Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Supplementary materials for

Xuebin LAI, Yan GUO, Ming HE, Hao YUAN, Wei LI, Xiaonan CUI, 2025. A UAV-enabled mobile edge computing paradigm for dependent tasks based on a computing power pool. *Front Inform Technol Electron Eng*, 26(4):623-638.
https://doi.org/10.1631/FITEE.2400465

1 Notations

Table S1 Summary of key notations

Notation	Definition	Notation	Definition
N	Number of UAVs	$t_{n,m}^{\text{off}}$	Offloading time the UAV n assigns to user m
M	Number of IoT device users	snr_n^w	SNR between UAV n and UAV w
($\boldsymbol{w}_m$, d_m, $\boldsymbol{D}_m$, t_m)	Task information of user m, where $\boldsymbol{w}_m$ is user m's coordinates, d_m is the amount of data, $\boldsymbol{D}_m$ is the DAG of dependent tasks, and t_m is the expected time to receive service	$\boldsymbol{D}_m$=(T_m, $\boldsymbol{E}_m$, $\boldsymbol{C}_m$)	DAG of user m's dependent tasks, where T_m is the set of subtasks, E_m is the dependency between subtasks and C_m represents the computing time of the subtask on UAVs
ΔT	The flight cycle of the UAVs	$t_{\text{u,comm}}$	Data transfer time between UAVs
K	The number of time slots	$T_{m,i}$	Subtask i of task m
T	Unit time slot length	$T_{\text{pre}(m,i)}$	Predecessor subtask of $T_{m,i}$
Q	Set of UAV trajectories	X	UAV–subtask association, which is the set of variables $\chi_{m,i,n}$
$\boldsymbol{q}_n[k]$	Horizontal coordinates of UAV n in k^{th} time slot	$\chi_{m,i,n}$	Indicate whether subtask $T_{m,i}$ is assigned to UAV n
$h_m^n[k]$	Channel gain between UAV n and user m	$E_n^{\text{f}}[k]$	Flight energy consumption of UAV n in k^{th} time slot
B	Bandwidth resource allocation	E_n^{h}	Total hover energy consumption of UAV n
$b_{n,m}[k]$	Bandwidth ratio of user m allocated by UAV n	E_n^{move}	Moving energy consumption of the UAV n
$\text{snr}_m^n[k]$	SNR between the user m and the UAV n in k^{th} time slot	F	Computational allocation, made up of the set of $f_n[k]$, which is the computing frequency in k^{th} time slot
$R_m^n[k]$	Data transmission rate	E_n^{c}	Computing energy of UAV n
A	UAV–user association, the set of $\alpha_{m,n}[k]$	SSC	Time slot that all the dispatched UAVs have completed the task collection
$\alpha_{m,n}[k]$	Whether user m's task is captured by UAV n in the k^{th} time slot	CFT_n	Time slot that UAV n completes task collection
H	Flight altitude of UAVs	SFC_n	Time slot that UAV n has completed all the assigned subtasks' computing
P_m	User m's transmitting power	γ	Energy consumption coefficient
P_{u}	The power required for the UAV to transmit data	$d_{\min}$	Collision avoidance distance between UAVs

UAV: unmanned aerial vehicle; DAG: directed acyclic graph; SNR: signal-to-noise ratio

2 Algorithms

This section shows the algorithms employed in this paper. Algorithm S1 is used to solve problem P1, Algorithm S2is used to solve problem P2, and Algorithm S3 is used to solve problem P.

Algorithm S1 Iterative algorithm for clustering and convex optimization

1: **Input**: the initial value of the UAV–user association A_n obtained by *K*-Means++, user information, UAV starting point, hovering point, computing energy consumption, and computing time.
2: Let the number of iterations j=1;
3: **while** $j \lesssim j_m$ **do**
4: **for** n=1: N
5: Initialization: B_n^1, $\boldsymbol{q}_n^1$, r_n=1, E_n^1=0
6: **while** (Obj$_1$(r−1) –Obj$_1$(r))$\geqslant \varepsilon$ and $E_n^r < E_n^0$ **do**
7: SCA: Obtaining the $B_m^{n,r}[k]$ by inequalities (5), (6), (23), and (25) and the trajectory $\boldsymbol{q}_n^{r+1}[k]$ by inequality (25);
8: Update: $B_m^{n,r+1}[k]$, $\boldsymbol{q}_n^{r+1}[k]$;
9: Update: r=r+1;
10: **end while**
11: n=n+1;
12: **end for**
13: **If** $|\mathrm{CFT}_n - \mathrm{CFT}_w| \leqslant \theta$
14: Update: Obj(j+1)=Obj(j);
15: **else**
16: Update: A_n^{r+1};
17: **end if**
18: j=j+1;
19: **end while**
20: **Output**: UAV–user association A, communication resource allocation B and UAV trajectories Q.

