A DECISION SUPPORT TOOL FOR STRATEGIC CONFLICT MANAGEMENT THROUGH ASSIGNMENT OF CALCULATED TAKE-OFF TIMES

Nina Schefers[(a)], Juan José Ramos González [(b)], Pau Folch[(c)]

[(a),(b)] Department of Logistics and Aeronautics, Universitat Autònoma de Barcelona, Spain
[(c)] Department of Research and Innovation, Aslogic, Spain

[(a)] NinaRebecca.Schefers@uab.cat, [(b)] JuanJose.Ramos@uab.cat
[(c)] pfolch@aslogic.es

ABSTRACT
Collaborative Decision Making (CDM) is part of the "Single European Sky" program for optimizing airspace and airport operations. It emphasizes the coordination of processes, the sharing of accurate information among agents and the improvement of real-time data exchange between airports and the Air Traffic Flow Management (ATFM) network. This enhanced cognitive decision making process supports the global performance ambitions for air traffic optimization. In this paper, an advanced tool is presented that enhances the design of Decision Support Tools (DST) by identifying concurrence events at network level to readjust the aircraft take-off times within their assigned nominal Calculated-Take-Off Time (CTOT) margins on ground. The overall goal is to reduce the probability of separation minima infringement. The tool is capable to identify concurrence events at 3D level and to filter the tightest concurrence events for each pair of aircraft. Furthermore, an efficient analysis method based on graph theory to cluster the detected concurrence events is presented to ensure an efficient conflict resolution.

Keywords: Air Transportation, Decision Support Tools, Conflict Detection, Air Traffic Management, Graph

1. INTRODUCTION

According to the Single European Sky program SESAR, one of the elements in the European air transport value chain that should be improved and innovated is the Air Traffic Management (ATM) due to its limitations in capacity and its costs. The key areas of European air traffic performance optimization lay in environmental sustainability, capacity improvement, cost efficiency, operational efficiency, safety and security. To support these global key performance areas, the ATM sector has defined focus areas to introduce changes and implement optimization techniques. The areas include an optimized ATM network services, high-performing airport operations, advanced air traffic services and improved aviation infrastructure. (Sesar 2015)

In this paper, we address all ATM key performance areas and propose an innovative CDM methodology to improve the ATM performance based on the concept of Trajectory Based Operations (TBO) using Reference Business Trajectories (RBT). The approach focuses on improving the air traffic dynamic demand capacity balance by using means of the prompt identification of concurrence events at network level and by readjusting the take-off times within the assigned nominal Calculated-Take-Off-Time (CTOT) margins of [-5, 10] minutes. This way, the amount of Air Traffic Control (ATC) interventions could be minimized by rearranging the departing sequence of aircraft at the involved airports. The approach can be considered as a short-term Air Traffic Flow and Capacity Management (ATFCM) measure, applied at local level and reducing traffic peaks for the whole European airspace.

The objectives and benefits of this approach are aligned with the Single European Sky ATM Research Programme (SESAR) and can be summarized as follows:

1. Reduction of the probability of separation minima infringement: The approach is based on RBTs that provide an excellent source of information to identify long time in advance situations in which 2 or more aircraft could require ATC directives to maintain the required separation minima. Applying the mitigation mechanism, robust clearances at hotspots for a certain rate of predicted conflicts could be achieved. This way we create a robust traffic in which a reduced amount of ATC interventions is considered as part of the solution.
2. Enhancement of Airport Collaborative Decision Making (A-CDM) processes: The tool, presented in this paper, will contribute to a smooth integration of the different DSTs implemented at airport level in the ATM system, in which information about turnaround and taxi-out delays could be used for a better use of airspace resources.
3. To improve Air traffic Navigation Service Provider (ANSP) predictive workload: The TBO mapping tool presented in this paper provides more accurate traffic information in terms of the management of the flight position in which task load at sector level could be estimated at micro-level.

