2020, Graduate Thesis

Unmanned Aircraft Traffic Deconfliction Using Greedy Coloring

Nol Peeravatanachart

(International Course)

Flight Dynamics Laboratory
Course of Aeronautics and Astronautics
Department of Mechanical and Aerospace Engineering
School of Engineering
Kyushu University
Supervisor A. Prof. Shin-Ichiro Higashino

Abstract

The objective of an unmanned traffic management (UTM) is to optimize the capacity of the high-density low-level uncontrolled airspace. One such method for optimization is to assign different vertical layers of altitude for unmanned aircraft systems (UAS) flight operations that are in conflict.

This study aims to analyze the deconfliction of unmanned aircraft traffic management at the high-density low-level uncontrolled airspace by utilizing the vertical layer assignment method. This was done by modeling the airspace and generating random UAS flight operations and applying the greedy coloring algorithm to resolve conflicts between operations.

60 simulations were performed by varying the independent variables number of operations from 1000 to 10 000, and conflict distance from 50 m to 300 m. The effects on average number of conflicts per operation, number of layers needed, and the distribution of operations among these layers were analyzed. We compare the cases for the single UTM architecture where operations are managed by one with one UTM service provider (UTMSP), to the federated UTM architecture where operations are split among multiple UTMSPs based on flight headings.

In the case of multiple UTMSPs, the idea of a shared layer to combine all layers with number of operations less than a 1% cut-off percentage of the total number of operations in each service provider was proposed to reduce the extensive use of layers by a low percentage of operations.

TABLE OF CONTENTS

Abstract 2

List of Figures 4

List of Tables 5

1 Introduction 7

2 Method 11

2.1 Simulation models in previous studies 11

2.1.1 Probabilistic models for flight operations 11

2.1.2 Deconfliction models 11

2.2 Simulation model 12

2.2.1 Airspace 12

2.2.2 Flight operations 13

2.2.3 Splitting operations based on flight headings 16

2.2.4 Definition of a conflict 17

2.3 Modeling Simulations as Networks 20

2.4 Metrics 26

3 Results 28

3.1 Average number of conflicts per operation 28

3.2 Number of layers needed 32

3.3 Comparison between average number of conflicts per operation and number of layers needed 36

3.4 Distribution of operations among layers 38

3.5 Number of layers needed adjusted to 1% cut-off point 44

4 Conclusion 48

References 49

List of Figures

Figure 1. 1 Two UTM architectures 8

Figure 2. 1 Model representing a 10 000 m by 10 000 m by 150 m airspace 13
Figure 2. 2 A single UAS flight operation..... 14
Figure 2. 3 Model of conflict between UAS i and UAS j 17
Figure 2. 4 Modeling simulations as networks..... 20
Figure 2. 5 Optimal coloring of a simple network 24
Figure 2. 6 Largest degree first vertex ordering strategy prioritizing vertices with large degrees 25

Figure 3. 1 Average number of conflicts per operation as a function of n and R. 30
Figure 3. 2 Average numbers of conflicts per operation as a function of n and R for all UTM architectures..... 31
Figure 3. 3 Number of layers needed as a function of n and R. 34
Figure 3. 4 Number of layers needed as a function of n and R for all UTM architectures 35
Figure 3. 5 Average number of conflicts per operation and number of layers needed for all UTM architectures..... 36
Figure 3. 6 Distribution of operations among layers as a function of n (R = 50 m) 40
Figure 3. 7 Distribution of operations among layers as a function of n (R = 300 m) 42
Figure 3. 8 Distribution of operations among layers as a function of n (R = 300 m) for the federated (4 UTMSPs) UTM architecture, adjusted to 1% cut-off point 44
Figure 3. 9 Average number of conflicts per operation and number of layers needed for all UTM architectures Adjusted to 1% cut-off point..... 45

List of Tables

Table 2. 1 Heading range for each UTMSP for all UTM architectures 16
Table 2. 2 List of UTM components of interest and its network counterparts..... 21

Table 3. 1 m and c for the line of best fit for average number of conflicts per operation and number of layers 37
Table 3. 2 Distribution of operations among layers 43
Table 3. 3 m and c for the line of best fit for average number of conflicts per operation and number of layers, adjusted to 1% cut-off point 46

Chapter 1
Introduction

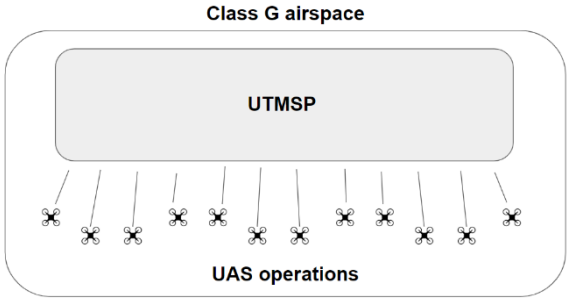
1 Introduction

Recently, one of the industries that have been gaining interest in aviation is the unmanned aircraft systems (UAS). In the past, UAS applications were mostly restricted to military purposes. But now with millions of commercial UAS registered worldwide, a large amount of research and development is being conducted to realize the commercial and leisure applications of UAS, especially in uncontrolled low altitude airspace at or below 500 feet above ground level. These applications include geofencing, infrastructure monitoring, package delivery, and disaster management. The expected number of the UAS fleet will grow rapidly, reaching 2 to 3 million in 2023 in the United States alone [1]. This large amount of UAS operations will lead to an increased usage of the uncontrolled low-level airspace beyond its current capacity as conflicts happen too frequently. Therefore, it is necessary to develop new deconfliction methods. This paper focuses on assigning different vertical layers of altitude for flight operations that are in conflict.

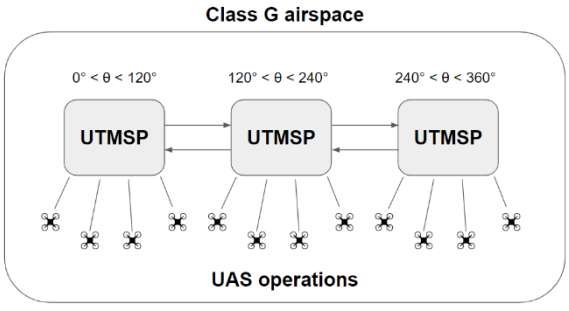
Currently, the unmanned traffic management (UTM) architecture relies solely on the UAS operators for deconfliction. In Japan, UAS flight operations are to follow the unmanned vehicle section of the Japanese Civil Aviation's Bureau's (JCAB) Aeronautical Act that was enforced since December 2015 [2]. The Aeronautical Act provides basic operational limitations such as weight restrictions and the 30m minimum distance between UAS and other structures. All flights are to be conducted within visual line-of-sight (VLOS) under 150m (500ft) from ground level. Requests for operations beyond the limitations of the Aeronautical Act such as beyond visual line-of-light (BVLOS) and night operations will have to be reviewed on a case-by-case basis. Because there are no specific flight rules to follow, the UAS operators are solely responsible for the deconfliction between their own UAS and other nearby UAS on a small local scale. The current law is effective because the current uncontrolled low-level airspace has extremely low traffic density. Once traffic density increases, an architecture which can support mass automation on managing and deconfliction will have to be utilized.

In order to realize the automation of high volumes of UAS operations, the concept of a UTM Service Provider (UTMSP) was introduced. A UTMSP should be able to provide services such as UAS registration, airspace management based on flight plans, and deconfliction. All UAS operators submit flight plans and the UTMSP is considered to have perfect information on all flight plans it is responsible for [3].

This paper aims to explore deconfliction in two architectures. A single architecture where a single UTMSP is responsible for all operations as shown in Figure 1.1(a), and a federated architecture where multiple UTMSPs split the management of all operations among themselves based on the heading of the operations as shown in Fig 1.1(b). By splitting operations to different UTMSPs based on their headings, we expect the overall number of conflicts between flight operations to decrease.



(a) Single architecture



(b) Federated (3 UTMSPs) architecture

Figure 1. 1 Two UTM architectures

(where theta represents the range of flight heading for each UTMSP)

Python was used as a tool to make simulations. Firstly, the airspace was modeled, and flight operations were generated in the airspace. Once the UAS operations were generated, the operations were assigned to a UTMSP based on their headings. Conflicts between pairs of operations were checked and a deconfliction method was applied to resolve all conflicts. This paper considers the assignment of vertical layers as the deconfliction method, and this was done through the greedy coloring algorithm. The metrics used for the analysis are the average number of conflicts per operation and the number of layers needed.

Overall, this paper aims to compare the single unmanned traffic management (UTM) architecture to the federated UTM architecture for unmanned aircraft systems (UAS) in high density low level uncontrolled airspace. Vertical layer assignment was applied as a deconfliction method. By treating the UAS operations in the airspace as a graph coloring problem, the layers were assigned based on the greedy coloring algorithm. The effectiveness of the federated UTM architectures will be shown by comparing the simulation results.

Chapter 2

Method

2 Method

2.1 Simulation models in previous studies

2.1.1 Probabilistic models for flight operations

Many models have been developed for the purpose of analyzing UAS operations in an airspace. The earliest model for simulating aircraft deconfliction that can be applied to UAS is a probabilistic setup proposed by Hoekstra [4], developed by Jardin [5], and further explored within the Metropolis project by TU Delft [6]. In this model later known as the Dutch model [7], the aircraft's starting positions are chosen uniformly at random in the airspace. All aircraft fly at the same speed and their headings are uniformly distributed. The model assumes a two-dimensional single layer of traffic.

Because the uniformly distributed positions may not reflect the real positions of UAS traffic, Bulusu and other researchers at UC Berkeley proposed the Cal model [7], lowering the scope to lower altitude airspace. The model expands upon the Dutch model by utilizing a population density model. Instead of uniformly distributed positions and headings, the flight endpoints are sampled in accordance to the population density. Like the Dutch model, this model also assumes a two-dimensional single layer of traffic and utilizes vertical maneuvers for deconfliction.

2.1.2 Deconfliction models

Deconfliction methods can be divided into three categories, based upon the domain in which the deconfliction happens. These are the horizontal plane domain (xy-plane), altitude domain (z-axis) and the time domain (t-axis). Unmanned aircraft are expected to rely on horizontal maneuvers more so than manned aircraft [8], whose all collision avoidance systems are vertical maneuvers, due to limited airspace size in the vertical direction. As for the time domain, similarly to manned aircraft, ground delay schemes could be utilized like models developed by Barnier [9]. Moreover, for UAS with

hovering capabilities, hovering mid-air to allow an approaching UAS to transit through is also a valid strategy [10].

This paper focuses on the studies that utilize the altitude domain, in particular, the idea of vertical layers where the airspace is split into different layers that are stacked on top of each other.

Sunil suggested that a layer should be restricted based on flight headings [11]. It was shown that the overall number of conflicts between the UAS operations are reduced by splitting into more layers based on flight headings in the following table [12].

Sedov suggested dividing the low-level airspace into conflict-free layers using graph coloring techniques [10]. By utilizing the graph coloring algorithm, the model aims to minimize the number of layers needed. The algorithm used does not account for restrictions on the maximum number of layers or flight heading.

