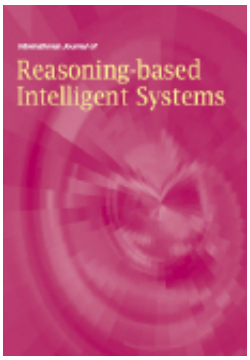
INDERSCIENCE PUBLISHERS
Linking academia, business and industry through research



International Journal of Reasoning-based Intelligent Systems

ISSN online: 1755-0564 - ISSN print: 1755-0556
https://www.inderscience.com/ijris

E-commerce cloud computing data migration method based on improved slime mould algorithm

Yujie Li, Ning Liu, Liming Dang, Yong Huang

DOI: 10.1504/IJRIS.2025.10071385

Article History:

Received:	30 March 2025
Last revised:	22 April 2025
Accepted:	23 April 2025
Published online:	11 June 2025

E-commerce cloud computing data migration method based on improved slime mould algorithm

Yujie Li, Ning Liu and Liming Dang

Economics and Management School,
Nanchang Institute of Science and Technology,
NanChang 330108, China
Email: liyujie@stu.ncpu.edu.cn
Email: 18720937405@163.com
Email: 18770051539@163.com

Yong Huang*

Business School,
Jiangxi Modern Polytechnic College,
NanChang 330000, China
Email: yong08250017@163.com
*Corresponding author

Abstract: The inherent dynamism of e-commerce cloud environments, characterised by massive data volumes and high-concurrency demands, poses significant challenges to traditional data migration approaches. For this reason, this paper firstly improves the sticky mushroom algorithm (SMA) (BOSMA) by replacing the anisotropy operator of the SMA with a balanced optimisation operator, and adding a stochastic difference variance operator to the SMA to avoid premature convergence. To reduce the migration energy consumption, the e-commerce cloud computing data migration task dependency graph is designed, and the objective function to lower the energy consumption during migration is constructed. The BOSMA is adopted to discover the finest solution to the target function to acquire the migration of the task with the largest reduction in energy consumption. The simulation outcome implies that the BOSMA method can quickly search for solutions with less migration energy consumption.

Keywords: e-commerce cloud computing; data migration; sticky mushroom algorithm; SMA; balancing optimiser; task dependency graph.

Reference to this paper should be made as follows: Li, Y., Liu, N., Dang, L. and Huang, Y. (2025) 'E-commerce cloud computing data migration method based on improved slime mould algorithm', *Int. J. Reasoning-based Intelligent Systems*, Vol. 17, No. 7, pp.1–10.

Biographical notes: Yujie Li received his PhD in Segi University of Malaysia. He has six-year experience in international trade and e-commerce in the enterprise, and is currently a teacher in the Economics and Management School at Nanchang Institute of Science and Technology. His research interests include e-commerce, international trade and data mining.

Ning Liu received her Bachelor's in Accounting from the Nanchang Institute of Science and Technology. She is currently a teacher in the Economics and Management School at Nanchang Institute of Science and Technology. Her research interests include machine learning and financial management.

Liming Dang received his Master's in Management from the University Sains Malaysia. He currently a teacher in the Economics and Management School at Nanchang Institute of Science and Technology. His research interests include international marketing, digital marketing and digital marketing.

Yong Huang received his PhD in Management from the University Sains Malaysia. He is currently a teacher in the Economics and Management School at Nanchang Institute of Science and Technology. His research interests include data mining, international marketing and digital marketing.

1 Introduction

At a time when the wave of digitisation is sweeping the world, the e-commerce industry is booming and generating huge amounts of data. Cloud computing technology, leveraging its powerful storage and computing capacity, serves as the backbone for e-commerce businesses in handling and managing data (Almarabeh and Majdalawi, 2019). The efficiency and quality of data migration, as an important aspect of cloud computing applications, directly affects the continuity and stability of e-commerce business (Shakya, 2019). Efficient, secure, and affordable data migration in complex cloud environments has become a key concern for researchers and practitioners alike. E-commerce cloud computing data migration is a complex optimisation problem involving multiple objectives and constraints such as migration time, cost, quality of service, and resource utilisation (Andrikopoulos et al., 2013). When addressing such problems, conventional mathematical optimisation approaches frequently encounter significant challenges, including prohibitive computational complexity and the inherent difficulty in obtaining globally optimal solutions (Song et al., 2019). Heuristic optimisation algorithms, especially population intelligence algorithms, show significant advantages in solving cloud computing data migration problems by virtue of their powerful global search capability and robustness (Huang and Abnoosian, 2020). Hence, optimising e-commerce cloud data migration using heuristic algorithms remains a critical and valuable research challenge.

The traditional cloud data transfer approaches include stop-and-copy, iterative migration, and on-demand replication migration (Gholami et al., 2016). Li et al. (2019) proposed an overhead-sensitive cloud data migration method based on the overhead-sensitive cloud data migration method, but the method is not applicable to real-time migration. Shekhar and Sharvani (2021) conducted a systematic evaluation of multi-tenant data models to address real-time database migration in multi-tenant cloud environments. Their proposed iterative migration technique achieved minimal downtime while maintaining optimal performance during live migration of multi-tenant databases. Teli et al. (2016) find the target node based on the resource access and the size of the data to be migrated, the query operation during migration is done at the source node, and the update operation is done at both the source and destination nodes, but the migration overhead is high. Fahmideh et al. (2019) implemented a priority-based migration strategy that executes higher-priority tasks pre-migration, with dynamic priority adjustment during and post-migration. However, the approach demonstrated performance limitations under high-load conditions.

