MULTI-AGENT BASED SOLUTION FOR FREE FLIGHT CONFLICT DETECTION AND RESOLUTION USING PARTICLE SWARM OPTIMIZATION ALGORITHM

Hojjat Emami[1], Farnaz Derakhshan[2]

The future management of air traffic allows aircraft choosing their own velocities, altitudes and directions in real time. In aviation industry, this possibility is known as "free flight". One of the most important issues in the free flight method is conflict detection and resolution problem. In successful free flight, conflicts while maintaining satisfactory performance must be avoided. This paper, presents a multi-agent based conflict detection and resolution approach for free flight. In this paper, aircraft and ground flight path controllers are selected as agents, respectively called Aircraft Agent or "AA" and Flight Path Controller Agent or "FPCA". This type of agent selection provides a proper balance between distributed and centralized authority in order to solve air traffic conflicts and this is one of the advantages of our proposed system. FPCA agents map the situation of traffic in their vision domain to a flow graph using negotiation with aircraft agents. After some conversion on mapped graph, agents color the corresponding graph using Particle Swarm Optimization (PSO) algorithm. The proposed method is implemented and tested by using five well-known test cases. The experimental results show the high capability and efficiency of our approach for conflict problem. The advantages of using our proposed system includes delay reduction, passenger comfort, safety and speed increase, travel time reduction and less fuel consumption. This system not only proposed for free flight but also can use along with the current air traffic management systems without completely replaces them.

Keywords: Aircraft conflict detection and resolution, free flight, graph coloring problem, particle swarm optimization algorithm

1. Introduction

The management of air traffic is a complex and dynamic problem. In recent years the aviation industry faced with many problems that impose many losses, such as inefficiency in managing traffic, long delay time in flights, more fuel consumption and many other important challenges. The free flight concept has been introduced as a potential solution to these problems.

[1] Dept. of Computer Engineering, Islamic Azad University, Miandoab Branch, Miandoab, Iran, email: hojjatemami@yahoo.com

[2] Faculty of Electrical and Computer Engineering, University of Tabriz, Tabriz, Iran, email: derakhshan@tabrizu.ac.ir

The free flight is an innovative and alternative concept introduced by NASA and FAA[3] in which pilots have more freedom to select their own route, speed and altitude in real time [1–4]. The free flight method has many advantages such as increase the efficiency of airspace usage, reducing fuel costs and travel time. Despite these advantages, one of the main challenges in free flight method is the prevention of losses of separation, that this issue is known as conflict problem [5].

In recent years, many approaches have been proposed to deal with conflict detection and resolution problem when many aircraft are involved; including centralized and decentralized methods. In 2000, Kuchar and Yang in [6] presented a complete overview of the proposed approaches. While classical models for air traffic management have good capability, but they are often centralized. Some of classical approaches include Lagrangian models (e.g. in [7, 8]), Eulerian models (e.g. in [9, 10]) and many other optimization and mathematical methods (e.g. in [11, 12]). Lagrangian model for air traffic flow management systems involves computing the trajectories for individual aircraft. Eulerian methods predict flow patterns in the airspace for aircraft.

Resolution of conflicts among n aircraft is a highly combinatorial problem [13] and can not be solved with classical approaches under realistic hypothesis. Therefore, in last few years, some researchers for solving conflicts between aircraft focused on multi-agent based and distributed approaches (e.g. in [14, 15]). A multi-agent based air traffic management systems applied different autonomous agents with different strategies. Some agents used learning strategies (e.g. Reinforcement Learning [16]), some others used game theory concept, and some others used algorithmic and optimization methods (e.g. mixed integer programming [17, 18], and many others). Here, the important issue is that autonomous agents attempt to perform their actions properly, improve system performance and finally propose a safe and optimal solution for conflict detection and resolution problem.

As the number of conflicts increases among n aircraft, we need to an automated system to prevent excessive workload for ground and centralized air traffic controllers. For this reason, in this paper we proposed a multi-agent approach for detecting and resolving conflicts between aircraft. In our proposed model, we selected flight path controller and aircraft as autonomous agents that respectively called Flight Path Controller Agent (FPCA) and Aircraft Agent (AA). Each FPCA agent has a predefined vision domain as a cylinder with 50 nautical miles radius and this agent has capability of interacting with the aircraft in its vision domain. The flight path controller agent is responsible to detect and resolve conflicts between aircraft in its vision domain. This agent maps the traffic situation in its vision domain to a flow graph. When a conflict is detected, this

[3] Federal Aviation Administration

agent uses the PSO algorithm to solve conflicts. In fact, the FPCA agent tries to map the conflict resolution process to the graph coloring problem with interact with the aircraft in its vision domain. Then, the agent uses the PSO algorithm to color this flow graph.

The novelty of this approach would be in using a PSO algorithm to select the best routes in conflict resolution process and discard the worst ones. Also the proposed method institutes a proper balance between decentralized and centralized authority. Moreover, the precision of conflict resolution procedure is guaranteed by utilizing the graph coloring problem concept. In this paper, we proposed a conceptual and abstract multi-agent model for conflicts problem, in addition, we implemented and tested our model on five well-known traffic patterns.

