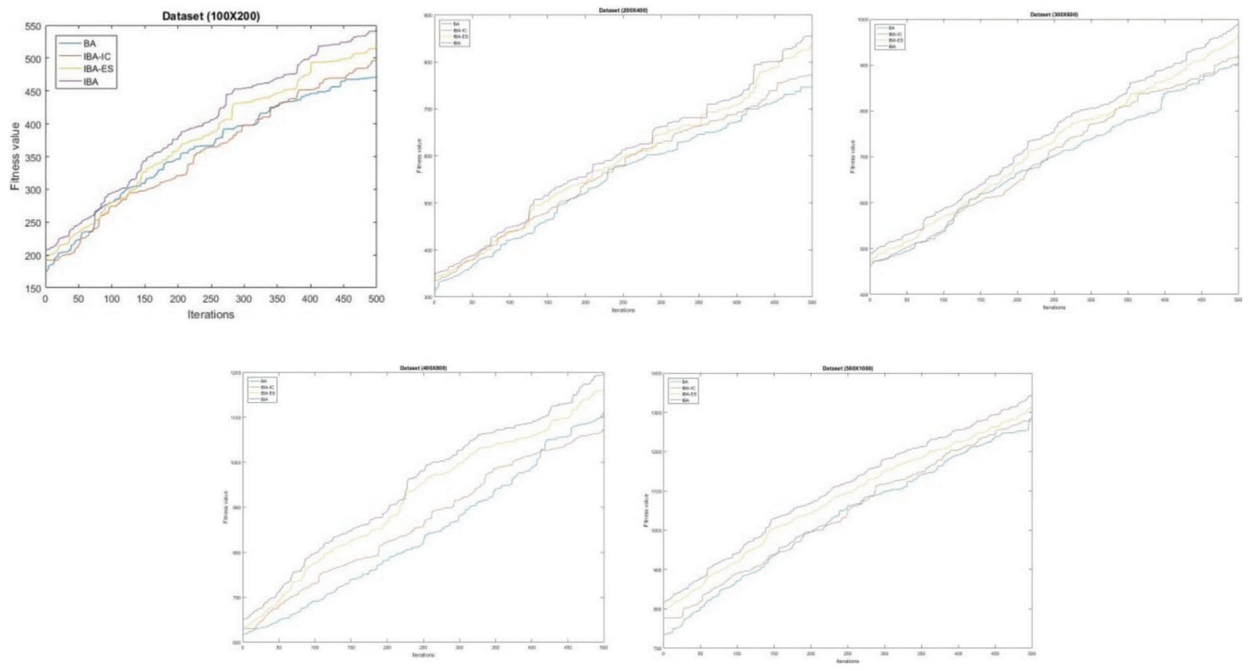

Figure 3: Fitness value curve obtained using the three IBA instances compared to BA on 500 iterations

The elitist strategy was added to the IBA-IC and notated as IBA-ES. In the IBA-ES, the elitist strategy behavior is tested along with the initialization cooperative on the BA to see the improvement of ES. It can be seen from Fig. 3 that the ES improves the convergence speed and stability of the IBA-IC among the large-scale datasets. These results support the abovementioned claim that the algorithm search pattern is the main factor of convergence speed, so the elitist strategy in this work is adapted and adjusted to accommodate the WSC problem.

Finally, the neighborhood search was added to the IBA-IC and IBA-ES and notated as IBA. NS aims to achieve a better tradeoff between exploration and exploitation of BA. The results show that the IBA performance is enhanced compared to the IBA-ES performance, and this is because of the proposed algorithm's best balancing of exploration and exploitation ability.

The results above prove the applicability of the three proposed enhancements to improve BA performance. The first enhancement aims to enhance the search starting points and convergence speed; the second enhancement was proposed. The exploration and exploitation were enriched using the third enhancement to improve the performance.

5.3 Convergence Speed of IBA Compared to Other Algorithms

To verify the stability of the IBA compared to other algorithms, the convergence speed of IBA was reached using 500 iterations in one run. Fig. 4 illustrates all algorithms' convergence rates overall algorithms.