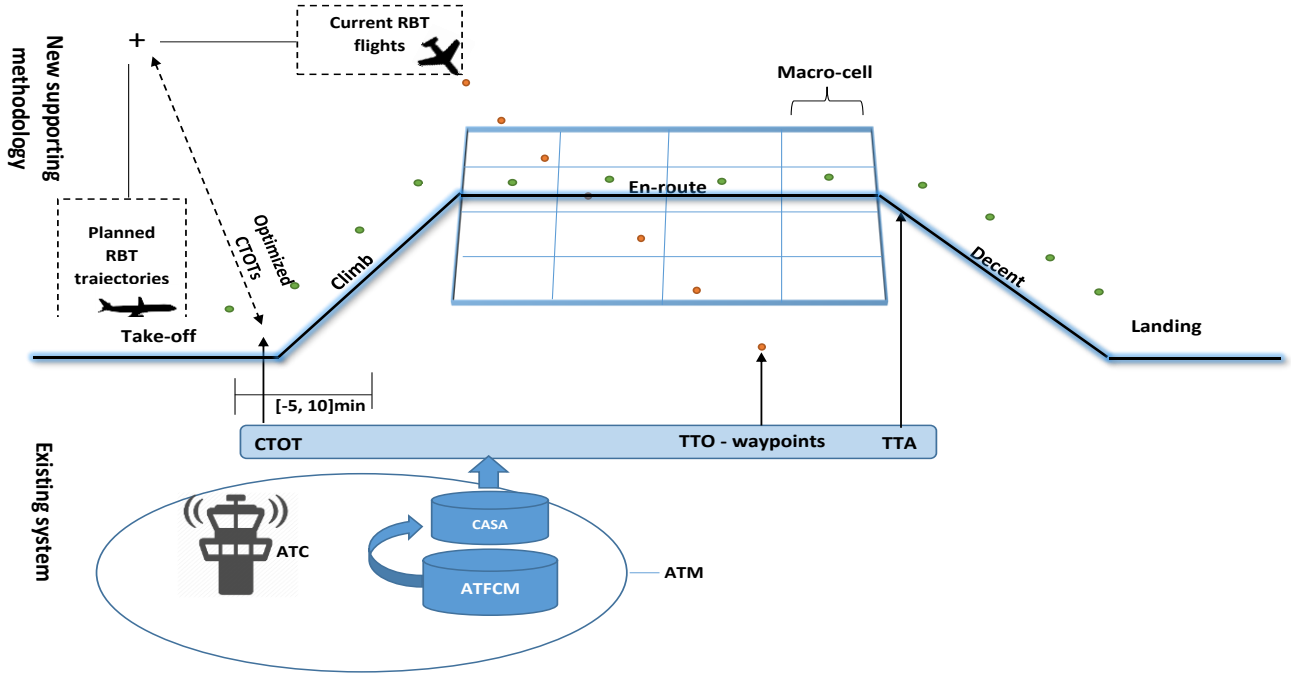

Figure 1: Aeronautical background in context of the used methodology

1.1. Aeronautical background

Europe has some of the busiest airspace in the world, compiled from 44 member states united in the European Civil Aviation Conference (ECAC) region. To safely operate the demand, any Airspace User that intents to depart from, arrive at or overfly one of the ECAC member states must submit a flight plan that must be approved in advance. Once the flight plan has been approved, its term changes to Reference Business Trajectory (RBT) and the aircraft is authorized to proceed in accordance with the agreed RBT consisting of predefined conflict free segments.

In Europe, an Air Traffic Flow and Capacity Management (ATFCM) service has been established to use the given capacity to the maximum extent possible keeping in mind the guiding principles safety, continuity and expeditious for the flow of air traffic. Information can be retrieved from the System Wide Information Management (SWIM) platform, an advanced technology program designed to facilitate greater sharing of ATM system information, such as airport operational status, weather information, flight data, and airspace restrictions.

Integrated in the ATFCM service is the Computer Assisted Slot Allocation (CASA) system that operates under the "First-Planned, First-Serve" policy. As it can be seen in Figure 1, the CASA system calculates the estimated Time to Overfly (TTO) for each point of entry in each sector and provides the Calculated-Take-Off-Time (CTOT) that must be followed within a slot window. (Cook 2007)

Attempting to improve the slot situation, new information processes and systems are under development to meet the current European capacity demands. The goal hereby is to improve the flight planning process and the supporting systems to create shorter routes, reduce emissions, reduce delays and improve the connectivity of trajectories. Thus, the ATFCM adherence measured at its efficiency and safety level can be revealed to decrease the overload of ATC workload in dense sectors.

To draw a connection between ATFCM and ATC as two components of the ATM, the concept of TBO was introduced. Short Term ATFCM Measures (STAM) tools and functionalities that smooth sector workloads by reducing traffic peaks through e.g. short-term application of minor ground delays [-5,10] minutes, rely on this TBO framework. The result is a synchronization of the trajectory prediction ensuring consistency between the trajectory and generic constrains that originate various ATM components and the various regions that shape this trajectory. Furthermore, it fosters the ground delay approach over the en-route delay approach since studies have reveal that holding aircraft on the ground contributes to less fuel consumption, less emissions and represents one of the simplest ways to leverage ATC workload as stated in (Barnier & Allignol 2008) and (Envisa 2017).

By empowering the concept of TBO as a flexible synchronization mechanism towards an efficient and competitive ATM service, a precise description of an aircraft path in space and time can be retrieved. Under this approach, airspace users should fly precise 4-

dimensional trajectory (4DT) paths, previously agreed upon with the network manager and in consistency with the agreed RBT.

In this paper, an advanced tool is presented that is compatible with the above described ATM tools and services to support the SESAR optimization objectives. The proposed algorithm allows the analysis of en-route trajectory interactions and can detect overlapping time windows to identify concurrence events between two or more aircraft on a 3D level (including different flight levels), see chapter 3. The approach is based on the idea to map the European airspace onto a grid and to identify "collective micro-cells" whose initial 2D concepts were studied in (Nosedal et al. 2014) and (Barnier & Allignol 2012). Furthermore, the computational efficiency and assurance of compatible conflict resolutions is of great importance. Therefore, an innovative methodology based on graph theory is proposed in chapter 4 to cluster the identified concurrence events into independent sub-graphs. This could support a mitigation tool to find fast and robust scheduling solutions measured by the amount of reduction ATC interventions.

2. CONFLICT DETECTION METHODOLOGY

The conflict detection is composed of two processes. First, the calculation of "overlap times" of aircraft within one microcell and second, the filtering of the tightest concurrence events, see chapter 2. Later, a graph theory based analysis is applied to cluster the hotspots of the detected concurrence events and to provide a robust set of data to a mitigation tool to reschedule the CTOTs of the detected aircraft, see chapter 3.

2.1. 3D Conflict detection

To detect the different ''collective microregions'' throughout the entire European airspace, each en-route trajectory is initially projected onto a discrete grid (square microcells of 6NM ground size) spanning longitudes of -20 to 30 degrees and latitudes of 0 to 80 degrees, representing the European airspace as presented in (Nosedal et al. 2014; Schefers et al. 2017).

After the initial mapping, microcells (cells of 6NM x 6NM ground size) with an occupancy rate equal or greater than two aircraft simultaneously are identified, as described in Figure 2. Then, for each aircraft pair, the "clearance time'' or ''overlap time'' is computed and the entry and exit times are stored to identify the aircraft that share one "microregion" as it can be seen in Figure 3.

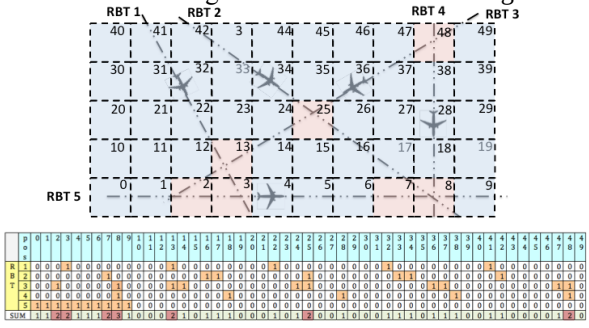

Figure 2: Occupancy rate matrix

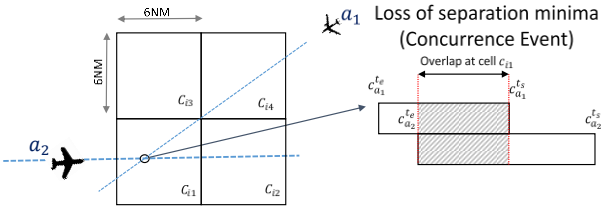

Figure 3: Conflict detection

To avoid missing concurrence events in neighbouring cells, a neighbourhood analysis is performed using a shifting process. To improve the reliability of the concurrence event identification, the original microcell will be shifted by 0,1 degree up, then 0,1 degree to the right, then 0,1 degree sown and back to its original position.

