Using Ant Algorithm to Arrange Taxiway Sequencing in Airport

Kamila B. Nogueira, Paulo H. C. Aguiar, and Li Weigang

***Abstract*—Over the last decade, air traffic flow at airports experienced an increase never seen before. Whereas several improvements have been achieved in enlarging the en-route traffic capacity, little has been done in order to decrease congestion on the airport surface. Aircraft taxiing is an expensive process, responsible for causing delays for both passengers and airlines and pollution problems. This paper presents an optimization solution for aircraft taxi-scheduling problem using Ant Colony Optimization, a method for search and optimization inspired by the behavior of real ants in nature. The simulations presented were based on real data from Brasilia International Airport. However, the developed model can be used in any airport for optimal taxiing routes searching. The paper has shown the optimization of taxiing to be efficient in achieving minimization of aircraft taxi time and complies with security constraints at airports.**

***Index Terms*—Ant algorithm, air traffic flow management, optimization, simulation, taxiway sequencing.**

I. Introduction

The increased demand at airports in the past decade has caused many problems both for passengers and airlines. The Air Traffic Flow Management (ATFM) is a problem that have been studied by scientists since the 1970's, who try to create more efficient air traffic control systems [1]. The air traffic bottleneck is located in airports, and it is important that their operators put effort into optimizing the use of their resources (consisting of runways, terminal gates, hangars).

Taxiing is defined as the movement of an aircraft on the ground, under its own power, excluding takeoff and landing [2]. This process is responsible for most of delays at airports. In Europe, for example, it is estimated that aircraft spend 10-30% of their flight time taxiing, and that a medium range A320 expends as much as 5-10% of its fuel on the ground [3].

In Brazil there isn't a system that automates this process, which is performed by air traffic controllers, professionals constantly suffering great pressure because of the responsibility they carry. Service quality at airports could not meet the increasing demand and has made it rather difficult for controllers to manage taxiing, a task that is performed mainly visually.

This paper presents a system to solve the problem of taxiing using the Ant Colony Optimization, an efficient search method inspired by the behavior of real ants in nature.

When ants leave the nest searching for food, they lay down on the ground a substance called pheromone. Other ants perceive the presence of pheromone through smell and tend to follow the path where its concentration is higher. The ants that chose the shortest route will return to the nest faster, causing this path to get larger amount of pheromone, while on the paths traveled down less frequently, the substance will evaporate over time [4].

Through this mechanism, ants are able to carry food to the nest with an incredible efficiency, making this an astonishing ability to find what computer scientists call a shortest path. This method is used for computational problems involving graph traversal and, in the case of taxiing problem, will apply to traversing the airport.

Manuscript received November 10, 2013; revised January 17, 2014.

The authors are with the TransLab, University of Brasilia, UnB, Brasilia, DF, Brazil (e-mail: weigang@unb.br).

II. Related Works

Aircraft taxiing management is a subject that has been studied by researchers all over the world, due to the impact that correct management has on avoiding delays, not only for the comfort of millions of passengers that every year use this means of transportation, but also to mitigate operating costs and reduce environmental impact from airports. In this section, some researches in this field are presented.

A. Optimization of the Taxi with the Method of Ant Colony

Ground operations optimization at an airport has NP-hard complexity [5] due to the challenge of efficiently manage traffic, reducing delays and avoiding conflicts in these operations. The ground operations are generally classified as management takeoff and landing and taxi planning and scheduling.

In 2011, Changyou Liu and Wang Yongcheng [5], from the University of China, proposed a pioneer model that applies the ant colony for the taxiing optimization problem. In this model, it is applied the first method proposed with ant colony, Ant System.

According to flight schedules, it is necessary specify routes to all landings and takeoffs in order to ensure that the taxiing will be calculated using the shortest time possible or shortest distance traveled, under certain security conditions. The taxi routes are represented by a graph $G = (V, E)$, where V is the set of vertices (including garages) and E is a set of edges that connect two nodes. The distance between a node i and a node j is represented as S_{ijk}, where k represents the aircraft. In this model, each artificial ant has a constant amount of pheromone.

When the ant goes from the starting point to its destination, i.e., follows a path between vertices S_1 and S_2, pheromone will be deposited on the route. If this route is a good choice, the distribution of pheromone will be higher. If, on the other hand, the route is bad, distribution of the pheromone element will be lower. In this case, the tracks on

DOI: 10.7763/IJCTE.2014.V6.889

the route will reduce until no ants choose this path.

