Genetic Algorithms for Storage- and Energy-Aware Caching and Trajectory Optimisation Problem in UAV-Assisted Content Delivery Networks

Thuong C. Lam[1], Nguyen-Son Vo[2(✉)], Thanh-Hieu Nguyen[3], Thanh-Minh Phan[4], and De-Thu Huynh[5]

[1] HUTECH University, Ho Chi Minh City, Vietnam
lc.thuong@hutech.edu.vn
[2] Institute of Fundamental and Applied Sciences, Duy Tan University, Ho Chi Minh City, Vietnam
vonguyenson@duytan.edu.vn
[3] Ho Chi Minh City University of Transport, Ho Chi Minh City, Vietnam
thanhhieu.nguyen@ut.edu.vn
[4] Vietnam Aviation Academy, Ho Chi Minh City, Vietnam
minhpt@vaa.edu.vn
[5] The Saigon International University, Ho Chi Minh City, Vietnam
huynhdethu@siu.edu.vn

Abstract. Trajectory and caching optimisation design is a promising joint solution for enhancing the quality of services in unmanned aerial vehicle (UAV) assisted content delivery networks (CDNs). In this paper, we review the problem of which contents to cache in the UAV and which trajectory to fly, i.e., where to stop and how to gain the shortest path over the stops, under the constraints of caching storage and energy resources, namely storage- and energy-aware caching and trajectory optimisation (SECTO) problem. The SECTO problem in UAV-assisted CDNs is formulated and solved by applying genetic algorithms (GAs) to maximise the content delivery capacity while minimising the flying distance. The simulation results are shown to demonstrate the benefits of GAs in terms of accuracy and time complexity performance compared to other conventional solutions such as exhausted and greedy search algorithms.

Keywords: Content delivery network · Genetic algorithm · travelling salesman problem · UAV caching · UAV trajectory

1 Introduction

One of the most important missions of 6 G networks is to take the non-terrestrial networks (NTNs) into account [1]. The NTNs enable a flying platform of satellites and unmanned aerial vehicles (UAVs) to connect more things in every remote

N.-S. Vo and H.-A. Tran (Eds.): INISCOM 2023, LNICST 531, pp. 34–50, 2023.
https://doi.org/10.1007/978-3-031-47359-3_3

corner on the earth. And thus, the physical, cyber, and biological worlds can further flourish in the era of 6G thanks to the present of flexible access points, relays, contents, helps, and many advanced applications and services (AASs) from the sky [2–6]. In this scenario, the models, techniques, and AASs of UAV caching and trajectory in content delivery networks (CDNs) have drawn a numerous studies from both industrial and academic sectors [7]. These studies demonstrate that UAV caching networks are able to partially alternate the CDNs so as to not only mitigate the workload of the terrestrial stations but also provide the end users (EUs) with better AASs by exploiting shorter and line-of-sight (LoS) transmission.

In UAV caching networks, the UAVs have to select a set of contents to cache in advance. It then flies to deliver the contents to the EUs in the error-prone areas in which, for example, the workload of the terrestrial stations is extremely high, the wireless links are not good due to high and dense obstacles, and the emergency communications are required for rescue operations and safety public activities. To serve the EUs the highest quality of service (QoS) while utilising the resources of UAVs, the optimal set of contents and the optimal trajectory must be found simultaneously. This so-called resource-aware caching and trajectory optimisation problem in UAV-assisted CDNs has been studied in the literature by different approaches from objective functions, constraints, to algorithms and solutions [8–12].

Particularly, the work in [8] proposed a storage- and energy-aware caching and trajectory optimisation (SECTO) problem to maximise the content delivery capacity while minimising the flight distance under the caching storage and energy consumption constraints. The SECTO is solved by using exhausted and greedy search algorithms. In [9], the authors tried to minimise only the delay under the constraint of caching storage by solving a tractable semi-definite programming problem reformulated from an original intractable distributionally robust stochastic optimization problem. Besides caching in the UAVs, the authors in [10] considered caching in the EUs to establish a cache-enabling UAV-device-to-device (UAV-D2D) network. A joint caching placement in UAV-D2D network and UAV trajectory optimization (CTO) problem under the constraint of caching storage is formulated for maximising the cache utility. The CTO problem, which is a mixed integer nonlinear programming problem, is decomposed into three sub-problems for being solved by a low complexity iterative algorithm. Interestingly, reinforcement learning approach is applied to minimise the sum content acquisition delay of EUs by jointly optimising the multiple EUs' association, cache placement, UAV trajectory, and transmission power [11,12].

We can see that the aforementioned optimisation solutions are for whether multiple objective functions of very high time complexity but exact results [8] or single objective function of lower time complexity with approximate results by reformulating and decomposing the original problems into solvable sub-problems [9–12]. However, these solutions are not flexible in terms of balancing the trade-off between accuracy and time complexity in accordance with the

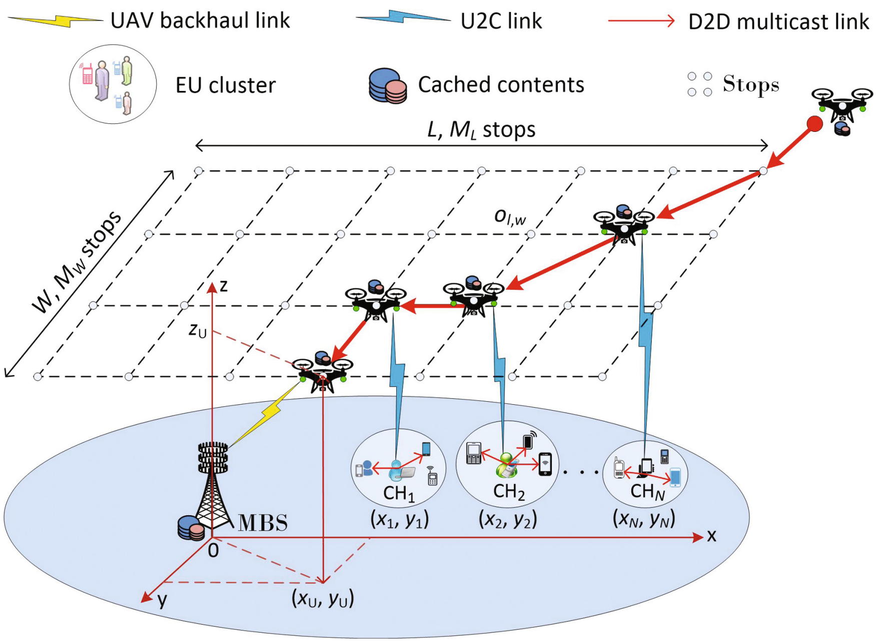

Fig. 1. UAV-assisted content delivery networks [8].

diverse requirements of AASs, especially in solving the complicated SECTO problem with multiple objective functions studied in [8].