The rest of the paper is organized as follows. Section 2 describes the theoretical foundation for this paper. Section 3 presents our proposed conflict detection and resolution model in detail. Section 4 illustrates experimental results. Finally, Section 5 makes some conclusions.

2. Background

In this section, we briefly describe the theoretical foundation for this paper including the concept of multi-agent system (MAS), particle swarm optimization (PSO) algorithm and graphic coloring problem (GCP).

A. Multi-Agent System

A multi-agent system is a computational system and composed of multiple interacting intelligent agents in which agents using some communication languages interact with one another to achieve some delegated goals. Using of multi-agent systems for solving large and complex problems are one of the most successful and efficient solutions. A multi-agent system is an appropriate choice to solve problems which are difficult or even impossible to solve by using individual agents [19, 20]. Obviously, we can assume the management system of airspace as a multi-agent system in which agents cooperate with each other to manage air traffic safely and efficiently [21].

B. Particle Swarm Optimization Algorithm

Particle Swarm Optimization (PSO) Algorithm [22] developed by Kennedy and Eberhart (1995) is a population based and swarm intelligence based evolutionary algorithm. This algorithm which simulates the social behavior of a flock of birds flying to resource, introduced to deal with many optimization tasks [23, 24]. In the PSO algorithm, the main goal is to find the global optimum of a goodness function defined in a given search space. In PSO, individuals called *"particles"* and a set of these particles called "*swarm"*. In fact a particle or a position in the solution space is a candidate solution for the problem to be

optimized. Particles cooperatively moved around in the search space to find the best position according to their velocities. The movement of the particle *i* is guided by own best position found by particle *i* and the best known position by the entire swarm so far. At each iteration, the position of each particle in the search space is computed using the following equations [23, 24]:

$$v_i^{t+1} = \alpha v_i^t + c_1.\omega_1.(pbest_i^t - x_i^t) + c_2.\omega_2.(gbest_i^t - x_i^t) \quad (1)$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \quad (2)$$

where in Eq. (1), x_i^t is the position of particle *i* at time *t* , v_i^t is the velocity of particle *i* at time *t*, α is an inertia weight scaling the previous time step velocity, cognitive parameter (c_1) and social parameter (c_2) are constant acceleration coefficients, ω_1 and ω_2 are random factors in the [0, 1] interval, $pbest_i^t$ is the best position found by particle *i* at time *t* and $gbest_i^t$ is the best known position found by the entire swarm so far. The pseudo code of the PSO is shown in Fig. 1.

```
Initialize a population.
(a population of particles with random velocities and positions)
while (stop conditions are not satisfied)
   {
     for each particle i= 1…Population Size
      {
        update the position of particle "i" according to equation (1).
        update the velocity of particle "i" according to equation (2).
        evaluate the fitness value for particle "i";
        if necessary, update pbest_i for particle "i", also update gbest_i .
      }
   }
```

Fig 1. The canonical version of PSO algorithm.

C. Graph Coloring Problem

The graph coloring problem (GCP) [25] is an optimization problem. Coloring a graph consists of finding an optimal and valid coloring for that graph such that the colors assigned to adjacent vertices to be different. Graph coloring problem is a practical method of representing many real world problems including time scheduling, frequency assignment, register allocation, bandwidth allocation, and circuit board testing [26]. In general, there exist two important challenges in the graph coloring problem. First, in coloring process the vertices of graph to be colored correctly, i.e., all the vertices of graph must be colored and different

colors are assigned to adjacent vertices. Second the total number of colors used in the coloring process to be minimized. This is most often implemented by using a conflict minimization algorithm. In the implementations of this paper, both of these goals are considered.

3. The Proposed Method

In the following sections, we describe the details of our proposed method, starting by an overview on the proposed system followed by discussion on the process of creating flow graph, and details pertaining to conflict detection and resolution process.

A. System Overview

Fig. 2 shows an abstract view of the proposed conflict detection and resolution process explained in the following sections. Also some of the main properties of the proposed method are indicated in Table 1, according to features described in the comparison framework presented in [6, 21]. For simplicity, in the implementation of proposed model we assumed some constraints. Firstly, each flight path controller agent (FPCA agent) is responsible only for aircraft that there exist in its vision domain, and can interact with them. Second the conflicts are resolved as locally (in each FPCA's vision area) and we not considered the interaction between different existing FPCA agents.

The process begins with existing agents (in our implementations, FPCA agents) in the system continuously monitoring the traffic situation in their areas (vision domains) and use nominal state propagation method to check for imminent conflicts. In this step, agents create a (directed) flow graph of the airspace situation. This graph also is called conflict graph. Each vertex of this graph indicates an aircraft in the airspace. Initially there is not any edge in this graph and it only contains vertices.