Traditional methods often fail to cope with dynamically changing cloud computing environments, resulting in migration solutions that are locally optimal but overall ineffective. Heuristic optimisation algorithms are able to search extensively in the solution space to avoid falling into locally optimal solutions, thus finding a better data migration scheme. Zhao et al. (2023) offered an enhanced multi-objective virtual machine (VM) allocation strategy using ant colony optimisation (ACO) algorithm, which effectively solves issues of physical host load balancing and energy and power consumption, but the algorithm converges slowly. Chawla et al. (2020) improved processes such as mutation and mating in genetic algorithm (GA) to increase the probability of inheriting high quality genes to the offspring and utilised the improved GA to achieve cloud computing data migration with improved migration efficiency. Singh and Dhir (2019) used GA-optimised neural networks to predict the load on servers and thus allocate a small number of VMs to achieve the goal of energy saving. However, the diversity of resources in cloud data centres is ignored. Kak et al. (2024) proposed a VM placement and migration method based on sparrow search algorithm (SSA) but with low utilisation of physical servers. Durairaj and Sridhar (2024) optimised data migration strategy for cloud computing by using mayfly algorithm (MA) but without considering the communication between VMs and migration cost.

In contrast to other heuristic optimisation techniques, the slimy mushroom algorithm (SMA) has a strong global search capability by simulating the positive and negative feedback mechanism of slimy mushrooms, which can adeptly manages the trade-off between global searching and local refining, ensuring it doesn't settle for a suboptimal solution. Guo et al. (2017) achieved efficient deployment of VMs in cloud environments based on the SMA approach, which improved application performance also developed an energy-efficient management method for cloud data centres, but neglected the impact of the optimal solution on future node loads. Kumar and Rajesh (2023) introduced the SMA algorithm and the idea of simulated annealing to propose a heuristic dynamic migration strategy for VMs. The algorithm optimises the optimal solution for each cycle but neglects the cost spend of migration.

Current research reveals that conventional cloud data migration approaches are inadequate for addressing the dynamic, heterogeneous complexities of e-commerce cloud environments, resulting in a significant increase in migration energy consumption. To address the above issues, SMA is firstly improved based on the balanced optimiser (BOSMA) and stochastic difference variance operator to address the problems such as the slow convergence speed of SMA. In place of the anisotropy operator in SMA, the search operator from BO is used. Use the historical optimal position of the population in the mining process of the SMA to guarantee the approach's convergence to the best possible solution. The introduction of a stochastic difference variance operator enhances the algorithm's exploration capability, reducing the probability of local optima entrapment and mitigating premature convergence issues. For the goal of reducing the migration energy consumption, the e-commerce cloud computing data migration task dependency graph is designed. And a mathematical model with the objective function of minimising the migration energy consumption is constructed. Finally, the BOSMA algorithm solves the objective function by migrating all the

tasks that can be migrated to the cloud as the initial solution, and then calculates the energy savings of the migratable tasks running on the mobile terminal one by one, and migrates the tasks with the largest savings to the mobile terminal in order. For each migrated task, the algorithm updates the energy savings of each task based on the communication time between the tasks. Simulation results indicate that the proposed method significantly reduces migration energy consumption and migration time, providing an effective way for e-commerce enterprises to realise fast and reliable data migration in cloud computing environment.

2 Relevant technologies

2.1 Overview of cloud computing related knowledge

Cloud computing refers to a paradigm that combines large-scale, economical hardware devices through virtualisation technologies to create an extensive resource pool. This architecture abstracts underlying infrastructure details, allowing users to provision tailored resources and services based on diverse needs (Diaby and Rad, 2017). The architecture of cloud computing is shown in Figure 1, which includes software service layer, platform layer, infrastructure layer and hardware layer (Jawed and Sajid, 2022).

- Software layer: similar to consumers shopping in the supermarket, only need to choose the preferred product, for the product of the pre-late issues do not need to consider, these tasks are responsible for the completion of the cloud service provider.
- Platform layer: the platform layer typically includes services such as operating systems, virtualisation technologies, storage, and platform services such as development, testing, deployment, monitoring, and management for applications.
- Infrastructure layer: the cloud computing infrastructure layer provides a reliable base environment for applications and services, enabling developers and users to rapidly deploy and manage applications and services.
- Hardware layer: these hardware components are integrated to form a large-scale distributed computing system, delivering cloud services characterised by high availability, scalability, flexibility, and robust security.

2.2 Slime mould algorithm

SMA represents the problem as a maze in which slime moulds are placed, and then observes the paths formed by the slime moulds as they search for food; these paths are used to optimise the solution of the problem (Son and Khoi, 2024). In practice, SMA can be used to solve optimisation problems such as the traveller's problem, data migration, etc. Compared with classical algorithms such as GA, PSO, ACO, etc., SMA has the advantage of being able to show excellent performance in dealing with large-scale problems, and it has strong parallelism. The approach encompasses three fundamental steps, as detailed below.

1 Proximity to food. The behaviour of slime moulds was modelled as a mathematical formula to simulate the contraction pattern according to equation (1).

$$X(t+1)=\begin{cases} X_b(t)+v_b\cdot\left(W\cdot X_A(t)-X_B(t)\right), r<p \\ v_c\cdot X(t), r\geq p \end{cases} \tag{1}$$

where t is the current iteration number, $X_b(t)$ is the current optimal individual position, there are two random individual positions, one is $X_A(t)$ and the other is $X_B(t)$, the mass of the slime mould is W, which represents the fitness weight, the control parameters are v_b and v_c, r is an arbitrary number between 0 and 1, and the control variable p and the parameter a are implied in equation (2) and equation (3).

$$p=\tanh\left|S(i)-DF\right| \tag{2}$$

$$a=arctanh\left(-\left(\frac{t}{t_{\max}}\right)+1\right) \tag{3}$$

The current individual fitness is chosen to represent $S(i)$, $i = 1, 2, \ldots, n$, the best fitness value is DF, and the maximum amount of epochs is t_{max}. The updating strategy for W is shown below.

$$W(smell(i))=\begin{cases} 1+r\cdot\log\left(\frac{bf-S(i)}{bf-wf}+1\right), if\ i=1 \\ 1-r\cdot\log\left(\frac{bf-S(i)}{bf-wf}+1\right), if\ i=0 \end{cases} \tag{4}$$

where $smell(i) = sort(S)$, bf is the best iterative fitness value.