In this paper, the typical SECTO problem is studied and then efficiently solved by applying genetic algorithms (GAs). The benefit of GAs is that they can capture the evolutionary principles of natural selection and genetic variation to find the exactly or approximately global optimal results without considering the fact that if the search space is unimodal or multimodal. To make the GAs more feasible to solve the SECTO problem with respect to two types of variables, i.e., 1) binary variables for caching decision and stop index and 2) integer variables for trajectory, we divide each original chromosome into two sub-chromosomes. The left sub-chromosome and the right sub-chromosome represent the binary variables and the integer variables, respectively. Each sub-chromosome is applied it own crossover and mutation operations with different probabilities. The performance of GAs is investigated in terms of flexible implementation, accuracy, and time complexity compared to other conventional exhausted and greedy search algorithms.

The rest of this paper is organised as follows. We introduce the UAV-assisted CDNs and describe how it works in Sect. 2. In Sect. 3, we review the SECTO problem formulations studied in [8] for the ease of following the GAs solution. Section 4 presents the GAs solution for SECTO optimisation problem. The GAs and system performance metrics are investigated in Sect. 5. Finally, Sect. 6 is dedicated to concluding the paper.

2 UAV-Assisted Content Delivery Networks

In this paper, we consider the model of UAV-assisted CDNs as shown in Fig. 1 [8]. The model consists of one MBS, one UAV, I contents, and N EU clusters. The clusters are established by applying the device-to-device (D2D) clustering scheme [13]. In each cluster, the EUs, who are interested in the same content, are represented by a cluster head (CH). The CHs, which are in charge of forwarding the contents to their EU members, are selected based on their powerful capacity of computing, storage, and energy. The MBS is able to acknowledge all the information of UEs for clustering, then formulating and solving the SECTO optimisation problem. After solving the SECTO optimisation, the UAV realises 1) the optimal set of contents for caching and 2) the optimal trajectory, i.e., optimal set of stops and optimal stop sequence, for flying and delivering. As a result, the content delivery capacity is maximised, meanwhile the flight distance is minimised. In addition, we further consider the constraints of storage and energy resources of the UAV to make the SECTO solution more efficient.

To fly, we assume that the UAV is placed at the horizontal coordinate of $o_{\mathrm{U}} = (x_{\mathrm{U}}, y_{\mathrm{U}})$ while the vertical coordinate is fixed at $z_{\mathrm{U}} = h_{\mathrm{U}}$. We then limit the flight area by defining a smallest rectangular, a length of L (m), and a width of W (m) that covers all the CHs. The rectangular is divided into a grid topology of $M_{\mathrm{L}} \times M_{\mathrm{W}}$ stops as shown in Fig. 1. The horizontal coordinates of the UAV's stops are $o_{\mathrm{U}} = o_{l,w} = (x_l, y_w), l = 1, 2, ..., M_{\mathrm{L}}, w = 1, 2, ..., M_{\mathrm{W}}$. Furthermore, we define a $F_{M_{\mathrm{L}} \times M_{\mathrm{W}}}$ binary matrix in which each pair of row and column is represented by a binary stop index $f(o_{l,w})$. The UAV stops at $o_{w,l}$ if $f(o_{l,w}) = 1$, otherwise $f(o_{l,w}) = 0$. So, given $N_{\mathcal{T}}$ stops, we can find a proper stop sequence $\mathcal{T}$ for the UAV to fly throughout with minimum distance.

3 SECTO Problem Formulations

In this section, we review the system formulations studied in [8] for the ease of following. Based on these formulations, we then introduce the SECTO optimisation problem that can be solved by using GAs.

3.1 Content Popularity

It is certain that each content has its own access rate depending on the EUs' behavior, and thus a given set of I contents follows a popularity pattern. In this paper, we assume that the popularity pattern is modeled by Zipf-like distribution [14]. In this distribution, the content i is ranked in accordance with its popularity expressed as

$$p_i = \frac{i^{-\alpha}}{\sum_{i=1}^{I} i^{-\alpha}}, \tag{1}$$

where α is the coefficient reflecting the skewness of the content popularity preference, i.e., $\alpha = 0$ means that there is no popularity skewness among different contents, while the higher value of α makes the first contents much higher popularity than the last ones.

3.2 Caching, Requesting, and UAV-Cluster Association Index

Due to the storage limit, the UAV is capable of caching a proper number of contents which are frequently requested by the EUs in different clusters. So, the UAV has to decide which contents to be cached before flying to deliver. We define a binary variable c_i for cache decision in which $c_i = 1$ indicates that the content i is cached, otherwise $c_i = 0$. Meanwhile, the number of clusters requesting the content i, which is directly proportional to the popularity p_i is given by

$$N_i = \lceil p_i N \rfloor, \tag{2}$$

where the operator $\lceil . \rfloor$ is used to round an arbitrary value the nearest integer.

However, this way may cause $\sum_{i=1}^{I} N_i \neq N$ due to the round operator. To make the equality held, without loss of generality, we do the following adjustment

- If $\sum_{i=1}^{I} N_i < N$, then the first smallest value in the array $\{N_i, i = 1, 2, ..., I\}$ is increased by 1.
- If $\sum_{i=1}^{I} N_i > N$, then the last highest value in the array $\{N_i, i = 1, 2, ..., I\}$ is decreased by 1.