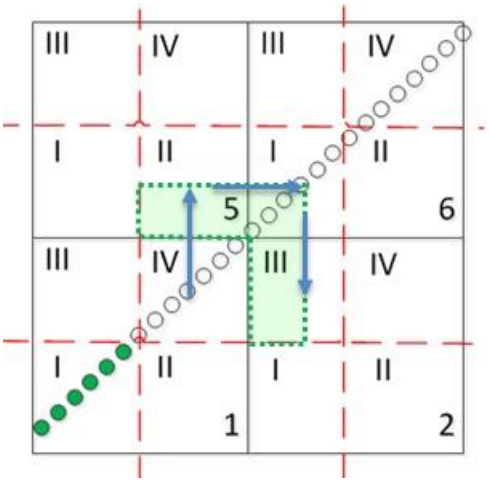

Figure 4: Neighborhood Analysis

The above presented approach described a 2D concurrence event detection approach. This article extends the 2D concurrence event detection approach and introduces a 3D concurrence event detection method. Therefore, the different flight levels must be integrated to achieve a multi-level mapping. The world's airspace is divided into three-dimensional segments that have been divided into different classes that are fundamentally defined by the International Civil Aviation Organization (ICAO)(Alphaaditya n.d.).

The approach that has been developed represents a rather conservative but safe method. Considering an aircraft a_i that changes its flight level from flight level FL_i to FL_{i+1}, the trajectory will be represented on both flight levels during the climbing manoeuvre, see Figure 5. If now an aircraft a_j causes a concurrent event during the climbing process of a_i in FL_i it might happen that a_i is already about to approach FL_{i+1} and no real conflict exists. On the other hand, this approach can be considered as very safe which is a primary objective in aeronautics.

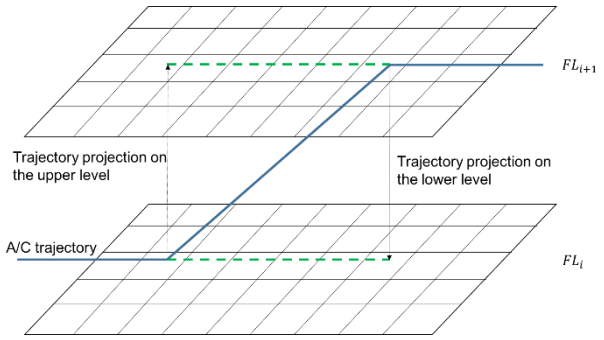

Figure 5: 3D Multi-flight level mapping

After computing all microcells with an occupancy rate of 2 or more aircraft, the output of the detection process is a list of all concurrence events, with all pairwise conflicts. The detection algorithm has $O(n \times m)$ complexity.

2.2. Filtering process

In the previous chapter it was mentioned that the entry and exit times of aircraft into a cell are stored. Now, these values are retrieved to calculate the temporal looseness of aircraft in one particular cell. As mentioned in (Nosedal et al. 2014) the temporal looseness H of two aircraft can be calculated by determining the minimum value of the exit time of the two aircraft and subtract the maximum entry time from this value.

$$H = \min_{t_j}\left(t_{j_x}; t_{j_y}\right) - \max_{t_i}(t_{i_x}; t_{i_y}) \quad (1)$$

In Equation 1, Min is a function that determines the minimum exit time t_j of two aircraft t_{j_x} and t_{j_y} in one cell. Max is a function that determines the maximum entry time t_i of two aircraft $t_{i_x}$and t_{i_y} in one cell.

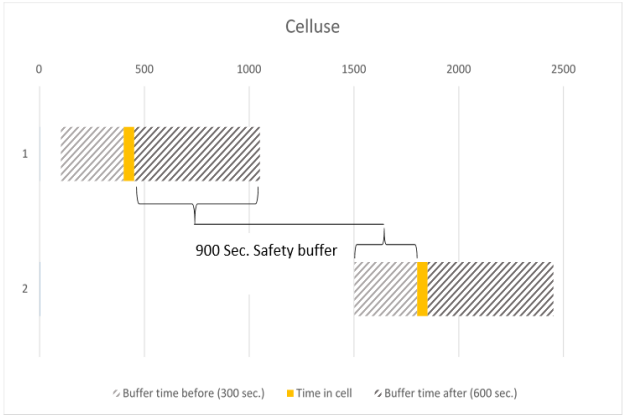

Figure 6: Celluse of one Microcell. X-Axis: Time in sec. Y-Axis: Aircraft

The detected concurrence events are filtered for each pair of aircraft. The result after the filtering process are tightest potential concurrence events for each pair of aircraft. Since the domain for rescheduling the CTOT is restricted to [-5,10]min, the greatest value is 15min (900sec). Therefore, the list of tightest potential occurrence events can be cut by all values exceeding this timeframe because their spatial separation is so great that even a maximum CTOT shift (900 sec.) would not have any effect to endanger the safety separation, see Figure 6.

The idea behind this process is to focus on the endangered pair of aircraft that could lose separation minima. Furthermore, this process makes it feasible to develop a conflict detection methodology, capable to outperform present NP-Hard algorithms (mainly pairwise oriented) and new Polynomial algorithms such as SDS (Spatial Data Structure) with a performance sensible to scalability problems.

3. CONFLICT DETECTION ANALYSIS

In order to provide a robust set of data for the mitigation tool capable to reschedule the initial CTOTs, this paper proposes an efficient analysis to identify those trajectories that will reduce the maximum clearance in order to provide a more robust departure coordination solution. The analysis is based upon Adjacency Matrix properties and Depth-first search (DFS) algorithm, connecting components are extracted from the initial graph in order to be processed independently. Such a formulation based on graphs allows representing in single manner concurrence and coupling interdependencies between detected pairwise conflicts obtained from the mapping and filtering tools as it can be seen in Figure 7.