UAV: unmanned aerial vehicle; SCA: successive convex approximation

First, we adopt the *K*-Means++ algorithm to obtain the initial value of the unmanned aerial vehicle (UAV) –user association. Then successive convex approximation (SCA) technology is used to convert the non-convex constraints into convex constraints. The convex optimization toolbox (CVX) solver is used to iteratively solve the UAV trajectory and communication resource allocation, and the solution set of UAV trajectory, and communication resource allocation is obtained. Next, according to the judgment condition of the completion time of the UAVs collection, the solutions with the expected mutual waiting time are screened out. If there is no solution, the *K*-Means++ algorithm is invoked again to update the UAV–user association and start the next iteration.

Algorithm S2 Improved GA

1: **Input:** DAG, UAV hovering position, parameters of the UAVs, population size, number of populations, maximum number of iterations max_iter;
2: Initialization: i=0, populations Pop(i);
3: Encoding: Subtasks, UAVs associated with the subtasks, and the computational frequency of UAVs are encoded to obtain the initial solution set Pop^s(i);
4: Substituting Pop^s(i) into the objective function, obtaining the fitness of the candidate solution X_{best}, the optimal solution F_{best} and the optimal objective function value $E_{\max}$;
5: **while** i<max_iter
6: Regeneration, crossover, mutation to produce new populations Pop^s(i+1);

7: Calculate the fitness of the Pop^s(i+1), update the optimal solution X_{best}, F_{best}, and the objective function value $E_{\max}$;
8: i=i+1;
9: **end while**
10: **Output**: Optimal value $E_{\max}$ and optimal solution X_{best}, F_{best}.

UAV: unmanned aerial vehicle; DAG: directed acyclic graph

For the coding part, we first index each subtask according to the directed acyclic graph (DAG). Considering the diversity of variables, we adopt a multi-digit encoding method. Therefore, the chromosome is composed mainly of four parts: the subtask index, the UAV to which the subtask is assigned, the time slot at which the task begins to perform, and the computing frequency assigned to that subtask.

Since the initial solution greatly influences the result of the genetic algorithm, we focus on the selection of the initial population. Considering that the UAVs at this stage are in a hovering state, the longer the hovering time, the more the increased cost. Therefore, without considering the energy balance, the solution with the shortest overall time is taken as the initial solution of the genetic algorithm (GA) proposed in this paper, and the corresponding population is taken as the initial population. Due to the dependency of subtasks (Hu ZZ et al., 2021), we adopt the priority to determine the assignment order of each subtask, which is determined mainly by the computing time and communication time, as shown in Eq. (S1).

$$\text{Pri}_{T_{m,i}} = \frac{1}{N}\sum_{n=1}^{N} C_m(i,n) + \max_{T_{m,j}}\{e_m(i,j) + \text{Pri}_{T_{m,j}}\}, \text{when } T_{m,i} = T_{\text{pre}(m,i)}, \forall m,n \tag{S1}$$

Each subtask is ranked in descending order according to the priority calculated by this method, and the subtasks are selected from highest to lowest, and the most appropriate associated UAV is selected based on the principle of minimum idle time. The algorithm complexity here is $O(NM^2)$.

Since we want to find a task allocation method under an energy consumption constraint, and the traditional GA is used to solve the maximum value, we use the inverse of the objective function as the fitness evaluation function. We adopt the classic roulette to pick out the paternal chromosomes from the current population, and the fitter individuals have a better chance of being selected. Then, the selected parents are randomly paired to imitate the process of deoxyribonucleic acid (DNA) replication, and the matched parents are cross-operated to exchange part of the genetic information with a crossover probability of 0.75 to obtain a new population. We also set the genes on each chromosome to change randomly with a mutation probability of 0.05. The genetic characteristics of the new population will be better than those of the previous generation, so the larger the value of the feasible solution function of the new population, the closer we are to the optimal solution. After a round of evolution is completed, we recalculate the fitness of each chromosome in the new population and update the current solution if the fitness of the new population is greater than the current one.