To obtain the UAV-cluster association index, let $r_{i,n}$ be the requesting index to indicate that $r_{i,n} = 1$ if there is at least one EU in the cluster n requesting the content i ($r_{i,n} = 1$), otherwise $r_{i,n} = 0$. Obviously, we have

$$N_i = \sum_{n=1}^{N} r_{i,n}, \tag{3}$$

and

$$N = \sum_{n=1}^{N} \sum_{i=1}^{I} r_{i,n}. \tag{4}$$

So far, we define the UAV-cluster association index as below

$$a_{i,n} = c_i r_{i,n}. \tag{5}$$

The UAV-cluster association index $a_{i,n} \in \{0, 1\}$ is used to match the content i cached in the UAV and the cluster n. This association index enables us to establish the transmission channel, and thus derive the delivering capacity between the UAV and the cluster n presented in the following subsections.

3.3 Content Delivery Capacity

Following the UAV-cluster association, we further compute the average delivery capacity which is obtained based on the wireless channel model from the UAV to the CHs in different clusters. The UAV utilises the licensed frequency band for backhaul link, while stipulating the unlicensed Industrial, Scientific, Medical

(ISM) WiFi band, i.e.,2.4 GHz with 802.11 b/g/n devices and 5 GHz with 802.11 a/n/ac devices, for communicating with the CH in each cluster [15]. By dominating the LoS links [16,17], the signal-to-noise ratio of the channel for delivering the content i from the UAV located at $\mathrm{o}_{l,w}$ to the CH n in the cluster n located at o_n is given by

$$\gamma_{l,w}^{i,n} = \frac{a_{i,n} P_{\mathrm{T}} \beta_0^{\mathrm{U}} (d_{l,w}^{n})^{-2}}{\sigma^2}, \tag{6}$$

where P_{T} is the transmission power of the UAV, β_0^{U} is the channel power gain at the reference distance of 1m, σ^2 is the additive white Gaussian noise power, and $d_{l,w}^{n}$, which is the distance between the UAV and the CH n, is computed as

$$d_{l,w}^{n} = \sqrt{h_{\mathrm{U}}^2 + \|o_{l,w} - o_n\|^2}, \tag{7}$$

where $\|.\|$ is the Euclidean norm.

From Eq. (6), we have the corresponding delivery capacity expressed as

$$C_{l,w}^{i,n} = B \log_2(1 + \gamma_{l,w}^{i,n}), \tag{8}$$

where B is the system bandwidth.

Finally, the overall average delivery capacity from the UAV to the CHs in all clusters, which is the objective function of the SECTO optimisation problem, is given by

$$\overline{C} = \frac{1}{N} \sum_{n=1}^{N} \sum_{l=1}^{M_L} \sum_{w=1}^{M_W} \sum_{i=1}^{I} p_i f(o_{l,w}) C_{l,w}^{i,n}, \tag{9}$$

where $f(o_{l,w}) \in \{0,1\}$ is the stop index used to indicate that if $f(o_{l,w}) = 1$, the UAV stops at $\mathrm{o}_{l,w}$, otherwise $f(o_{l,w}) = 0$, it does not stop. And thus, the number of stops is expressed as

$$N_{\mathcal{T}} = \sum_{l=1}^{M_L} \sum_{w=1}^{M_W} f(o_{l,w}). \tag{10}$$

In Eq. (9) and Eq. (10), $\overline{C}$ is the function of caching index (c_i) and the flying strategy $f(o_{l,w})$. So, besides finding the optimal caching index, the SECTO finds the optimal flying strategy via $f(o_{l,w})$, which includes: 1) where to stop among $N_{\mathcal{T}}$ out of N stops ($N_{\mathcal{T}} \leq N$) to maximise $\overline{C}$ and 2) how to fly throughout $N_{\mathcal{T}}$ stops to minimise the distance for energy saving.

3.4 Storage and Energy Consumption

Storage Consumption. Given the caching index c_i and the size s_i of the content i, the total caching storage consumed is given by

$$S = \sum_{\substack{i=1 \\ r_{i,n}=1, \forall n}}^{I} c_i s_i. \tag{11}$$

Energy Consumption. The energy is consumed by transmission and flying. To compute the total energy consumption, we need to derive the transmission duration and flying duration. It is noted that when flying, the UAV must stop to transmit the contents, and thus the standstill duration is actually equal to the transmission duration. We first compute the transmission/standstill duration at the stop $o_{l,w}$ for transmitting the content i to the CH n which is given by

$$t_{l,w}^{i,n} = \frac{a_{i,n} f(o_{l,w}) s_i}{\max\{\epsilon, C_{l,w}^{i,n}\}}, \tag{12}$$

where the infinitesimal value ϵ is added to avoid the situation of dividing by zero if $C_{l,w}^{i,n} = 0$. And then, the total transmission/standstill duration is

$$t_{\mathrm{T}} = \sum_{n=1}^{N} \sum_{l=1}^{M_{\mathrm{L}}} \sum_{w=1}^{M_{\mathrm{W}}} \sum_{i=1}^{I} t_{l,w}^{i,n}. \tag{13}$$

For the flying duration, given a minimum distance $d_{\min}$ found by solving the SECTO optimisation problem (23) for minimum value of d (17) and a fixed flying velocity V, it is simply expressed by

$$t_{\mathrm{F}} = d_{min}/V. \tag{14}$$

Finally, the total energy consumption is given by

$$E = P_{\mathrm{F}} t_{\mathrm{F}} + (P_{\mathrm{S}} + P_{\mathrm{T}}) t_{\mathrm{T}}, \tag{15}$$

where P_{F}, P_{S}, and P_{T} are the transmission powers used for flight, standstill, and transmission, respectively.

3.5 SECTO Optimisation Problem

As aforementioned, the SECTO solution includes 1) which contents to cache and where to stop (WCS) for maximising the average content delivery capacity and 2) how to fly over the stops for minimising the distance, i.e., the shortest path (STP) or travelling salesman problem. So, we first present the two STP and WCS optimisation subproblems separately and then we combine them to formulate the SECTO optimisation problem as below.