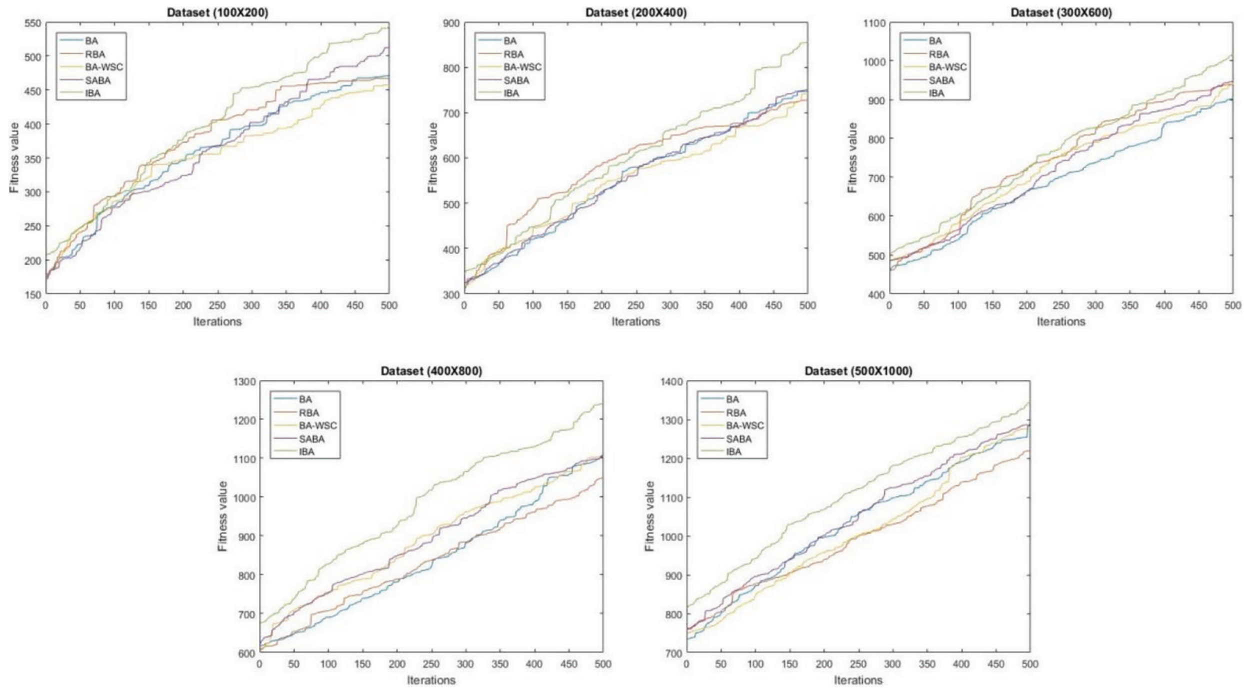

Figure 4: Fitness value curve obtained using the IBA, BA, RBA, BA-WSC, and SABA over all datasets

It is clearly seen that the IBA outperforms all competitors where it has a fast convergence rate. The figure also depicts that the IBA has the best initial value, which confirms the effectiveness of the IC method in improving the initialization process of the bats. Furthermore, the figure illustrates that the competitors suffer from local optima with a different level where their convergence rates have fluctuated. For example, the results on dataset 200×400 show that the RBA algorithm achieves better results than IBA; however, this achievement didn't continue because it suffers from local optima after approximately iteration number 300, and IBA then overcomes it. The reasonable IBA convergence rate is because of the ES and NS mechanism's collaboration, which helps IBA avoid the local optima and achieve better exploration and exploitation behavior.

5.4 Result Comparison of IBA Compared to Other Algorithms

In this subsection, the simulation results are discussed of the IBA, standard BA, and three bat-based algorithms of related works using the five large-scale datasets with 30 different runs and 500 maximum iterations. The results are evaluated the average fitness value, an average execution time, and the Standard Deviation (Std) of the fitness value. Tab. 3 reports the fitness values average and standard deviation by implementing the five algorithms to address the WSC problem overall datasets obtained from 30 independent runs. The best results are highlighted in bold. As shown in Tab. 3, The IBA outperforms BA, RBA, BA-WSC, and SABA in all datasets. Furthermore, we can observe that the proposed mechanism, especially the elitist strategy, effectively improves the convergence rate and the solution quality.