In conflict prediction process, (FPCA) agents look 2 minutes into the future. In this step, if the agent determines that a conflict is going to occur, the agent creates an edge or edges between corresponding vertices of aircraft that are involved in the conflict. In the flow graph, each edge between two vertices indicates an impending conflict between two corresponding aircraft in the future prediction time. Also FPCA agent can assign a weight to edges according to distance of each aircraft agent to an impending conflict point. But we not considered this feature in the implementation of this paper.

Afterwards, the created directed flow graph was converted to an undirected flow graph by using a simple method. In the next stage, each (FPCA) agent initiates a negotiation session by sending a message to the aircraft agents that there exists in its vision domain. This message contains some guideline instructions and all necessary information about traffic situation and predicted

conflicts in the future time. When each aircraft agent arrives in a FPCA vision area, receives a message from that FPCA agent. Each aircraft agent must send back all requested information to that FPCA. This information include: aircraft's position, current velocity, direction, origin, destination and etc.

After creating the undirected flow graph by using collected information, the FPCA agent uses PSO algorithm to color this graph. In fact, in the coloring process the agent uses a prescribed method to generate a number of possible alternative routes for aircraft which there exist in the congestion area, in order to prevent of conflicts. In coloring process, each color for coloring the flow graph is equivalent to an alternative safe route, and for each aircraft the algorithm can assign five alternative routes (four prescribed deviations are right, left, up, and down of main route and fifth route is the main course or no deviation) provided that these routes have not conflict with other aircraft' allocated routes [27]. In each agent update cycle the alternative routes for each aircraft can be shown as a binary routes array similar to: [r1, r2, r3, r4, r5]. Each element in this array indicates an alternative route and has two possible values: 0 or 1. If an alternative route previously not occupied by another aircraft then this element is equal to 1, otherwise it is equal to 0. Initially this array for each aircraft is equal to [1, 1, 1, 1, 1]. Obviously, each FPCA agent for *n* aircraft (that there exist in its vision domain) holds a matrix of alternative routes as below.

$$\text{Alternative Route Matrix} = \begin{matrix} \text{aircraft}_1 \\ \text{aircraft}_2 \\ . \\ . \\ \text{aircraft}_n \end{matrix} \begin{bmatrix} r_{1,1} & . & . & . & r_{1,5} \\ . & . & & & . \\ . & & . & & . \\ . & & & . & . \\ r_{n,1} & . & . & . & r_{n,5} \end{bmatrix}_{n\times 5} \tag{3}$$

In our implementation we numbered these alternative routes in order as follows.

$$\text{Alternative Routes} = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \end{bmatrix}_{n\times 5} \tag{}$$

Each aircraft agent only can occupy one of these (alternative) routes provided that has not conflict with other aircraft's selected routes in current and future predicted times. The FPCA agent in an informational message sends this information (route matrices and all information about all adjacent aircraft in its vision domain) to each aircraft agent in its vision area. Then, each aircraft agent selects its options (alternative routes) and sends back the message to the FPCA agent. The FPCA agent by considering some constraints runs PSO algorithm. The PSO technique uses a fitness function to evaluate these routes and then determines

a cost value for each route. The algorithm tries to select the routes that impose less cost on the system. After finding a valid and optimal solution, the FPCA agent sends the new flight plan to all involved aircraft. After receiving acceptance messages from all involved aircraft, this plan can be executed. In this paper, in the PSO algorithm a particle is a vector of natural numbers of size k, where k is the number of vertices in the flow graph. A particle can be shown as Fig. 3. Each element in this particle indicates an (alternative) route. Also, Fig. 4 shows a simple conflict resolution process for three involved aircraft.

B. Creating the Flow Graph

Each vertex of the flow graph indicates an aircraft in the airspace (i.e., all necessary information about an aircraft such as position, speed, direction, flight level, origin, destination and etc) and each edge between any two vertices (aircraft) indicates a conflict that is going to occur in future prediction time. In each agent update cycle (each 5 seconds) this flow graph is updated and the structure of this graph may be changed in every update cycle.

C. Conflict Detection

In our proposed model, we considered a protected zone for each aircraft. The protected zone is a cylinder of height $2V$ and radius H around each aircraft, where V and H respectively indicate the least safe vertical and horizontal separation distance between aircraft. For instance, in this paper H is considered 1.5 nautical miles and V is considered 1000 feet [28]. In this section, it is assumed that a conflict between two aircraft will occur when the protected zone of the two aircraft overlap. Let $P_a(x, y, z)$ be the position of aircraft A and $P_b(x, y, z)$ be the position of aircraft B. These two aircraft are in loss of separation if the following predicate holds.

$$Loss\_Separation(P_a, P_b, H, V) = \sqrt{(P_a(x,y) - P_b(x,y))^2} < H \text{ and } \left|P_a(z) - P_b(z)\right| < V \quad (5)$$