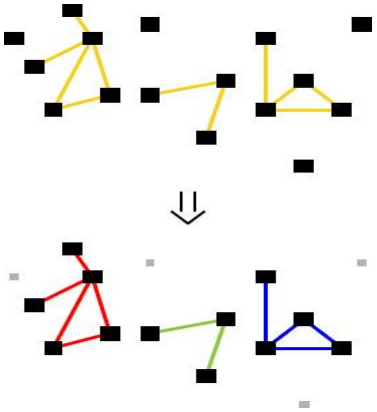

Figure 7: Graph representation of coupled concurrence events

This section is organized as follows: chapter 3.1 will introduce the preliminary definitions needed to understand the graph theory that was applied to analyze the conflict interdependencies. Chapter 3.2 describes methods of graph representations. Finally, chapter 3.3 explains the concept for the analysis of conflict interdependency based on graph theory.

3.1. Preliminary definitions for graph theory

A graph $G(V, E)$ is a triple consisting of a vertex set $V(G)$, an edge set $E(G)$, and a relation that associates with each edge two vertices called its endpoints (not necessarily distinct) see Figure 8.

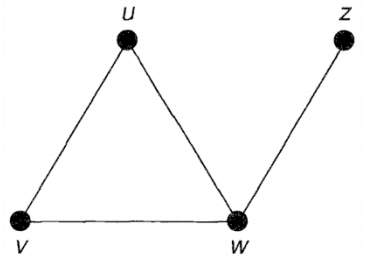

Figure 8: Simple graph

For our purposes, all graphs will be finite graphs where $V(G)$ and $E(G)$ are finite sets. However, this definition does not exclude the possibility that two endpoints of an edge are the same vertex (which is called a loop) and we may have multiple edges. A simple graph $G(V, E)$ is a graph having no loops or multiple edges.

For a graph G, we denote, with $v_G = |V(G)|$ and $\varepsilon_G = |E(G)|$. The number of the vertices v_G is called the order of G, and ε_G is the size of G. Vertices u and v are adjacent or neighbours if $uv \in G$, and u and v are then incident with such an edge, see Figure 9. Similarly, two edges $e_1 = uv$ and $e_2 = uw$ having a common endpoint are adjacent with each other.

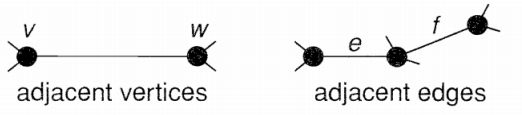

Figure 9: Adjacency in vertex and edges

For our purposes, we need to introduce also the concept of connectedness. Two graphs can be combined to make a larger graph. If the two graphs are $G_1 = (V(G_1), E(G_1))$ and $G_2 = (V(G_2), E(G_2))$, where $V(G_1)$ and $V(G_2)$ are disjoint, then their union $G_1 \cup G_2$ is the graph with vertex set $V(G_1) \cup V(G_2)$ and edge family $E(G_1) \cup E(G_2)$.

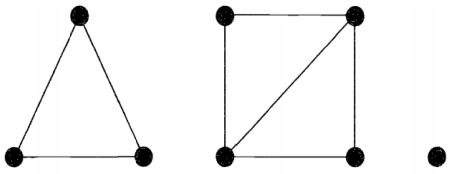

Figure 10: Disconnected graph with three components

A graph is connected if it cannot be expressed as the union of two graphs and disconnected otherwise. More formally, a graph G is connected if for every $u, v \in V(G)$ there exists a u,v-path in G. Otherwise G is disconnected. The maximal connected subgraphs of G are called its components. (Trudeau & Trudeau 1993)

Finally, in a formal way, the degree of a vertex v of G can be defined as follow: Let $v \in G$ be a vertex a graph G. The neighborhood of v is the set:

$$N_G(v) = \{u \in G \mid vu \in G\} \quad (2)$$

The degree of v is the cardinality of its neighborhood:

$$d_G(v) = |N_G(v)| \quad (3)$$

If $d_G(v) = 0$, then v is said to be isolated in G, and if $d_G(v) = 1$, then v is a leaf of the graph. The minimum degree and the maximum degree of G are defined as:

$$\delta(G) = \min\{d_G(v) \mid v \in G\} \quad (4)$$
and
$$\Delta(G) = \max\{d_G(v) \mid v \in G\}$$

3.2. Graph representations

The two main graph representations used in graph problems are the adjacency list and the adjacency matrix. An adjacency list is a list of lists. Each list corresponds to a vertex u and contains a list of edges uv that originate from u (the neighborhood of u). Thus, an adjacency list takes up $O(V + E)$ space.
An adjacency matrix is a $|V| \times |V|$ matrix of bits where element (i, j) is 1 if and only if the edge $v_i v_j$ is in E, and 0 otherwise. Thus, an adjacency matrix takes up $O(|V|^2)$ storage (note that the constant factor here is small since each entry in the matrix is just a bit).

The worst-case storage of an adjacency list is when the graph is dense, i.e. $E = (|V|^2)$. This gives us the same space complexity as the adjacency matrix representation. The $O(|V| + |E|)$ space complexity for the general case is usually more desirable, however. Furthermore, adjacency lists give you the set of adjacent vertices to a given vertex quicker than an adjacency matrix $O(neighbors)$ for the former vs. $O(|V|)$ for the latter.

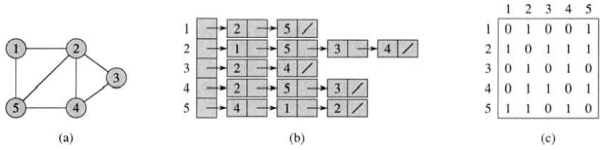

Figure 11: Two representations of an undirected graph. (*a*) An undirected graph G having five vertices and seven edges. (*b*) An adjacency-list representation of G. (*c*) The adjacency-matrix representation of G.