2 Surrounding food. Slime moulds update their own state and adjust their own movement direction and speed to avoid obstacles and optimise their paths, and ultimately achieve the encirclement of food.

$$X(t+1)=\begin{cases} rand\cdot(UB-LB)+LB, rand<z \\ X_b(t)+v_b\cdot\left(W\cdot X_A(t)-X_B(t)\right), r<p \\ v_c\cdot X(t), r\geq p \end{cases} \tag{5}$$

where UB and LB represent the upper and lower bounds of the seek region, respectively, rand is the number of randomisers, and z is a custom parameter.

3 Acquisition of food. The organism employs chemical signalling to detect food presence in its vicinity, dynamically adjusting both locomotion direction and velocity in response to chemical concentration gradients to navigate toward the food source.

Figure 1 Cloud computing architecture (see online version for colours)

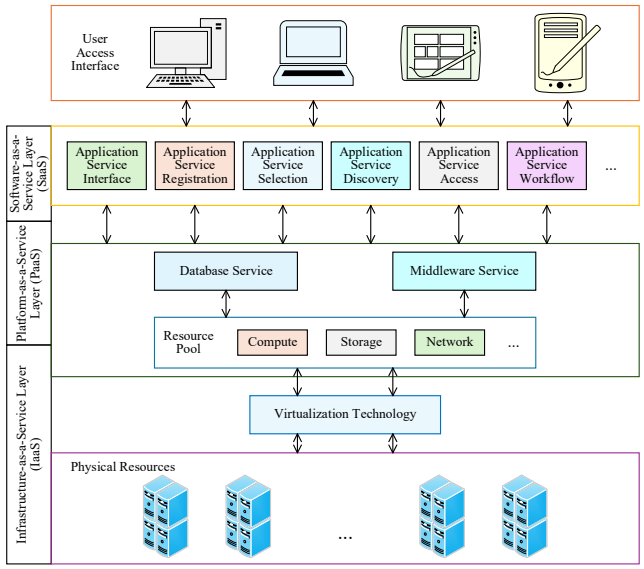
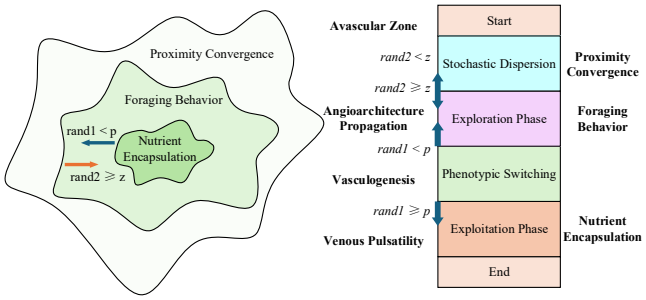


Figure 2 SMA search process (see online version for colours)



3 Optimisation of slime mould algorithm based on balance optimiser

Intending to the issues of sluggish convergence rate, low initial population diversity, and poor global optimisation of standard SMA, this paper uses the balanced optimiser (BO) (Jia and Peng, 2021) and stochastic differential variational operator (Kharazmi and Zayernouri, 2019) to improve the conventional SMA (BOSMA) to equilibrate the algorithm's ability to explore and exploit effectively, and increase the probability of the approach to seek out the best solution within the entire solution space, the flow of the approach is shown in Figure 3. First, the search operator from BO takes the place of SMA's anisotropy operator. Then, in the exploration phase of the SMA, the individual historical optimum is used instead of the population historical optimum for updating, and during the SMA's exploitation stage, the population's historical best solution serves as a means to ensure the algorithm converges to the whole optimal solution. Finally, a stochastic differential variance operator is augmented with this element to improve the chances of leaping out of the local optimum and steer clear of premature convergence.

BO represents a fresh approach to optimisation, influenced through the physical process of preserving mass balance in a control volume. The mass balance equations depict the physical occurrences of mass entering, leaving, and being generated within a defined volume. The updated formulation of BO is as follows.

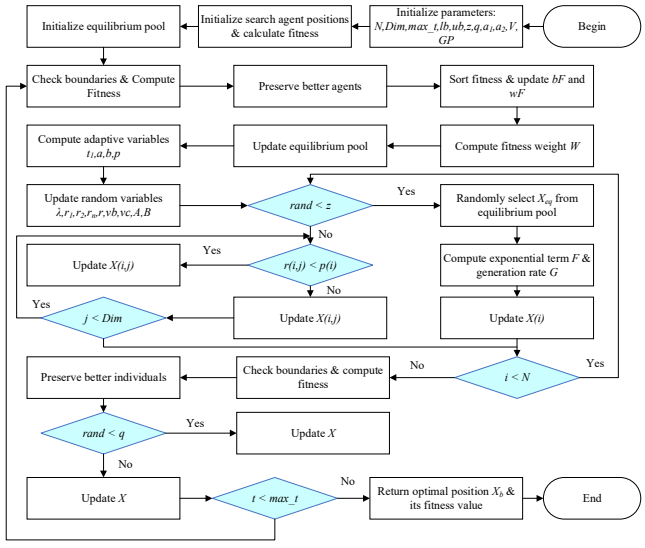
$$C = C_{eq} + \left(C - C_{eq}\right)F + \frac{G}{\lambda V}(1-F) \tag{6}$$

where C represents the present solution, C_{eq} is chosen at random from the balanced pool of solutions, F serves as a value to equilibrate partial and entire seek efforts, G serves as a measure of quality generation, thereby amplifying the algorithm's local search effectiveness, λ is a vector of arbitrary numbers between 0 and 1, and $V = 1$ represents the standard volume unit. F is adjusted dynamically based on the iteration count, as outlined below.

$$F = a_1 \cdot sign(r - 0.5) \cdot \left(e^{-\lambda t} - 1\right) \tag{7}$$

where a_1 is the fixed weight value assigned to the global search component. As a_1 increases, the exploration capability strengthens while the utilisation ability diminishes, and vice versa. *sign*() is the sign operation. r and λ are arrays filled with stochastic numbers ranging between 0 and 1, t is the search time. The mass generation rate G is shown below.

Figure 3 The flow of BOSMA (see online version for colours)



$$G = \begin{cases} 0.5 r_1 \left(C_{eq} - \lambda C\right) F, & r_2 \geq GP \\ 0, & r_2 < GP \end{cases} \tag{8}$$

where r_1 and r_2 are random numbers in the range [0, 1] and GP stands for a generation probability value of 0.5.

