# Drone Base Stations Transmission Power Control and Localization

Salim Janji(✉) and Adrian Kliks

Institute of Radiocommunications, Poznan University of Technology, Poznań, Poland
salim.janji@doctorate.put.poznan.pl

**Abstract.** We target the problem of deploying drone base stations (DBSs) to serve a set of users with known locations after dividing it into two problems: users clustering along with 2D localization of drones, and transmission power control. For clustering, we propose an algorithm inspired by the expectation-maximization (EM) algorithm that boosts link reliability. As for transmission power control, we utilize the Monte Carlo tree search (MCTS) algorithm. Furthermore, we show that the resulting mechanism can also adapt to user changes without rebuilding the search tree, and it also intrinsically avoids collision between DBSs. Also, our simulations show that our proposed algorithm improves the link reliability and system energy efficiency substantially in comparison to another mechanism mentioned in the literature. The results show that controlling the power of DBSs does not only reduce power consumption but also improves the users' links quality. Finally, we show the improvement in system performance relative to the baseline method where DBSs are randomly distributed and users are changing their location according to the random waypoint mobility model.

**Keywords:** Ad hoc networks · Interference management · Drone base stations · Users clustering

## 1 Introduction

Drone base stations (DBS) are considered as promising solutions for the barriers faced by future wireless networks [1]. In areas where fixed infrastructure with high capacity is not always needed, DBSs can provide on-demand cost-effective coverage when such a capacity is required [2]. For example, they can also offload traffic from base stations in areas with crowded events or flash mobs [3]. Moreover, in cases where a fixed infrastructure is unavailable in isolated or disaster-struck areas, a number of DBSs can provide adequate connectivity for the required duration [4]. Compared to ground base stations, a DBS can establish Line-of-Sight (LOS) communication links to the users with reduced path loss, and it can continuously adapt its location to improve the performance of the communication system. Finally, due to their altitude, DBSs can leverage the use of free space optical (FSO) communications in the backhaul layer which can achieve higher data rates, longer transmission range, and less interference [5].


S. Longfei and P. Bodhi (Eds.): MobiQuitous 2022, LNICST 492, pp. 356–377, 2023.
https://doi.org/10.1007/978-3-031-34776-4_19

However, the use of DBSs also has its limitations. Multiple DBSs can cause severe co-channel interference when operating close to each other. Another issue is the limited onboard energy which restricts the transmission power, capacity, and deployment period of the DBS. Determining the location of the DBS should take into consideration the aforementioned factors. In this paper, we deal with the problem of 2D localization and transmission power control of DBSs while optimizing for users' signal-to-interference-plus-noise ratio (SINR) and satisfying the drones capacity constraint.

Our contributions are summarized as follows:

- Given the settings of drones' transmission powers, we find the 2D location for each DBS such that the path loss is minimized along with reducing interference between clusters served by different DBSs. We do so by proposing a novel algorithm derived from the standard expectation-maximization clustering algorithm (EM). We call it SINR-EM.
- We find the settings of drones' transmission powers such that the number of served users achieving a minimum signal-to-interference-plus-noise ratio (SINR), after applying SINR-EM, is maximized while satisfying the drones capacity constraint. This problem is modeled as a game tree in which the solution is found using Monte Carlo tree search. We call our algorithm Transmission Power Monte Carlo Tree Search (TP-MCTS). We also show that TP-MCTS can learn from its previous tree history for building a new tree whenever the deployment parameters of DBSs are to be updated.
- By simulations we show that our proposed algorithm performs substantially better in terms of link reliability and energy efficiency when compared to another algorithm that uses particle swarm optimization (PSO) [6].
- While utilizing the random waypoint mobility model (RWP) [7] we simulate our proposed algorithm against the baseline scenario in which DBSs are randomly distributed in a given area and show that our algorithm can adapt to the changes in users' locations.
- Finally, we discuss the insights that our results provide into the trade-off between the number of DBSs deployed, their transmission power settings, system performance in terms of efficiency, and the users' quality of service.

The rest of the paper is structured as follows. Section 2 gives an overview of related work in the literature. Section 3 presents the system model including the scenario considered, channel model, and the performance indicators that are to be achieved or maximized. Section 4 begins by introducing the standard EM algorithm then we introduce our proposed SINR-EM solution along with illustrating the effect of the threshold parameter which we introduced. Monte Carlo Tree Search is then introduced in Sect. 5, followed by an illustration of our version which is adapted to the problem in hand to which we also present a complete algorithmic description. Furthermore, Sect. 6 presents the simulation results for two scenarios: TP-MCTS against the PSO solution [6] for a static distribution of dense users; and TP-MCTS against the random deployment of DBSs in a given area for moving users. Finally, Sect. 7 concludes the paper.

## 2 Related Work

UAVs applications in wireless networks were the focus of extensive research in recent years [1,2,8–10]. In the matter of a single DBS deployment, the authors in [11] formulated a circle placement and smallest enclosing circle problems such that the transmit power is minimized while maximizing coverage. Similarly in [12], the scenario considers users with different signal-to-noise ratio (SNR) requirements where the goal is again to maximize their coverage. In [13] the optimal altitude for the DBS is derived for optimal coverage. On the other hand, in [14] a grid search algorithm was proposed to maximize the number of served users and sum-rates as far as the backhaul capacity permits.

Moreover, the problem of deploying multiple DBSs introduces further complexity and numerous solutions were proposed of which we mention a few. In [15], the authors proposed deploying DBSs sequentially along a spiral path to minimize the required number of DBSs and cover users. In [16], the authors proposed Q-learning and deep Q-learning (DQL) algorithms for the deployment of a single DBS and multiple DBSs respectively such that the fairness in coverage is maximized along with reward penalties for collision avoidance and interference minimization between different DBSs. Similarly, the authors in [17] designed the state representation, action, and reward for training a deep Q-network using prioritized replay which eventually determines the locations of DBSs such that user coverage is maximized using an obstacle-aware channel model while disregarding interference between DBSs. Furthermore, Liu et al. in [18] utilize Deep Deterministic Policy Gradient (DDPG) to train actor-critic networks such that the movement of DBSs is determined in continuous space and the individual coverage, coverage fairness, and energy consumption are optimized. A subset of UAVs serve as gateways through satellite links, and a penalty is applied whenever a DBS is far from any gateway and is therefore offline. However, instead of considering ground users, the algorithm considers points of interest and interference between DBSs is not taken into account. In [19], the authors consider a street graph model where only outdoor users are considered and represented by their densities along the graph edges and vertices. Moreover, they formulate an unconstrained optimization problem whose objective is a combination of three functions: the first calculates the total coverage achieved by DBSs, while the other two are heuristic functions that introduce a penalty based on the interference between DBSs and the distance to the nearest charging pole. As explained, the problem can be solved distributively whereas each DBS moves only when its local objective function can be improved.

Besides the specified differences, all of the aforementioned works with a focus on multiple DBSs deployments do not consider the capacity of DBSs which will, in reality, be limited due to bandwidth or backhaul constraints. Furthermore, the mobility of users is also neglected whereas we show that our proposed solution is able to continuously adapt to the changes in the users' distribution without restarting the state. Moreover, instead of considering the coverage in terms of SNR we directly target the optimize the signal-to-interference-plus-noise ratio (SINR) which takes into account the interference between DBSs and plays an

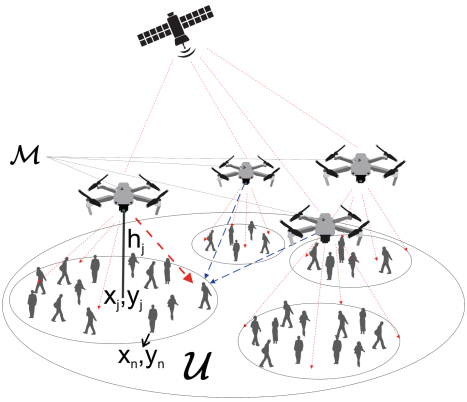



**Fig. 1.** The considered scenario in which users with known locations are served by a set of DBSs.

important role in the achieved link quality. We do so because we consider that the available number of DBSs is sufficient to provide coverage to the considered area and optimize the location and transmission power of DBSs. The authors of the work in [6] to which we compare our results make similar assumptions and use particle swarm optimization (PSO) to determine the 3D locations of DBSs.