In this paper, a similar setup to Sedov's multi-layer design's vertical layer assignment was used. This paper's main contribution is the application of the greedy coloring algorithm to both the single and federated UTM architecture.
To demonstrate how the vertical layer assignment algorithm can be applied to UAS flight operations in a high-density low-level airspace, simulations made in Python were built and analyzed.

2.2 Simulation model

2.2.1 Airspace

Firstly, a proper airspace in which the UAS flight operations will take place was modeled. The airspace was modeled as an object in Python that stores information of all current active flight operations. For every time tick, it updates and makes a new instance

of itself. In the simulations, a time tick of 1 s was used. The time frame for the simulations is 10 000 s. The airspace is assumed to be an empty cuboid space with no terrain or obstructions. Following the Japanese Civil Aviation's Bureau's (JCAB) Aeronautical Act where operations must be conducted under 150m from ground level, the z dimension in the vertical direction is set to 150m. As for the xy dimensions in the horizontal plane, the value is set to 10 000 m as shown in Figure 2.1.

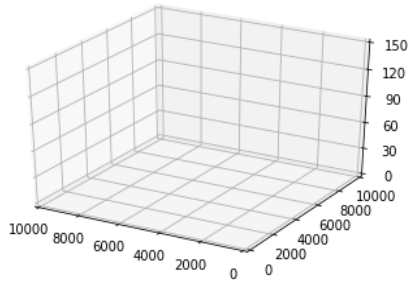


Figure 2. 1 Model representing a 10 000 m by 10 000 m by 150 m airspace

2.2.2 Flight operations

In this paper, all UAS have hovering capabilities and travel at a constant cruise speed of 20 m/s. This value for speed was picked because it serves as a good scope for the maximum velocities of most commercial drones that are being used currently. All flight operation paths are that of a linear flight from the takeoff point to the landing point by utilizing a single cruising altitude z without any altitude changes as shown in Figure 2.2. Each operation consists of four points, namely, P_T, P_A, P_B, P_L.

P_T: Takeoff point at (x_T, y_T, 0)

P_A: Linear starting point at (x_T, y_T, z)

P_B: Linear ending point at (x_L, y_L, z)

P_L: Landing point at (x_L, y_L, 0)

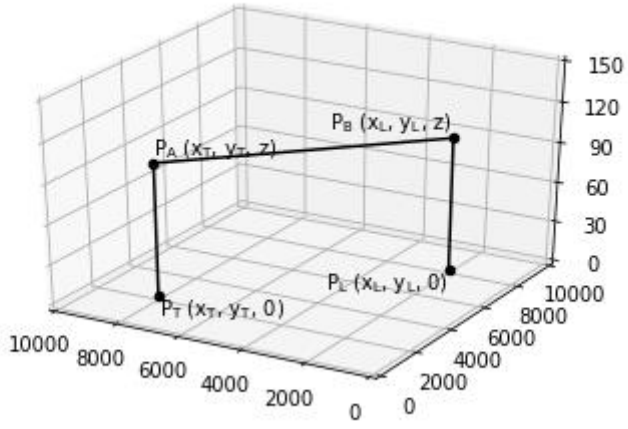


Figure 2. 2 A single UAS flight operation.

The movement of a UAS is simplified such that: it starts off at the takeoff point P_T, ascends vertically by hovering from P_T to its designated altitude level z at P_A, travels linearly from the linear starting point P_A to the linear ending point P_B, and descends vertically from P_B to the landing point P_L. The vertical motion section of the flight is assumed to be instantaneous. For the linear section of the flight (from P_A to P_B), the UAS starts from rest and instantly accelerates to cruising speed v_{cruise} and remains at that speed until instantly decelerating to rest at point P_B.

For the positions of points P_T and P_L, a random integer between 0 and 10 000 m (the dimension of the airspace) was assigned to the x and y coordinates of the points. For the takeoff time t_T, a random integer between 0 and 10 000 s (the time frame of the airspace) was assigned to the t coordinate of the points.

By defining P_T (x_T, y_T, 0) and P_L (x_L, y_L, 0), the flight heading becomes equation (1)

$$\theta = arctan\left(\frac{y_L - y_T}{x_L - x_T}\right) \quad (1)$$

This flight heading will be useful in determining which UTMSP the flight operation belongs to in the case of the federated UTM architectures.

By assuming that the UAS travels at a constant cruise velocity, the landing time t_L can be simplified to equation (2)

$$t_L = t_T + \frac{\sqrt{(x_L - x_T)^2 + (y_L - y_T)^2}}{v_{cruise}} \quad (2)$$

And the x and y positions of each UAS can be described parametrically for every time tick as equations (3) and (4), respectively. Where t represents the time elapsed since t_T specifically for each flight operation.

$$x(t) = x_T + v_{cruise} cos\theta \cdot t \quad (3)$$

$$y(t) = y_T + v_{cruise} sin\theta \cdot t \quad (4)$$

With this basic set of rules, an n amount of operations was generated into the airspace. The number of operations n was varied between 1000 and 10 000 (with intervals of 1000). All operations were initially given the altitude z. However, if the UAS were to come to close to each other, i.e. be in conflict, the algorithm will assign it a new layer.

Assumptions

The vertical takeoff and landing sections of the flight were assumed to be clear of conflict. Conflicts can arise at the beginning of an operation when two or more UAS have takeoff points P_T and landing points P_L that are within the conflict distance of each other. These conflicts can be resolved through other deconfliction methods in the xy-plane and t-axis such as horizontal maneuvers, ground delays, or hovering. However, in these simulations, if this case were to happen, P_T would be randomly generated again until the two P_T are not in conflict with each other.

A large airspace means the probability of generating takeoff points within conflict distance within time decreases, as the dimension of the conflict distance gets smaller

relative to the dimension of the airspace. However, in real life, it can be said that the distribution of takeoff points and landing points are not randomly generated like in this simulation, but rather they are more condensed in popular spots and sparser in lesser populated areas.

2.2.3 Splitting operations based on flight headings

Four different architectures were used: Single, Federated (2 UTMSPs), Federated (3 UTMSPs), and Federated (4 UTMSPs).

For the single architecture, all conflicts were resolved by one system. For the federated architectures, all operations were divided into UTMSPs based on their allowed heading range θ as shown by Table 2.1. Past studies have shown that conflict reduces as smaller heading ranges are used, but at diminishing rates [11]. In this paper, we are both limiting the heading range and lowering the number of operations by splitting them into multiple UTMSPs.

Table 2. 1 Heading range for each UTMSP for all UTM architectures

UTM architecture	Number of UTMSPs	Heading range allowed
Single	1	$0° \leqslant \theta < 360°$ (all operations)
Federated (2 UTMSPs)	2	$0° \leqslant \theta < 180°$ $180° \leqslant \theta < 360°$
Federated (3 UTMSPs)	3	$0° \leqslant \theta < 120°$ $120° \leqslant \theta < 240°$ $240° \leqslant \theta < 360°$
Federated (4 UTMSPs)	4	$0° \leqslant \theta < 90°$ $90° \leqslant \theta < 180°$

		$180° \leqslant \theta < 270°$ $270° \leqslant \theta < 360°$

2.2.4 Definition of a conflict

Because the main objective of the simulation is to designate the flight operations to different vertical layers to resolve the conflicts, the proper definition of a conflict will have to be defined.

For each flight operation, the linear starting point P_A and the linear ending point P_B are defined. Another property specific to a UAS is its buffer radius r_i, where $1 \leqslant i \leqslant n$. A conflict is defined when two buffer radii overlap in time. Figure 4 illustrates a conflict arising from two operations.

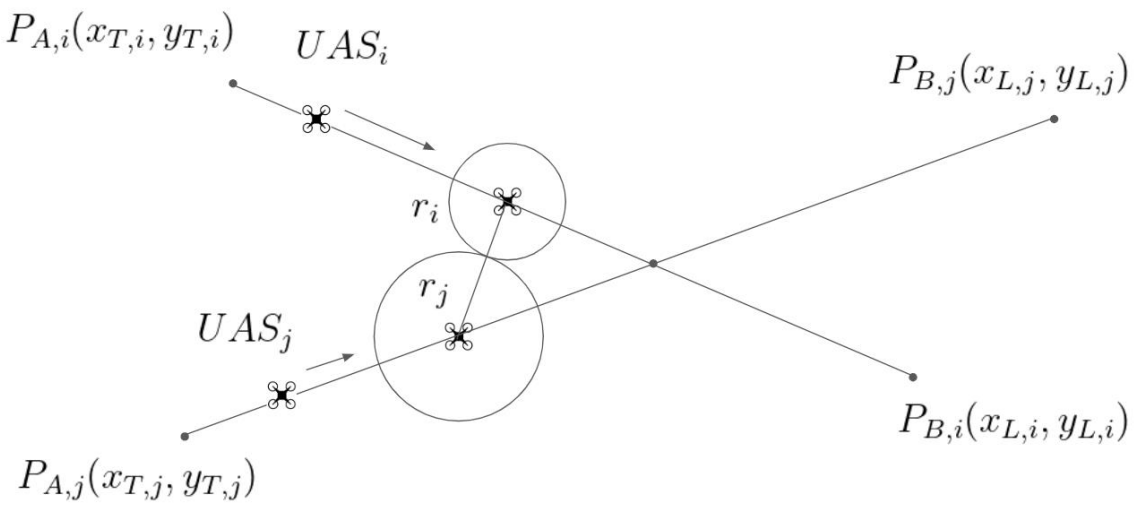


Figure 2. 3 Model of conflict between UAS i and UAS j

The minimum distance that two UAS can be far apart is called the conflict distance R. In Figure 2.3, the conflict distance is $r_i + r_j$. R is checked every time tick in the simulations. Because R depends on r_i from both UAS, it is important to know how r_i is

calculated. Ideally, the size of the buffer radius r_i is some function of the UAS's size, cruising speed v_{cruise}, and current GPS accuracy. A bigger UAS would require a larger r_i. Similarly, a faster travelling UAS would require a larger r_i. Additionally, a UAS whose GPS accuracy is poor would require a larger r_i to compensate for the uncertainty of its current position. Because many UAS were generated in these simulations (n from 1000 to 10 000) and they are assumed to be identical, it is impractical to account for all the fine properties of the UAS to determine r_i on a case-by-case basis. Therefore, R was set to specific values. R was varied between 50 and 300 (with intervals of 50). These values were selected to be in line with past studies [4] [7] [10]-[12].