To enhance seek performance, the easy arbitrary seek operator in SMA can be replaced by the BO operator, and the formula for updating positions in BOSMA is as follows.

$$X(t+1) = \begin{cases} X_{eq}(t) + \left(X(t) - X_{eq}(t)\right) \cdot F + G \cdot (1-F)/\lambda V & rand < z \\ X(t) + vb \cdot \left(W \cdot X_A(t) - X_B(t)\right) & r < p \\ X_b(t) + vc \cdot \left(W \cdot X_A(t) - X_B(t)\right) & r \geq p \end{cases} \tag{9}$$

where X_{eq} is a randomly selected solution from the equilibrium pool, X is the location of the search instance, $X_b(t)$ emerges as the peak solution in the equilibrium landscape, and z is an empirical value, which usually takes the value of 0.6. Equation (9) demonstrates the stochastic differential variance operator, which performs an anisotropic operation on the dimensions of 4% of the search individuals in the population. As iterations accumulate, the exploration becomes less extensive. To improve BOSMA's ability to explore and prevent it from getting stuck in local optima, a variation operation is performed on 80% of the searched individuals. This is designed to diversify the algorithm's problem-solving techniques in the exploration phase to better search the solution space of the issue.

To improve the possibility that BOSMA will leap out of the local optimum's confines, the stochastic differential variational operator is added after the update of equation (9). The variational operator can be mathematically modelled as described below.

$$X(t+1)=\begin{cases}X(t)+BF\cdot(rand\cdot(UC-LB)+LB)\cdot L & rand<q\\ X(t)+R\cdot\left(X_A(t)-X_B(t)\right) & rand\geq q\end{cases} \tag{10}$$

where CF serves as an adjustable shrinking coefficient, R varies randomly within the range of 0.2 to 1, L stands for a matrix composed of instances that are either 0 or 1, and the adjustable value of q is set to 0.2. X_A and X_B represent two solutions randomly sampled from the population. CF and L are calculated as in equation (11) and equation (12), respectively.

$$CF=\left(1-\frac{t}{\max\limits_{t}}\right)^{\frac{a_1\cdot t}{\max\limits_{t}}} \tag{11}$$

$$L_{i,j}=\begin{cases}1 & rand<q\\ 0 & rand\geq q\end{cases} \tag{12}$$

where N denotes the number of individuals in the population and Dim represents the dimensionality of the issue. For the goal of preventing inefficient searching, the limits of the search agent's exploration are validated using equation (13) after each positional change.

$$X_{i,j}(t+1)=\begin{cases}\left(X_{i,j}(t)+UB\right)/2 & X_{i,j}(t+1)>UB\\ \left(X_{i,j}(t)+LB\right)/2 & X_{i,j}(t+1)<LB\\ X_{i,j}(t+1) & others\end{cases} \tag{13}$$

In this paper, the computational complexity of BOSMA algorithm is compared with that of ACO, SSA, MA and SMA algorithms, as shown in Table 1, where N is the number of populations, n is the problem dimension, and D is the problem dimension, and it can be seen through Table 1 that BOSMA algorithm has lower computational complexity in terms of time complexity and space complexity than the other four algorithms.

Table 1 Computational complexity of different heuristic optimisation algorithms

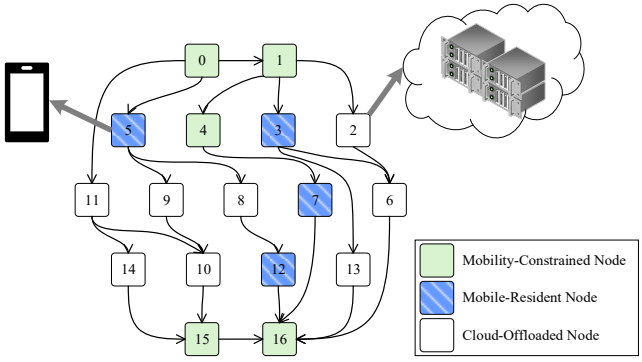
Algorithm	*Time complexity*	*Space complexity*
ACO	$O(N{\cdot}D^2)$	$O(D^2)$
SSA	$O(D{\cdot}N + N\log N)$	$O(D\,N)$
MA	$O(D\,N+N^2)$	$O(D\,N)$
SMA	$O(D\,N)$	$O(D\,N)$
BOSMA	$O(N)$	$O(N)$

4 Blockchain technology application and optimisation algorithm in enterprise supply chain management

4.1 Ecommerce cloud data migration task dependency graph design

For the goal of reducing the migration energy consumption, this paper first designs the e-commerce cloud computing data migration task dependency graph, and construct a mathematical model with the objective function of minimising the migration energy consumption, and migrate all the tasks that can be migrated to the cloud as the initial solution, and then calculate the energy savings of the migratable tasks running in the mobile terminal one by one, and migrate the tasks with the largest savings to the mobile terminal in order. For each migrated task, the algorithm updates the energy savings of each task based on the communication time between the tasks. Finally, the BOSMA algorithm is used to solve the objective function and migrate the task that reduces energy consumption the most.

Figure 4 Ecommerce cloud data migration task map (see online version for colours)



E-commerce cloud computing data migration involves both mobile and cloud, the dependency between the tasks that will be performed at both ends can be represented by a directed acyclic graph as shown in Figure 4. Each node in Figure 4 represents a task, and if there is a link from i to j between nodes i and j, it means that there is a dependency relationship between tasks i, j. Task i is said to be the father node of task j.

As shown in Figure 4, the green node indicates that the task can only be executed on the mobile. Except for the

green node, all other nodes can be migrated, either to the cloud or to the mobile. A white node indicates that the task was migrated to the cloud; a blue node indicates that the task was completed on mobile. For two tasks with dependencies, if both tasks are realised in the cloud, the communication time and energy consumption between the two tasks are negligible; if two tasks with communication dependencies are realised on different ends, the communication time and energy consumption between the tasks need to be considered.