Table 3: The fitness values average and standard deviation of the IBA, BA, RBA, BA-WSC, and SABA over all datasets

		100*200	200*400	300*600	400*800	500*1000
BA	Mean	483.19	734.26	930.87	1108.22	1250.07
	Std	18.45	38.51	28.18	32.13	35.75
RBA	Mean	485.10	731.83	919.21	1110.16	1293.46
	Std	24.83	28.84	24.45	39.68	30.15
BA-WSC	Mean	478.28	733.09	951.83	1126.43	1267.67
	Std	19.72	31.44	33.57	37.44	37.82
SABA	Mean	551.27	767.49	944.29	1130.06	1297.38
	Std	24.67	32.37	35.46	46.48	41.69
IBA	Mean	593.20	810.39	995.52	1239.25	1365.92
	Std	32.73	33.66	24.78	39.14	39.76

Figs. 5 and 6 illustrate the Boxplot relative to 30 different runs of the IBA, BA, RBA, BA-WSC, and SABA over all datasets. The Boxplots are utilized to analyze the optimizers' variability to get the average fitness value and average execution time in MilliSeconds (MS) in all the runs. In Fig. 5, the Boxplots justify and confirm the better performance of IBA compared to the competitors in terms of the average fitness value. In Fig. 6, the Boxplots show the competitive performance of IBA compared to the competitors in terms of the average execution time.

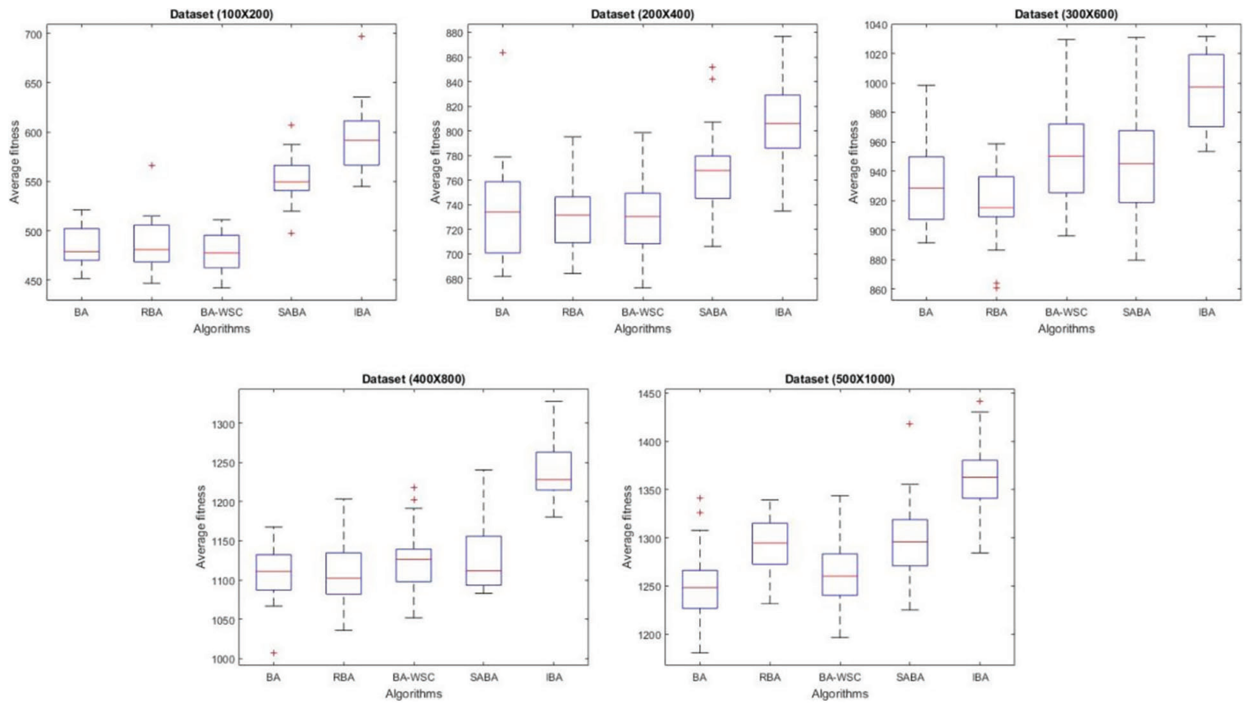

Figure 5: Boxplots representation of the average fitness value of the IBA, BA, RBA, BA-WSC, and SABA over all datasets