In order to check whether flight operations i and j are in conflict, the time frame where both the operations are active $t_{active\ i,j}$ was considered, as shown in equation (5). Given the takeoff and landing times of UAS i and UAS j: t_{Ti}, t_{Li}, t_{Tj}, t_{Lj}, the time frame is between the maximum value of the lower bound and the minimum value of the upper bound with respect to the takeoff and landing times, as shown in equation (5).

$$max(t_{Ti}, t_{Tj}) \leq t_{activei,j} \leq tmin(t_{Li}, t_{Lj}) \tag{5}$$

Secondly, the relative distance $\rho_{i,j}$ between the positions of UAS i and UAS j must be calculated. Because the condition to find the minimum relative distance can be realized with the square of the relative distance, the squared term will be kept for ease of calculation, as shown in equation (6).

$$\rho_{i,j}(t) = \sqrt{\left[x_j(t) - x_i(t)^2\right] + \left[y_j(t) - y_i(t)^2\right]}$$

$$\rho_{i,j}(t)^2 = \left[x_j(t) - x_i(t)^2\right] + \left[y_j(t) - y_i(t)^2\right] \tag{6}$$

Equation (6) becomes equation (7) by substituting in equations (3) and (4) into equation (6) considering UAS i and j and time t.

$$\rho_{i,j}^2(t) = \left[\left(x_{Tj} - x_{Ti1}\right) + v_{cruise}t\left(cos\theta_j - cos\theta_i\right)\right]^2 + \left[\left(y_{Tj} - y_{Ti}\right) + v_{cruise}t\left(sin\theta_j - sin\theta_i\right)\right]^2 \tag{7}$$

The condition when $\rho_{i,j}$ is minimized is given in equation (8).

$$\frac{d\left(\rho_{i,j}^2(t)\right)}{dt} = 0 \tag{8}$$

The time t that satisfies the minimum relative distance condition is given by equation (9).

$$t_{mindist\ i,j} = \left[\frac{1}{v_{cruise}} \cdot \frac{(x_{Ti} - x_{Tj})(cos\theta_j - cos\theta_i) + (y_{Ti} - y_{Tj})(sin\theta_j - sin\theta_i)}{(cos\theta_j - cos\theta_i)^2 + (sin\theta_j - sin\theta_i)^2}\right] \tag{9}$$

Now we check whether $t_{mindist\ i,j}$ lies in the active time frame of both operations i and j. If it is outside the active time frame $t_{active\ i,j}$, we can deem that there is no conflict and stop further calculations for the next step.

If $t_{mindist\ i,j}$ lies in $t_{active\ i,j}$, we substitute the time when $\rho_{i,j}$ is minimized to find $\rho_{i,j}$ and check whether it is smaller than conflict distance R or not as in equation (10).

$$\rho_{i,j}\left(t_{mindist\ i,j}\right) \leq R \tag{10}$$

If this condition is satisfied, then operations i and j are in conflict.

2.3 Modeling Simulations as Networks

The idea of modeling air traffic as networks has been proposed in the past [16]. By representing the simulations as networks, methods that are effective for time scheduling and slot assignment applications such as graph coloring become available for analysis [17] [18]. A simple model of a network is considered where the focus is only on whether operations are in conflict with each other or not. In this model, an operation can be represented as a vertex, and a conflict between two operations can be represented as an edge between two vertices as shown in Figure 2.4. Using these two definitions of the vertex and edge, the rest of the UTM components that we are interested in analyzing can be represented by their network counterparts. Table 2.2 provides a summary of the analogy. Further details on each component is discussed after Table 2.2.

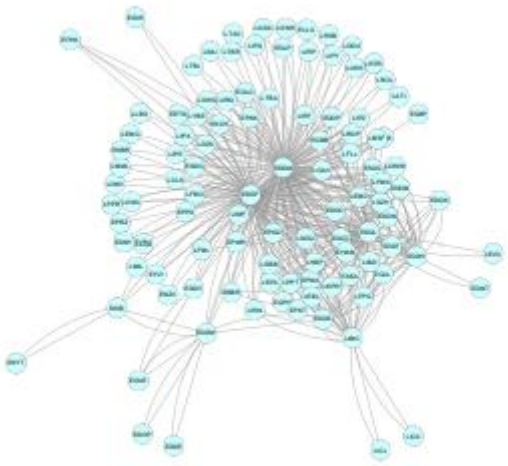


Figure 2. 4 Modeling simulations as networks

(Taken from Zanin, 2013) [16]

Table 2. 2 List of UTM components of interest and its network counterparts.

UTM components	Network counterpart
Simulation with certain R and n	Network
List of all conflicts between flight operations	Adjacency matrix
Flight operation	Vertex
Conflict between two operations	Edge between two vertices
Layer	Color
Distribution of operations among layers	Distribution of vertices among colors
Number of conflicts per operation	Degree of a vertex
Average number of conflicts per operation	Average degree of network

Adjacency matrix

The list of all combinations of i and j and whether they are in conflict is stored in an adjacency matrix G as shown in equation (11). For each element in G, the value of 0 is given if there is no conflict and 1 is given if there is a conflict. The adjacency matrix represents the list of all conflicts in a network.

$$G = \begin{bmatrix} g_{1,1} & g_{1,2} & \cdots & g_{1,n} \\ g_{2,1} & g_{2,2} & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ g_{n,1} & \cdots & \cdots & g_{n,n} \end{bmatrix} \tag{11}$$

In the case where there are n number of operations, a single operation will have to be checked against all other operations. The selection of the pair of drones were done in terms of combinations such that there are no overlaps between i and j drones as shown in equation (12).

$$C(n,2) = \frac{n!}{2!\,(n-2)!} = \frac{n(n-1)}{2} \tag{12}$$

Because the conflict between operation i and operation j is the same as the conflict between operation j and operation i, the adjacency matrix is a symmetrical matrix with diagonal terms equal to 0 and only n(n-1)/2 calculations have to be performed.

Vertex matrix

A vertex matrix V is created for each vertex v_i up to n flight operations as shown in equation (13). This represents the list of flight operations.

$$V = [v_1, v_2, v_3, \ldots, v_n] \tag{13}$$

Degree of a vertex

Using the adjacency matrix, the degree for each vertex i $\deg(v_i)$ is represented by the number of elements of row i that is equal to 1 as shown in equation (14).. This represents the number of conflicts for each flight operation. Then a degree matrix D is created for each degree $\deg(v_i)$ up to n flight operations as shown in equation (15).

$$deg(v_i) = \sum_{g_{i,j} \in row\ i} 1_1(g_{i,j}) \tag{14}$$

$$D = [deg(v_1), deg(v_2), deg(v_3), \dots, deg(v_n)] \tag{15}$$

Number of edges

Summing the degrees and dividing by 2 to ignore duplicate terms of a symmetrical matrix we get the number of edges m in a network as shown in equation (16). This represents the number of conflicts in a simulation.

$$m = \frac{1}{2} \cdot \sum_{i=1}^{n} deg(v_i) \tag{16}$$

Average degree of a network

The average degree δ is the number of edges over the number of vertices as shown in equation (17). This represents the average number of conflicts per operation of each simulation.

$$\delta = \frac{m}{n} \tag{17}$$

Graph coloring problem

Using the network analogy, a layer can be represented as a color. The act of assigning different vertical layers to operations in conflict is akin to assigning different colors to vertices. Therefore, the concept of deconfliction by layer assignment can be viewed as a graph coloring problem [18]. We are interested in finding the number of layers needed for certain conditions and architectures and the distribution of operations among layers. Furthermore, we are also interested in the average number of conflicts per operation for each network

The problem of coloring a network with k colors (k-coloring) has been studied extensively. Figure 2.5 shows an example of an optimal coloring of a simple network. The goal is to find a solution that colors the vertices in such a way that no two adjacent vertices in the network have the same color. A chromatic number $\chi(G)$ of a network is the smallest integer for which the network is colorable. An optimal coloring is when k equals the chromatic number.

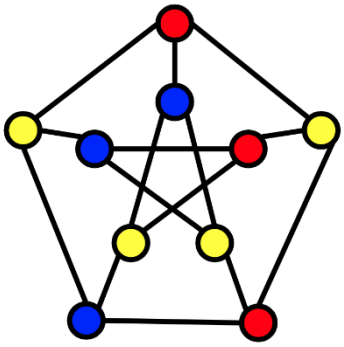


Figure 2. 5 Optimal coloring of a simple network

It is known that a solution for the case where $k = 2$ exists and can be solved in polynomial time. A solution for the case $k \geq 3$ cannot be solved in polynomial time but can be verified in polynomial time, making it an NP-complete problem. However, the optimal coloring to the graph coloring problem is NP-hard. Because of the hardness of the problem, much of the research on graph coloring is dedicated to developing new algorithms to find a solution that is as close to optimal coloring as possible. These non-optimal coloring algorithms are also NP-hard [19].

Finding optimal coloring is difficult in practice. Even some networks with vertex counts as low as 125 cannot be optimally solved by the best performing exact brute-force algorithms [19]. Therefore, heuristics are necessary to solve problems with larger networks.

In this paper, because the number of operations, i.e., vertices, simulated are large (n from 1000 to 10 000), we decide to use a heuristic called the greedy coloring algorithm.

Unlike k-coloring where the number of colors is set to k, the greedy coloring algorithm used here does not have a limit on the number of colors. The algorithm assigns a color to a vertex v_i in order, from v_1 to v_n. The assigned color should be the smallest color that is not used in any adjacent vertices.

Because the quality of the solution (closeness to optimal solution) depends on the selected vertex ordering, an effective strategy to determine vertex ordering should be used. For example, a crown graph with n vertices can have two vastly different solutions depending on the vertex ordering used. For example, a crown graph with n vertices could be filled with as low as 2 colors or as high as n/2 colors.

The vertex ordering strategy used in the simulations in this paper is the largest degree first strategy. This means that the vertices matrix is sorted by vertices v_i with the largest degrees first. Figure 2.6 shows how the assignment of colors start from vertices with higher degrees

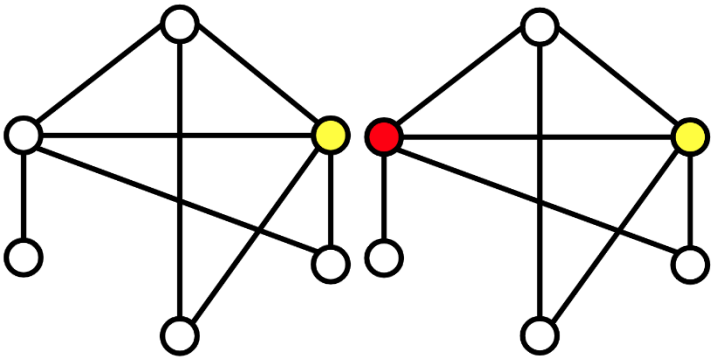


Figure 2. 6 Largest degree first vertex ordering strategy prioritizing vertices with large degrees

2.4 Metrics

We decided to vary two independent variables: conflict distance R and number of operations n. Six values for R were used from 50 m to 300 m (with intervals of 50 m). Ten values for n were used from 1000 to 10 000 (with intervals of 1000). A total of 60 simulations were run.

Operations will be assigned into their appropriate UTMSP based on their headings. For each UTMSP, the average number of conflicts per operation and the number of layers needed will be calculated. We would like to test whether the number of layers needed are appropriate for the context of UTM. Because there is 150m worth of airspace to work with in the vertical direction, we deem the practical value for the maximum number of layers to be 20. Furthermore, the number of layers assigned from the greedy coloring algorithm will be calculated, as well as the distribution of operations among the layers.

For each federated UTM architecture, the results for all systems in each UTMSP will be combined. The overall average degree for each federated UTM architecture is the mean of the average degree of each UTMSP. The overall number of layers needed is the sum of the number of layers needed from all UTMSPs.