One of the most fundamental problems in graph theory is graph traversal (also known as graph search). This problem refers to the process of visiting (checking and/or updating) each vertex in a graph. There are two standards of traversing all vertices/edges in a graph in a systematic way: Depth-First Search (DFS) and Breadth-First search (BFS).

The main idea of the Depth-First Search algorithm is to make a path as long as possible, and then go back (backtrack) to add branches also as long as possible. A

complete DFS ends when we traverse back to the root and we have visited every vertex or when we have found the desired edge/vertex. During the search DFS divides the edges of *G* into tree edges and back edges. Obviously, the tree edges form a spanning tree of *G*, also known as a DFS-tree. (Chartrand & Zhang 2006)(Gibbons 1985)

3.3. Formulation for conflict interdependencies

A set of pairwise concurrence events between aircraft is produced by mapping and filtering tools as described in chapter 2. The final output of these tools is a set of pairwise potential concurrence events. This list is made by the following information:

1. An identification number of the cell where the potential concurrence events occurs.
2. The Flight Level where the concurrence event is detected.
3. Identification numbers of the two involved aircraft.
4. The times t_s and t_e of the two aircraft.
5. And two Boolean values that describes if the involved aircraft will take off before the time window analyzed or not. This information will be used in the mitigation phase in order to detect in which aircraft we are able to do a shifting in its CTOT and in which not.

The process to detect potential concurrence events analyses a scenario that can contain potential conflicts between more than two aircraft. However, the obtained pairwise list of potential conflicts does not represent intuitively the real state space of the processed scenario. Hence, we need new formulation able to correct this issue and, at the same time represent efficiently possible interdependencies between these listed pairwise potential conflicts.

Therefore, the potential concurrence events detected in one cell at the same flight level will form a node or vertex of a graph and its edges will represent the interdependencies between potential conflicts.

More formally, the nodes of graph $G(V,E)$ will be constructed in the following way:

1. By using the output list aforementioned, the potential conflicts are grouped by cells and by flight level. This way, clusters of potential conflicts that share physical location in the space are formed. Notice that the cell fixes the Cartesian coordinates x and y, and Flight Level (FL) fixes the last coordinate z, which represents the altitude.
2. Once the clusters are made, we must distinguish which pairwise conflicts really form a unique potential concurrence event using time information. This step is conceptually important because it transforms the spatial representation of the potential conflicts induced by the grid to an adimensional representation.
   To distinguish between potential conflicts in a cluster we use the following criterion:
   a) Create an empty list L, and add to it the earliest potential conflict in the cluster.
   b) Search in the cluster if there is another pairwise potential conflict that involves one of the aircraft in the earlier conflict, let be a_i the shared aircraft (AC), and check that the times t_e and t_s of a_i are the same in the two conflicts. Repeat this step until no more pairwise conflicts are found.
   c) Add all conflicts found in step b) in L, ordering them by earlier t_e.
   d) Repeat b) and c) considering this time the earlier element of L not used yet. Repeat until all elements in L has been cheeked.
   e) Construct using list L an ordered list of AC by its t_e.

   By regrouping the elements in the output list the potential concurrence events that really occurs in the analyzed scenario independently of the number of AC involved can be reconstructed. (see Figure 12).

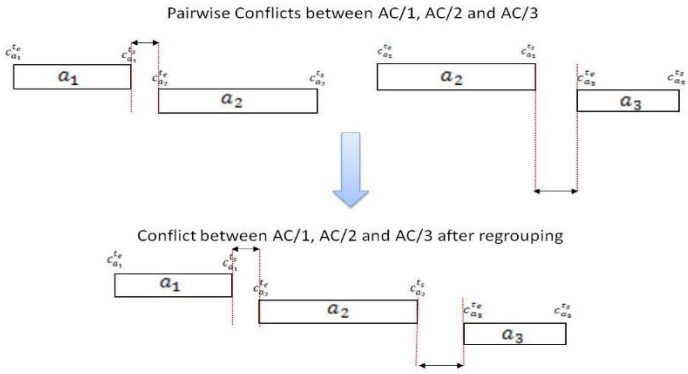

Figure 12: Regrouping two pairwise conflicts to form one conflict node

All the nodes in graph *G* are outputs of this procedure, thus, one node which represent one potential conflict could involve more than one aircraft. Furthermore, two different nodes may represent two potential concurrence events in the same cell, but involving different aircraft in different time.

Remembering, the main objective of the tool is to enhance the airspace demand-capacity balance by trying to reduce the number of potential concurrence events en-route. Towards this goal, concurrence interdependencies between aircraft trajectories are identified at the network level, and are removed by rescheduling take-off times in such a way that target times of arrival are preserved within a one-minute margin. Then, the interdependencies between potential conflicts are in some sense the repercussions that the rescheduling takes-off times could produce later at en-route phase.

To construct *G*, we need to define also the edge set $E(G)$. As aforementioned, the edges in *G* must represent the interdependencies between potential concurrence events

and the rescheduling takes-off times. Hence, edges must be related to RBTs of the aircraft involved in the nodes of *G*.

Formally, we add to $E(G)$ an edge uv if and only if there are at least one aircraft which is involved in node u and in node v. That is an edge $uv \in E(G)$ if and only if u and v shares an aircraft.

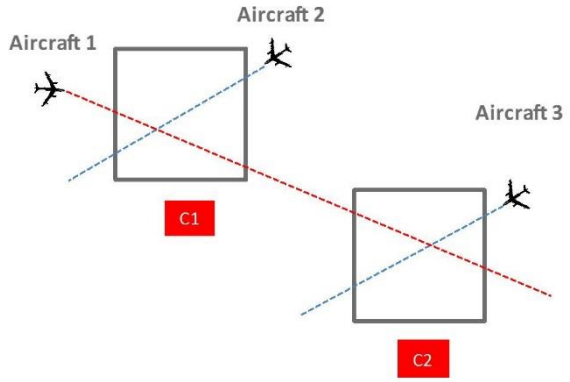

Figure 13: Two potential conflicts shearing aircraft 1.

Figure 13 shows two potential concurrence events that occur in different cells. Let be u the node of G that represents the potential conflict listed in cell c_1 and v the node for the conflict in c_2. Then, we must add to $E(G)$ the edge uv because u and v shares Aircraft 1.