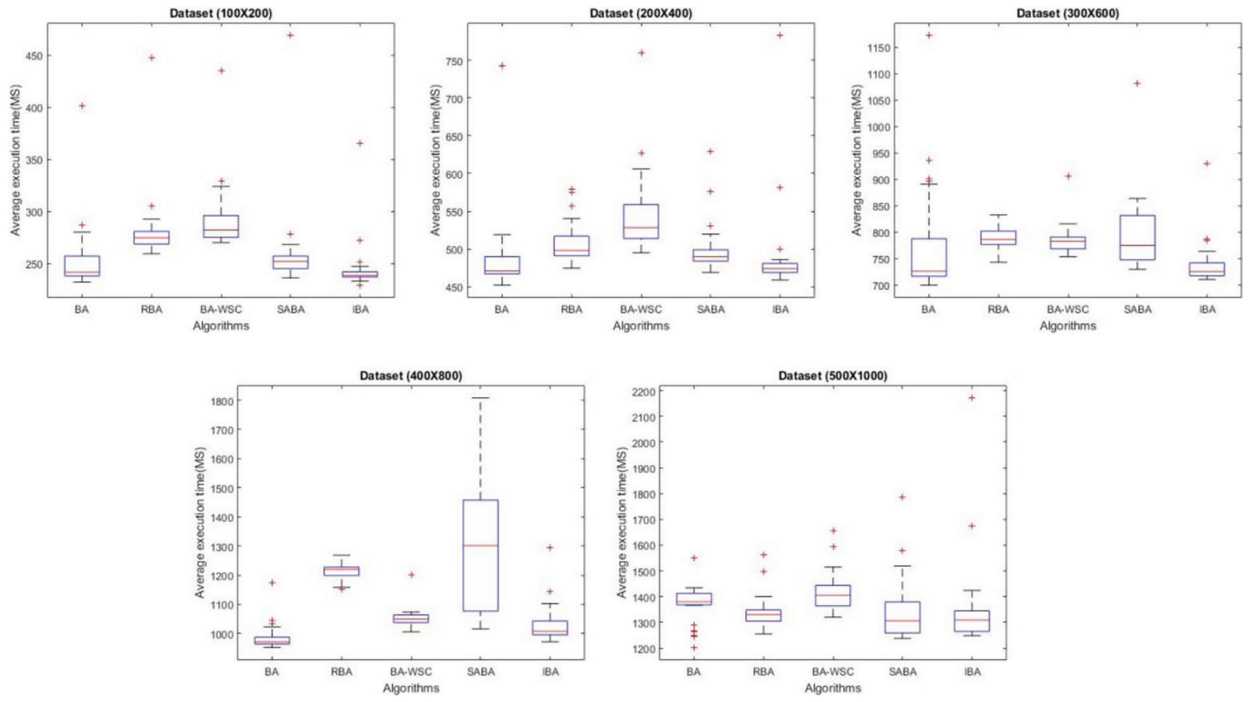

Figure 6: Boxplots representation of the average execution time of the IBA, BA, RBA, BA-WSC, and SABA over all datasets

The abovementioned results depict, on average, that the IBA outperforms the standard BA and other competitors. Wilcoxon’s rank-sum statistical test is conducted to confirm the significance of IBA compared to the competitors in each independent run and evaluate the overall performance. Tabs. 4 and 5 show the Wilcoxon test results at a significance level of 0.05 in terms of average fitness value and execution time. In the table, “–”, “+”, “=” denotes that the competitors’ performance is worse, better than, or equal to the version of IBA, respectively.