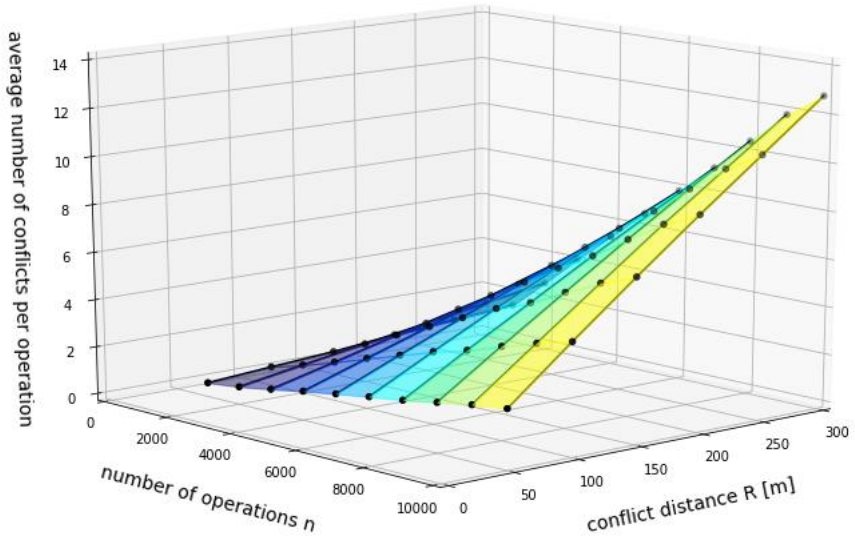
Chapter 3

Results

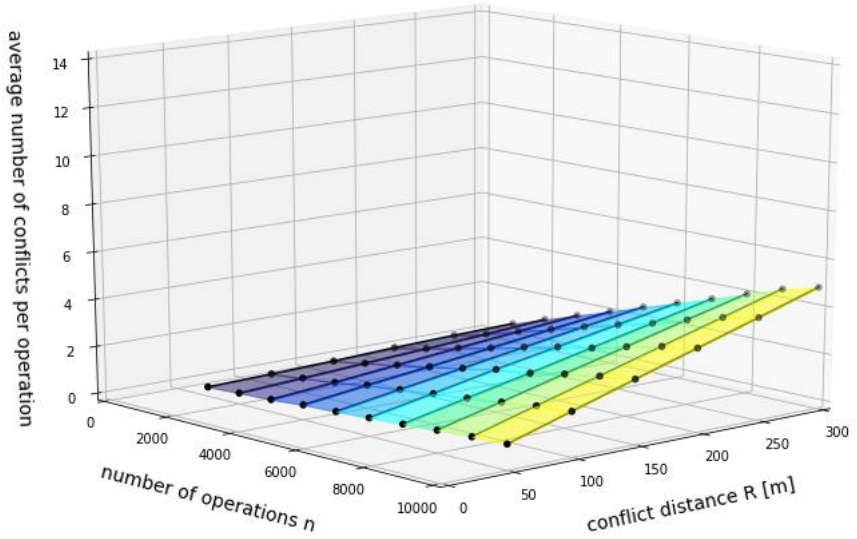
3 Results

3.1 Average number of conflicts per operation

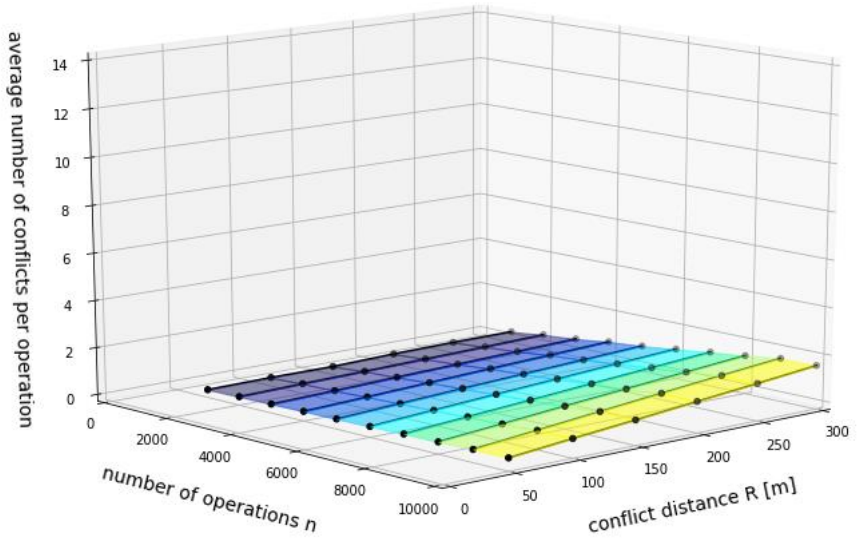
Firstly, the values of the average number of conflicts per operation for all 60 simulations were plotted in Figure 3.1 for all four UTM architectures.



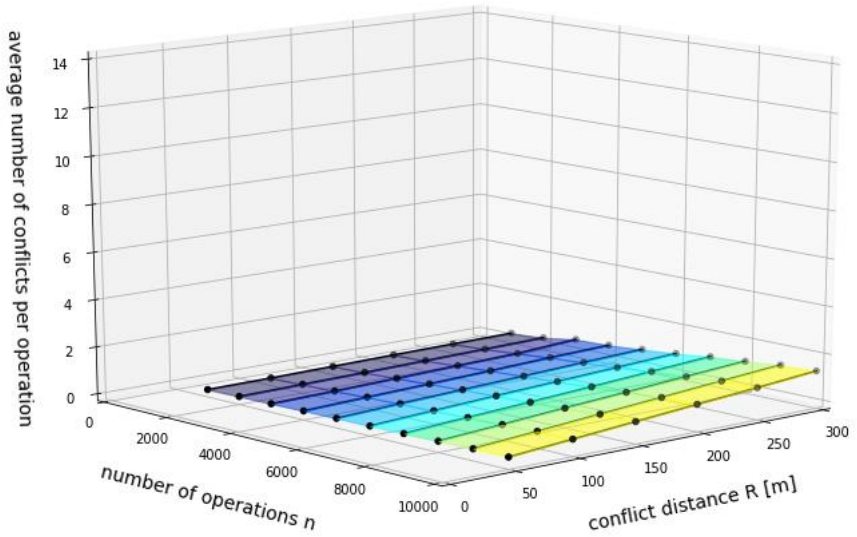
(a) single UTM architecture



(b) federated (2 UTMSPs) UTM architecture



(c) federated (3 UTMSPs) UTM architecture



(d) federated (4 UTMSPs) UTM architecture

Figure 3. 1 Average number of conflicts per operation as a function of n and R.

Figure 3.2 combines the average numbers of conflicts per operation of all UTM architectures into one figure. Black: single UTM architecture, Blue: federated (2 UTMSPs) UTM architecture, Red: federated (3 UTMSPs) UTM architecture, Green: federated (4 UTMSPs) UTM architecture.

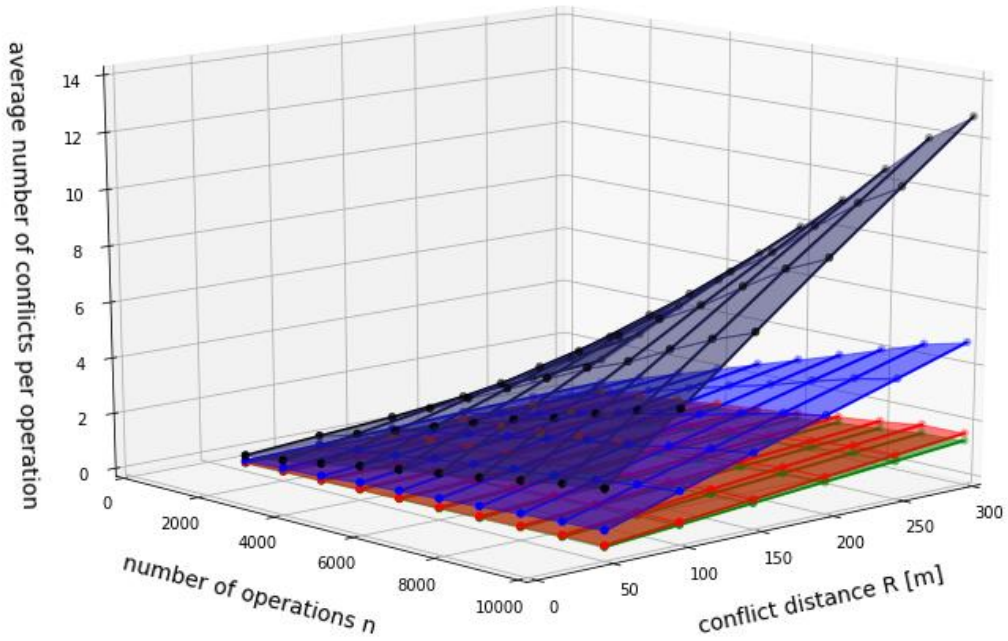


Figure 3. 2 Average numbers of conflicts per operation as a function of n and R for all UTM architectures

As shown in Figure 3.2, the average number of conflicts per operation obtained experimentally increases linearly with both the number of operations n and conflict distance R. The linear relationship between the number of operations n and average numbers of conflicts per operation, is in line with experimental and theoretical relations presented in past studies [11] [12].

The effect of increasing UTMSPs can be seen as the average number of conflicts per operation decreases. However, the decrease is not linear but diminishing returns can be observed. The values for average number of conflicts per operation of the 3 UTMSPs and 4 UTMSPs architectures are very close.

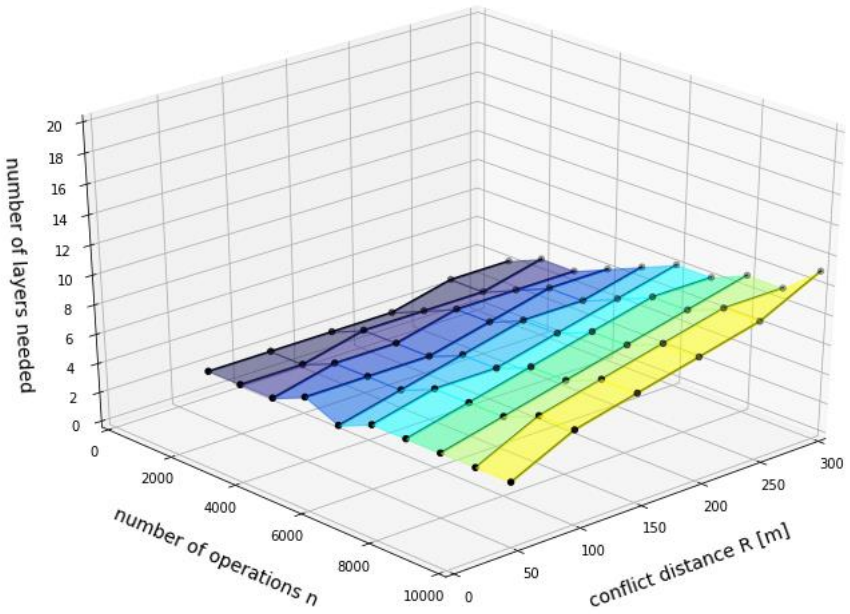
This non-linear behavior stems from the criteria used to split the operations into multiple UTMSPs. By splitting the UTMSPs based on flights heading, the distribution is expected to be equal among the UTMSPs in each architecture as the number of

operations become large. This means that a UTMSP will have less operations to manage. As shown in the individual plots in Figure 3.1, the average number of conflicts per operation has a linear relationship with the number of operations. This means that the average number of conflicts per operation for a federated UTM architecture with h number of UTMSPs will decrease by a factor of h compared to the single UTM architecture. This is called the “spreading effect” by Hoekstra and it translates to a linear decrease [12]. This linear decrease would ideally occur if operations are simply split uniformly and randomly among the UTMSPs without any restrictions.