4.2 Mathematical model construction for data migration in e-commerce cloud computing

For a task graph with N tasks, which tasks are completed in the mobile and which tasks are completed in the cloud, and by scheduling the execution order of the tasks at both ends, the energy consumption of the mobile can be reduced under the constraint that the completion time T of the task graph is satisfied. The energy consumption on the mobile side consists of the energy used to perform both mobile tasks and transmission tasks. Assuming that $l = 1$ denotes that the task is executed on the mobile, $l = 0$ denotes that the character is executed on the cloud, and $x_j^l(t)$ denotes whether task j is executed on the mobile at time t, the following equations are in place, where m_j denotes whether task j is realised on the mobile, and c_j denotes whether task j is migrated to the cloud for realisation.

$$m_j = \sum_{t=1}^{T} x_j^0(t) \tag{14}$$

$$c_j = \sum_{t=1}^{T} x_j^1(t) \tag{15}$$

The communication time required for cloud task i to transmit data to mobile task j is denoted as τ_{ij}^{cm}. Similarly, the communication time required for task i on mobile to upload data to task j in the cloud is denoted as τ_{ij}^{cm}. In particular, when $i = j$ or when both i and j are in the cloud or mobile, the communication time and energy consumption between them is 0. The total communication energy consumption caused by the task migration is as follows.

$$E_{com} = P_T \sum_{i=1}^{N}\sum_{j=1}^{N} \tau_{ij}^{mc} + P_R \sum_{i=1}^{N}\sum_{j=1}^{N} \tau_{ij}^{cm} \tag{16}$$

The objective function for minimising mobile energy consumption based on the decision variables $x_j^l(t)$, $l \in \{0, 1\}$, $i, j, = 1, 2, \ldots N$, and $t = 1, 2, \ldots, T$ can be defined as follows.

$$\min\left\{\sum_{j=1}^{N} P \cdot m_j k_j^m + E_{com}\right\} \tag{17}$$

The objective function established above must satisfy the following constraints.

1 The total time required to complete all tasks does not exceed T, i.e., $0 < \sum_{t=1}^{T} t \cdot x_N^0(t) \leqslant T$.

2 Each task node j can be executed only on the cloud or on the mobile, i.e., $m_j + c_j = 1$.

3 All nodes of the task graph must satisfy the predecessor-successor dependency, such that $\theta_{jk} = w_k + \tau_{jk}^{\mathrm{mc}} + \tau_{jk}^{\mathrm{cm}}$. Then we have equation (18), where $j < k$, $t = w_j, \ldots, T-\theta_{jk}$.

$$\sum_{l=0}^{1}\sum_{s=1}^{t+\theta_{jk}} x_k^l(s) \leq \sum_{l=0}^{1}\sum_{s=1}^{t+\theta} x_j^l(s) \tag{18}$$

4 At any moment t, multiple tasks are executed serially on the mobile side, such that $L = min\{t + w_j - 1, T\}$, there is $\sum_{j=1}^{N}\sum_{s=1}^{L} x_j^0(s) \leqslant 1$.

5 For any task k, before k is executed, it has to wait for the completion of the previous task before it can be executed, and the completion time of k is the time of the completion of the previous task plus the time needed for its own completion, as follows.

$$\sum_{l=0}^{1}\sum_{\iota=1}^{T} t \cdot x_j^l(t) + \tau_{jk}^{cm} + \tau_{jk}^{mc} + w_k \leq \sum_{l=0}^{1}\sum_{\iota=1}^{T} t \cdot x_k^l(t) \tag{19}$$

6 Constraint on the number of tasks that can only be executed on mobile: the number of tasks that can only be executed on mobile is n, and the serial numbers of such tasks are $b_1, b_2, \ldots, b_n$, then there are $m_{b_i} = 1$, where $i = 1, 2, \ldots, n$.

4.3 Mathematical model solving based on improved slime mould algorithm

With the above constraints satisfied, the BOSMA algorithm is used to solve the objective function and migrate to the task that reduces energy consumption the most. The migration is repeated until the time constraints cannot be met. When a task node is migrated, the communication time and energy consumption associated with the task node change. The total energy consumption on the mobile includes the energy used to perform tasks on the mobile and the total energy used for communication. When a task node is migrated from the mobile to the cloud, the energy used to perform the task on the mobile decreases; conversely, when a task is migrated from the cloud to the mobile, the energy used to perform the task on the mobile increases.

1 The tent mapping (Nagaraj, 2022) was used to initialise the slime mould population as a representative of the chaotic mapping, and the resulting chaotic sequences were uniformly distributed in [0, 1], so that the slime mould population was uniformly distributed in the

search space. The mathematical model of the tent mapping is as follows.

$$X_{k+1} = \begin{cases} \dfrac{X_k}{h}, 0 \leqslant X_k < h \\ \dfrac{(1-X_k)}{1-h}, h \leqslant X_k \leqslant 1 \end{cases} \tag{20}$$

where k is the number of mapping, $k = 1, 2, \ldots, n$ is the value of the k^{th} mapping function; h is the chaos parameter, which is a random value between 0 and 1. When $h = 1/2$, tent has the best performance, and the resulting chaotic sequence is uniformly distributed in the search space.

2 Using equation (9) to update the position of the population, the best individual $X_b(t)$ and two random individuals $X_A(t)$ and $X_B(t)$ are used to guide the search direction of the population individuals during the global search.

3 According to the search direction, for the tasks marked as "undivided", the task node that meets the following conditions is migrated to the mobile terminal. After this node is migrated, the optimal solution calculated by BOSMA satisfies the time constraint. Under the condition that the constraint is satisfied, the task node that saves the most energy consumption on the mobile terminal after migration is found.

4 At the end of the iteration, the individuals of the population will gradually approach the current optimal solution X_b, which may lead to the stagnation of the population search. Therefore, BOSMA introduces a random traceless σ-point variant to promote diversity within the population as iterations progress, so as to increase the local optimisation at the stagnation stage of the search, as shown below.

$$X'_{i+d} = X_b(t) - r'\left(\sqrt{(d+k)SD_X}\right)_i \tag{21}$$

$$SD_X = \frac{1}{2d+1}\sum_{i=1}^{2d+1}\left(X_i - X_{avg}\right)\left(X_i - X_{avg}\right)^T \tag{22}$$

where SD_x is the population covariance matrix, r is the scale factor, d is the dimension, and X_{avg} is the mean. The algorithm will select the best individual from the final $2d + 1$ individuals generated as follows.