## 3 System Model

The considered scenario is shown in Fig. 1 in which a set $\mathcal{U}$ of $U$ users are served by the set $\mathcal{M}$ containing $M$ drones. Such scenarios can include temporary events or crowded areas that are not covered by fixed infrastructure (e.g., due to destruction). For simulations, two users distribution models are considered. The first is a static distribution with different uniform densities in different regions similarly to [6], and the second utilizes RWP [7] to model the movement of users. A 3D Cartesian coordinates system is used such that the location of each drone is given by $\mathbf{m}_j = (x_j, y_j, h_j)$ where $j \in \{1, 2, .., M\}$. Similarly, the 2D location of each user is given by $\boldsymbol{u}_n = (x_n, y_n)$ where $n \in \{1, 2, .., U\}$. The users are assumed to be at ground level height ($h_n = 0$) and the drone height, $h_j$, is given above ground level (AGL). Furthermore, we assume that each drone $j$ is transmitting at a controllable power of $P_{T_j} \leq P_{\max}$, where $P_{\max}$ is the maximum transmission power of the DBS. Finally, we assume that the DBSs are backhaul-connected via satellite links [15].

### 3.1 Channel Model

The distance between DBS $j$ and user $n$ is calculated as

$$d_{j,n} = \sqrt{(x_j - x_n)^2 + (y_j - y_n)^2 + h_j^2}. \tag{1}$$

As presented in [13], the probability of a LOS connection between the $j$-th DBS and $n$-th user can be given by

$$Pr_{\mathrm{LOS}_{j,n}} = \frac{1}{1 + \alpha \times e^{-\beta(\omega - \alpha)}}, \tag{2}$$

where $\alpha$ and $\beta$ are environment-dependent variables that can be set according to the environment type (i.e., suburban, dense urban, highrise urban, etc.), and $\omega = \arctan(\frac{h_j}{d_{j,n}})$. Accordingly, the probability of a non-line-of-sight connection can be obtained from (2) as $Pr_{\mathrm{NLOS}_{j,n}} = 1 - Pr_{\mathrm{LOS}_{j,n}}$. Furthermore, the path loss is a combination two components: a free-space path loss component, and an average path loss that depends on link type (LOS or NLOS). The path loss is therefore represented as

$$L^{[\mathrm{dB}]} = 20 \log \left( \frac{4\pi f_c d}{c} \right) + \eta_{path}, \tag{3}$$

where $f_c$ is the carrier frequency, $c$ is the speed of light, $d$ is the distance, and $\eta_{path}$ is the average loss depending on the link type. The expected path loss between a DBS $j$ and user $u$ can therefore be given as

$$L_{j,n}^{[\mathrm{dB}]} = 20 \log \left( \frac{4\pi f_c d_{j,n}}{c} \right) + Pr_{\mathrm{LOS}_{j,n}} \times \eta_{\mathrm{LOS}} + Pr_{\mathrm{NLOS}_{j,n}} \times \eta_{\mathrm{NLOS}}, \tag{4}$$

whereas $\eta_{\mathrm{LOS}} < \eta_{\mathrm{NLOS}}$. Given the total bandwidth of a DBS $B_D$ and the allocated bandwidth to the user $B_n \leq B_D$, the received power for user $n$ can be obtained by

$$P_{R_{j,n}} = \frac{B_n P_{T_j} 10^{\frac{-L_{j,n}}{10}}}{B_D}. \tag{5}$$

Using (5), the received signal-to-noise ratio (SNR) is given by

$$\mathrm{SNR}_{j,n} = \frac{P_{R_{j,n}}}{\sigma_0^2}, \tag{6}$$

where $\sigma_0^2$ is the receiver noise power. The signal-to-interference-plus-noise ratio (SINR) is therefore given on the user side as

$$\Gamma(j,n) = \frac{P_{R_{j,n}}}{\sigma_0^2 + \sum_{i \neq j}^{M} P_{R_{i,n}}}. \tag{7}$$

### 3.2 Problem Formulation

In the considered scenario it is assumed that the transmission power of each drone can be varied up to the maximum limit specified by $P_{\max}$. Also, the 2D location of each drone is denoted by $\mathbf{m}_j = (x_j, y_j)$ where the height $h_j$ is omitted and assumed to be constant with a value of $h$. To balance the load, we assume

that each DBS can serve up to a maximum number of $U_{\max}$ users that are given a bandwidth share of $B_n = \frac{B_D}{U_{\max}}$.

A considered figure of merit is the system energy efficiency, $\zeta$, which can be derived as the total bit rate divided by the total transmission power of the drones as follows

$$\zeta = \frac{R_{\text{total}}}{P_{\text{total}}}. \tag{8}$$

Using Shannon's capacity formula, the total bit rate in Kbps is

$$R_{\text{total}} = \frac{1}{1000} \sum_{j=1}^{M} \sum_{n=1}^{U} W(j,n) B_n \ \log_2(1 + \Gamma(j,n)), \tag{9}$$

where

$$W(j,n) = \begin{cases} 1 & \text{if } j = \underset{j'}{\text{argmax}}\ \Gamma(j',n) \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

maps the association of user $n$ to drone $j$ which provides the highest SINR value. The total transmitted power in Watts is given by

$$P_{\text{total}} = \sum_{j=1}^{M} P_{T_j}. \tag{11}$$

Another performance metric is the number of served users achieving an SINR value greater than or equal to a predefined threshold $\Gamma_{Th}$,

$$N_j = \sum_{n=1}^{U} W(j,n) H\left(\Gamma(j,n) - \Gamma_{Th}\right), \tag{12}$$

where $H(x)$ is the Heaviside step function. Taking into consideration the capacity constraint of $U_{\max}$, the total number of users served with an SINR requirement or $\Gamma_{Th}$ is

$$N_{\text{SU}} = \sum_{j=1}^{M} N_{Sj}, \tag{13}$$

where

$$N_{Sj} = \begin{cases} N_j, & \text{if } N_j \leq U_{\max} \\ U_{\max}, & \text{otherwise.} \end{cases} \tag{14}$$

Let $N_{th} = \Phi U$ be the minimum required number of served users achieving an SINR value greater than or equal to $\Gamma_{Th}$ where $\Phi$ is the desired percentage of served users (or link reliability), then the problem is formulated as

$$\max_{P_{T_j}, \mathbf{m}_j} \ \zeta \tag{15a}$$

$$\text{s.t.} \quad N_{\text{SU}} \geq N_{th} \tag{15b}$$

$$0 \leq P_{T_j} \leq P_{\max} \tag{15c}$$

The formulation in (15) aims to maximize the system energy efficiency after ensuring that the minimum required number of users are served with an SINR value satisfying $\Gamma_{Th}$ which is ensured by the constraint in (15b). Both the location of the $j$-th drone and its power can be varied whereas the latter is constrained to be less than $P_{\max}$ as dictated by the second constraint in (15c).