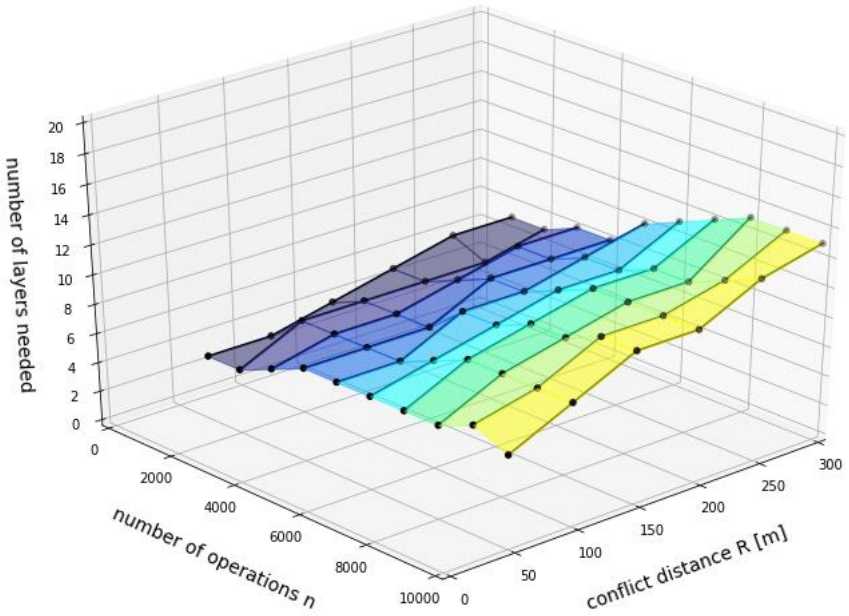
However, we must consider another effect caused by restricting the flight headings. Because of this restriction, there is a higher probability for UAS to approach each other with a smaller angle difference, which results in a higher probability of having a lower relative velocity [12]. This “reduction of relative velocity effect” on top of the “spreading effect” results in the non-linear behavior observed in Figure 3.2.

3.2 Number of layers needed

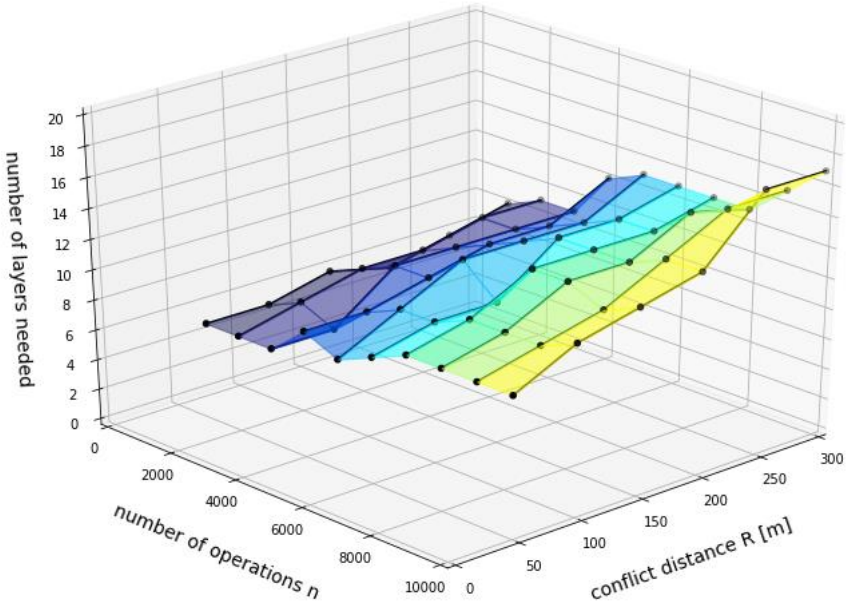
The values of the number of layers needed for all 60 simulations were plotted in Figure 3.3 for all four UTM architectures.



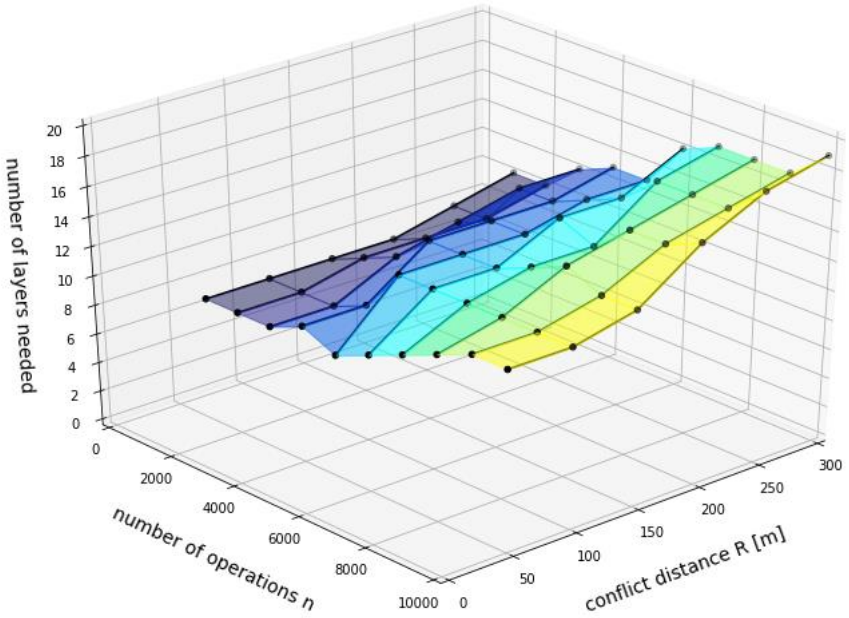
(a) single UTM architecture



(b) federated (2 UTMSPs) UTM architecture



(c) federated (3 UTMSPs) UTM architecture



(d) federated (4 UTMSPs) UTM architecture

Figure 3. 3 Number of layers needed as a function of n and R.

Figure 3.4 combines the number of layers needed of all UTM architectures into one figure. Black: single UTM architecture, Blue: federated (2 UTMSPs) UTM architecture, Red: federated (3 UTMSPs) UTM architecture, Green: federated (4 UTMSPs) UTM architecture.

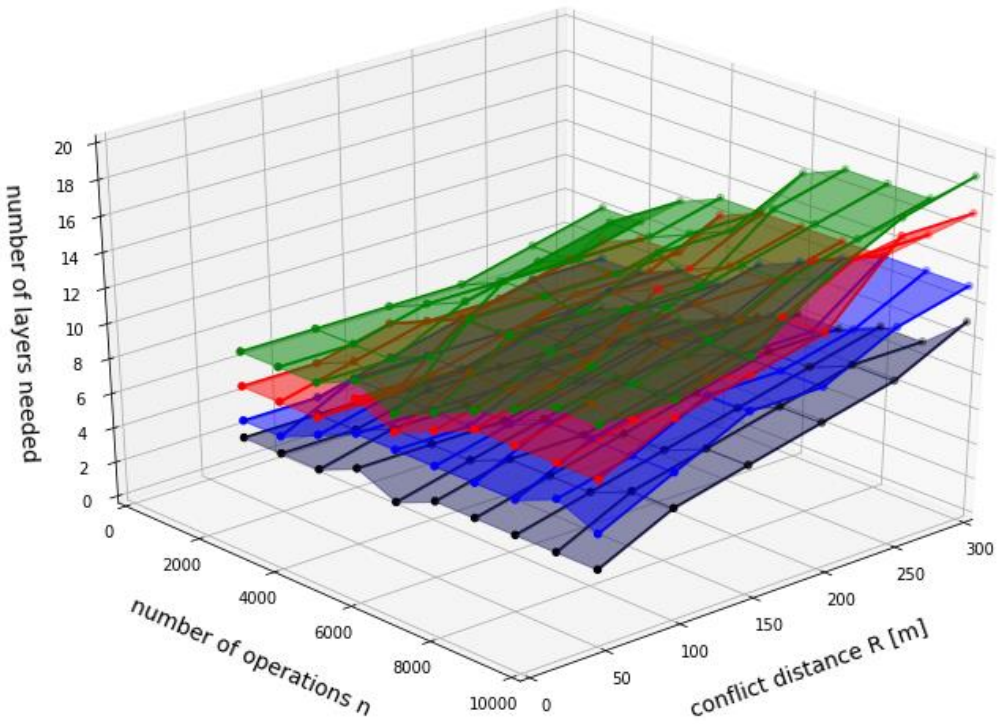


Figure 3. 4 Number of layers needed as a function of n and R for all UTM architectures

In contrast to the average number of conflicts per operation that decreases with increasing UTMSPs, the number of layers needed increases. The trend on Figure 3.4 is not as smooth because, unlike the average number of conflicts per operation, the number of layers needed is always an integer with a discrete value. “Jumps” in the surface plot where the number of layers needed are 1 higher than the adjacent simulation with higher n or R occur due to the randomness of the probabilistic setup used in the simulations. Randomly generated flight operations can result in some local points having high density of traffic (vertices with high degrees) and therefore, requiring another layer to achieve deconfliction. This discrete nature of the number of layers needed having to be an integer is amplified and explained in the next section when the number of layers needed is compared to the average number of conflicts per operation.

3.3 Comparison between average number of conflicts per operation and number of layers needed

Data for average number of conflicts per operation in Figure 3.2 was plotted against data for the number of layers needed in Figure 3.4, as shown by Figure 3.5.