The idea behind this is to remove by rescheduling take-off times the potential concurrence event encoded in u considering the modified CTOT of Aircraft 1 that may result in a reduction of the clearance H in v and vice-versa. In case where u and v represent only potential conflicts which means that there is a positive clearance H in both cases, a rescheduling take-off in Aircraft 1 may remove one potential conflict but, producing at same time a real conflict later on.

This formulation based on graphs allows representing real conflicts and their interdependencies, and visualize complex situations. For example, the connection of nodes v and w by a vw-path of length greater than 1 means that the potential conflicts in v and w even they do not share any aircraft there is interdependence between them which is given by some intermediate nodes that we must consider when rescheduling take-off times.

Once the formulation has been presented along this section we will then introduce a couple of definitions regarding the nature of the interdependencies.

The presented approach induces some differences between the elements listed in the output of the mapping and filtering tools. To be more precisely, we will differentiate between interdependencies that relate aircraft inside a node of *G* and those that relate conflicts by edges in *G*:

- Concurrent interdependencies are those which appear between aircraft that are in the same node in *G*. That is, potential concurrence events between two or more aircraft result in a concurrent interdependence between these aircraft. This definition induces to introduce a criterion or metric to measure the hardness of that interdependence.
- If there exist in *G* a uv-path between two nodes u and v each one encoding a potential conflict then, there is and interdependence between them. Since the resolution of one of them propagate some restrictions in the time stamp domains for the resolution of the other one. We namely this kind of interdependencies coupling interdependence.

Concurrent interdependencies must be removed from the system as much as possible but considering the existence of the coupling interdependencies. It is possible to expand these concepts by introducing criterions or metrics that help us describe the degree of these interdependencies. For the concurrent interdependencies, we introduce the saturation concept which refers to the trade-off between cell occupancy and cell capability. That is, we define the level of saturation as a metric to indicate how occupied is a cell in terms of its capability.

Finally, we introduce one more concept which is the coupling level that describes how much the propagation of one resolution between concurrent interdependencies may affect future resolutions. This metric deals with concepts like tight interdependence which appears in scenarios where the coupling interdependencies between the concurrent ones produce a complex over-constrained system where solutions maybe strongly relegated or even removed.

4. APPLICATION AND EXPERIMENTAL RESULTS

The model was applied to DDR2 data obtained from one day of traffic. The scenario is composed of a set of 2584 real 4D trajectories in the European airspace that reveals 4222 conflicts. In this work, we assumed TBO without uncertainties. In this context, the trajectories were discretized at each second, and each position was specified in terms of geographic coordinates and a time stamp.

Figure 7 at the beginning of this chapter showed graphically the idea of identifying connected components in *G*. In this way, the whole system is partitioned in independent sub problems of less size. This allows the mitigation tool, read more in (Schefers et al. 2017), to work faster and in a parallelizable way.

The way we carried out the partitioning of *G* is to modify the Depth-First Search algorithm to be able to extract the connected components of *G*. This minor modification is based on the idea of colouring each visited node of one connected component using the same colour and changing it each time the DFS starts to visit another non

visited node. At the end of the DFS phase we obtain which node belongs to each connected component and the number of components that *G* has.

	A	B	C	D	E	F	G	H	I	J	K
1	Total_File_Conflicts	Total_File_RBTs	# Connected C.								
2	4222	2584	68								
3	Cell_ID	Flight_Level	A1_ID	A2_ID	A1_Entry	A1_Exit	A2_Entry	A2_Exit	A1_Flying	A2_Flying	CComponent
4	1493	190	81668	79624	1483555319	1483555372	1483555464	1483555523	0	0	1
5	14546	270	81668	81932	1483555580	1483555629	1483555798	1483555846	0	0	1
6	14546	270	79624	81932	1483555745	1483555797	1483555798	1483555846	0	0	1
7	76473	220	79888	82028	1483560709	1483560769	1483560806	1483560817	0	0	2
8	136534	340	80507	82895	1483600132	1483600187	1483600297	1483600341	0	0	1
9	379434	330	78260	82927	1483576482	1483576521	1483576600	1483576631	0	0	-1
10	630650	360	83156	82373	1483572288	1483572336	1483572566	1483572610	0	0	3
11	667384	270	82845	83175	1483560833	1483560922	1483560904	1483561002	0	0	1
12	667385	290	82612	86493	1483570136	1483570228	1483570343	1483570350	0	0	4
13	712440	190	76857	79398	1483548141	1483548154	1483548317	1483548352	0	0	1
14	712440	190	78416	79398	1483548246	1483548312	1483548317	1483548352	0	0	1
15	715050	220	76857	78416	1483548208	1483548231	1483548318	1483548371	0	0	1
16	720270	240	78416	79677	1483548480	1483548490	1483548685	1483548687	0	0	1
17	805712	370	80313	81585	1483568480	1483568530	1483568507	1483568552	0	0	1
18	828993	390	84979	82362	1483576955	1483576998	1483576970	1483576975	0	0	3
19	883516	340	83995	84428	1483586465	1483586518	1483586467	1483586518	0	0	1
20	912838	190	85970	82790	1483565446	1483565507	1483565455	1483565516	0	0	1

Figure 14: Output of the analytic tool

The output of the analytic tool has one more column than the data list created in the mapping phase. That column determines in which component belongs each pairwise conflict and can be seen in column K in Figure 14. If the potential conflict is an isolated node in *G* the value is set to -1.

4.1. Experiments

The mitigation of the conflicts is tested by a Constraint Programming model (read more in has been implemented with the ILOG Optimization Suite (IBM 2015). The optimization goal introduced in the mitigation tool which is formulated by the means of Constraint Programming is set up in a way that the solution searches to maximize the minimum clearance. This way, the minimum distance between two aircraft that take part in a concurrence event is maximized as much as possible. The idea behind this is to provide a set of safe trajectories that reduce the possibility of an ATC intervention. The following results were obtained:

As it can be seen in Figure 15, there are in total 68 subgraphs. The proportion of the amounts of relating components in *G* show that most subgraphs consists of one isolated node in G, followed by 24 subgraphs composed of 2 nodes and 10 subgraphs composed by either 3 or 5 nodes. Subgraphs that are constructed out of more than 5 relating components and up to 30 related components only occur once or twice in the data series. Finally, there is one subgraph that has the most components which is subgraph 1 with 3721 nodes, see Figure 16.