## 4 Expectation-Maximization Algorithm

### 4.1 Standard EM

EM algorithm usually utilizes Gaussian Mixture Models (GMMs) that model the set of sample points as being drawn from a "mixture" of Gaussian distributions [20]. A GMM gives the probability of a sample point being drawn from the mixture and it can be defined as:

$$\pi(\boldsymbol{u};\theta) = \sum_{j=1}^{M} P(j) N(\boldsymbol{u};\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j), \tag{16}$$

where the parameters $\theta = \{P, \boldsymbol{\mu}, \boldsymbol{\Sigma}\}$ include mixing proportions $P$, means of Gaussian components $\boldsymbol{\mu}$, and covariance matrices $\boldsymbol{\Sigma}$, and $N(\boldsymbol{u};\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)$ gives the probability of sample point $\boldsymbol{u}$ taken from the $j$-th Gaussian distribution.

The EM algorithm learns the parameters $\theta$ given a set of sample points, $\mathcal{U}$, and the number of mixtures, $M$. In this paper, the notation for the number of Gaussian components, $M$, and the set of sample points, $\mathcal{U}$, is consistent with the mapping of these parameters to the total number of drones and the set of 2D locations of users respectively. Moreover, each mean, $\boldsymbol{\mu}_j$, corresponds to the 2D location of drone $j$. At each iteration the algorithm performs two steps:

1. Estimation step (E-step). Evaluate the posterior assignment probabilities given by:

$$p^{(l)}(j|n) = p(j|\boldsymbol{u}_n,\theta^{(l)}) = \frac{P^{(l)}(j) N\left(\boldsymbol{u}_n;\boldsymbol{\mu}_j^{(l)},\boldsymbol{\Sigma}_j^{(l)}\right)}{\pi(\boldsymbol{u}_n;\theta^{(l)})}, \tag{17}$$

   where $(l)$ denotes iteration number, and $\boldsymbol{u}_n = \begin{bmatrix} x_n \\ y_n \end{bmatrix}$ is the sample $n \in \{1,..,U\}$, where $U$ is the total number of samples.
2. Maximization step (M-step). Update parameters according to:

$$P^{(l+1)}(j) = \frac{\hat{n}(j)}{U}, \quad \text{where } \hat{n}(j) = \sum_{n=1}^{U} p^{(l)}(j|t). \tag{18a}$$

$$\boldsymbol{\mu}_j^{(l+1)} = \frac{1}{\hat{n}(j)} \sum_{n=1}^{U} p^{(l)}(j|n)\boldsymbol{u}_n. \tag{18b}$$

$$\boldsymbol{\Sigma}_j^{(l+1)} = \frac{1}{\hat{n}} \sum_{n=1}^{U} p^{(l)}(j|n) \left(\boldsymbol{u}_n - \boldsymbol{\mu}_j^{(l+1)}\right)\left(\boldsymbol{u}_n - \boldsymbol{\mu}_j^{(l+1)}\right)^T. \tag{18c}$$

The E-step obtains the posterior probabilities, $p^{(l)}(j|n)$, that denote the probability of having the sample observed ($\boldsymbol{u}_n$) being drawn from the $j$-th Gaussian component. In the M-step, the parameters of each component $j$ are updated to increase the total likelihood of the GMM. It is worth noting that the EM algorithm is guaranteed to converge to a local optimal solution [20].

### 4.2 EM-Based Clustering Algorithm

We propose a new clustering algorithm inspired by EM. Since we are interested in obtaining the 2D locations of DBSs we only consider the estimation step given by (17) and the mean update given by (18b). In particular, we modify (17) such that we replace the posterior probabilities $p(j|n)$ by weights that are proportional to the achieved SINR $\Gamma(j,n)$. The proposed solution aims to:

1. Directly incorporate the transmission power of DBSs into the posterior probabilities calculations and mean updates. The result is that a higher transmission power for DBS $j$ entails a cluster with a larger area (i.e., for any $\boldsymbol{u}_n$ within that area, $p(j|n) > p(k|n)$ for every $k \neq j$). By building such a relationship we can then delegate the task of choosing the best transmission power settings to TP-MCTS.
2. Allow each $\boldsymbol{u}_n$ to contribute to the mean update of cluster $j$ with a value of $p(j|n)$ proportional to the SINR received from that cluster (i.e., DBS) relatively to the other clusters. Furthermore, completely prevent users from contributing to the mean updates of clusters that will not serve them.

**Proposal 1:** The posterior probabilities obtained in (17) are replaced by probabilities that directly map the achieved SINR given by (7). Specifically, the vector of new probabilities for each user is obtained by

$$\boldsymbol{q'}^{(l)}(\cdot|n) = s(\boldsymbol{\Gamma}_n^{(l)}), \tag{19}$$

where different entries in this vector are associated with different DBSs and $\boldsymbol{\Gamma}_n^{(l)}$ is the vector of achieved SINR values (see (7)) for user $n$ with each DBS. That is,

$$\boldsymbol{\Gamma}_n^{(l)} = \left[\Gamma(1,n)^{(l)}, \Gamma(2,n)^{(l)}, .., \Gamma(M,n)^{(l)}\right]^T, \tag{20}$$

and $s(\boldsymbol{z})_j$ is the Softmax function defined as

$$s(\boldsymbol{z})_j = \frac{e^{Cz_j}}{\sum_{j'=1}^{M} e^{Cz_{j'}}}, \tag{21}$$

where $C$ is an arbitrary scaling constant. Using (19) for determining the posterior probabilities instead of (17) means that each user contributes to the parameters update of each DBS, $j$, with a magnitude proportional to how likely it is for the user to be served by this particular drone and not the others. The probability or magnitude of association between drone $j$ and user $n$ is directly proportional to the achieved SINR given by (7) relatively to the achieved SINR by other drones.

**Proposal 2:** We define a dynamic SINR threshold below which $q(j|n)$ will be set to zero if $\Gamma(j,n) < \Gamma_{\text{th}_n}^{(l)}$. This threshold is defined as

$$\Gamma_{\text{th}_n}^{(l)} = \max\left\{\boldsymbol{\Gamma}_n^{(l)}\right\} - S_{\text{th}}, \tag{22}$$

where the first term on the right-hand side denotes the highest achieved SINR between all drones in dB, and $S_{\text{th}}$ is the SINR dropout value that determines how fast the users are decoupled from non-serving drones. After obtaining the SINR threshold value from (22), the decoupling mask is defined as

$$\boldsymbol{M}_n^{(l)} = \left[H\left(\Gamma(1,n)^{(l)} - \Gamma_{\text{th}_n}^{(l)}\right), .., H\left(\Gamma(M,n)^{(l)} - \Gamma_{\text{th}_n}^{(l)}\right)\right]. \tag{23}$$

Finally, the normalized masked probabilities can be obtained using

$$\boldsymbol{q}^{(l)}(\cdot|n) = \frac{\boldsymbol{M}_n^{(l)T} \odot \boldsymbol{q'}^{(l)}(\cdot|n)}{||\boldsymbol{M}_n^{(l)T} \odot \boldsymbol{q'}^{(l)}(\cdot|n)||}, \tag{24}$$

where $\odot$ denotes the dot product operation and $||\cdot||$ denotes the L2 norm of a vector. The numerator discards DBSs with SINR below the threshold and the denominator normalizes the results so that their summation is equal to one.