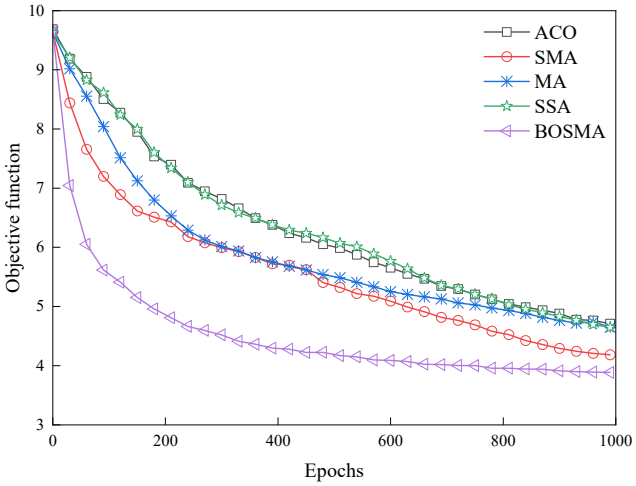
$$X_b(t+1) = \arg\min_{0 \leqslant i \leqslant 2d}\left\{f\left(X'_i\right)\right\} \tag{23}$$

5 Migrate the best individuals first and mark the tasks that have been migrated from the cloud to the mobile as "migrated". If the selected task nodes increase the total energy consumption of the migrated mobile, the algorithm process ends.

5 Experimental results and analyses

The experiments were done on Cloudsim's simulation platform, which simulates a cloud computing environment consisting of multiple e-commerce cloud computing data centres and mobile. In the paper, 600 data centres and 50 mobile sites are created, which are connected to each other through a high-speed network with different bandwidths. The simulation environment is Win11 64-bit operating system, Gen Intel(R) Core (TM) i5-12450H CPU with 2.00 GHz, 16.0 GB of RAM, and MATLAB R2022b simulation software. The maximum number of iterations was set to 100, the dimension size to 30, and the population size to 30. The data used in the experiments are averages of 100 experiments with the same settings.

Figure 5 Comparison of convergence of different algorithms for solving objective function (see online version for colours)

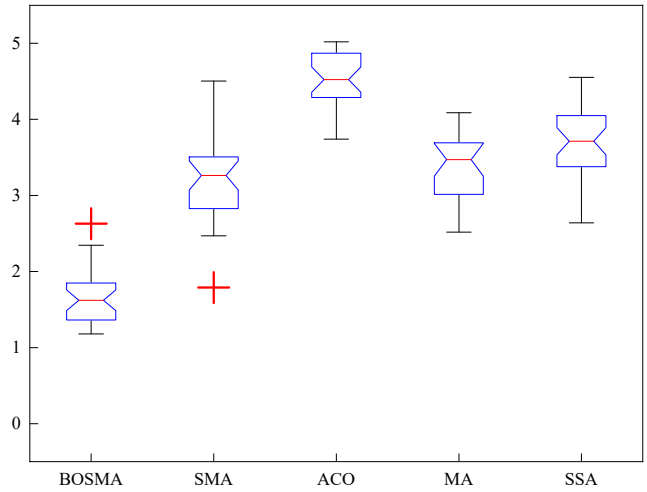


The convergence comparison of the suggested BOSMA algorithm with the ACO, SSA, MA, and SMA algorithms for solving the objective function is shown in Figure 5. From Figure 5, it can be clearly seen that both SAM and BOSAM accelerate the convergence speed of the original optimal solution to a certain extent and improve the solution accuracy, among which the optimisation performance of SMA based on the BO and the stochastic difference variational operator contribute the most to the optimisation performance of the SMA, and the use of the stochastic difference variational operator to initialise the population mainly improves the uniformity of the distribution of the individuals in the SMA, so that the probability of the optimal solution to be found is increased. The convergence curve of BOSMA is at the bottom, which indicates that the BOSMA algorithm has higher convergence speed and accuracy than the other four algorithms, and also shows that the introduction of multiple optimisation strategies is effective in improving the optimisation performance of SMA.

To investigate the distribution pattern of optimisation results obtained by BOSMA and the competing approaches, Figure 6 shows the best fitness gained through performing 100 independent runs for each of the five algorithms. The

outcome indicates that BOSMA is with smaller medians, upper and lower quartiles, fewer outliers, and a narrower distributional frame than most algorithms. In addition, the gaps in the box plots of BOSMA do not exhibit overlapping features with the BO and SMA, thus the BOSMA's median can be assessed to be smaller with a 95% confidence level, which surpasses other algorithms by a substantial amount.

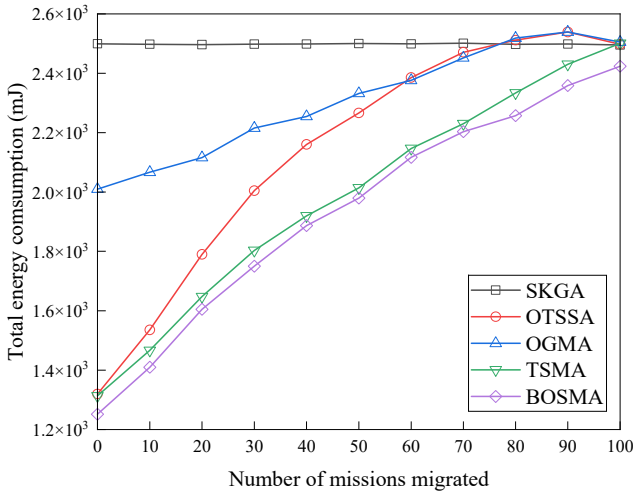
Figure 6 Optimal fitness of different algorithms (see online version for colours)



In addition, the mean (Mean), standard deviation (STD) and minimum (Min) metrics are used in this paper to evaluate the performance of the algorithms and to compare the Friedman average rank (FAR) of the algorithms on different functions. Then Wilcoxon rank sum test is used to assess whether there is a significant difference between BOSMA and the compared algorithms and the results obtained for the five algorithms on the CEC2019 function are shown in Table 1. The dimensions, search ranges and theoretical optimal values of the five functions TF1, TF2, TF3, TF4 and TF5 in CEC2019 are shown in the literature []. It can be seen that BOSMA obtains better average and minimum values on TF3, and only worse than SMA on TF1, which indicates that BOSMA has stronger mining ability than SMA. The mean and STD of BOSMA for the five functions are larger than those of other algorithms, indicating that BOSMA can find the optimal value of the function and quickly obtain a solution with higher accuracy.