First, the set of data was analyzed without making use of the in chapter 3 described analysis tool. The result of solving the 4222 conflicts achieve the maximum minimum clearance of 2 seconds.

In a second experiment, the data output of the analysis tool that can be seen in Figure 14 was applied. In the data, it can be seen which pairwise conflicts belong to each subgraph and the number of connected component that the graph has. The results that were achieved can be seen in Table 1:

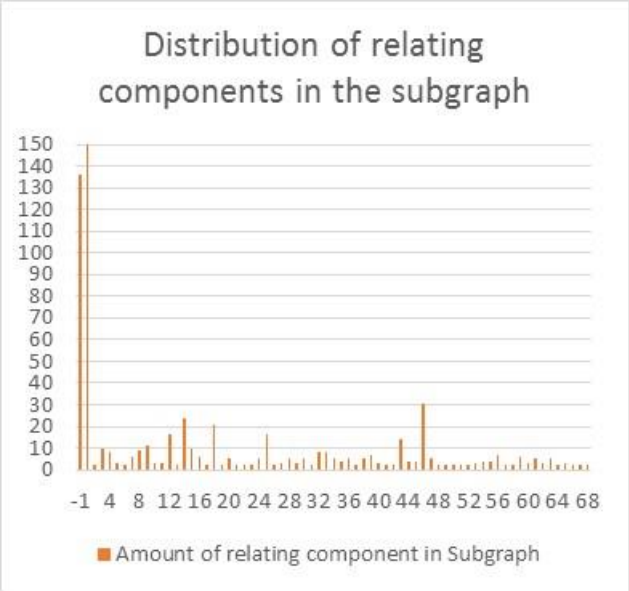

Figure 15: Amount of relating components for each subgraph in *G*

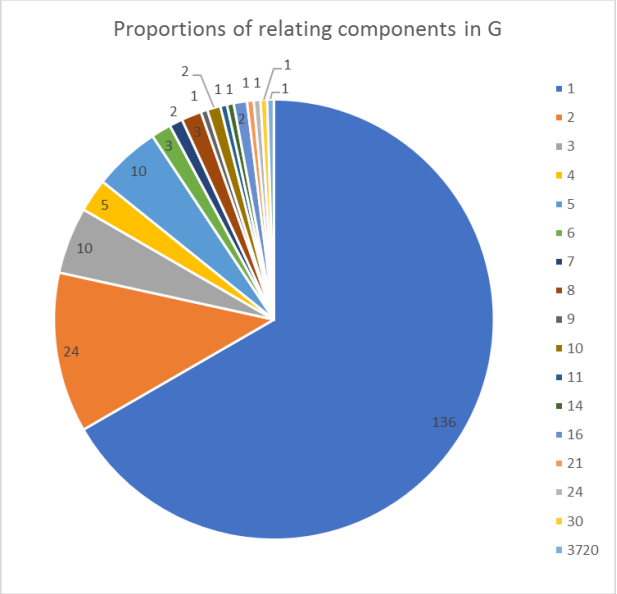

Figure 16: Proportion of distribution of relating components in *G*

Connecting Components	Max. Min. Clearance in sec.	Running Time
3721	2	00:09:26:65
30	119	00:00:06:39
24	147	00:03:05:12
21	21147	00:03:06:02
16	16144	00:00:06:62
14	14280	00:00:07:64
11	11420	00:00:06:13
10	10537	00:00:10:14
9	362	00:00:05:85
8	389	00:00:05:66
7	478	00:00:05:87

6	729	00:00:08:37
5	328	00:03:04:62
4	322	00:00:05:75
3	1176	00:00:05:46
2	1525	00:00:04:87
Without analysis tool	2	00:15:49:80

Table 1: Results of the experiments

Solving the subgraph with 3720 connecting components also achieves a maximum minimum clearance time of 2 seconds. This is no improvement with respect to the first experiment, however, there are still 501 pairwise conflicts remaining distributed among different subgraphs. Solving the second biggest subgraph combined of 30 nodes already achieves a clearance time of 119 seconds. The biggest clearance time of 1525 seconds was achieved in a subgraph with 2 nodes.

4.2. Solution Analysis

There is a significant improvement in both, achieved clearance times and running time using the analysis tool based on graph theory. As it can be seen in Figure 17, the bottleneck of the maximum minimum clearance time lays in the subgraph with 3721 nodes. All other subgraphs achieved a great improvement in their clearance time.

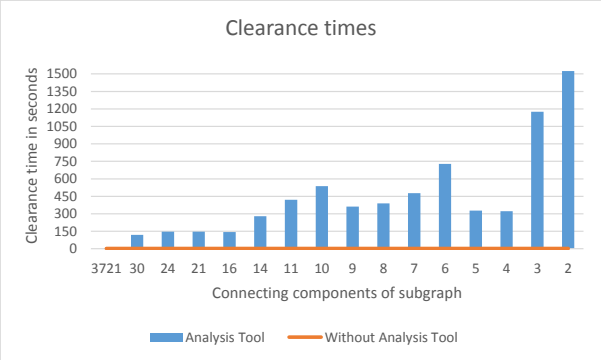

Figure 17: Clearance times

By dividing the whole problem into sub problems using graph theory, the clearance times within the sub problems can be significantly improved.

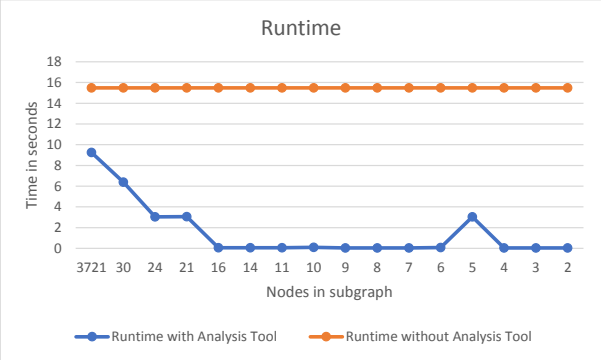

Figure 18: Runtime

Furthermore, the runtime of each sub problem can be drastically improved as it can be seen in Figure 18. While executing the whole problem continuously takes 15 minutes and 49 seconds, the runtime can be improved to 9 minutes and 26 seconds for solving the subgraph with 3721 nodes and subgraphs with only 2 nodes which represent the second biggest type of subgraph in the problem (see Figure 16) could be solved within 4 seconds.