For the STP optimisation problem, given a set of $N_{\mathcal{T}}$ valid stops $\mathcal{T}_s = \{1, 2, ..., N_{\mathcal{T}}\}$, $\mathcal{T}$ is the set of stop sequence (order) to fly, i.e., $\mathcal{T}$ is an arbitrary combination of $\mathcal{T}_s$. For example, if $\mathcal{T}_s = \{1, 2, 3\}$, $\mathcal{T} = \{2, 1, 3\}$, i.e., $\mathcal{T}_1 = 2, \mathcal{T}_2 = 1, \mathcal{T}_3 = 3$, is a valid stop sequence, meaning that the UAV flies to the stop 2 first, then to the stop 1, and finally to the stop 3. The distance to fly from the stop $n-1$ to the stop n is expressed as

$$d_n = \begin{cases} d\big(\mathcal{T}_{n-1}, \mathcal{T}_n\big), \text{if } \mathcal{T}_n \in \mathcal{T}_s, \mathcal{T}_s = \mathcal{T}_s \setminus \mathcal{T}_n, \\ 0, \text{if } \mathcal{T}_n \notin \mathcal{T}_s. \end{cases} \tag{16}$$

It is noted in (16) that if $\mathcal{T}_n \in \mathcal{T}_s$, after flying to the n-th stop with an actual distance added, and then this stop is removed from the set of valid stops. Otherwise, the UAV cannot fly to the n-th stop, and thus no distance is added but with violence. For example, considering an invalid stop sequence $\mathcal{T} = \{1, 1, 3\}$, it is easy to see that the violence occurs when $n = 2$ at $\mathcal{T}_2$. Consequently, the total distance for the UAV to fly over $N_{\mathcal{T}}$ stops is given by

$$d = \sum_{n=1}^{N_{\mathcal{T}}} d_n. \tag{17}$$

For the ease of grasping the violence degree of (16) if $\mathcal{T}_n \notin \mathcal{T}_s$, we further add an independent penalty function βd_p to (17); where $\beta > 0$ is the violent degree and d_p starts with 0, then increases by 1 if the violence occurs. So, the minimum value of d holds since $d_p = 0$. The STP integer combination problem is formulated as

$$\min_{\mathcal{T}} \,(d + \beta d_p). \tag{18}$$

For the WCS optimisation problem, we take into account the constraints of caching storage (S), energy consumption (E), and number of stops ($N_{\mathcal{T}}$), it is formulated as below

$$\max_{c_i, f(o_{l,w})} \overline{C} \tag{19a}$$

$$\text{s.t.} \qquad S \leq S^* \tag{19b}$$

$$E \leq E^* \tag{19c}$$

$$N_{\mathcal{T}} \leq N \tag{19d}$$

Aiming at maximising the average content delivery capacity (19) and minimising the distance (18) simultaneously, the SECTO optimisation problem is formulated as

$$\max_{c_i, f(o_{l,w}), \mathcal{T}} \left[\overline{C} - (\gamma d + \beta d_p)\right], \tag{20}$$

$$\text{s.t. } (19b), (19c), (19d),$$

where $\gamma > 0$ is used to adjust the balance between the two objective functions $\overline{C}$ and d; and S^* and E^* are to limit the caching storage and energy consumption.

4 GAs for SECTO Optimisation Problem

In this paper, we apply GAs [18] to solve the SECTO optimisation problem. To do so, we apply penalty function [19] which requires to rewrite (19b), (19c), and

(19d) as below

$$\begin{cases} \varDelta S = S^* - S \geq 0, \\ \varDelta E = E^* - E \geq 0, \\ \varDelta N = N - N_{\mathcal{T}} \geq 0. \end{cases} \tag{21}$$

As we have mentioned that βd_p in (18) is the penalty function, thus we extract it from (20), and then add it to the whole penalty function of (22) which is given by

$$F = \lambda_1 \big(\min\{0, \varDelta S\}\big)^2 + \lambda_2 \big(\min\{0, \varDelta E\}\big)^2 + \lambda_3 \big(\min\{0, \varDelta N\}\big)^2 + \beta d_p, \tag{22}$$

where λ_1, λ_2, and λ_3 are the constraint violation degrees which are properly selected to punish the individuals in the current generation if they violate the constraints.

Finally, the GAs can be used to solve the unconstrained SECTO optimisation problem given by

$$\max_{c_i, f(o_{l,w}), \mathcal{T}} \overline{C}_{\mathrm{F}} = \overline{C} - (\gamma d + F). \tag{23}$$

In (23), if d is minimised and too small (~ 0), the UAV flies over small number of stops or even it stops at the original location to transmit the contents. This makes the capacity significantly decreased. Inversely, if d is minimised but too large, the capacity increases but limited by E^*. The balance of maximum value of $\overline{C}$ and minimum value of d yields maximum value of $\overline{C}_F$ when F approaches 0. So, together with λ_1, λ_2, and λ_3, γ is used to balance the impact of F and d on the process of maximising the fitness function $\overline{C}_{\mathrm{F}}$.

To implement GAs, it is noted that the SECTO optimisation problem has two different types of variables. The binary variables (c_i and $f(o_{l,w})$) for caching and stop indexes, and meanwhile the integer variables ($\mathcal{T}$) for stop sequence. These two types have different requirements of crossover and mutation, i.e., different operations and probabilities. So, we use two chromosome types, one for binary variables and the other for integer variables. Correspondingly, we have to create two separated binary and integer populations, each of the same number of individuals is performed by its own crossover and mutation operations.