The energy consumption of BOSMA algorithm applied to e-commerce cloud computing data migration with four migration methods SKGA (Chawla et al., 2020), OGMA (Kak et al., 2024), OTSSA (Durairaj and Sridhar, 2024), and TSMA (Kumar and Rajesh, 2023) is are compared as shown in Figure 7. As the amount of migrated tasks rises, so does the total energy consumption of the mobile, because fewer tasks can be migrated and more tasks can be performed by the mobile. Nevertheless, the BOSMA algorithm is able to give lower energy consumption than the other four algorithms, indicating that the BOSMA algorithm is still feasible and efficient as the number of tasks changes.

Figure 7 Comparison of migration energy consumption of different methods



Comparison of migration time of each method for different number of tasks is shown in Table 3, when the number of migrated tasks is 60, the migration time of BOSMA is 130.27 ms, which is reduced by 41.15%, 33.06%, 23.5%, and 18.71% compared to SKGA, OGMA, OTSSA, and TSMA, respectively. SKGA implements cloud computing data migration through GA, but the search speed is relatively slow because GA is not able to utilise the feedback information from the network in time. More training time is needed to get more accurate solutions. OGMA is the introduction of SSA for cloud migration, but SSA falls into a local optimum at the initial stage, which affects the global search capability. OTSSA uses MA to optimise cloud migration strategies, but the method has limited exploration capability in the search space, resulting in the inability to jump out of the current local optimal region. TSMA implements migration policy optimisation based on traditional SMA without improving SMA, which leads to long search time for optimal migration direction. BOSMA not only accelerates the convergence speed, but also increases the probability of jumping out of the local optimum by adding the stochastic differential variance operator, avoiding premature convergence and improving the migration efficiency.

To further verify the influence of each component in the BOSMA model on the model migration efficiency, this paper conducts ablation experiments on each component. The BOSMA algorithm is replaced with SMA algorithm denoted as – SMA, without using any optimisation algorithm denoted as – OR, and the complete migration method using the improved SAM algorithm is BOSMA. The results of the ablation experiments for each module are shown in Table 4. The migration efficiency of cloud computing data migration optimisation using SMA algorithm reaches 88.9%, the migration efficiency without applying any optimisation algorithm is only 81.2%, and BOSMA which incorporates all the components achieves the best migration efficiency.

Table 2 Comparison of migration times for various methods (ms)

Algorithm	*Norm*	*TF1*	*TF2*	*TF3*	*TF4*	*TF5*	*FAR*	*Rank*
BOSMA	Mean	1.3682	3.6864	1.0408	1.6386	9.0106	1.94	1
	STD	0.1248	1.0170	0.0274	0.3698	9.9786	2.75	3
	Min	1.0000	1.9950	1.0000	1.1522	1.0000	1.97	1
ACO	Mean	4.0884	11.4327	1.2002	3.1699	21.0416	6.25	11
	STD	2.1187	3.8315	0.0950	0.5418	0.0409	5.81	9
	Min	1.4092	5.9751	1.0922	1.7539	20.9843	6.78	14
SSA	Mean	1.3682	9.3230	9.3230	2.7763	17.7164	4.16	6
	STD	0.1248	3.7318	0.0208	0.6744	7.6038	5.44	6
	Min	1.0000	3.9849	1.0074	1.3442	1.0000	3.00	2
MA	Mean	1.1227	6.8371	1.0357	2.6977	14.9981	2.06	2
	STD	0.1907	1.7081	0.0203	0.3208	9.3206	2.69	2
	Min	1.0000	1.9950	1.0000	2.0859	1.0000	3.41	4
SMA	Mean	1.3281	4.2502	1.0080	2.1428	21.1104	2.31	3
	STD	0.1669	1.2513	0.0107	0.4034	0.0327	1.75	1
	Min	1.0000	1.9950	1.0000	1.5203	21.0302	3.28	3

Table 3 Comparison of migration times for various methods (ms)

Number of missions migrated	*SKGA*	*OTSSA*	*OGMA*	*TSMA*	*BOSMA*
20	110.69	81.32	66.34	50.11	30.12
40	180.93	148.16	124.81	110.58	80.34
60	221.35	194.62	170.29	160.25	130.27
80	362.94	335.81	305.98	281.58	240.58
100	470.81	432.69	410.35	389.63	326.91

Table 4 Results of ablation experiments

Method	*–SMA*	*–OR*	*BOSMA*
Migration efficiency	88.9%	81.2%	95.6%

6 Conclusions

This paper suggests an e-commerce cloud computing data migration method based on improved SMA, which aims to reduce migration energy consumption and improve data migration efficiency. Firstly, to address the problem of poor global optimisation of SMA, a balanced optimisation operator is incorporated into SMA, which helps the approach to strike a good balance between exploration and exploitation. Then, to improve the probability of the algorithm jumping out of the local optimal, the probabilistic divergence mutation operator is added in the iterative process of the approach, which boosting the algorithm's capacity for exploration, increases the diversity of the population, and avoids premature convergence of the algorithm. To reduce the migration energy consumption, the dependency diagram of data migration task for e-commerce cloud computing is designed, and a mathematical model is constructed to minimise the objective function of migration energy consumption. The BOSMA algorithm is used to solve the objective function and realise the migration of the tasks that reduce the energy consumption the most. As an initial solution, all tasks that can be migrated are migrated to the cloud, and then the energy savings of the migratable tasks running on the mobile are calculated one by one, and the tasks with the highest savings are migrated to the mobile in order. For each migrated task, the algorithm updates the energy savings of each task in time based on the communication time between tasks. The experimental outcome indicates that the BOSMA algorithm not only has higher convergence speed and convergence accuracy, but also can quickly search for the solution that minimises the energy consumption of migration.