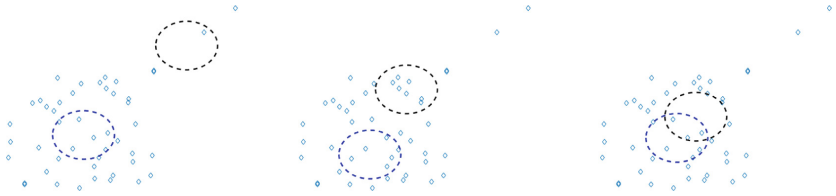
At every iteration after obtaining the probabilities defined in (24) the 2D locations of the DBSs can be updated using (18b) but with using $q^{(l)}(j|n)$ instead of $p^{(l)}(j|n)$. The pseudocode for SINR-based EM algorithm is shown in Algorithm 1 where the clusters locations are initialized using a set of unique initial locations $\mu_{\text{init}}$. Furthermore, the convergence is determined whenever an iteration results in a mean update less than some predefined threshold $\epsilon$ for all clusters.

**Algorithm 1.** SINR-Based EM Algorithm.

**Input:** $\mathcal{U}$, $M$, $\boldsymbol{P}_T$
**Output:** $\boldsymbol{\mu} = [\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, .., \boldsymbol{\mu}_M]$
*initialisation*: $\boldsymbol{\mu} = \boldsymbol{\mu}_{\text{init}}$
**repeat**
Calculate SINR-based probabilities using (24)
Obtain new DBSs 2D locations using (18b)
**until** convergence

### 4.3 On Outliers and the Choice of $S_{\text{th}}$

$S_{\text{th}}$ can be chosen such that DBSs will not be placed in sub-optimal locations where outliers might exist. By introducing $S_{\text{th}} > \epsilon$ for some $\epsilon > 0$ we allow the points that do not belong to the subset of outliers to *attract* cluster $j$ to some extent. Figure 2 illustrates the difference in results obtained from SINR-EM for the cases of $S_{\text{th}} = 0$, $S_{\text{th}} > 0$, and $S_{\text{th}} = \infty$ for a given users distribution. On the other hand, increasing $S_{\text{th}}$ indefinitely allows more users that are not served



(a) $S_{\text{th}} = 0$. (b) $S_{\text{th}} = 0.2 > 0$. (c) $S_{\text{th}} = \infty$.

**Fig. 2.** Comparison of SINR-EM resulting clusters overlap for different values of $S_{\text{th}}$.

by a cluster $j$ to contribute to the mean update of (18b) of $j$, and therefore results in an increased overlap between different clusters which in turn degrades the overall achieved link quality of the users due to interference (see Fig. 2c).

### 4.4 On Collision Between DBSs

Collision avoidance is an important aspect to consider in multi-UAV applications and it has received substantial attention in literature [8]. In the case of SINR-EM, and since all DBSs are flying at the same height, a collision can be modeled as the event when the 2D distance between any two DBSs is below a certain threshold. However, we know that by introducing the dynamic SINR threshold inside SINR-EM (see (22)) the clusters served by different DBSs are disjointed in the sense that users served by one DBS do not contribute to the location update of other DBSs as previously discussed.

**Lemma 1.** *For any two points $u_n$ and $u_t$ and given any two DBSs $j$ and $k$, if $\Gamma(j,n) > \Gamma(k,n)$ and $d_{k,t} > d_{j,n}$ then $\Gamma(j,t) > \Gamma(k,t)$.*

*Proof.* Since the heights of the DBSs are all equal and fixed, and $\eta_{\text{LOS}} < \eta_{\text{NLOS}}$ then $\Gamma(j,n)$ monotonically decreases as $d_{j,n}$ decreases for any $P_{T_j}$. This can be directly deduced from (2), (3), and (7).

**Theorem 1.** *Assuming $U >> M$, and at any SINR-EM iteration $l$, and for any two clusters $j$ and $k$, $k \neq j$, after obtaining the sets of points associated with each cluster defined as $\mathcal{U}_j^{(l)} = \{\boldsymbol{u}_n : q^{(l)}(j|n) > q^{(l)}(k|n)\}$ and $\mathcal{U}_k^{(l)} = \{\boldsymbol{u}_n : q^{(l)}(k|n) > q^{(l)}(j|n)\}$, there exists an $\epsilon$ such that for any $S_{th} \leq \epsilon$ the location update of (18b) will not result in $\boldsymbol{u}_j = \boldsymbol{u}_k$ in any further iteration.*

*Proof.* As a consequence of Lemma 1, and regardless of any other DBS, the 2D space is split into two regions $R_1 \supseteq \mathcal{U}_j^{(l)}$ and $R_2 \supseteq \mathcal{U}_k^{(l)}$. Furthermore, we can always find $S_{\text{th}} = \epsilon$ such that a subset of points in $\mathcal{U}_j^{(l)}$ discards cluster $k$ by applying the mask in (22). Therefore, given that the mean result in (18b) is the weighted average of the locations, then we are certain that $\boldsymbol{u}_j^{(l+1)} \neq \boldsymbol{u}_k^{(l+1)}$ when not all the points are included in the mean update of each cluster because of the 2D spatial distribution of the points in $\mathcal{U}_j^{(l)}$ and $\mathcal{U}_k^{(l)}$ and the proof is complete.

Given that clusters locations are initialized to distinct locations given by $\mu_{\text{init}}$ as mentioned previously, then Theorem 1 states that we can always choose $S_{\text{th}}$ to be small enough to avoid the case where SINR-EM results in equal locations for different clusters. In this paper, we ignore the case where two DBSs collide during their travel to the new locations proposed by TP-MCTS as this can be avoided through appropriate control by the central unit (e.g., each DBS can travel at a different height, or a collision avoidance algorithm could be implemented [21]). In Subsect. 6.2 we verify the aforementioned deductions by simulation.

## 5 Monte Carlo Tree Search

Monte Carlo Tree Search (MCTS) is a tree search algorithm. It tries to find optimal decisions in a given domain through sampling the decision space and building a search tree according to the results [22]. In its simplest form, MCTS incrementally builds an asymmetric tree where each node defines a unique decision at a given state. At each iteration the algorithm begins at the root node and progresses to a specific node through a series of selections that conform to a certain policy which balances between exploration of new states and exploitation of previous results. Next, the selected node is expanded by adding a child node to the tree. Starting from this child node a simulation (or playout) is run until a terminal state is reached. In the context of games this can be a *win* or *loss*. Finally, the result is back-propagated from the terminal state through its ancestors to the root node by updating the states of those nodes [22].

### 5.1 Transmission Power MCTS

An adapted MCTS algorithm is proposed to determine each DBS transmission power. The algorithm is denoted as Transmission Power Monte Carlo Tree Search (TP-MCTS). In TP-MCTS each node is uniquely defined by a set of transmission power settings for all DBSs. Starting from an initial setting of maximum power for all DBSs (i.e., $\boldsymbol{P}_T = [P_{T_1} = P_{\max}, .., P_{T_M} = P_{\max}]$) a root node is defined. A child node is created by decreasing the transmission power of one DBS by a predefined step of $P_\Delta$ (e.g., $\boldsymbol{P}_T = [P_{\max}, P_{\max} - P_\Delta, .., P_{\max}]$). For each selected node, SINR-EM is performed given the specified transmission powers, and two figures of merit are obtained as a result: the first is the number of served users, $N_{\text{SU},node}$, given by (13); and the second is the achieved energy efficiency, $\zeta_{node}$, given by (8). Both are used to obtain the score (reward) of each node. Let $N_{\text{th}} = \Phi U$ be the minimum required number of served users achieving an SINR value greater than or equal to $\Gamma_{\text{th}}$, where $\Phi$ is the desired percentage of served users (or link reliability). Then, the score of each node in the $i$-th iteration is obtained as

$$S^i_{node} = \begin{cases} \frac{N^i_{SU,node}}{U} & \text{if } N^i_{\text{SU,node}} < N_{\text{th}}. \\ \Phi + \frac{\zeta^i_{node}}{B_n} & \text{otherwise.} \end{cases} \tag{25}$$

If $N_{\text{th}}$ is achieved, then the score would also include a value proportional to the achieved energy efficiency, independently of the assigned bandwidth, in order to

search for solutions that further optimize the system's energy efficiency since the score guides the tree search as will be explained later below.