In the SECTO optimisation problem, because the binary variables are c_i and $f(o_{l,w})$ of length $I + M_{\mathrm{L}} \times M_{\mathrm{W}}$ bits, each individual represented by a binary chromosome (binary string) has $N_{\mathrm{B}} = I + M_{\mathrm{L}} \times M_{\mathrm{W}}$ bits. The maximum number of stops in the sequence $\mathcal{T}$ is $N_{\mathcal{T}} = N$, each individual represented by an integer chromosome (integer string) therefore has $N_{\mathrm{I}} = N$ integers. When GAs terminate, the first $N_{\mathcal{T}}$ stops in the final $\mathcal{T}$ are derived for the optimal stop sequence. The detailed implementation of GAs is shown in Algorithm 1. In Algorithm 1, GAs terminate if one of the three following conditions of TC holds: 1) $F = 0$ in 3 successive generations, 2) $\overline{C}_{\mathrm{F}}$ does not change while $F \leq 10^{-3}$ in 3 successive generations, and 3) $gen = N_G$.

Algorithm 1. GAs implementation

Input: System parameters listed in Table 1

$N_{\mathrm{P}} = 10,000$: Number of individuals in both the binary and integer populations
$N_{\mathrm{B}} = I + M_L \times M_W$: Number of binary variables (bits) for each binary individual
$N_{\mathrm{I}} = N$: Number of integer variables for each integer individual
$N_{\mathrm{G}} = 100$: Number of generations
$P_{\mathrm{G}} = 0.8$: Generation gap
$P_{\mathrm{CB}} = 0.6$: Crossover probability for binary individuals
$P_{\mathrm{CI}} = 0.9$: Crossover probability for integer individuals
$P_{\mathrm{MB}} = 10^{-6}$: Mutation probability for binary individuals
$P_{\mathrm{MI}} = 10^{-10}$: Mutation probability for integer individuals
$\beta = 1$: Violence degree applied if $\mathcal{T}$ is invalid
$\gamma = 0.1$: Coefficient used to balance the two objective functions $\overline{C}$ and d
$\{\lambda_{1,2,3}\} = \{1, 10^3, 1\}$: The method to choose the proper λs is presented in [20]
TC: Termination conditions
$Gen = 1$: Generation count

Output: $\overline{C}_F^*$ and X^*

1: Randomly generating N_P binary strings, each of length N_{B} bits to represent the individual k ($\{X_{\mathrm{B}}^k\}$), $k = 1, 2, ..., N_P$
2: Randomly generating N_P integer strings, each of length N_{I} integers to represent the individual k ($\{X_{\mathrm{I}}^k\}$)
3: Mapping $\{X_{\mathrm{B}}^k\}$ into the binary variables c_i^k and $f(o_{l,w}^k)$ and mapping $\{X_{\mathrm{I}}^k\}$ into the integer variables $\mathcal{T}^k$
4: Calculating N_P fitness values $\overline{C}_F\big(c_i^k, f(o_{l,w}^k), \mathcal{T}^k\big)$
5: **while** TC does not hold **do**
6: Putting $\{X_{\mathrm{B}}^k\}$, $\{X_{\mathrm{I}}^k\}$, and $\overline{C}_F\big(c_i^k, f(o_{l,w}^k), \mathcal{T}^k\big)$ in the mating pool for ranking
7: Selecting $N_{\mathrm{G}} = N_{\mathrm{P}} \times P_{\mathrm{G}}$ better individuals with higher $\overline{C}_F\big(c_i^k, f(o_{l,w}^k), \mathcal{T}^k\big)$ for breeding the next generation by using stochastic universal sampling operator [18]
8: Dividing the selected individuals into a set of N_{G} binary individuals $\{X_{\mathrm{B}}^{k,*}\}$ and a set of N_{G} integer individuals $\{X_{\mathrm{I}}^{k,*}\}$
9: Selecting a pair of parents to make a pair of offsprings by using 1) multiple point crossover with probability P_{CB} for $\{X_{\mathrm{B}}^{k,*}\}$ and 2) single point crossover with probability P_{CI} for $\{X_{\mathrm{I}}^{k,*}\}$ [18]
10: Mutating the offsprings of $\{X_{\mathrm{B}}^{k,*}\}$ with probability P_{MB} and the offsprings of $\{X_{\mathrm{I}}^{k,*}\}$ with probability P_{MI}, to recover the good genetic materials likely lost in the previous operations
11: Merging the two sets of offsprings into one, evaluating the fitness values of the merged set, reinserting it into the present generation
12: $Gen = Gen + 1$
13: **end while**
14: Finding the best fitness value $\overline{C}_F^*$ with respect to the best individual X^* in the last generation

5 Performance Evaluation

5.1 GAs Performance

To evaluate the performance of GAs, we compare it to the other two search algorithms namely exhausted - exhausted (EE) search algorithm and exhausted - greedy (EG) search algorithm which are corresponding to TSM and NSH respectively studied in [8]. We deploy GAs, EE, and EG by a desktop computer with detailed information listed in Table 2. The CHs are randomly distributed within a circle of 100-meter radius. The UAV is originally located at $(x_{\mathrm{U}}, y_{\mathrm{U}}) = $ (100m, 100m) and its vertical coordinate is fixed at $h_{\mathrm{U}} = 50$m.

We first evaluate the convergence rate of GAs by considering 100 generations. Figure 2 plots the results of the best fitness value $\overline{C}^*_F$, the mean of all the fitness values with respect to all individuals in each generation, and the penalty value (F). The results show that the convergence situation of GAs starts from the 35-th generation when the mean value is closer to the best value and $F = 0$. It indicates that all the individuals become better after each generation while ensuring the given constraints for optimal solutions.

The performance of GAs is further evaluated by investigating the accuracy and time complexity in comparison with EG and EE. The Algorithm 1 is exe-

Table 1. Parameters setting.

Symbols	Specifications
$\{N, I\}$	{10 clusters, 5 videos}
s_i	[300, 100, 200, 400, 900] Mbits
S^*	1000 Mbits
α	1
V	15 m/s
$\{P_{\mathrm{T}}, P_{\mathrm{F}}, P_{\mathrm{S}}\}$	$\{0.5, 50, 0.25 \times P_{\mathrm{F}}\}$ W
E^*	5000 Joules
B	10 MHz
β_0^{U}	10^{-5}
σ^2	10^{-13} W

Table 2. Computer information.