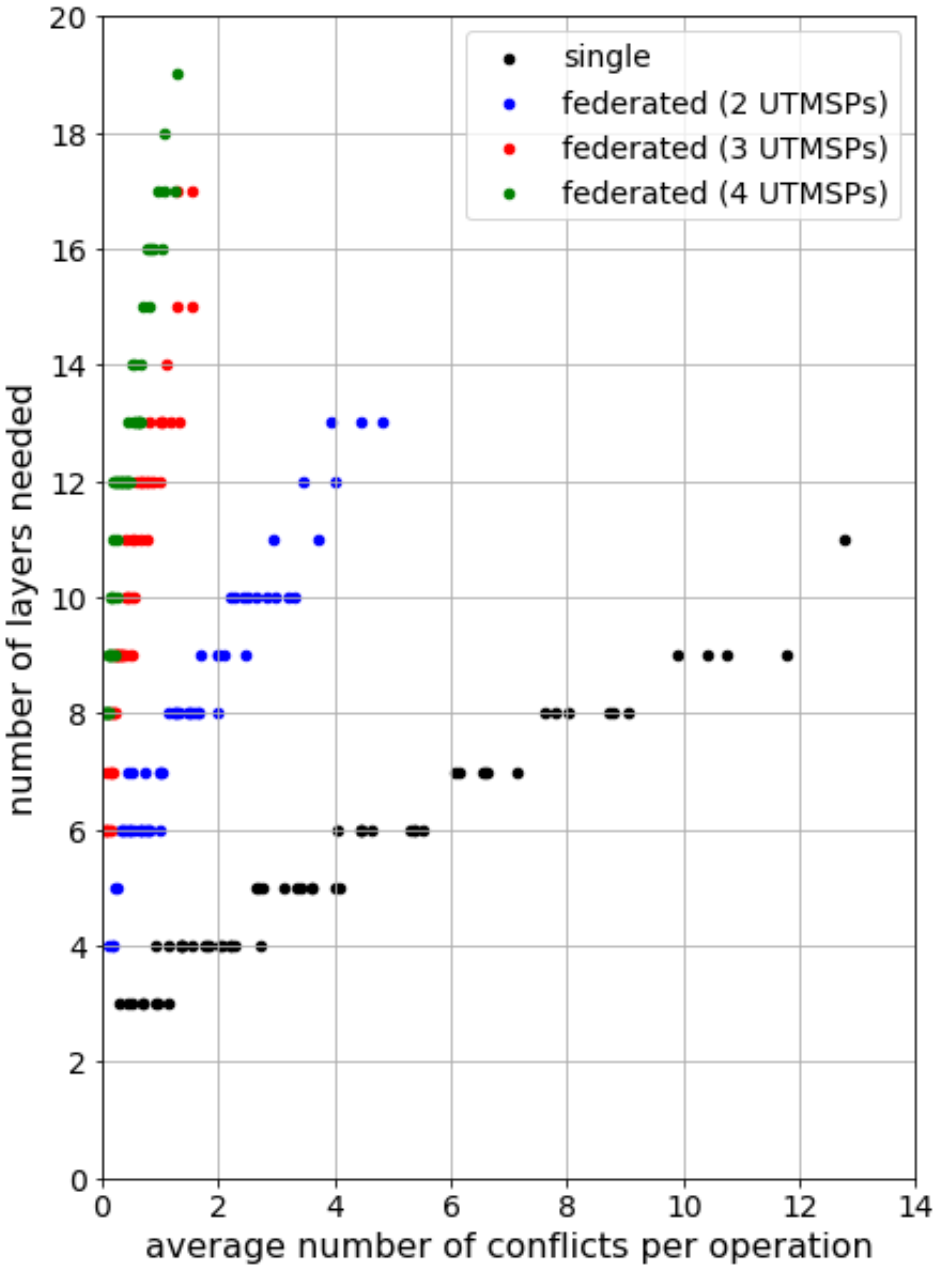


Figure 3. 5 Average number of conflicts per operation and number of layers needed for all UTM architectures

Figure 3.5 represents a tradeoff between the average number of conflicts per operation and the number of layers needed as the number of UTMSPs changes. For the range of independent variables used in the simulations (50 < R < 300 and 1000 < n < 10 000), the number of layers needed and average number of conflicts per operation exhibits a linear relationship. The line of best fit can be represented as equation (18)

$$k = m\delta + c \tag{18}$$

Where k is the number of layers needed, δ is the average number of conflicts per operation, m is the gradient, and c is the y-intercept. Table 3.1 records the m and c for the line of best fit according to linear regression from Figure 3.5's data.

Table 3. 1 m and c for the line of best fit for average number of conflicts per operation and number of layers

UTM architecture	m	c
Single	0.599	2.932
Federated (2 UTMSPs)	1.812	5.070
Federated (3 UTMSPs)	6.353	6.711
Federated (4 UTMSPs)	8.492	8.351

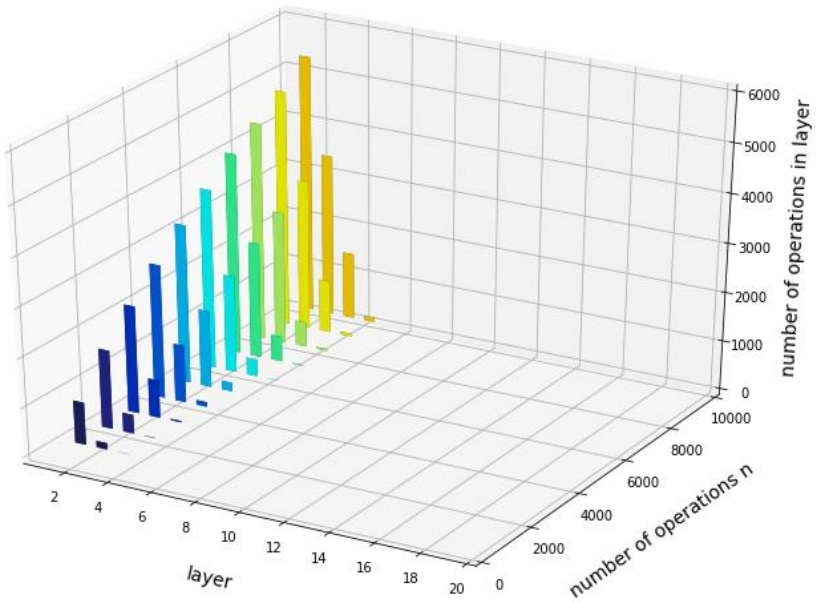
It can be seen that even in the case that requires the most number of layers in these simulations (R = 300, n = 10 000), the single UTM architecture only utilizes 11 layers at a relatively high average number of conflicts per operation of 12.776, whereas the federated (4 UTMSPs) UTM architecture utilizes 19 layers at a relatively low average number of conflicts per operation of 1.299. This tradeoff between the average number of conflicts per operation and number of layers needed can favor the federated UTM architecture's more UTMSPs because the benefits of lower traffic caused by a lower average number of conflicts per operation can outweigh the extra number of layers needed. For all values of n and R used in the simulations, even with the extra layers generated by more UTMSPs, the number of layers needed is 19, which is below the maximum of 20 layers for the low level airspace that has 150 [m] of vertical airspace.

From Figure 3.5, it can be observed that there is a high concentration of data points where the number of layers k needed is at (h x p) layers, where h is the amount of UTMSPs used and p is any integer. For example, high concentration of points where k = 6, 8, 10 for 2 UTMSPs, k = 9, 12 for 3 UTMSPs, and k = 12, 16 for 4 UTMSPs. This is because the distribution of operations among h UTMSPs is almost equal, which means the combined k is equal to the k for each UTMSP multiplied by the amount of UTMSPs or h.

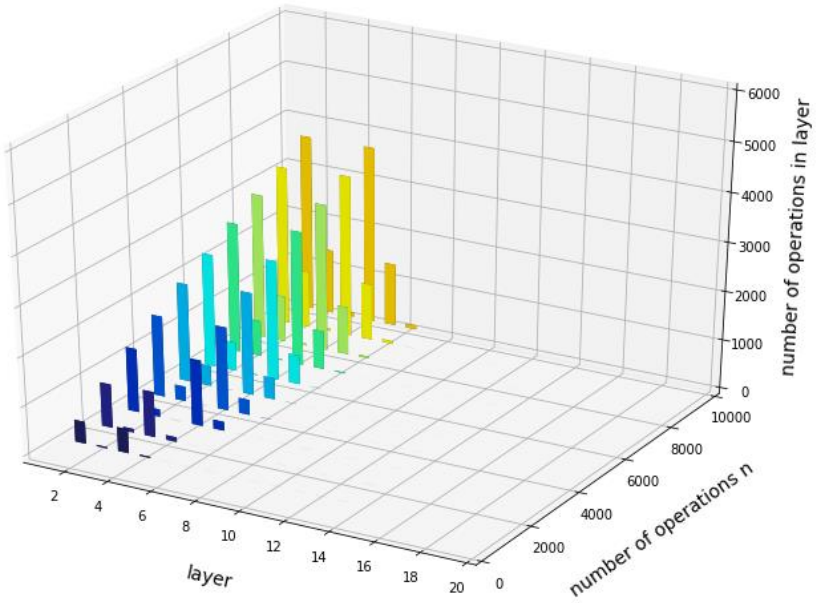
The only exception to this is when "busy" local points with high density of traffic (vertices with high degrees) occur, resulting in the need to make a new layer. Because simulations with lower values for n and R have relatively lower overall traffic, the concentration of points where k - (h x p) is higher for those simulations. Ideally, we would like to have a case where operations are distributed more evenly among UTMSPs and produce the same number of layers needed which fits k = (h x p). In the next section, the distribution of operations among layers is analyzed to achieve this goal.

3.4 Distribution of operations among layers

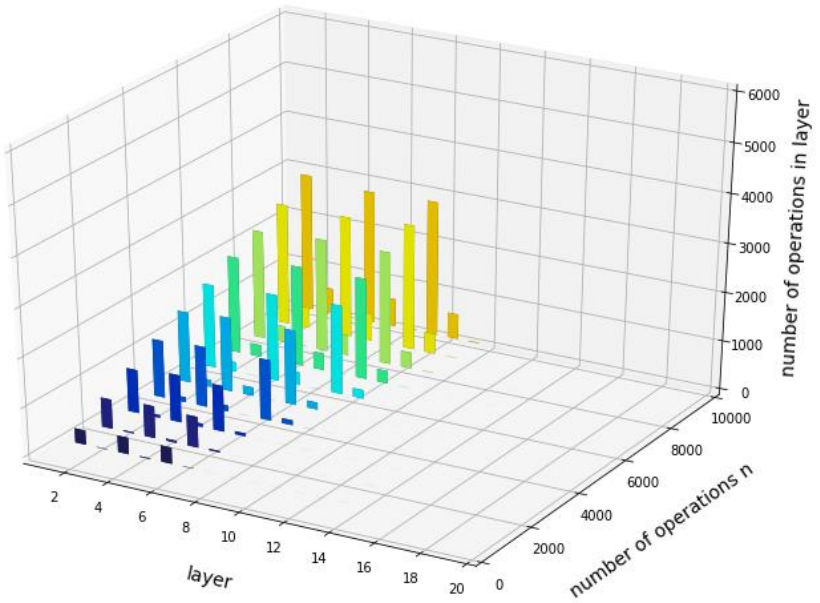
The distributions of operations among layers for 20 simulations (cases where R = 50 m and R = 300 m) were plotted in Figure 3.6 for all four UTM architectures.