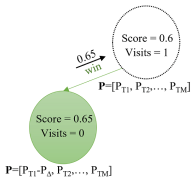
Figures 3a to 3d illustrate an example of TP-MCTS for four iterations. Each node is defined by the set of specified transmission powers below it, and its state contains the score and number of visits. Note that the number of visits is incremented only when a child of that node is visited through it, not when it is selected. For the sake of demonstration, we assume that the root node initially has a score of 0.6 obtained from the deployment results of SINR-EM with the initial maximum transmission power settings. The algorithm proceeds as follows:

- **Iteration 1**: The child node which reduces the transmission power of the first drone is selected (the selection criterion will be presented later). After performing SINR-EM, a result of 0.65 is achieved which is higher than the node's sole ancestor (i.e., the root) score so a *win* has occurred and the result is back-propagated upwards.
- **Iteration 2**: The root node score is updated with the resulting *win* from its child. When a *win* occurs, all of the winning node ancestors' scores are updated after receiving the score (see (25)) achieved by the winning child. The new scores are calculated according to the cumulative average

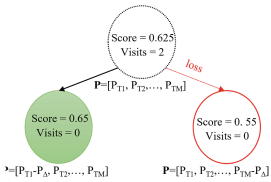
$$S_{node}^{i+1} = S_{node}^{i} + \left( \frac{S_{\text{win}}^{i} - S^{i}}{V_{node}^{i} + 1} \right), \tag{26}$$

  where $V_{node}^{i}$ is the visits count and $S_{\text{win}}^{i}$ is the winning child node's score. The next selected node reduces the transmission power of DBS $M$ and results in a score lower than its highest ancestor score (the root in this case) and therefore a *loss* occurs. The score is not back-propagated in the case of *loss*.
- **Iteration 3**: The child node with the reduced transmission power for the first DBS is selected again. Given that this node was selected before then it is not considered a *win* and therefore the algorithm continues. That is, a *win* or *loss* occur only when a node selected for the first time. Next, a further child node is created by again reducing the transmission power of the first drone ($j = 1$), and SINR-EM results in a score of 0.65. Since this score is equal to the highest ancestor score (0.65 belonging to the parent in this case) then again no *win* or *loss* occur. Finally, another child node is selected which reduces the transmission power of the second drone ($j = 2$) and the obtained score is 0.7, which is higher than 0.65, and therefore a *win* has occurred and the result is back propagated. The optimal state (shaded) is the one that has so far achieved the highest score during creation.
- **Iteration 4**: Assuming that the losing node of the second iteration is selected again, we observe that no loss occurs. Next, a further child node is created by an additional decrease of the last DBS power ($j = M$) and the achieved score is 0.58. Although the score is higher than its parent node, a loss occurs because the highest ancestor score is 0.644 belonging to the root node which is higher than the score achieved by this child.

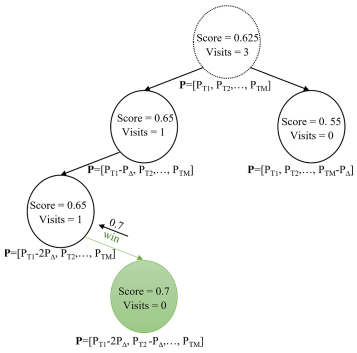
In TP-MCTS the selection, expansion, and simulation steps of the standard MCTS are lumped together, and at each iteration, the algorithm progresses by



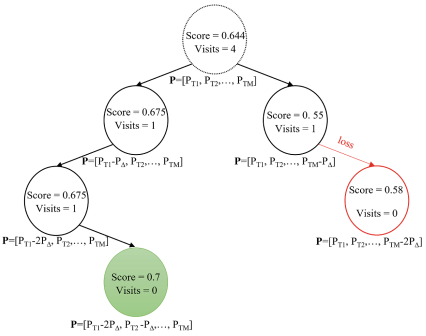

(a) iteration 1: first win occurred after reducing $P_{T_1}$.




(b) iteration 2: reducing $P_{T_M}$ results in a loss.




(c) iteration 3: reducing $P_{T_2}$ after reducing $P_{T_1}$ for the second time caused a win.




(d) iteration 4: the second loss occurred after reducing $P_{T_M}$ again.

**Fig. 3.** MCTS example. Note how the ancestors scores are updated after each iteration.

selecting child nodes until reaching a terminal node which is selected for the first time such that it terminates with a *win* or *loss* if the achieved score of the considered node is higher or lower respectively than the highest score in the traversed nodes. The score is back-propagated only in the case of *win*.

### 5.2 Selection Criterion

To select the next node, the scores of all possible child nodes are taken into consideration, while the missing scores of the nodes that are not yet created are substituted by the parent score (i.e. current). We select the child node with the maximum upper confidence bound (UCB) value given by [23]

$$\mathrm{UCB}_{child} = S_{child} + K\sqrt{\frac{\ln(visits_{parent})}{visits_{child}+1}}, \tag{27}$$

where $K$ is an arbitrary constant controlling the level of exploration.

Algorithm 2 shows the object-oriented programming (OOP) pseudocode for TP-MCTS which is implemented recursively. Each node is an object (line 7) which can be simulated by invoking the SIMULATE method (line 12). Each child receives its ancestor's highest score to check for a *win* or *loss* after performing SINR-EM with the given transmission power $\boldsymbol{P}_T$ and set of users $\mathcal{U}$. Starting

with the *root* node, and for a given number of max iterations ($iteration_{\max}$), the SIMULATE method is invoked. On the first call to each node's SIMULATE method, we obtain the score after performing SINR-EM and we compare it to the global highest score to keep track of the optimal simulated state (line 16). Then we instantiate each possible child of the current node by decreasing the transmission power of a uniquely selected drone and the score of each child is initiated to the parent achieved score as mentioned previously. Then, we compare the achieved score to the highest ancestor score, $S_{ancestor}$, which is the highest score between the nodes on the path from the parent of the current node to the root. In the case of a *win* (line 26) we return the achieved score; in the case of a *loss* we return 0. If it is not the first simulation of the node or no *win* or *loss* occurred we proceed by selecting a child of the current node according to the UCB policy (see (27)) and invoking its SIMULATE method. Whenever a positive score resulting from a win is back-propagated from the child simulation the parent node updates its score according to (26).