Processor	Intel(R) Core(TM) i7-7700 CPU @ 3.60 GHz, 3601 Mhz
Processor type	x64-based PC
Processor cores	4
Logical processors	8
Total PHY memory	15,9 GB
Operating System	Windows 10

cuted versus different population sizes N_P from 1000 to 20,000. For each population size, the Algorithm 1 is repeated 50 times, and then we calculate the average of 50 optimal results of $\overline{C}$ and d in accordance with the average of time complexities. As shown in Table 3, the accuracy of GAs increases if N_P increases. We select $N_P = 10,000$ for obtaining the optimal results because if $N_P > 10,000$, the accuracy does not increase significantly but time complexity becomes too high, i.e., up to 1,009.36 s. At $N_P = 10,000$, GAs provide a reasonable accuracy of 99.69% for $\overline{C}$ and 99.51% for d compared to EE. However, GAs have a much less time complexity of 356.01 s compared to that of 1,320.17 s introduced by EE. Related to EG, although it can provide the exact result of $\overline{C}$ thanks to using exhausted search algorithm, the accuracy of d is lower than GAs done at $N_P \geq 5,000$. We can see that the accuracy and time complexity of EG is equivalent to GAs done at $N_P = 5,000$. So, GAs are flexible to obtain the exact or approximate optimal results with reasonable accuracy and time complexity.

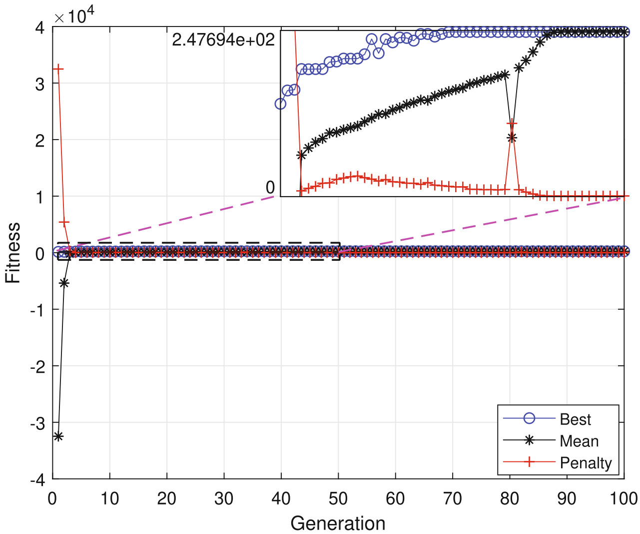

Fig. 2. Convergence rate of GAs.

Table 3. Accuracy and time complexity comparison

Metric	N_P (GAs)						
	1,000	5,000	10,000	15,000	20,000	EG	EE
$\overline{C}$ (Mbps)	310.55	310.91	310.97	311.14	311.15	311.95	311.95
Accuracy of $\overline{C}$ (%)	99.55	99.67	99.69	99.74	99.74	100.00	100.00
d (m)	696.26	653.93	645.64	641.25	641.94	657.93	642.53
Accuracy of d (%)	91.64	98.22	99.51	99.80	99.91	97.60	100.00
Time (s)	19.23	117.55	356.01	635.52	1,009.36	111.65	1,320.17

5.2 System Performance Metrics

Storage-Aware Results. To have the storage-aware results, we run the Algorithm 1 by changing the skewness coefficient of content popularity preference α with different values of caching storage constraint S^* (19b). As shown in Fig. 3, if S^* is large enough ($S^* \geq 400$ Mbits), $\overline{C}$ increases with respect to the increase of α commonly (Fig. 3(a)), meanwhile d is mostly kept the same to provide the highest value of $\overline{C}$ (Fig. 3(b)). It is noted that when α is high, the results become

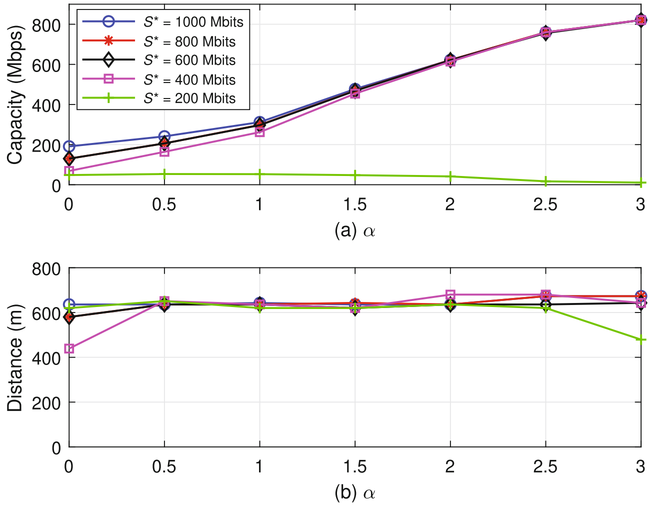

Fig. 3. Capacity and distance versus α with different S^*.

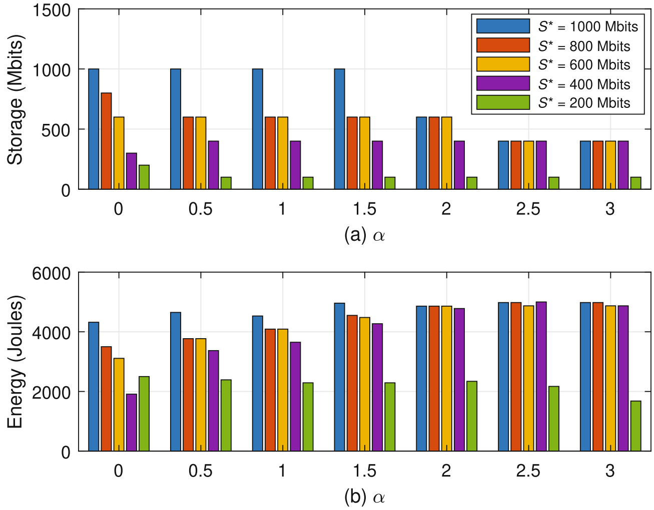

Fig. 4. Storage and energy consumption versus α with different S^*.

equally saturated because the system just focuses on less contents with higher popularity to serve the CHs for the highest capacity performance. However, if S^* is too small ($S^* = 200$ Mbits), it is clear that $\overline{C}$ significantly decreases since the UAV cannot cache the most popular contents with $s_i > 200$ Mbits, e.g., $s_1 = 300$ Mbits.