B. Optimization of the Taxi Using Genetic Algorithms

Genetic algorithms are global optimization heuristic techniques inspired by the biological process of natural evolution and genetics. In this algorithm, first of all, it is created a population consisting of a random set of individuals that might be seen as possible solutions for the problem. During the process of evolution, is population is evaluated: for each individual is assigned a score, or index, reflecting its ability to adapt to a particular environment (fitness). A percentage among those better adapted is kept, while the others are rejected, a characteristic that goes back to the process of natural selection [6].

Individuals chosen by selection may change in their fundamental features through mutation and crossover or genetic recombination generating offspring for the next generation. This process, called reproduction, is repeated until a satisfactory solution is found. Using this metaphor for the taxiing problem, genetic recombination represents attempts to determine more optimal paths in the airport [7].

Using the schedule and points of departure and destination of planes, L feasible routes are generated along the set of vertices of the mesh of possibilities. To ensure the implementation of this algorithm, the gene of each chromosome is one of the series of numbers selected randomly from L taxiing routes for aircraft k. The selection is made for each gene of the chromosome and the length of the chromosome is the total number of flights. For example, the string of chromosome 2 5 2 4 7 5 6 5 8 presents the number of paths for 9 aircrafts. It is important to note that L is the total number of feasible taxiway routes, and the total number of flights N_f is the length of the chromosome, while a possible trajectory for the aircraft k is a position of a gene in this algorithm.

The fitness function measures the quality of each generation from the presented population. It is used to assign reproductive features to individuals in population and then works as a quality measure that must be increased, i.e., individuals with a higher fitness value have a higher probability of being selected as candidates for further examination. The chromosomes are crossed with a probability P_c and mutation occurs with a probability P_m. Therefore, the generations keep evolving so that the best solutions are chosen.

As it is difficult to determine the scope of the optimal solution, the maximum number of generations is adopted to terminate the algorithm. Finally, the best chromosome is found, i.e., the one that presents the best solution among all the non-deterministic solutions found.

In the Translab - Laboratory of Computational Model for Air Transportation – at the University of Brasilia, a model using genetic algorithm was also developed [6] for taxiway sequencing, and now the ant colony approach comes as another alternative for the taxiway arrangement. This research proposes an alternative method for this problem.

III. Ant Colony Optimization

The ACO (Ant Colony Optimization), formulated in the 1990s by Marco Dorigo in his PhD thesis, is a heuristic inspired by the behavior of real ants [8], regarding their skills in finding the shortest path between nest and food.

The interactions between individuals (ants) are based on simple behavioral rules that exploit only local information that individuals exchange with each other or the environment. The overall behavior of the system is the result of interactions between individuals and those with the environment, i.e., the group self-organizes.

In ACO, a set of software agents called artificial ants search good solutions to a given optimization problem. To apply the ACO, the optimization problem is transformed into the problem of finding the best path in a weighted graph. Incrementally, ants build solutions by moving in the graph. The construction process of solution is random and is based on a model of pheromones, i.e., a set of parameters associated with graph components (nodes or points), whose values are modified at runtime by the ants.

The moment an ant finds a node i, and having constructed a partial solution s_p, the probability to go to node j is presented as follows [9]:

$$P_{ij} = \frac{\left[T_{ij}\right]^{\alpha}\left[\eta_{ij}\right]^{\beta}}{\Sigma_{C_{il}\in N(s_p)}[T_{il}]^{\alpha}[\eta_{il}]^{\beta}}$$

where $N(s_p)$ is a set of possible solutions, that is, vertices (i, l), where l is a node not yet visited by ant k. α and β are constants and control the relative importance of the pheromone and the length of the route, respectively. The heuristic information η_{ij} is given by:

$$\eta_{ij} = \frac{1}{d_{ij}}$$

where d_{ij} is the distance between i and j, which shows that the shorter the path, the greater is its pheromone concentration.

Two important features in this algorithm are the *local pheromone update* and *offline pheromone update*.

The local pheromone update is a pheromone decay performed by all ants at each step of the iteration. Each ant applies it only to the last edge traversed:

$$T_{ij} = (1-\phi)\times T_{ij} + \phi \times T_0$$

where $\varphi \varepsilon (0,1]$ is the pheromone decay coefficient, and τ_0 is the initial value of pheromone.

The main goal of the local update is to diversify the search performed by subsequent ants during an iteration: by decreasing the concentration of pheromone on the traversed edges, ants encourage the following to choose other routes and thus to produce different solutions.

The offline pheromone update is applied at the end of each iteration by only one ant, which is the best-so-far. The update formula is as follows:

$$\Delta T_{ij}^{k} = \begin{cases} \dfrac{Q}{L_k} & \text{se a formiga } k \text{ utilizou a rota}(i,j); \\ 0 & \text{caso contrário} \end{cases}$$

and the amount of pheromone deposited at the edge (i, j) is given by:

$$\Delta T_{ij} = \frac{Q}{L_b}$$

where Q is a constant and L_b is the length of the best route found.