### 5.3 Time-Based Updates

Assuming that DBSs are deployed to serve non-stationary users, the mobility of drones can be utilized to adapt to the movement of users and periodically update the locations of DBSs so that the overall performance of the system is improved. Instead of initializing the TP-MCTS tree at each update interval and performing a substantial number of iterations to approach the optimal solution, we can make use of the experience embedded in the current tree structure. To proceed further we make the following assumption: after a duration of $t_{\text{update}}$, the new optimal solution lies in the vicinity of the previously found solution. That is if $t_{\text{update}}$ is small enough relative to the speed of movement of users, then the existing branches weights (scores) can still be utilized to guide the search to the next optimal solution, and last winning node can possibly serve as a good starting point for the next search. Accordingly, after each $t_{\text{update}}$ we reset the *simulated* attribute of each node so that when a previously simulated node is selected, EM-SINR will be performed again for the current users' distribution and the node's score is updated accordingly. The *visits* count is also reset for all nodes. Furthermore, consider that the previously winning node is given by $node_{\text{win}}$, then after simulating this node and the *root* node again we compare

**Algorithm 2.** Transmission Power Monte Carlo Tree Search.

**Input:** $\mathcal{U}$, $M$, $P_{\max}$, $P_{\Delta}$,
**Output:** $\boldsymbol{P}_T$, $\boldsymbol{\mu} = [\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, .., \boldsymbol{\mu}_M]$
1: $root \leftarrow \text{NODE}([P_{\max}, .., P_{\max}], 0)$
2: $S_{max} \leftarrow 0$
3: $\boldsymbol{P_{best}} \leftarrow [P_{\max}, .., P_{\max}]$
4: **for** $iteration < iteration_{max}$ **do**
5: $\quad root.\text{SIMULATE}(0, \mathcal{U})$
6: **end for**

```
object NODE(P_T, S_parent)
    visits ← 0
    simulated ← false
    childs ← { }
    S ← S_parent
    function SIMULATE(S_ancestor, U)
        if simulated = false then
            perform SINR-EM (see Algorithm 1).
            S = obtain node score according to (25).
            if S > S_max then
                S_max ← S
                P_best ← P_T
            end if
            for i ← 1, M do
                P_{T_child} ← P_T
                P_{T_child,i} ← P_{T_child,i} − P_Δ
                childs ← {childs, NODE(P_{T_child}, S)}
            end for
            simulated ← true
            if S > S_ancestor then return S                    ▷ win occurred
            else if S < S_ancestor then return 0               ▷ loss occurred
            end if
        end if
        visits ← visits + 1
        child ← child with highest UCB value obtained according to (27)
        result ← child.SIMULATE(max{S, S_parent}, U)
        if result > 0 then update score according to (26)      ▷ update if win
        end if
        return result
    end function
end object
```

the achieved scores and select the node with the highest score as the current root to search from. We call this *best root selection* (BRS) (see Algorithm 3).

## 6 Simulation and Discussions

In this work we perform two sets of simulations each against a different baseline method and with different environment settings:

1. **Scenario 1:** Through extensive simulations, we compare the results achieved by TP-MCTS to the results of a popular PSO-based solution mentioned in [6] for uniformly distributed users.
2. **Scenario 2:** Using the random way-point mobility model (RWP) [7], we simulate TP-MCTS against the baseline settings in which DBSs are randomly distributed. On the other hand, TP-MCTS updates the location of drones every $t_{\text{update}}$ seconds in accordance with Subsect. 5.3.

**Algorithm 3.** TP-MCTS with best root selection.

**Input:** $\mathcal{U}$, $M$, $P_{\max}$, $P_{\Delta}$,
**Output:** $\boldsymbol{P}_T$, $\boldsymbol{\mu} = [\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, .., \boldsymbol{\mu}_M]$

Perform TP-MCTS as defined in Algorithm 2.
At every $t = Nt_{\text{update}}$ s.t. $N \in \{1, 2, 3..\}$
    Set $visits \leftarrow 0$, $simulated \leftarrow false$ for all nodes.
    $node_{\text{win}} \leftarrow$ the node with highest score so far.
    $root$.SIMULATE$(0, \mathcal{U})$
    $node_{\text{win}}$.SIMULATE$(0, \mathcal{U})$
    **if** $node_{\text{win}}.score > root.score$ **then**
        **for** $iteration < iteration_{\max}$ **do**
            $node_{\text{win}}$.SIMULATE$(0, \mathcal{U})$
        **end for**
    **else**
        **for** $iteration < iteration_{max}$ **do**
            $root$.SIMULATE$(0, \mathcal{U})$
        **end for**
    **end if**

In both cases, we discuss the results and deduce insights into the trade-off between different deployment parameters such as transmission power settings, users' quality of service, number of DBSs, and the achieved energy efficiency. Also, in both cases, given the required rate, $R$, for users and the capacity of DBS, $C_{\text{BS}} = B_D \times \gamma$ where $\gamma$ is the system average spectral efficiency, the maximum number of users per DBS can be obtained as

$$U_{max} = \left\lfloor \frac{C_{\text{BS}}}{R} \right\rfloor. \tag{28}$$

### 6.1 Scenario 1 (PSO)

**Simulation Settings (PSO).** In this work we compare the results of the aforementioned PSO solution to our TP-MCTS algorithm assuming an area of 1 km $\times$ 1 km with 500 users. We simulate using different numbers of serving DBSs, beginning with the minimum number estimated by dividing the total number of users by the maximum number of users per DBS obtained in (28). Also, we vary the SINR requirement of $\Gamma_{\text{th}}$. The maximum number of iterations is limited to $I = 200$. Table 1 lists the simulation parameters. The PSO-related parameters are listed in Table 1(c) using the same notations as in [6]. $\phi$ (also called the inertia weight) is set as a linearly decreasing weight (LDW) [24] obtained as $\phi = \phi_{\max} - \frac{\phi_{\max} - \phi_{\min}}{I} i$, where $i$ is the current iteration count, and $\phi_1$ and $\phi_2$ are randomly drawn uniformly from the specified ranges.

**Table 1.** (a) System parameters, (b) TP-MCTS parameters, (C) PSO parameters.

(a)

| Parameter | Value |
|---|---|
| Number of users U | 500 |
| Path loss parameters $\alpha$, $\beta$, $\eta_{\mathrm{LOS}}$, and $\eta_{\mathrm{NLOS}}$ | 9.61, 0.16, 1 dB, and 20 dB |
| Carrier frequency $f_c$ and drone bandwidth $B_D$ | 2 GHz and 20 MHz |
| Noise power $\sigma_0^2$, required rate $R$, and spectral efficiency $\gamma$ | -110 dBm, 1 Mbps, and 2 bps/Hz |
| Link reliability $\Phi$ and SINR threshold $\Gamma_{\mathrm{th}}$ | 0.95 and -2/ 0 dB |

(b)

| Parameter | Value |
|---|---|
| Softmax scale C and dropout SINR difference $S_{\mathrm{th}}$ | 4 and 0.5 |
| Drone height $h$ and exploration constant $K$ | 70 and 0.01 |
| Power step $P_{\Delta}$ and maximum power $P_{\max}$ | 100 mW and 500 mW |

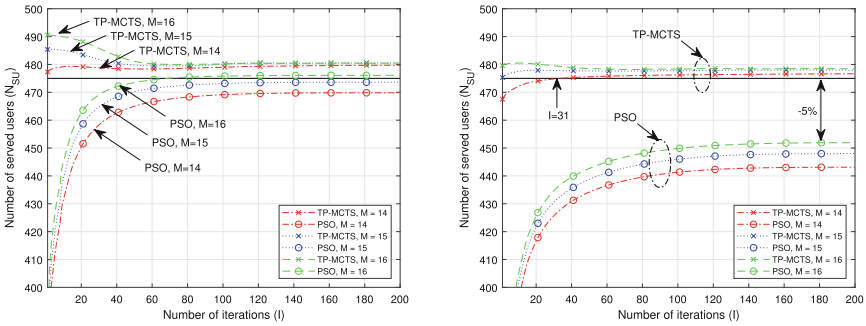
(c)

| Parameter | Value |
|---|---|
| $\phi_{\min}$, $\phi_{\max}$, $c_1$, and $c_2$ | 0.9, 0.95, 0.095, and 0.15 |
| $\phi_1$ and $\phi_2$ | $\in$ [0.8 1.8] |

**Number of Served Users ($N_{\mathrm{SU}}$).** Figure 4 shows the value of $N_{\mathrm{SU}}$ per iteration achieved by each algorithm for different numbers of serving DBSs, $M$, and two different conditions of $\Gamma_{\mathrm{th}}$. In Fig. 4a it can be observed that for $M = 14$ and $M = 15$ the PSO solution was, on average, not able to provide the required $N_{\mathrm{th}} = 475$ (indicated by the straight line). On the other hand, it appears that clustering users using EM-SINR without power control did already satisfy the requirement and TP-MCTS changes the transmission power of DBSs in such way that $N_{\mathrm{SU}}$ is rather decreased while optimizing for the achieved energy efficiency $\zeta$ (see (25)). Additionally, Fig. 4b plots the results given $\Gamma_{\mathrm{th}} = 0$ dB. Evidently, the PSO algorithm was far from finding a solution that satisfies the requirements whereas TP-MCTS was able to find a solution for $M = 14$, and EM-SINR had already satisfied the requirements in the cases of $M = 15$ and $M = 16$.