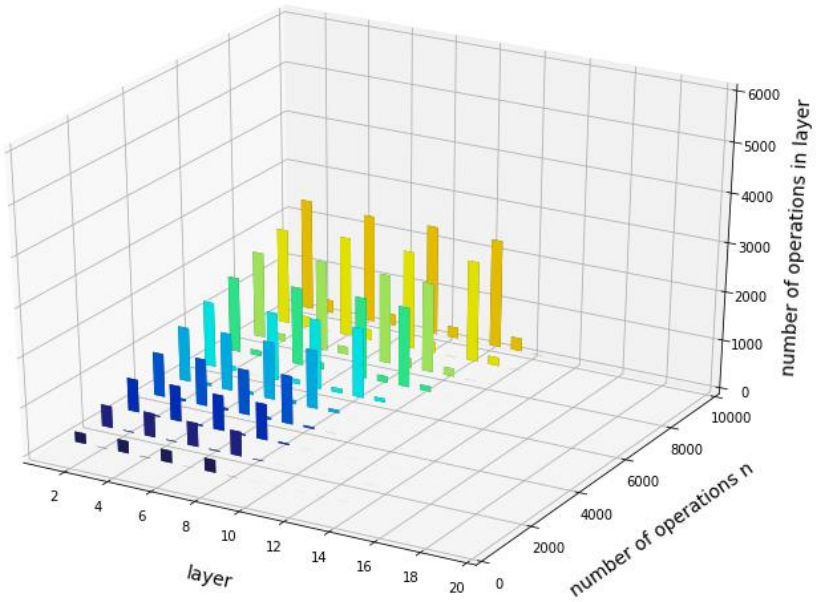
(a) single UTM architecture



(b) federated (2 UTMSPs) UTM architecture

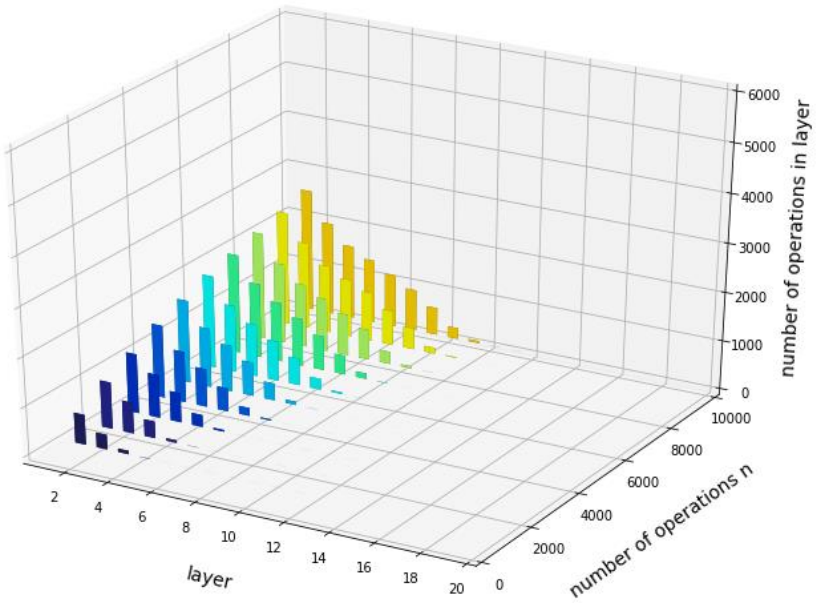


(c) federated (3 UTMSPs) UTM architecture

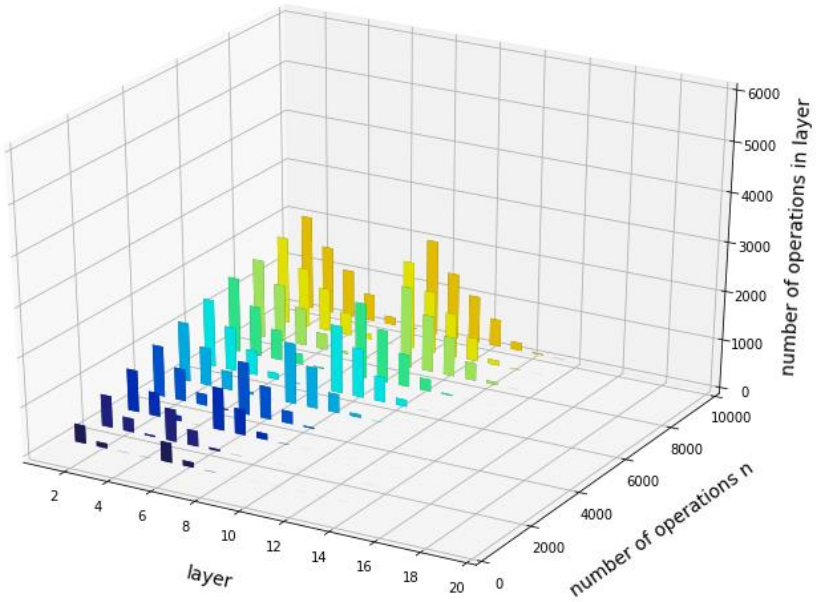


(d) federated (4 UTMSPs) UTM architecture

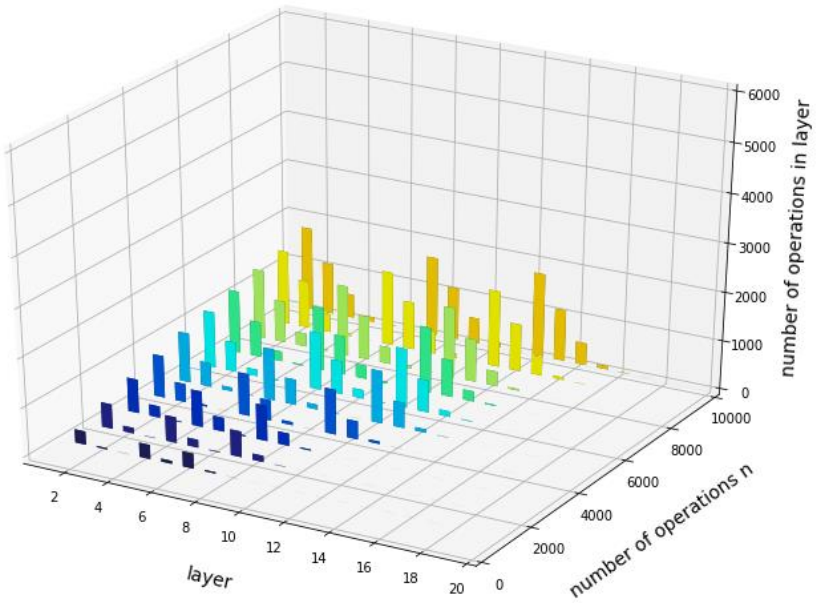
Figure 3. 6 Distribution of operations among layers as a function of n (R = 50 m)



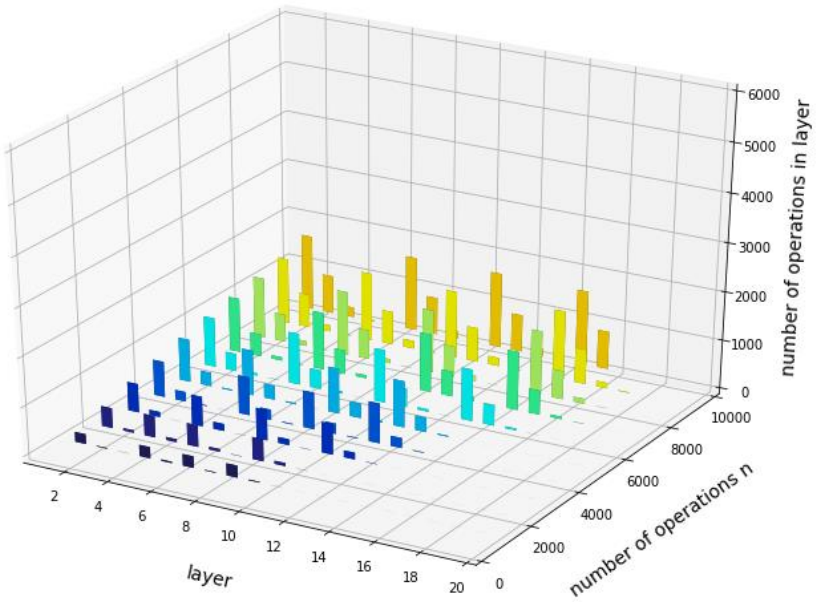
(a) single UTM architecture



(b) federated (2 UTMSPs) UTM architecture



(c) federated (3 UTMSPs) UTM architecture



(d) federated (4 UTMSPs) UTM architecture

Figure 3. 7 Distribution of operations among layers as a function of n (R = 300 m)

When raw data was examined, there were many instances where a very small amount of "busy" UAS occupies new layers. This can clearly be seen in Figures 3.6 and 3.7 For example, in Table 3.2, only three operations from a total of 5000 operations cause two more layers to be generated.

Table 3. 2 Distribution of operations among layers

(simulation n = 5000, R = 150 m)

	Federated (3 UTSPs) UTM architecture		
	UTMSP 1 ($0° \leqslant \theta < 120°$)	UTMSP 2 ($120° \leqslant \theta < 240°$)	UTMSP 3 ($240° \leqslant \theta < 360°$)
Number of operations	1602	1706	1692
Number of layers needed	4	4	3
Layer 1	1239	1272	1295
Layer 2	348	401	381
Layer 3	14	31	16
Layer 4	1	2	

For federated UTM architectures, we propose the idea of grouping all these "busy" operations from all UTMSPs together into one shared layer to reduce unnecessary layers. In order to ensure that these "busy" operations really account for a small portion of the total amount of operations in a simulation, the cut-off point is set to 1%. In other words, the new layer must contain less than 1% of the total number of operations in each UTMSP for the operations to be moved to the new shared layer. Because the order of magnitude of the operations in the new shared layer is extremely small, we assume that there are no conflicts in this layer, or that these few conflicts can be solved by other deconfliction methods besides vertical layer assignment.

3.5 Number of layers needed adjusted to 1% cut-off point

Figure 3.8 uses the same dataset as Figure 3.7(d) but was adjusted to the 1% cut-off point. In the case where n = 10 000 and R = 300 m, the number of layers needed is significantly reduced from 19 to 13.

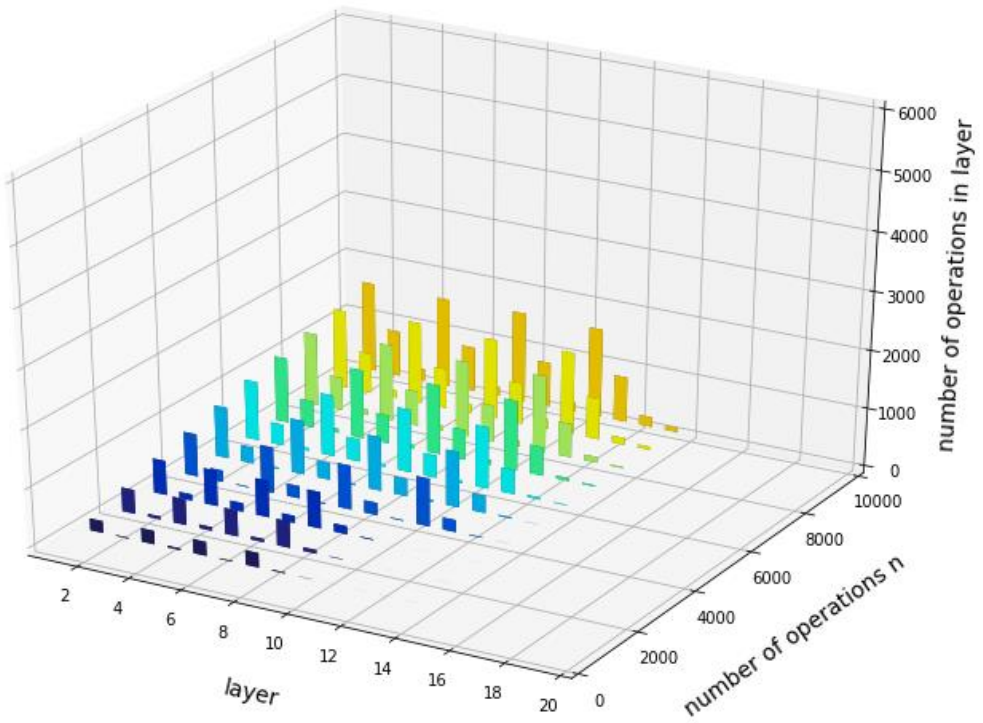


Figure 3. 8 Distribution of operations among layers as a function of n (R = 300 m) for the federated (4 UTMSPs) UTM architecture, adjusted to 1% cut-off point

Likewise, Figure 3.9 uses the same dataset as Figure 3.5 but was adjusted to the 1% cut-off point.