In this paper, for predicting conflicts we used a simple conflict prediction tool called *nominal state propagation method* [6]. We used this method to determine the future location of aircraft using current situation of traffic. In each agent update cycle we predict next position of aircraft at 2 minutes into the future. Let T_o and T_i to be the aircraft A and B predicted routes, respectively. A conflict between two aircraft occurs when there exist a point $q_{(x,y,z)} \in [0, T]$ (T determines the size of anticipation window – for instance in this paper is set to 5 next minutes) on the predicted trajectories such that $T_o(q_{(x,y,z)})$ and $T_i(q_{(x,y,z)})$ are in loss of separation. In other words a conflict occurs if the following predicate holds.

$$Conflict(T_a, T_b, H, V, 0, T) = \exists\, q \in [0,T] : Loss\_Seperation(P_a(q), P_b(q), H, V) \quad (6)$$

Table 1

The main characteristics of the proposed conflict detection and resolution system

Feature	Value	Feature	Value
Agent selection	Flight path controller aircraft	Managing of Multiple conflicts	Global; due to using flow graph
Agent actions	Mapping the traffic situation to a flow graph, detecting conflicts, solving conflicts and some improvement Guidelines	maneuvers	Horizontal Maneuvers (lateral maneuver) Vertical Maneuvers (change flight level)
Conflict resolution	using constraint satisfaction problem concept optimization by using PSO algorithm	Interaction between agents	Message passing
Conflict detection	Violation from minimum horizontal or vertical distance (overlap in protected zone)	Plan Dimensions	Horizontal Vertical
Type of Multi-agent system	Combined (agents helps ground controllers)	strategy	Cooperation
Programming language	JAVA	Tools	JADE

Step1: Set the initial parameters including minimum horizontal (H) and vertical (V) separation distance, the maximum iterative count of PSO algorithm, the population size (P_{size}), c_1 , c_2 , v and ω .

Step 2: Monitor traffic

Step 3: Predict next states of the traffic situation using nominal state propagation method

Step 4: Map the traffic situation to an undirected flow graph

Step 5: if (conflict detected)

Step 5.1: (PSO method)

Initialize a population of candidate solutions of size P_{size} (i.e. generate alternative trajectories and calculate the cost function).

Step 5.2: for iterative counts execute PSO algorithm.

Step 6:

if (conflict resolved - flow graph colored**?**)

Send new (conflict free) flight plan to aircraft

else

Select fallback plan or contact ground controllers for more guidelines

Fig 2. Overview of the conflict detection and resolution process.

n_1	n_2	n_3	…	n_N

Fig 3. The representation of a particle in the PSO algorithm.