Algorithm S3 Iterative optimization based energy balancing algorithm for mobile edge computing in UAVs

1: **Input:** System parameter.
2: Initiate: A^r, X^r, O^r, Q^r, B^r, F^r, the value of the objective function Obj(r), the maximum number of iterations max_iter, and the number of iterations r=1;
3: When r<max_iter, perform steps 4–7; otherwise, output the results;
4: Given X^r and F^r, obtain A^{r+1}, B^{r+1}, Q^{r+1} by solving problem P1;
5: Given A^{r+1}, B^{r+1}, Q^{r+1}, obtain X^{r+1} and F^{r+1} by solving problem P2;
6: Calculate Obj(r+1) corresponding A^{r+1}, X^{r+1}, Q^{r+1}, B^{r+1}, F^{r+1}, and make r=r+1;
7: if |Obj(r+1) – Obj(r)|$\leqslant \xi$, end the loop and output the result; otherwise, return to step 3;
8: **Output**: Optimal objective function values and corresponding solutions.

UAV: unmanned aerial vehicle

In Algorithm S1, we adopt the K-Means++ algorithm to solve the UAV–user association, the complexity of which is $O(\log N)$. Then we adopt SCA to obtain the UAVs' trajectories and communication resource allocation. The complexity of this algorithm is related to the size of the problem. In the worst case the complexity is

$O(L\sqrt{N(4MK+2K)})$, where L is the number of iterations, so the complexity of Algorithm S1 is $O(L\sqrt{N(4MK+2K)}\log N)$. In Algorithm S2, the computational complexity of the improved GA comes from population initialization, selection, crossover, variation, and fitness evaluation operations. After calculation, the overall complexity is about $O(GNM^2)$, where G is the number of generations. Therefore, the algorithm complexity of Algorithm S3 does not exceed $O(I(L\sqrt{N(4MK+2K)}\log N+GNM^2))$, where I is the number of iterations of Algorithm S3.

3 Simulation setting and results

The imulation parameters are shown in Table S1.

Table S2 Simulation parameter settings

Parameter	Value
Maximum computing frequency of UAV	1 GHz
CPU capacitance coefficient κ	10^{-28}
CPU cycle C_k	1000
Max bandwidth resources B	1 MHz
Channel power gain	−50 dB
Noise power density	−100 dBm
Path loss index α	2
Energy consumption coefficient γ	10^{-4}
Maximum flying speed $V_{\max}$	30 m/s
Flight altitude H	50 m

UAV: Unmanned aerial vehicle; CPU: central processing unit

To make the experiments more comprehensive, we incorporate the variations in the moving and computing energy consumption of the UAV, task collecting and task computing time as the number of users increases. The performance of the different algorithms is shown in Figures S1−S4.