Table 4: Wilcoxon’s results of all algorithms regarding the average fitness value

	BA	*RBA*	*BA-WSC*	*SABA*
−	5	5	5	5
+	0	0	0	0
=	0	0	0	0

Table 5: Wilcoxon’s results of all algorithms regarding the average execution time

	BA	*RBA*	*BA-WSC*	*SABA*
−	2	5	4	4
+	2	0	1	1
=	1	0	0	0

Tab. 4 shows that the IBA provides the best results for all datasets, and the results also show that the IBA exhibits better stability and an increase in the scale of the datasets. Furthermore, Tab. 5 indicates that the IBA algorithm provides the best execution time for all the datasets. It gives the best execution time than BA-WSC and SABA on four datasets. The IBA has competitive results compared to BA.

6 Conclusion

In this work, three improvements are proposed to improve the BA performance for the web service composition problem. The proposed improvements efficiently deal with BA's population initialization, convergence speed, exploration, and exploitation balancing issues. A cooperative and adaptive population initialization mechanism is developed to resolve the population initialization problem. An elitist strategy is utilized to address the BA convergence rate. Furthermore, a neighborhood search mechanism is designed to achieve a better tradeoff between exploration and exploitation searching mechanisms. The IBA performance is tested using five large-scale datasets compared to the standard BA and three bat-based algorithms of related works. The IBA achieves better quality solutions results in average fitness value, standard deviation fitness value, and average execution time.

Furthermore, the effect of the three proposed improvements was investigated, and their applicability was proved. It is observed from the simulation results that introduced enhancement obtains significant results compared to four bat-inspired algorithms. In the future, we are planning to continue the investigation of using different nature-inspired algorithms to solve the WSC problem and their hybridization.

Acknowledgement: The author would like to express their gratitude to Prince Sattam bin Abdulaziz University, Al-Kharj 11942, Saudi Arabia for providing the support.

Funding Statement: The author extend their appreciation to Deputyship for research & Innovation, Ministry of Education in Saudi Arabia for funding this research work through the Project Number (IF-PSAU-2022/01/19619).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] H. Fekih, S. Mtibaa and S. Bouamama, "Local-consistency web services composition approach based on harmony search," *Procedia Computer Science*, vol. 112, pp. 1102–1111, 2017.

[2] M. Masdari, M. Nouzad and S. Ozdemir, "QoS-driven metaheuristic service composition schemes: A comprehensive overview," *Artificial Intelligence Review*, vol. 54, pp. 1–68, 2021.

[3] H. Kurdi, F. Ezzat, L. Altoaimy, S. H. Ahmed and K. Youcef-Toumi, "Multicuckoo: Multi-cloud service composition using a cuckoo-inspired algorithm for the internet of things applications," *IEEE Access*, vol. 6, pp. 56737–56749, 2018.

[4] M. Masdari, S. ValiKardan, Z. Shahi and S. I. Azar, "Towards workflow scheduling in cloud computing: A comprehensive analysis," *Journal of Network and Computer Applications*, vol. 66, pp. 64–82, 2016.

[5] M. Dorigo, V. Maniezzo and A. Colorni, "Ant system: Optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 26, no. 1, pp. 29–41, 1996.

[6] T. Cura, "A particle swarm optimization approach to clustering," *Expert Systems with Applications*, vol. 39, no. 1, pp. 1582–1588, 2012.

[7] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Technical Report-tr06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.

[8] A. Kaur and Y. Kumar, “Neighborhood search based improved bat algorithm for data clustering,” *Applied Intelligence*, vol. 52, pp. 1–35, 2022.

[9] X. S. Yang, “A new metaheuristic Bat-inspired algorithm,” *Studies in Computational Intelligence*, vol. 284, pp. 65–74, 2010.

[10] J. Senthilnath, S. Kulkarni, J. A. Benediktsson and X. -S. Yang, “A novel approach for multispectral satellite image classification based on the bat algorithm,” *IEEE Geoscience and Remote Sensing Letters*, vol. 13, no. 4, pp. 599–603, 2016.

[11] S. Neelima, N. Satyanarayana and P. Krishna Murthy, “Minimizing frequent itemsets using hybrid ABCBAT algorithm,” *Data Engineering and Intelligent Computing*, vol. 542, pp. 91–97, 2018.

[12] F. Dahan, H. Mathkour and M. Arafah, “Two-step artificial bee colony algorithm enhancement for QoS-aware Web service selection problem,” *IEEE Access*, vol. 7, pp. 21787–21794, 2019.