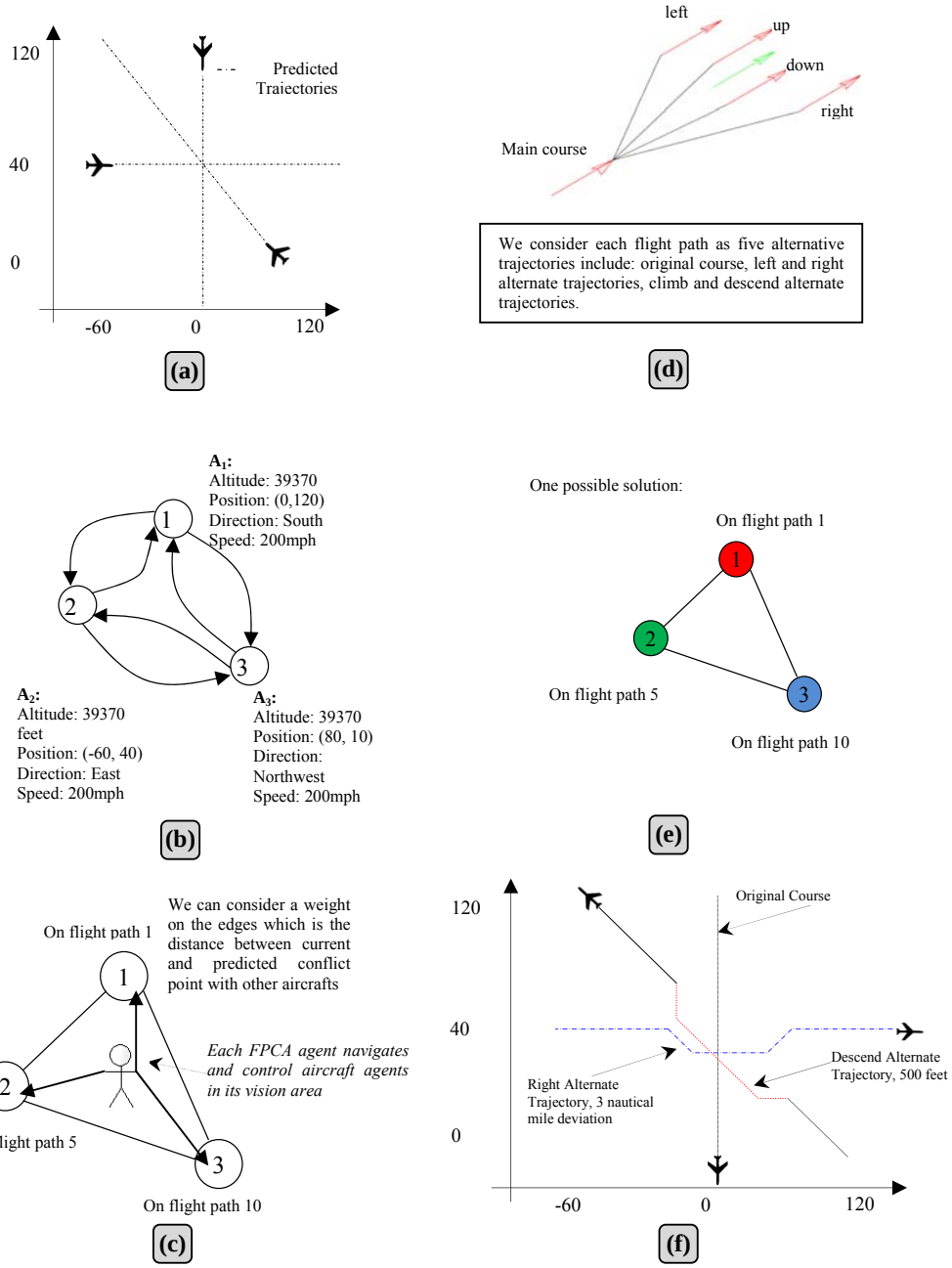


Fig 4. An example, the conflict resolution process for three involved aircraft; ***a***: An example scenario (initial traffic condition), ***b***: the airspace mapped into a directed graph, ***c***: the directed graph converted into an undirected graph and some necessary information are stored, ***d***: alternative trajectories is shown, ***e***: a valid, optimal and possible solution (the flow graph is colored), ***f***: the airspace condition after conflicts resolution process.

D. Conflict Resolution Process

We converted the conflict resolution process to the graph coloring problem concept. After creating the flow graph of agents, PSO technique is used to color that flow graph. The FPCA agent tries to conduct each aircraft in a safe flight path. When the FPCA agent detects that a conflict is going to occur between aircraft, it uses a predefined process to generate alternative conflict free routes (i.e. new flight plan). The five prescribed alternative routes are four deviations which include *right*, *left*, *up*, and *down* of the main route, and *no deviation* or *main route*. In other words this resolution process allows to aircraft for climbs, turns, and no deviation. Each of these alternative routes can be considered as a color in coloring the flow graph.

The conflict resolution (coloring the flow graph) process consists of assigning different free conflict alternative routes (colors) to aircraft (vertices) in the airspace (corresponding flow graph). The PSO algorithm initially creates a population of possible and valid candidate solution; then PSO uses a fitness function to evaluate the goodness of each individual. Here we used *"delay time"* metric as the objective function. How much the delay time of an individual is lower its fitness value is higher and vice versa, if delay time of an individual is higher, then its fitness value is lower. At the end, among the valid candidate solutions the FPCA agent selects the best solution (i.e. the solution with the least delay time). Then, the FPCA agent sends the new free conflict flight plan to the aircraft exist in congestion area.

4. Experiments and Results

The proposed method described earlier has been implemented using the JAVA programming language and JADE tool [29]. The Java Agent DEvelopment framework or shortly JADE, involves a runtime environment to host and run agents, a graphical user interface to monitor the activity of executing agents, and a library including Java classes utilized to create agents. In other words, JADE is a software infrastructure that permits a simple implementation of agents. All the simulations took place on a Pentium (R), CPU 3.00 GHz and 512 MB RAM computer. The parameters of PSO algorithm used by flight path controller agent are as follows. The number of generations ($T_{\max}$) and the initial population size (N_{pop}) are set to be 200 and 50 respectively. The lower bound ($v_{\min}$) and upper bound ($v_{\max}$) of velocity are set to 0.05 and -0.05 respectively. Also both c_1 and c_2 parameters are set to 1.50. The inertia weight factor (ω) is set as a time variant linear function decreasing with the increase of number of iterations where in each iteration *i*, the value of ω is given by:

$$\omega = 0.4 + 0.9 \times (\text{number of generation - i}) / (\text{number of generation - 1})$$
$$\text{where } \omega_{min} = 0.4 \quad \text{and} \quad \omega_{max} = 0.9 \tag{7}$$

Also to discover the potential of the proposed algorithm, we used five well-known benchmark scenarios. A description of these scenarios is given in the following.

A. Conflict Scenarios

The proposed conflict detection and resolution system was evaluated using five well-known scenarios. These scenarios are similar to the test cases used in other studies [18, 30]. These scenarios include traffic patterns with both random flight and fixed geometries traffic patterns. In all scenarios, aircraft travel at speed of range 300-500 mph. Also the standard values of safety horizontal and vertical distance between aircraft are set to be 5nmi and 1000 feet. At each 5 seconds time step, each aircraft receives all necessary information about other aircraft within its vision domain of size 50nmi and choose its route based on the received information. Then each aircraft updates its flight plan and distributes the new modified information to all aircraft in its vision domain. These scenarios are as follows. *N* is the number of aircraft, *P* is the number of initial impending conflicts in the scenario.