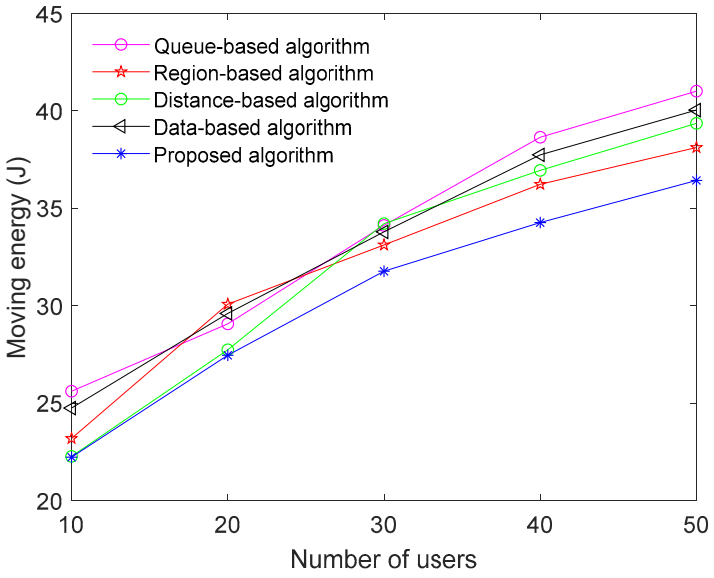

Fig. S1 Moving energy consumption of a UAV as the number of users increases

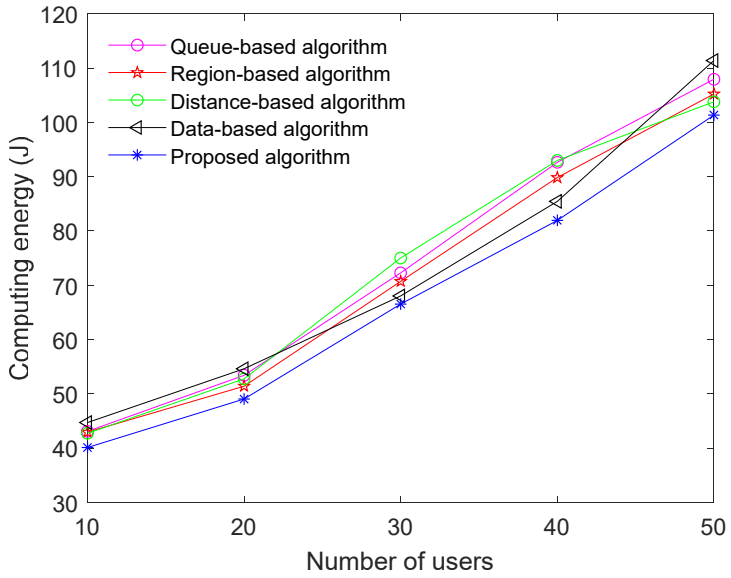

Fig. S2 Computing energy consumption of a UAV as the number of users increases

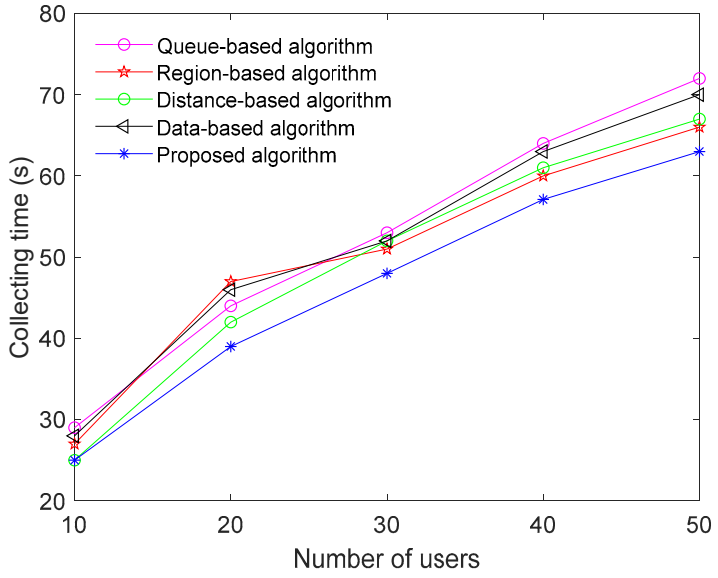

Fig. S3 Collecting time consumption of a UAV as the number of users increases

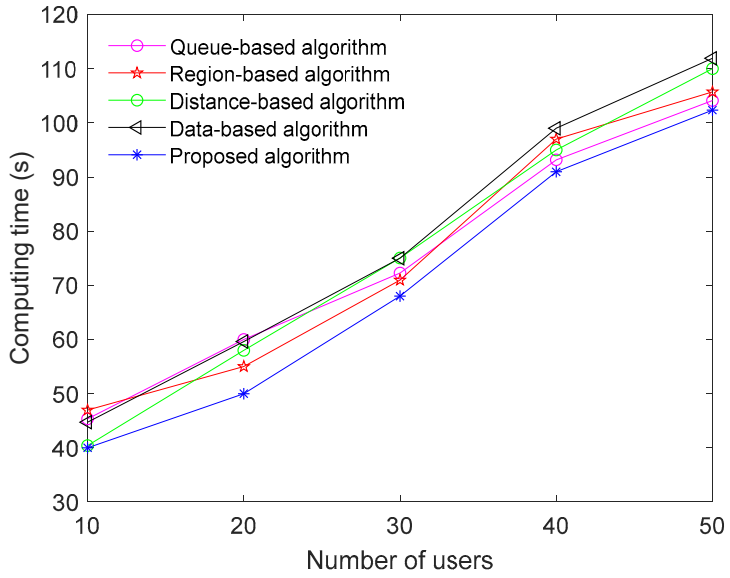

Fig. S4 Computing time consumption of a UAV as the number of users increases