[13] F. Dahan, K. El Hindi and A. Ghoneim, “Enhanced artificial bee colony algorithm for QoS-aware web service selection problem,” *Computing*, vol. 99, no. 5, pp. 507–517, 2017.

[14] S. R. Boussalia, A. Chaoui and A. Hurault, “Qos-based web services composition optimization with an extended bat inspired algorithm,” *Communications in Computer and Information Science*, vol. 538, pp. 306–319, 2015.

[15] P. Podili, K. K. Pattanaik and P. S. Rana, “BAT and hybrid BAT meta-heuristic for quality of service-based web service selection,” *Journal of Intelligent Systems*, vol. 26, no. 1, pp. 123–137, 2017.

[16] S. R. Boussalia, A. Chaoui, A. Hurault, M. Ouederni and P. Queinnec, “Multi-objective quantum inspired cuckoo search algorithm and multi-objective bat inspired algorithm for the web service composition problem,” *International Journal of Intelligent Systems Technologies and Applications*, vol. 15, no. 2, pp. 95–126, 2016.

[17] B. Xu and Z. Sun, “A fuzzy operator based bat algorithm for cloud service composition,” *International Journal of Wireless and Mobile Computing*, vol. 11, no. 1, pp. 42–46, 2016.

[18] B. Xu, J. Qi, X. Hu, K. -S. Leung, Y. Sun *et al.,* “Self-adaptive bat algorithm for large scale cloud manufacturing service composition,” *Peer-to-Peer Networking and Applications*, vol. 11, no. 5, pp. 1115–1128, 2018.

[19] N. El Allali, M. Fariss, H. Asaidi and M. Bellouki, “A web service composition framework in a heterogeneous environment,” *Journal of Ambient Intelligence and Humanized Computing*, vol. 13, pp. 1–25, 2022.

[20] Z. Wang, “Optimization of resource service composition in cloud manufacture based on improved genetic and ant colony algorithm,” *Smart Innovation, Systems and Technologies*, vol. 268, pp. 183–198, 2022.

[21] T. Li, Y. Yin, B. Yang, J. Hou and K. Zhou, “A self-learning bee colony and genetic algorithm hybrid for cloud manufacturing services,” *Computing*, vol. 104, pp. 1–27, 2022.

[22] F. Seghir, “FDMOABC: Fuzzy discrete multi-objective artificial bee colony approach for solving the non-deterministic QoS-driven web service composition problem,” *Expert Systems with Applications*, vol. 167, pp. 114413, 2021.

[23] X. Teng, Y. Luo, T. Zheng and X. Zhang, “An improved whale optimization algorithm based on aggregation potential energy for QoS-driven web service composition,” *Wireless Communications and Mobile Computing*, vol. 2022, pp. 1–13, 2022.

[24] V. Rajendran, R. K. Ramasamy and W. -N. Mohd-Isa, “Improved eagle strategy algorithm for dynamic web service composition in the IoT: A conceptual approach,” *Future Internet*, vol. 14, no. 2, pp. 56, 2022.

[25] J. Dogani and F. Khunjush, “Cloud service composition using genetic algorithm and particle swarm optimization,” in *Proc. 2021 11th Int. Conf. on Computer Engineering and Knowledge (ICCKE)*, Mashhad, Iran, pp. 98–104, 2022.

[26] D. H. Wolpert and W. G. Macready, “No free lunch theorems for optimization,” *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.

[27] X. Zhang, W. Zhang, W. Sun, H. Wu, A. Song *et al.,* “A real-time cutting model based on finite element and order reduction,” *Computer Systems Science and Engineering*, vol. 43, no. 1, pp. 1–15, 2022.

[28] X. Zhang, J. Zhou, W. Sun and S. K. Jha, “A lightweight CNN based on transfer learning for COVID-19 diagnosis,” *Computers, Materials & Continua*, vol. 72, no. 1, pp. 2475–2491, 2022.

[29] X. Wang, Z. Wang and X. Xu, “An improved artificial bee colony approach to QoS-aware service selection,” in *Proc. IEEE 20th Int. Conf. on Web Services, ICWS 2013*, Santa Clara, CA, USA, pp. 395–402, 2013.