1) Randomly Generated Scenarios

Each scenario of this type consists of several aircraft whose initial speeds, altitudes and headings would cause a conflict. In these scenarios, each aircraft travelling at randomly determined speeds ranging from 300–500 mph in a randomly allocated direction for a predetermined period of time. Aircraft are flying at a straight line at a constant flight level (altitude). The origin and destination of each aircraft is determined randomly in the beginning of simulation. Test cases 1–3 are based on this type of scenario.

Test case 1 ($N = 4,\ P = 2$*);* as shown in Fig. 5-a, this scenario consist of 4 aircraft in a shared airspace that involve in two conflicts. Each aircraft has generic initial and final configuration.

Test case 2 ($N = 6,\ P = 4$*);* in this scenario two pairs of aircraft are involved in four conflicts. Similar to test case 1, each aircraft travelling at randomly determined speeds ranging from 300 to 500 mph in a randomly assigned direction. Fig. 5-b illustrates this scenario.

Test case 3 ($N = 8,\ P = 4$*);* this scenario is another example of randomly generated patterns that we tackle. As shown in Fig. 5-c, four pairs of aircraft are involved in three conflicts. This scenario is more difficult than test case 1 and 2.

2) Circular scenario

Test case 4 ($N = 5,\ P = 10$); this scenario is relatively similar to Choke Point model used in [30]. In this scenario, aircraft arranged on a circle with radius 50nmi, and their destination points are located on the opposite direction of their origin. Therefore, aircraft ideal paths coincide at the center of the circle. Fig. 5-d shows an example of circular type with five involved aircraft.

3) Perpendicular Scenario

Test case 5 ($N = 7,\ P = 3$); this scenario is similar to the perpendicular flows model used in [30]. This scenario is called "crossing the street" in which two linear traffic flows intersect at left angles, one moving from right to left and the another flow moving from bottom to top. An example of this scenario is illustrated in Fig. 5-f. All aircraft fly on a same altitude with speed of 500 mph, and all aircraft are aware of all other aircraft within 100 nautical miles. At the beginning of simulation, in each traffic flow, each aircraft has an initial separation just over 5nmi from the proceeding aircraft.

B. Performance Metrics

The main goals of any conflict detection and resolution system are the maintenance of safe separation between aircraft, and increased efficiency [30, 31]. There exist various metrics to evaluate conflict detection and resolution systems. In this paper, to discover the potential of the proposed method we defined performance metric as below.

δ_i: Ideal flight time from origin to destination for *i*–th aircraft, which determines when the aircraft first arrives in the simulation

γ_i: Factual flight time for *i*–th aircraft

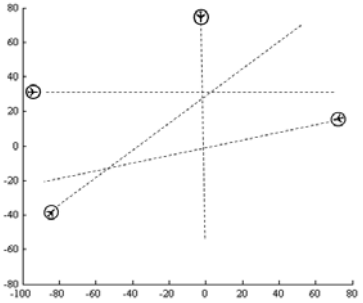
λ_i: Accumulated delay time for *i*–th aircraft which is defined as: $\lambda_i = \gamma_i - \delta_i$

In general, to avoid conflicts the system uses some conflict resolution maneuvers. These maneuvers cause each aircraft to deviate from its ideal flight path and experience somewhat delay time [30, 31]. The four prescribed deviations are left, right, up and down of main route. Since in our simulations all aircraft are similar and have the same characteristics, the performance of the system can be calculated tracking the time it takes aircraft to get from origin point to their destination point. So, the performance of the system (*P*) can be defined as below:

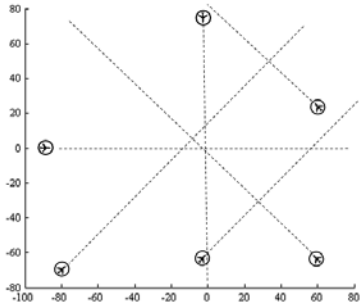
$$P = 100 \times \left(\frac{1}{N_a} \sum_{a=1}^{N_a} \frac{\delta_i}{\delta_i + \lambda_i} \right) \tag{8}$$

Where N_a is the total number of aircraft in the system. How much the value of this criterion is closer to *"1"* indicates good performance of the system and how much this value is closer to *"0"* indicates poor performance of the system. According to Eq. (12), the value of performance is directly related to the sum of delays

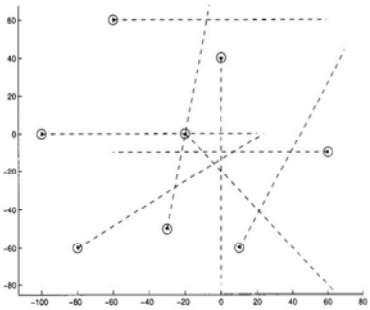
experienced by all aircraft. How much the total delay time experienced by aircraft in the system is lower then the system performance is higher. High performance is equal to the high accuracy in conflict resolution process, the increase in passengers' comfort and the decrease in flight delays.