5. CONCLUSION AND FUTURE RESEARCH

In this work a powerful analysis tool is presented based on graph theory. The model was applied on DDR2 traffic data and has been able to highly improve the maximum minimum clearance time of aircraft.

The analysis tool translates pairwise potential concurrent events described by using 4 dimensional coordinates into a planar representation allowing its visualization. This simplification is itself a useful tool to graphically analyses the whole system, since the information encoded in the mapping output list is presented now as an interdependence graph.

Furthermore, the graph representation allows identifying concurrent and coupling interdependencies, discarding useless information such as in which cell potential concurrence events take place.

Moreover, finding connected components reduce the problem size respecting all the identified interdependencies. The partitioning of the system does not eliminate or add any solution, being the solution space after partitioning the problem the same as before. Furthermore, this partitioning on the mitigation phase allows finding better solutions in less time.

Finally, induced metrics such as the saturation level and the coupling level can be extracted from the intrinsic information encoded in the graph representation and used later by the mitigation tool. That is, saturation level will be a function of the number of aircraft occupying in a particular cell, in other words, this level will be the size of the potential conflict encoded by the node. The coupling level of one aircraft, which will be used as a weight in the objective function of the constraint programming model for the resolution phase, will be a function of the degree of the nodes where it passes and its path length.

Regarding future research, up to now, two different research topics that require further developed/ discussion were identified. First, the objective function of the mitigation tool, on which the output of the analysis based on graph theory is based, could be optimized. There should be found a metric that defines and calculates the weight distribution that can be used for the objective function and therefore guides the search. This parameter depends on the characteristics of the graph theory as for

example the degree of the node, the size of the connected component, the length, the saturation level etc.

Furthermore, a criterion should be defined that clearly states how aircraft that cause a certain level of tightness are treated and delegated. As an example, either, special rules like relaxing the aircraft' domain could be applied for an aircraft that is imperiling the clearance or the identified aircraft could be delegated to ATC.

6. ACRONYMS

AC	Aircraft
ATC	Air Traffic Control
ATFM	Air Traffic Flow Management
ATFCM	Air Traffic Flow and Capacity Management
ATM	Air Traffic Management
AU	Airspace User
BFS	Breadth-First search
CASA	Computer Assisted Slot Allocation
CP	Constraint Programming
CSP	Constraint Satisfaction Problems
CTOT	Calculated-Take-Off Time
DFS	Depth-First Search algorithm
DST	Decision Support Tool
ECAC	European Civil Aviation Conference
FL	Flight Level
JSSP	Job Shop Scheduling Problem
TBO	Trajectory Based Operation
TTA	Target Time of Arrival
TTO	Time-To-Overfly
RBT	Reference Business Trajectory
STAM	Short Term ATFCM Measures
SWIM	System Wide Information Management
4DT	4-dimensional trajectories

ACKNOWLEDGMENTS

This research is supported by the European Union' s Horizon 2020 research and innovation program. The Project related to this research is PARTAKE "Cooperative departures for a competitive ATM network service" with Grant Agreement No. 699307. Opinions expressed in this paper reflect author's views only.

REFERENCES

Alphaaditya, Airspace classification - AviationKnowledge. Available at: http://aviationknowledge.wikidot.com/aviation:airspace-classification [Accessed July 14, 2017].

Barnier, N. & Allignol, C., 2008. Deconfliction with constraint programming. , p.pp xxxx. Available at: https://hal-enac.archives-ouvertes.fr/hal-00938416 [Accessed May 15, 2017].

Barnier, N. & Allignol, C., 2012. Trajectory deconfliction with constraint programming. , 27(3), pp.291–307. Available at: https://hal-enac.archives-ouvertes.fr/hal-00935206/document [Accessed April 23, 2017].

Chartrand, G. & Zhang, P., 2006. *Introduction to graph theory*, Tata McGraw-Hill Pub. Co.

Cook, A., 2007. *European air traffic management : principles, practice and research*, Available at: https://books.google.es/books/about/European_Air_Traffic_Management.html?id=vQ9LG6TWl9oC&redir_esc=y [Accessed May 11, 2017].

Envisa, 2017. Environmental Costs and Benefits of Ground Delay - Envisa - Aviation & Environmental Solutions. Available at: http://www.env-isa.com/en/expertise/environmental-costs-and-benefits-of-ground-delay-2/ [Accessed May 15, 2017].

Gibbons, A. (Alan M.., 1985. *Algorithmic graph theory*, Cambridge University Press.

IBM, 2015. *IBM ILOG CPLEX Optimization Studio OPL Language User's Manual*,

Nosedal, J. et al., 2014. An efficient algorithm for smoothing airspace congestion by fine-tuning take-off times. *Transportation Research Part C: Emerging Technologies*, 44, pp.171–184.

Schefers, N. et al., 2017. Causal Analysis of Airline Trajectory Preferences to Improve Airspace Capacity. *Procedia Computer Science*, 104, pp.321–328. Available at: http://www.sciencedirect.com/science/article/pii/S1877050917301424 [Accessed June 7, 2017].

Sesar, 2015. THE ROADMAP FOR DELIVERING HIGH PERFORMING AVIATION FOR EUROPE European ATM Master Plan. , Edition 20. Available at: http://ec.europa.eu/transport/modes/air/sesar/doc/eu-atm-master-plan-2015.pdf [Accessed July 18, 2016].

Trudeau, R.J. & Trudeau, R.J., 1993. *Introduction to graph theory*, Dover Pub. Available at: https://books.google.co.uk/books/about/Introduction_to_Graph_Theory.html?id=8nYH5OYEW24C&redir_esc=y [Accessed July 14, 2017].