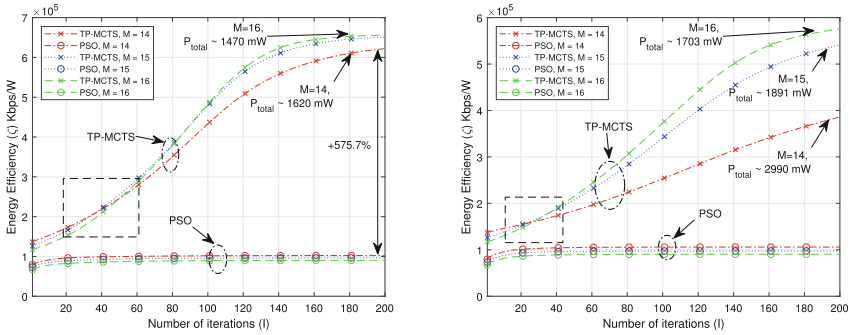
**Energy Efficiency ($\zeta$).** For $\Gamma_{\mathrm{th}} = -2$ dB, Fig. 5a shows that TP-MCTS is able to substantially improve the energy efficiency by around 575%. Furthermore, increasing the number of DBSs from $M = 14$ to $M = 16$ decreases the total required transmission power which is expected when cells are more dense. We observe that a higher number of DBSs initially results in degraded system performance in terms of $\zeta$ but later improves further than the cases with less DBSs as TP-MCTS optimizes the transmission powers.

**Transmission Power ($P_T$).** The average transmission power for all DBSs per iteration is plotted in Fig. 6. For the case when $\Gamma_{\mathrm{th}} = -2$ dB, TP-MCTS seem to substantially decrease the transmission power of all DBS to almost 100 mW. This improves the interference levels between DBSs as can be deduced from




(a) $N_{\mathrm{SU}}$ per iteration for $\Gamma_{\mathrm{th}} = -2$ dB, $N_{\mathrm{th}} = 475$ (solid line). (b) $N_{\mathrm{SU}}$ per iteration for $\Gamma_{\mathrm{th}} = 0$ dB, $N_{\mathrm{th}} = 475$ (solid line).

**Fig. 4.** TP-MCTS vs. PSO by $N_{\mathrm{SU}}$. TP-MCTS always achieves the required $\Gamma_{\mathrm{th}}$ and substantially outperforms the PSO solution.



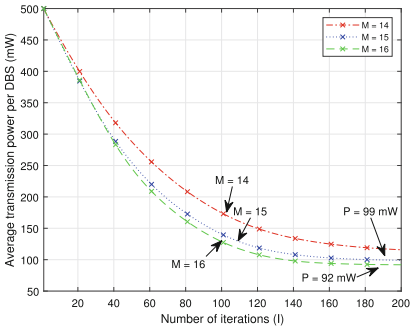

(a) $\zeta$ per iteration for $\Gamma_{\mathrm{th}} = -2$ dB. (b) $\zeta$ per iteration for $\Gamma_{\mathrm{th}} = 0$ dB.

**Fig. 5.** TP-MCTS vs. PSO by $\zeta$.

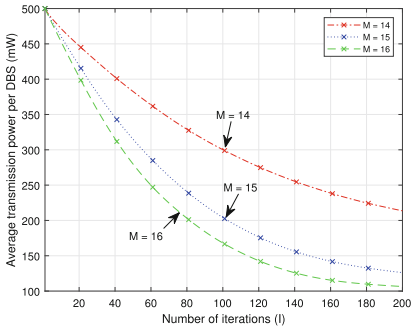
previous plots. By examining the results when $M = 16$, we observe that after a substantial number of iterations the average power per DBS is below 100 mW. Since $P_{\Delta} = 100$ mW (see Table 1) this indicates that TP-MCTS results in completely turning off a DBS by setting its transmission power to zero.

### 6.2 Scenario 2 (RWP)

**Simulation Settings.** We simulated the proposed TP-MCTS solution for a moving set of users generated according to the RWP model [7] against the baseline scenario where DBSs are distributed randomly. 200 moving users were considered in an area of $0.5 \times 0.5$ Km$^2$ that are served using 5 or 6 DBSs. TP-MCTS is run every $t_{\mathrm{update}} = 5$ seconds for 20 iterations (see Subsect. 5.3). 1000 trials are run and the duration of each simulation trial is 120 seconds (i.e., 120 s of network functioning in real life). Table 2 lists the simulation parameters.




(a) Average $P_T$ for $\Gamma_{\text{th}} = -2$ dB.




(b) Average $P_T$ for $\Gamma_{\text{th}} = 0$ dB.

**Fig. 6.** Average transmission power per DBS. Increasing the number of DBSs allows less transmission power per DBS.

**Table 2.** (a) System parameters, (b) TP-MCTS parameters.

(a)

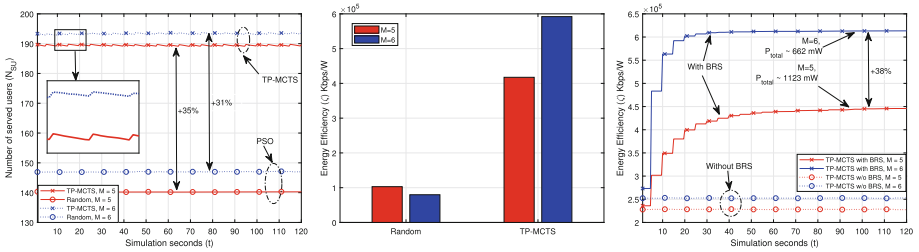
| Parameter | Value |
|---|---|
| Number of users U | 200 |
| Path loss parameters $\alpha$, $\beta$, $\eta_{\text{LOS}}$, and $\eta_{\text{NLOS}}$ | 9.61, 0.16, 1 dB, and 20 dB |
| Carrier frequency $f_c$ and drone bandwidth $B_D$ | 2 GHz and 20 MHz |
| Noise power $\sigma_0^2$, required rate $R$, and spectral efficiency $\gamma$ | -110 dBm, 1 Mbps, and 2 bps/Hz |
| Link reliability $\Phi$ and SINR threshold $\Gamma_{\text{th}}$ | 0.95 and 0 dB |

(b)

| Parameter | Value |
|---|---|
| Softmax scale C and dropout SINR difference $S_{\text{th}}$ | 4 and 0.5 |
| Drone height $h$ and exploration constant $K$ | 70 and 0.01 |
| Power step $P_\Delta$ and maximum power $P_{\max}$ | 100 mW and 500 mW |
| $t_{\text{update}}$ and number of iterations | 5 s and 20 |

**Number of Served Users ($N_{\text{SU}}$).** Figure 7a shows the achieved $N_{\text{SU}}$ for 5 and 6 serving DBSs. The random approach achieves a constant performance on average whereas TP-MCTS performance seems to decay between update intervals, and increases up to a maximum value after every update. TP-MCTS seems to also preserve a steady performance which means that the tree is indeed able to adapt to the new users' locations and find new solutions. Figure 7b shows the average achieved $\zeta$ for 5 and 6 serving DBSs along with using the random distribution approach versus TP-MCTS. Clearly, when adding an extra DBS the achieved energy efficiency decreases, but in TP-MCTS it can be intelligently deployed to improve the performance of the system. Figure 7c plots $\zeta$ per second

obtained using TP-MCTS with and without best root selection. The improvement resulting from using BRS is evident. The lack of such improvement in the case of no BRS is expected for a low iterations number of 20 since the tree search would never reach deep states that sufficiently reduce the transmission powers of DBSs.