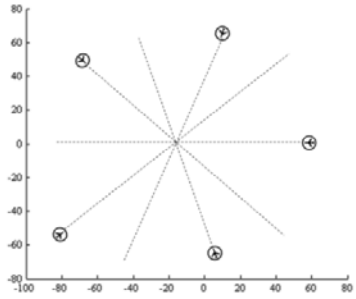
a. View of test case 1; two pairs of aircraft are involved in two conflicts.



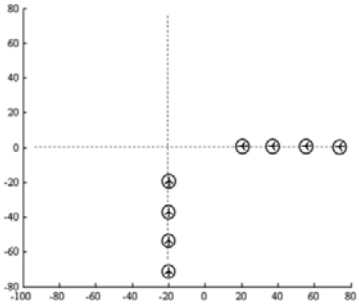
b. View of test case 2; two pairs of aircraft are involved in four conflicts.



c. View of eight aircraft in a shared airspace; three pairs of aircraft are involved in four conflicts [18].



d. An example case of circular airspace scenario with 5 aircraft; the aircraft distributed on a circle centered in the origin and of radius 50 nmi. All the aircraft will cross the origin at the same time.



f. Perpendicular flow with 8 aircraft.

Fig 5. Aircraft conflict scenarios

C. Results and Discussion

Table 2 shows simulation results for a range of problem sizes. The experimental results are averages of 10 runs of simulation. In the table, *n* indicates the number of aircraft considered in the simulations, *type* indicates the type of scenario, *p* is the efficiency value averaged, *sigma* is the standard deviation, and *m* indicates the average number of deviations during conflict resolution process over 10 runs.

From simulation results presented in Table 2, it can be seen that the proposed method can completely resolve conflicts while it is able to reach high efficiency. Also the processing times required to solve conflicts are quite low and this is due to the high potential of the proposed method. In other words, the proposed method in conflict resolution process uses the least number of deviation maneuvers for each aircraft and this is an ideal improvement.

Table 2

The results of applying the proposed method on different conflict scenarios

No.	***type***	***n***	***m***	***p***	***sigma***
1	Random flights	4	1.3	98.05	0.0063
2	Random flights	6	3.9	97.50	0.0105
3	Random flights	8	5.9	94.50	0.0077
4	Circular pattern	5	4.3	95.20	0.02
5	Perpendicular flow	8	5.4	95.70	0.0005

2. Conclusions

This paper investigates a simple, conceptual and abstract conflict detection and resolution model based on the multi-agent system and PSO algorithm. This system uses two types of agents namely flight path controller agent (FPCA) and aircraft agent (AA), providing a proper balance between distributed and centralized authority in order to resolve aircraft conflicts. The proposed system can be used beside current air traffic management systems and especially in free flight method. Since in the conflict resolution process, the PSO algorithm considers some optimization aspects such as having least total delay time and least total distance traveled, the selected valid solutions have minimum cost as an improvement. The proposed model is implemented and tested on five well–known air traffic conflict patterns to assess its efficiency. The reported results from simulation runs indicate the system has the capability of detecting and avoiding conflicts. This means our proposed model is a systematic and reliable method and safety options can be met by this system.

Our feature work will focus on the cooperation of different agents (especially the cooperation of the flight path controller agents) and implementing this model in the real world.

REFERENCES

[1]. Federal Aviation Administration, "Advancing Free Flight through human factors", www.hf.faa.gov/docs/508/docs/freeflt.pdf, Last access August 1th, 2011.

[2]. *J. van Gent, R. Ruigrok*, "Conceptual design of free flight with airborne separation assurance", In AIAA guidance, navigation, and control conference, Boston, MA, 1998, pp. 807–817.

[3]. *S. Wollkind, J. Valasek, and T. Ioerger*, "Conflict resolution for air traffic management using cooperative multiagent negotiation", In AIAA guidance, navigation, and control conference. Providence, RI, 2004.

[4]. *G. Valkanas, P. Natsiavas, and N. Bassiliades*, "A collision detection and resolution multi-agent approach using utility functions", Fourth Balkan Conference in Informatics, 2009.

[5]. *Agogino, and K. Tumer*, "Improving air traffic management with a learning multi-agent system", IEEE Intelligent. Systems., vol. 24, no. 1, 2009, pp. 18–21.

[6]. *J. K. Kuchar, L. C. Yang*, "A Review of Conflict Detection and Resolution Modeling Methods", IEEE Transactions on Intelligent Transportation Systems, Vol. 1, No. 4, December, 2000, pp. 179-189.

[7]. *M. Bayen, P. Grieder, G. Meyer, and C. J. Tomlin*, "Lagrangian delay predictive model for sector-based air traffic flow", AIAA Journal of Guidance, Control, and Dynamics, 28, 1015–1026, 2005.

[8]. *McNally, and C. Gong*, "Concept and laboratory analysis of trajectory-based automation for separation assurance", In AIAA guidance, avigation and control conference and exhibit. Keystone, CO, 2006.

[9]. *Bertsimas, and S. Stock-Patterson*, "The air traffic management problem with enroute capacities", Operations Research, 46(3), 406–422, 1998.