IV. Proposed Solution for Taxi Aircraft Sequencing

This section presents the solution to aircraft taxiing using ant colony optimization.

First, the operator must submit to the system a text file containing information regarding the structure of the airport, identification of each node and the distance between adjacent nodes. The map of the airport used is shown in Fig. 1.

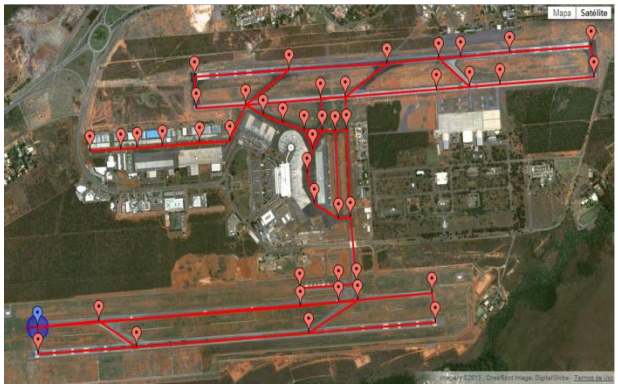

Fig. 1. Points from the airport represented in the graph.

In addition, the controller must submit the schedule of flights, informing the starting position of each aircraft and schedules for takeoff or landing, information on which the optimal routes for taxiing will be calculated.

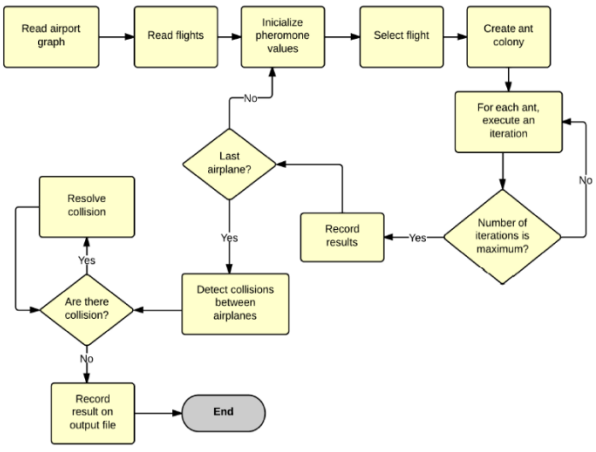

Fig. 2. Operation of the algorithm.

A. Modeling and Architecture

The modeling of the problem was made so that the airplane trajectory from the gate to the runway (or the opposite way, to the gate) is represented as a weighted graph, where the weight is calculated in time.

The system's basic operation is represented by the diagram in Fig. 2. This flowchart can be described as follows:

1) First, the input file that represents the airport, including the name of each node and the distance between adjacent nodes is read and then the graph is created. The file containing the scheduled flights is read, with time and nodes of origin and destination of each aircraft.
2) The pheromone values are initialized for each route according to the value of $\tau_{inicial}$. The first scheduled flight is selected and the ant colony is established, in this case featuring a number of 60 ants.

The agents start searching through multiple iterations, defined as each move by all the 60 ants on the graph. When the maximum number of iterations is achieved, the best solution is recorded. This process is repeated for each aircraft.

3) With the best routes already calculated for all aircrafts, conflicts are handled. If two aircrafts are less than 150 meters away from each other, the second one to arrive waits until it is safe to proceed.
4) After all conflicts are resolved, the taxiing route of each aircraft is written to an output file.

B. Implementation

This section presents the procedures of data optimization, ant algorithm execution and conflict detection. The main program uses several auxiliary programs for the process of assigning taxiing routes for a group of aircrafts at an airport. The solution was developed using the C language.

1) Entry data optimization

The first program called by the algorithm has the function to read the graph with distances between nodes in meters, in the form of floating point and turn them into seconds, considering that each plane has a constant average speed of 5 m/s. After reading the file, the optimized graph is written to a new file that will be used as input for each time an airplane has chosen a taxi route. This file contains the node's identification, the nodes that have connection and the distances between them.

2) Route calculation for each airplane

Performed individually for each flight, the program *Colony.exe* is executed several times and each time returns the partial result for each aircraft choice. The main program collects each of these partial results and save them in text file *output.txt*. This program performs the reading of the graph, i.e., the origin, destination and time of the flight, then calculates the best route using the optimization principle of ant system.

3) Conflicts detection and resolution

Once a solution is calculated for each aircraft, it is necessary to ensure the minimum safety requirements between aircrafts, since none of them should closely approach each other. In this case, we defined a minimum of 30 seconds (which corresponds to approximately 150 meters).