Figure 4 plots the storage and energy consumption which strictly satisfies the constraints (19b) and (19c). In Fig. 4(a), if α is high enough, i.e., $\alpha \geq 2.5$,

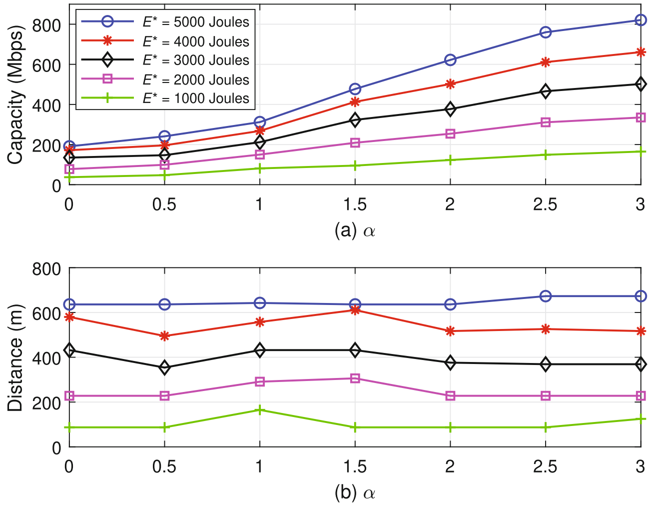

Fig. 5. Capacity and distance versus α with different E^*.

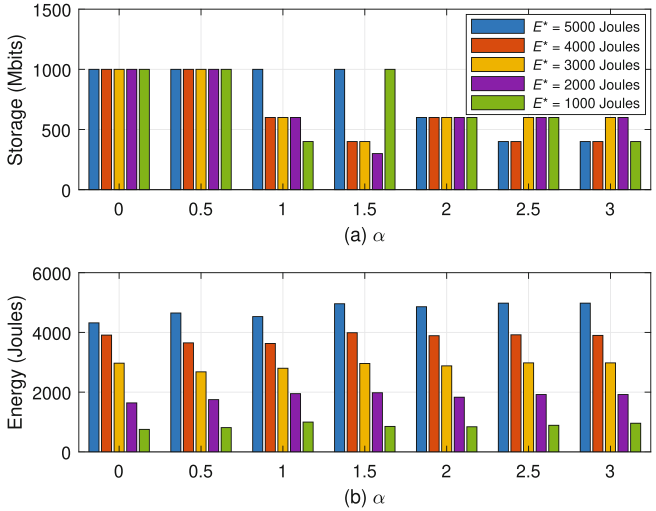

Fig. 6. Storage and energy consumption versus α with different E^*.

the system just focuses on less contents with higher popularity as we mentioned earlier, thus the caching storage consumption decreases. However, the energy consumption increases if α increases because more CHs request the most popular content with relatively large size, i.e., $s_1 = 300$ Mbits, leading to the fact that the transmission and standstill duration becomes longer requiring higher energy consumption (Fig. 4(b)), except for $S^* = 200$ Mbits making the distance decreased.

Energy-Aware Results. To have the energy-aware results, we run the Algorithm 1 by changing the skewness coefficient of content popularity preference α with different values of energy constraint E^* (19c). As shown in Fig. 5, the decrease of E^* significantly reduces both capacity and distance. The results of caching storage and energy consumption in Fig. 6 always satisfy the constraints (19b) and (19c). The behavior of all the system performance metrics is similar to that of the storage-aware results with respect to α.

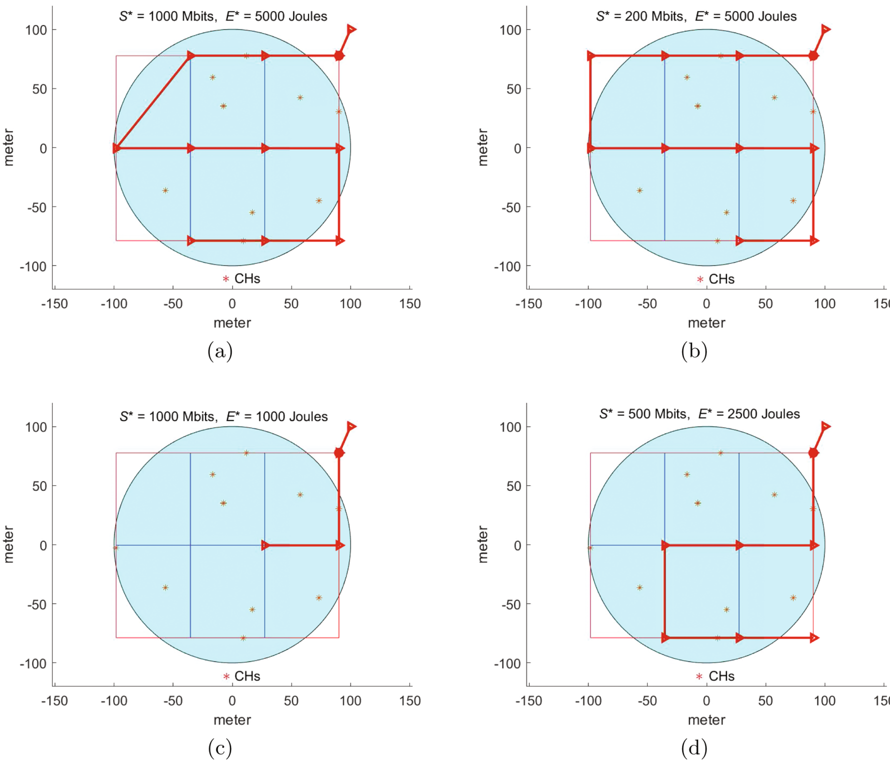

Fig. 7. Optimal trajectories with different constraints of S^* and E^*.