[10]. M. Christodoulou, and C. Costoulakis, "Nonlinear mixed integer programming for aircraft collision avoidance in free flight", In IEEE MELECON, Vol. 1. Dubrovnik, Croatia, 2004, pp. 327–330.

[11]. *K. Bilimoria, K. Sheth, H. Lee, and S. Grabbe*, "Performance evaluation of airborne separation assurance for free flight", In AIAA guidance, navigation, and control conference. Denver, CO, 2000.

[12]. *P.K. Menon, G.D. Sweriduk and B. Sridhar*, "Optimal strategies for free flight air traffic conflict resolution", Journal of Guidance, Control, and Dynamics, 1999, 22(2), 202–211.

[13]. *N. Durand, J-M. Alliot*, "Ant Colony Optimization for Air Traffic Conflict Resolution", 8th USA/Europe Air Traffic Management Research and Development Seminar, ATM 2009.

[14]. *J.C. Hill, F.R. Johnson, J.K. Archibald, R.L. Frost, and W.C. Stirling*, "A cooperative multi-agent approach to free flight", In AAMAS '05 Proceedings of the fourth international joint conference on autonomous agents and multiagent systems. New York, NY, USA, 2005, pp. 1083–1090.

[15]. *M. Pechoucek, and D. Sislak*,"Agent-based approach to free-flight planning, control, and simulation", IEEE Intelligent Systems, 2009, 24(1), 14–17.

[16]. *K. Tumer, and A. Agogino*, "Adaptive Management of Air Traffic Flow: A Multiagent Coordination Approach", Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence (2008), pp. 1581-1584.

[17]. *S. Cafieri, P. Brisset, and N. Durand*, "A mixed-integer optimization model for air traffic deconfliction", Proceedings of the Toulouse Global Optimization Workshop (TOGO 2010), Toulouse, September 2010, pp. 27-30.

[18]. *L. Pallottino, E. Feron, and A. Bicchi*, "Collision Resolution Problems for Air Traffic Management Systems Solved With Mixed Integer Programming", IEEE Transactions on Intelligent Transportation Systems, vol. 3, no. 1, March 2002, pp. 3-11.
[19]. *M. Wooldridge*, "An Introduction to MultiAgent Systems", Wiley & Sons, 2002.
[20]. The Multi-Agent Systems Lab. Last access April 5th, 2012. [Online]: http://mas.cs.umass.edu.
[21]. *H. Emami, F. Derakhshan*, "An Overview on Conflict Detection and Resolution Methods in Air Traffic Management using Multi-agent Systems", 16th Artificial Intelligence and Signal Processing (AISP) Conference, Shiraz, Iran, 2012, pp. 293-298.
[22]. *J. Kennedy, R. Eberhart*, "Particle Swarm Optimization", Proceedings of IEEE International Conference on Neural Networks. 1995, IV. pp. 1942–1948.
[23]. *F. Yang, T. Sun, C. Zhang*, "An efficient hybrid data clustering method based on K-harmonic means and Particle Swarm Optimization", Expert Systems with Applications, 2009, 36, 9847–9852.
[24]. *Y. Shi, R.C. Eberhart*, "A modified particle swarm optimizer", Proceedings of IEEE International Conference on Evolutionary Computation. 1998, pp. 69–73.
[25]. *T.R. Jensen, and B. Toft*, "Graph Coloring Problems", Wiley interscience Series in Discrete Mathematics and Optimization, 1995.
[26]. *H. Emami, Sh. Lotfi*, "Graph Coloring Problem based on Discrete Imperialist Competitive Algorithm", International Journal in Foundations of Computer Science & Technology (IJFCST), Vol. 3, No.4, July 2013, pp. 1-12.
[27]. *H. Emami and F. Derakhshan*. "Conflict Detection and Resolution in Air Traffic Management Based on Graph Coloring Problem Using Imperialist Competitive Algorithm" International Journal of Information Technology, Control and Automation (IJITCA), 2012, Vol.2, No.2, April 2012, pp. 27–42.
[28]. *M. S. Nolan,* "Fundamentals of Air Traffic Control". Brooks Cole, 4 Edition, July 2003.
[29]. *Bellifemine, A. Poggi, and G. Rimassa*, JADE – A FIPAcompliant agent framework, Telecom Italia internal technical report. http://jade.tilab. com/papers/PAAM.pdf, Last access August 10th 2011.
[30]. *J.K. Archibald, J.C. Hill, N.A. Jepsen, W.C. Stirling, and R.L. Frost*, "A Satisficing Approach to Aircraft Conflict Resolution", IEEE Transactions On Systems, Man, And Cybernetics–Part C: Applications And Reviews, Vol. 38, No. 4, July 2008, pp. 510-521.
[31]. *H. Emami, F. Derakhshan, and S. Pashazadeh*, "A New Prioritization Method for Conflict Detection and Resolution in Air Traffic Management", Journal of Emerging Trends in Computing and Information Sciences, Vol. 3, No. 7, 2012, pp. 1042–1049.