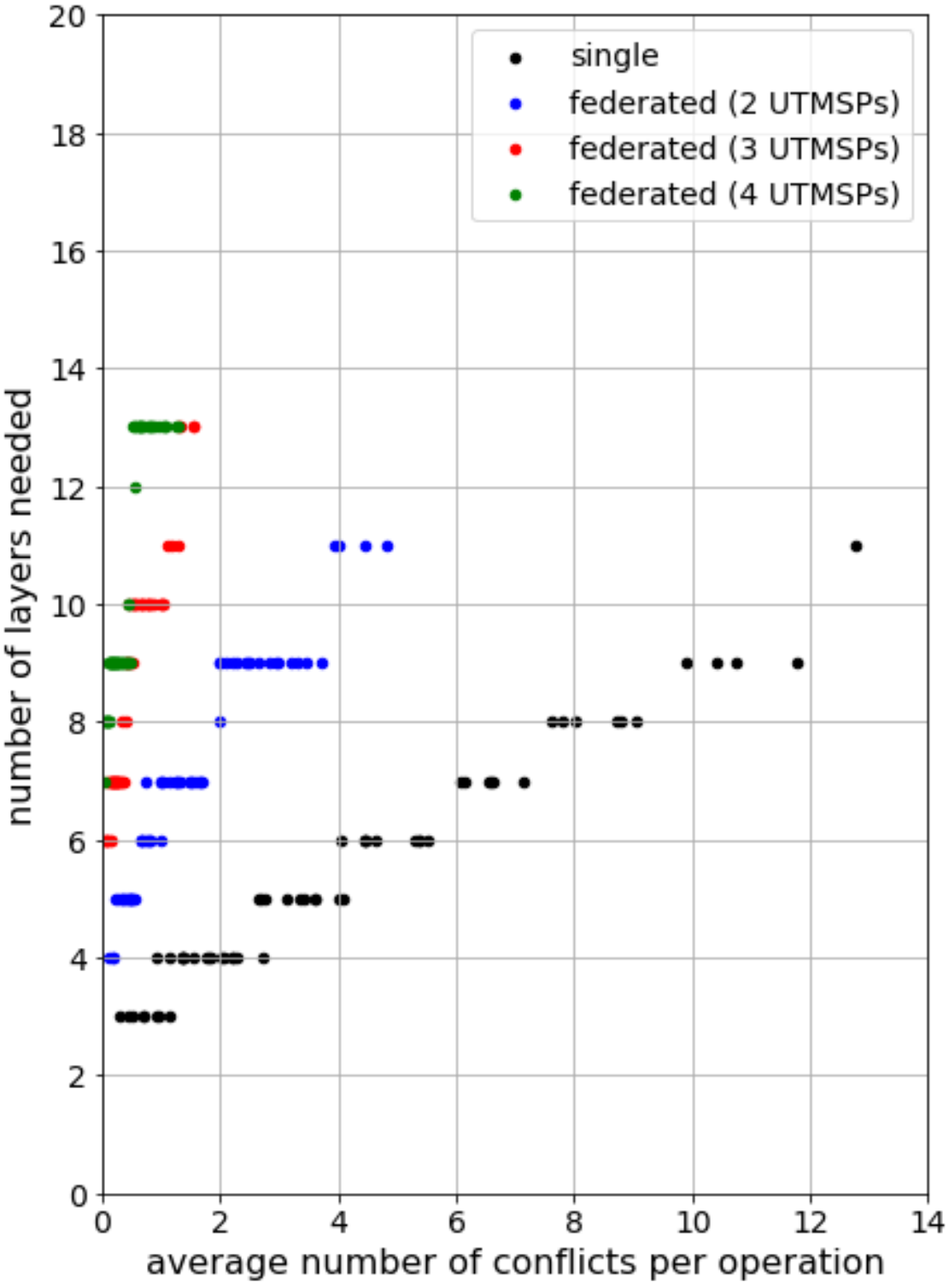


Figure 3. 9 Average number of conflicts per operation and number of layers needed for all UTM architectures Adjusted to 1% cut-off point

For the pre-1% cut-off point adjusted model (Figure 3.5) , there is a high concentration of data points in where the number of layers needed k = (h x p)
In comparison, for the 1% cut-off point adjusted model (Figure 3.9), there is an overwhelming high concentration of points on k = (h x p) + 1. This suggests that almost all operations with high degrees are grouped and put together in the new shared layer. As the number of UTMSPs increase, more layers can be grouped into the new shared layer, thus increasing the effectiveness of the adjusted model. We can start to observe overlaps between the performance of 3 UTMSPs and 4 UTMSPs. This is an advantage for the 4 UTMSPs architecture because it benefits from the lower average number of conflicts per operation, as well as the number of layers needed.

Table 3.2 records the m and c for the line of best fit according to linear regression from Figure 3.5's data.

Table 3. 3 m and c for the line of best fit for average number of conflicts per operation and number of layers, adjusted to 1% cut-off point

UTM architecture	m	c
Single	0.599	2.932
Federated (2 UTMSPs)	1.493	4.773
Federated (3 UTMSPs)	4.569	6.250
Federated (4 UTMSPs)	5.539	7.870

Chapter 4

Conclusion

4 Conclusion

This study aims to analyze the deconfliction of unmanned aircraft traffic management at the high-density low-level uncontrolled airspace by utilizing the vertical layer assignment method. This was done by modeling the airspace and generating random UAS flight operations and applying the greedy coloring algorithm to resolve conflicts between operations. The effect of using different number of UTMSPs were compared by assigning operations to their respective UTMSPs based on their flight headings. 60 simulations were performed by varying the independent variables number of operations from 1000 to 10 000, and conflict distance from 50 m to 300 m. The effects on average number of conflicts per operation, number of layers needed, and the distribution of operations among these layers were analyzed.

For the range of the independent variables used, a positive linear relationship was observed between the number of operations and the average number of conflicts per operation, as well as the conflict distance and the average number of conflicts per operation. A non-linear decrease in the average number of conflicts per operation was observed as more UTMSPs are used due to the "spreading effect" and the "reduction of relative velocity effect".

A tradeoff between the average number of conflicts per operation and the number of layers needed was observed where more UTMSPs utilized more layers at the cost of a lower average number of conflicts per operation. Because the number of layers needed for all 60 simulations remained within the acceptable 20 layers for low-level uncontrolled airspace, the tradeoff could favor the federated UTM architecture's more UTMSPs because the benefits of lower traffic caused by a lower average number of conflicts per operation could outweigh the extra number of layers needed.

Furthermore, it was shown that the number of layers needed could be reduced further in the case of federated UTM architectures by introducing the concept of a 1% cut-off point where extra layers caused by a minority of operations (less than 1%) could be combined to a shared layer.

References

[1] Federal Aviation Administration (2019). FAA Aerospace Forecast Fiscal Years 2019-2039. Retrieved from https://www.faa.gov/data_research/aviation/aerospace_forecasts/media/FY2019-39_FAA_Aerospace_Forecast.pdf

[2] Ministry of Land, Infrastructure, Transport and Tourism (n.d.). Japan's safety rules on Unmanned Aircraft (UA)/Drones. Retrieved from https://www.mlit.go.jp/koku/koku_tk10_000003.html

[3] Kopardekar, P., Rios, J., Prevot, T., Johnson, M., Jung, J., & Robinson, J. E. (2016). Unmanned aircraft system traffic management (UTM) concept of operations.

[4] Hoekstra, J. M., van Gent, R. N., & Ruigrok, R. C. (2002). Designing for safety: the 'free flight'air traffic management concept. *Reliability Engineering & System Safety*, *75*(2), 215-232.

[5] Jardin, M. R. (2005). Analytical relationships between conflict counts and air-traffic density. *Journal of guidance, control, and dynamics*, *28*(6), 1150-1156.

[6] Sunil, E., Hoekstra, J., Ellerbroek, J., Bussink, F., Nieuwenhuisen, D., Vidosavljevic, A., & Kern, S. (2015, June). Metropolis: Relating airspace structure and capacity for extreme traffic densities.

[7] Bulusu, V., Polishchuk, V., Sengupta, R., & Sedov, L. (2017). Capacity estimation for low altitude airspace. In *17th AIAA Aviation Technology, Integration, and Operations Conference* (p. 4266).

[8] Tompa, R. E., Wulfe, B., Owen, M. P., & Kochenderfer, M. J. (2016, September). Collision avoidance for unmanned aircraft using coordination tables. In *2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)* (pp. 1-9). IEEE.

[9] Barnier, N., & Allignol, C. (2009, June). 4D-trajectory deconfliction through departure time adjustment. In *8th USA/Europe Air Traffic Management Research and Development Seminar (ATM 2009)* (pp. 1-10).

[10] Sedov, L., & Polishchuk, V. (2018). Centralized and distributed UTM in layered airspace. In *8th International Conference for Research in Air Transportation (ICRAT 2018)* (pp. 1-8).

[11] Sunil, E., Ellerbroek, J., Hoekstra, J., & Maas, J. (2017). Modeling airspace stability and capacity for decentralized separation. *DEP*, *vol. 3*, R1.
[12] Hoekstra, J. M., Maas, J., Tra, M., & Sunil, E. (2016, June). How do layered airspace design parameters affect airspace capacity and safety?. In *Proceedings of the 7th International Conference on Research in Air Transportation (ICRAT 2016)* (pp. 1-8).
[13] Bulusu, V., Polishchuk, V., & Sedov, L. (2017, November). Noise Estimation for future large-scale small UAS Operations. In *INTER-NOISE and NOISE-CON Congress and Conference Proceedings* (Vol. 254, No. 2, pp. 864-871). Institute of Noise Control Engineering.
[14] Bulusu, V., Sengupta, R., Polishchuk, V., & Sedov, L. (2017, June). Cooperative and non-cooperative UAS traffic volumes. In *2017 International Conference on Unmanned Aircraft Systems (ICUAS)* (pp. 1673-1681). IEEE.
[15] Bulusu, V., Sengupta, R., & Liu, Z. (2016, June). Unmanned aviation: To be free or not to be free?. In *7th International Conference on Research in Air Transportation (ICRAT 2016)* (pp. 1-8).
[16] Zanin, M., & Lillo, F. (2013). Modelling the air transport with complex networks: A short review. *The European Physical Journal Special Topics*, *215*(1), 5-21.
[17] Marx, D. (2004). Graph colouring problems and their applications in scheduling. *Periodica Polytechnica Electrical Engineering*, *48*(1-2), 11-16.
[18] Barnier, N., & Brisset, P. (2004). Graph coloring for air traffic flow management. *Annals of operations research*, *130*(1-4), 163-178.
[19] Galinier, P., Hamiez, J. P., Hao, J. K., & Porumbel, D. (2013). Recent advances in graph vertex coloring. In *Handbook of optimization* (pp. 505-528). Springer, Berlin, Heidelberg.