In future work, this aspect will continue to be studied in depth by considering the impact of environmental fluctuations (i.e., network latency, bandwidth fluctuations, and server failures) on BOSMA application migration decisions. This paper will consider the optimisation objectives in a comprehensive manner in the later work by integrating multiple optimisation objectives (e.g., cost, energy consumption, and load balancing) into a single multi-objective optimisation problem study. Moreover, in this paper, simulation experiments will be conducted on multiple datasets and real scenarios to validate the scalability of the proposed approach.

Acknowledgements

This work is supported by the Science and Technology Research Project of Jiangxi Provincial Department of Education in 2024 (No. GJJ2402804).

Declarations

All authors declare that they have no conflicts of interest.

References

Almarabeh, T. and Majdalawi, Y.K. (2019) 'Cloud computing of E-commerce', *Modern Applied Science*, Vol. 13, No. 1, pp.27–35.

Andrikopoulos, V., Binz, T., Leymann, F. and Strauch, S. (2013) 'How to adapt applications for the cloud environment: challenges and solutions in migrating applications to the cloud', *Computing*, Vol. 95, pp.493–535.

Chawla, N., Kumar, D. and Sharma, D.K. (2020) 'Improving cost for data migration in cloud computing using genetic algorithm', *International Journal of Software Innovation* (*IJSI*), Vol. 8, No. 3, pp.69–81.

Diaby, T. and Rad, B.B. (2017) 'Cloud computing: a review of the concepts and deployment models', *International Journal of Information Technology and Computer Science*, Vol. 9, No. 6, pp.50–58.

Durairaj, S. and Sridhar, R. (2024) 'MOM-VMP: multi-objective mayfly optimization algorithm for VM placement supported by principal component analysis (PCA) in cloud data center', *Cluster Computing*, Vol. 27, No. 2, pp.1733–1751.

Fahmideh, M., Daneshgar, F., Rabhi, F. and Beydoun, G. (2019) 'A generic cloud migration process model', *European Journal of Information Systems*, Vol. 28, No. 3, pp.233–255.

Gholami, M.F., Daneshgar, F., Low, G. and Beydoun, G. (2016) 'Cloud migration process – a survey, evaluation framework, and open challenges', *Journal of Systems and Software*, Vol. 120, pp.31–69.

Guo, L., Zhang, Y. and Zhao, S. (2017) 'Heuristic algorithms for energy and performance dynamic optimization in cloud computing', *Computing and Informatics*, Vol. 36, No. 6, pp.1335–1360.

Huang, L. and Abnoosian, K. (2020) 'A new approach for service migration in cloud-based e-commerce using an optimization algorithm', *International Journal of Communication Systems*, Vol. 33, No. 14, p.e4457.

Jawed, M.S. and Sajid, M. (2022) 'A comprehensive survey on cloud computing: architecture, tools, technologies, and open issues', *International Journal of Cloud Applications and Computing* (*IJCAC*), Vol. 12, No. 1, pp.1–33.

Jia, H. and Peng, X. (2021) 'High equilibrium optimizer for global optimization', *Journal of Intelligent and Fuzzy Systems*, Vol. 40, No. 3, pp.5583–5594.

Kak, S.M., Agarwal, P., Alam, M.A. and Siddiqui, F. (2024) 'A hybridized approach for minimizing energy in cloud computing', *Cluster Computing*, Vol. 27, No. 1, pp.53–70.

Kharazmi, E. and Zayernouri, M. (2019) 'Operator-based uncertainty quantification of stochastic fractional partial differential equations', *Journal of Verification, Validation and Uncertainty Quantification*, Vol. 4, No. 4, p.41006.

Kumar, K.V. and Rajesh, A. (2023) 'Multi-objective load balancing in cloud computing: a meta-heuristic approach', *Cybernetics and Systems*, Vol. 54, No. 8, pp.1466–1493.

Li, C., Zhang, J., Ma, T., Tang, H., Zhang, L. and Luo, Y. (2019) 'Data locality optimization based on data migration and hotspots prediction in geo-distributed cloud environment', *Knowledge-Based Systems*, Vol. 165, pp.321–334.

Nagaraj, N. (2022) 'The unreasonable effectiveness of the chaotic tent map in engineering applications', *Chaos Theory and Applications*, Vol. 4, No. 4, pp.197–204.

Shakya, S. (2019) 'An efficient security framework for data migration in a cloud computing environment', *Journal of Artificial Intelligence*, Vol. 1, No. 1, pp.45–53.

Shekhar, C.A. and Sharvani, G. (2021) 'MTLBP: a novel framework to assess multi-tenant load balance in cloud computing for cost-effective resource allocation', *Wireless Personal Communications*, Vol. 120, No. 2, pp.1873–1893.

Singh, N. and Dhir, V. (2019) 'Hypercube based genetic algorithm for efficient vm migration for energy reduction in cloud computing', *Statistics, Optimization and Information Computing*, Vol. 7, No. 2, pp.468–485.

Son, P.V.H. and Khoi, L.N.Q. (2024) 'Application of slime mold algorithm to optimize time, cost and quality in construction projects', *International Journal of Construction Management*, Vol. 24, No. 13, pp.1375–1386.

Song, Z., Sun, Y., Wan, J., Huang, L. and Zhu, J. (2019) 'Smart e-commerce systems: current status and research challenges', *Electronic Markets*, Vol. 29, pp.221–238.

Teli, P., Thomas, M.V. and Chandrasekaran, K. (2016) 'Big data migration between data centers in online cloud environment', *Procedia Technology*, Vol. 24, pp.1558–1565.

Zhao, H., Feng, N., Li, J., Zhang, G., Wang, J., Wang, Q. and Wan, B. (2023) 'VM performance-aware virtual machine migration method based on ant colony optimization in cloud environment', *Journal of Parallel and Distributed Computing*, Vol. 176, pp.17–27.