Optimal Trajectories. Figure 7 depicts the optimal trajectories of the UAV when $\alpha = 1$ versus different constraints of S^* and E^*. Figure 7(a) shows the optimal trajectory when S^* and E^* are relaxed to 1000 Mbits and 5000 Joules, respectively. In Fig. 7(b), if we strictly limit the caching storage constraint S^* to 200 Mbits, the UAV has to change the caching policy, and thus change the set of stops to serve the CHs with a different optimal trajectory. The trajectories significantly change to the short flight distances if we further limit the energy constraint E^* or/and the caching storage constraint S^* as plotted in Fig. 7(c) and Fig. 7(d).

6 Conclusion

In this paper, we have introduced GAs to solve the SECTO problem for UAV-assisted CDNs to maximise the content delivery capacity while minimising the flight distance. Unlike other common optimisation problem having only one objective function with respect to (w.r.t) only one type of variable, i.e., binary, real, or integer variable, the SECTO problem consists of two objective functions w.r.t both binary and integer variables. To solve the SECTO problem, we modify the GAs by applying penalty method and separating the whole string (chromosome) of each individual into binary sub-string and integer sub-string. These sub-strings are applied different crossover and mutation schemes (operations and probabilities). The results are shown to demonstrate the benefits of GAs in terms of flexible implementation, accuracy, and time complexity compared to other conventional exhausted and greedy search algorithms.

References

1. Strinati, E., et al.: 6G in the sky: on-demand intelligence at the edge of 3D networks. ETRI J. **42**, 643–657 (2020)
2. Chen, M., Mozaffari, M., Saad, W., Yin, C., Debbah, M., Hong, C.S.: Caching in the sky: proactive deployment of cache-enabled unmanned aerial vehicles for optimized quality-of-experience. IEEE J. Sel. Areas Commun. **35**, 1046–1061 (2017)
3. Zeng, Y., Wu, Q., Zhang, R.: Accessing from the sky: a tutorial on UAV communications for 5G and beyond. Proc. IEEE **107**, 2327–2375 (2019)
4. Lu, R., Zhang, R., Cheng, X., Yang, L.: Relay in the sky: a UAV-aided cooperative data dissemination scheduling strategy in VANETs. In: Proceedings IEEE International Conference on Communications (Shanghai, China), pp. 1–6 (2019)
5. Erdelj, M., Natalizio, E., Chowdhury, K.R., Akyildiz, I.F.: Help from the sky: leveraging UAVs for disaster management. IEEE Pervasive Comput. **16**, 24–32 (2017)
6. Bilen, T., Canberk, B.: Content delivery from the sky: drone-aided load balancing for mobile-CDN. EAI Endorsed Trans. Ind. Netw. Intell. Syst. **9**, 1–7 (2022)
7. Duong, T.Q., Kim, K.J., Kaleem, Z., Bui, M.-P., Vo, N.-S.: UAV caching in 6G networks: a survey on models, techniques, and applications. Phys. Commun. **51**, 1–19 (2022)

8. Vo, N.-S., Lam, T.C., Bui, M.-P., Phan, T.-M., Tran, Q.-N.: UAV assisted video multicasting in 6G networks: a joint caching and trajectory optimisation. J. Aviat. Sci. Technol. **1**, 37–42 (2022)
9. Li, X., Liu, J., Zhao, N., Wang, X.: UAV-assisted edge caching under uncertain demand: a data-driven distributionally robust joint strategy. IEEE Trans. Commun. **70**, 3499–3511 (2022)
10. Zhang, T., Wang, Y., Yi, W., Liu, Y., Nallanathan, A.: Joint optimization of caching placement and trajectory for UAV-D2D networks. IEEE Trans. Commun. **7**, 5514–5527 (2022)
11. Xu, H., Ji, J., Zhu, K., Wang, R.: Reinforcement learning for trajectory design in cache-enabled UAV-assisted cellular networks. In: Proceedings IEEE Wireless Communications and Networking Conference (Austin, TX), pp. 1–6 (2022)
12. Ji, J., Zhu, K., Cai, L.: Trajectory and communication design for cache- enabled UAVs in cellular networks: a deep reinforcement learning approach. IEEE Trans. Mob. Comput., 1–15 (2022)
13. Gyawali, S., Xu, S., Ye, F., Hu, R.Q., Qian, Y.: A D2D based clustering scheme for public safety communications. In: Proceedings of IEEE 87th Vehicular Technology Conference (Porto, Portugal), pp. 1–5 (2018)
14. Breslau, L., Cao, P., Fan, L., Phillips, G., Shenker, S.: Web caching and Zipf-like distributions: evidence and implications. In: Proceedings of IEEE International Conference on Computer Communications (INFOCOM) (New York, NY), pp. 126–134 (1999)
15. Lin, D., Zuo, P., Peng, T., Qian, R., Wang, W.: Energy-efficient UAV-based IoT communications with WiFi suppression in 5 GHz ISM bands. IEEE Trans. Veh. Technol., 1–16 (2022)
16. Xu, X., Zeng, Y., Guan, Y.L., Zhang, R.: Overcoming endurance issue: UAV-enabled communications with proactive caching. IEEE J. Sel. Areas Commun. **36**, 1231–1244 (2018)
17. Wu, H., Lyu, F., Zhou, C., Chen, J., Wang, L., Shen, X.: Optimal UAV caching and trajectory in aerial-assisted vehicular networks: a learning-based approach. IEEE J. Sel. Areas Commun., 1–14 (2020)
18. Chipperfield, A., Fleming, P., Pohlheim, H., Fonseca, C.: Genetic algorithm TOOLBOX for using with Matlab Ver 1.2 users guide. University of Sheffield (1994)
19. Fang, T., Chau, L.P.: GOP-based channel rate allocation using genetic algorithm for scalable video streaming over error-prone networks. IEEE Trans. Image Process. **15**, 1323–1330 (2006)
20. Vo, N.-S., Duong, T.Q., Tuan, H.D., Kortun, A.: Optimal video streaming in dense 5G networks with D2D communications. IEEE Access **6**, 209–223 (2017)