(a) $N_{\mathrm{SU}}$ per second. (b) Average $\zeta$. (c) $\zeta$ per second.

**Fig. 7.** Results for the second simulation scenario.

**Collision Avoidance.** In support of the arguments mentioned in Subsect. 4.4 we measured the minimum distance that occurred between any DBSs during all trials. In the case of $M = 5$, the minimum distance recorded was 119.5 $m$, whereas for $M = 6$ the minimum distance was 107 $m$ (see Fig. 8).

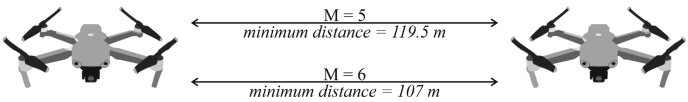



**Fig. 8.** Minimum distance between any two DBSs during simulation.

## 7 Conclusions

We showed that TP-MCTS outperformed the other baseline methods in terms of link reliability and energy efficiency. TP-MCTS is flexible and can optimize for whatever goals we set by defining the score appropriately. For example, the score given by (25) can be modified to include the system spectral efficiency instead of $\zeta$ depending on the design requirements. Also, simulations showed that TP-MCTS can remove redundant DBSs by setting their transmission power to zero. Furthermore, our results proved that intelligent densification of cells can improve the system performance if properly managed. Finally, we showed that TP-MCTS can utilize its tree history to guide the search towards more promising nodes.

## References


1. Mozaffari, M., Saad, W., Bennis, M., Nam, Y.-H., Debbah, M.: A tutorial on UAVs for wireless networks: applications, challenges, and open problems. IEEE Commun. Surv. Tutor. **21**(3), 2334–2360 (2019)
2. Zeng, Y., Zhang, R., Lim, T.J.: Wireless communications with unmanned aerial vehicles: opportunities and challenges. IEEE Commun. Mag. **54**(5), 36–42 (2016)
3. Osseiran, A., et al.: Scenarios for 5G mobile and wireless communications: the vision of the metis project. IEEE Commun. Mag. **52**(5), 26–35 (2014)
4. Liu, X., Ansari, N.: Resource allocation in UAV-assisted M2M communications for disaster rescue. IEEE Wirel. Commun. Lett. **8**(2), 580–583 (2019)
5. Gu, Z., Zhang, J., Sun, X., Ji, Y.: Optimizing networked flying platform deployment and access point association in FSO-based fronthaul networks. IEEE Wirel. Commun. Lett. **9**(8), 1221–1225 (2020)
6. Kalantari, E., Yanikomeroglu, H., Yongacoglu, A.: On the number and 3D placement of drone base stations in wireless cellular networks. In: 2016 IEEE 84th Vehicular Technology Conference (VTC-Fall), pp. 1–6 (2016)
7. Bettstetter, C., Resta, G., Santi, P.: The node distribution of the random waypoint mobility model for wireless ad hoc networks. IEEE Trans. Mob. Comput. **2**(3), 257–269 (2003)
8. Hayat, S., Yanmaz, E., Muzaffar, R.: Survey on unmanned aerial vehicle networks for civil applications: a communications viewpoint. IEEE Commun. Surv. Tutor. **18**(4), 2624–2661 (2016)
9. Fotouhi, A., Qiang, H., Ding, M., Hassan, M., Giordano, L.G., Garcia-Rodriguez, A., Yuan, J.: Survey on UAV cellular communications: practical aspects, standardization advancements, regulation, and security challenges. IEEE Commun. Surv. Tutor. **21**(4), 3417–3442 (2019)
10. Cicek, C.T., Gultekin, H., Tavli, B., Yanikomeroglu, H.: UAV base station location optimization for next generation wireless networks: overview and future research directions. In: 2019 1st International Conference on Unmanned Vehicle Systems-Oman (UVS), pp. 1–6 (2019)
11. Alzenad, M., El-Keyi, A., Lagum, F., Yanikomeroglu, H.: 3-D placement of an unmanned aerial vehicle base station (UAV-BS) for energy-efficient maximal coverage. IEEE Wirel. Commun. Lett. **6**(4), 434–437 (2017)
12. Alzenad, M., El-Keyi, A., Yanikomeroglu, H.: 3-D placement of an unmanned aerial vehicle base station for maximum coverage of users with different QoS requirements. IEEE Wirel. Commun. Lett. **7**(1), 38–41 (2018)
13. Al-Hourani, A., Kandeepan, S., Lardner, S.: Optimal lap altitude for maximum coverage. IEEE Wirel. Commun. Lett. **3**(6), 569–572 (2014)
14. Kalantari, E., Shakir, M.Z., Yanikomeroglu, H., Yongacoglu, A.: Backhaul-aware robust 3D drone placement in 5G+ wireless networks. In: IEEE International Conference on Communications Workshops (ICC Workshops) 2017, pp. 109–114 (2017)
15. Lyu, J., Zeng, Y., Zhang, R., Lim, T.J.: Placement optimization of UAV-mounted mobile base stations. IEEE Commun. Lett. **21**(3), 604–607 (2017)
16. Abeywickrama, H.V., He, Y., Dutkiewicz, E., Jayawickrama, B.A., Mueck, M.: A reinforcement learning approach for fair user coverage using UAV mounted base stations under energy constraints. IEEE Open J. Veh. Technol. **1**, 67–81 (2020)
17. Qiu, J., Lyu, J., Fu, L.: Placement optimization of aerial base stations with deep reinforcement learning. In: ICC 2020–2020 IEEE International Conference on Communications (ICC), pp. 1–6 (2020)

18. Liu, C.H., Chen, Z., Tang, J., Xu, J., Piao, C.: Energy-efficient UAV control for effective and fair communication coverage: a deep reinforcement learning approach. IEEE J. Sel. Areas Commun. **36**(9), 2059–2070 (2018)
19. Huang, H., Savkin, A.V.: A method for optimized deployment of unmanned aerial vehicles for maximum coverage and minimum interference in cellular networks. IEEE Trans. Industr. Inf. **15**(5), 2638–2647 (2019)
20. Singh, R., Jaakkola, T., Mohammad, A.: 6.867 machine learning (2006). MIT OpenCourseWare. https://ocw.mit.edu
21. Abeywickrama, H.V., Jayawickrama, B.A., He, Y., Dutkiewicz, E.: Algorithm for energy efficient inter-UAV collision avoidance. In: 2017 17th International Symposium on Communications and Information Technologies (ISCIT), pp. 1–5 (2017)
22. Browne, C.B., et al.: A survey of Monte Carlo tree search methods. IEEE Trans. Comput. Intell. AI Games **4**(1), 1–43 (2012)
23. Auer, P.: Using upper confidence bounds for online learning. In: Proceedings 41st Annual Symposium on Foundations of Computer Science, pp. 270–279 (2000)
24. He, Y., Ma, W., Zhang, J.: The parameters selection of PSO algorithm influencing on performance of fault diagnosis. In: MATEC Web of Conferences, vol. 63, p. 02019 (2016)