The second safety requirement is to assure that two aircrafts do not collide frontally. A solution set for a group of aircrafts is analyzed and a search for approximation problems is made. This program checks each route assigned to each plane, compares with the ones assigned to previous planes and, in case a conflict occurs, it is solved so that the priority is given to the first plane scheduled.

The main program executes each of the conflict detection programs three times so as to eliminate all the conflicts. In cases of larger aircraft flow, in which more collisions may occur, this amount of executions might have to be higher.

V. Simulations and Results

The tests were made in the system by changing properties so that results could be compared. These changes consist of scenery changes (variation in aircraft flow) and algorithm parameters. This section presents the results obtained with the model.

A. Case Study

Using data from Brasilia airport, containing all the nodes with their geographical coordinates, it was possible to calculate the distances between them.

	Flight	Date	Sched	Actual	Destination	Type	Delay	Orig. node	Final node	Inicial time (s)	Final time (s)
2	6300	31/jul	12:02	12:08	Recife	Departure	00:06	a16	a5	120	494
3	6222	31/jul	12:04	12:04	Aracaju	Departure	00:00	a17	a5	240	650
4	3578	31/jul	12:06	12:06	Guarulhos	Departure	00:00	a19	a5	360	802
5	6125	31/jul	12:09	12:09	Juazeiro do Norte	Departure	00:00	a16	a5	540	914
6	3845	31/jul	12:13	13:48	Belo Horizonte	Departure	01:35	a17	a5	780	1190
7	1760	31/jul	12:21	12:15	Cuiabá	Departure	00:00	a19	a5	1260	1702
8	6322	31/jul	12:26	12:26	Petrolina	Departure	00:00	a15	a5	1560	1891
9	6306	31/jul	12:28	13:09	Maceió	Departure	00:41	a16	a5	1680	2054
10	3718	31/jul	12:01	11:57	São Paulo	Arrival	00:00	a37	a16	60	345
11	3597	31/jul	12:05	12:05	Manaus	Arrival	00:00	a37	a17	300	549
12	3024	31/jul	12:07	12:19	Rio de Janeiro	Arrival	00:12	a37	a19	420	637
13	1684	31/jul	12:30	12:48	Guarulhos	Arrival	00:18	a37	a16	1800	2085
14	Total time:						02:52			9120	13313

Fig. 3. Flight data.

Default.txt

```
a16 120 a15 163 a14 205 a1 225 a2 313 a3 334 a4 455 a5 494
a17 240 a16 276 a15 319 a14 361 a1 381 a2 469 a3 490 a4 611 a5 650
a19 360 a17 392 a16 428 a15 471 a14 513 a1 533 a2 621 a3 642 a4 763 a5 802
a16 540 a15 583 a14 625 a1 645 a2 733 a3 754 a4 875 a5 914
a17 780 a16 816 a15 859 a14 901 a1 921 a2 1009 a3 1030 a4 1151 a5 1190
a19 1260 a17 1292 a16 1328 a15 1371 a14 1413 a1 1433 a2 1521 a3 1542 a4 1663 a5 1702
a15 1560 a14 1602 a1 1622 a2 1710 a3 1731 a4 1852 a5 1891
a16 1680 a15 1723 a14 1765 a1 1785 a2 1873 a3 1894 a4 2015 a5 2054
a37 60 a38 102 a27 180 a26 210 a25 243 a19 277 a17 309 a16 345
a37 300 a38 342 a27 420 a26 450 a25 483 a19 517 a17 549
a37 420 a38 462 a27 540 a26 570 a25 603 a19 637
a37 1800 a38 1842 a27 1920 a26 1950 a25 1983 a19 2017 a17 2049 a16 2085
```

Fig. 4. Output for an average flow of aircraft.

The flight schedules used on the tests were obtained from Brasilia airport website, on July 31, 2013, at varying times. Fig. 3 shows flight data used for testing.

TABLE I: Values for Tests

Parameter	Value
α	1
β	1
Q	100
τ_0	10
φ	0.01
ρ	0.01
$\tau_{inicial}$	50
N ºIterarions	50.000

The output presenting the taxiing routes for all 12 aircrafts is shown in Fig. 4, following the format [node] [time in seconds]. As noticed, no collisions occur between planes. For the first aircraft, for example, we have the origin node as a16, where the aircraft is at time t = 120s. Then it leaves for node a15, arriving there at t = 163s and so on, until it reaches its destination, the node a5, at t = 494s, spending, therefore, a total time of 374 seconds or 6.2 minutes to complete the path.

Table I shows the default values of parameters that were used in the simulations with varied flows.

B. Aircraft Flow Variations

For simulations using low flow of aircrafts, flights were chosen between 6:00 a.m. and 6:45 a.m. The total taxiing time for 6 flights was 32 minutes, an average of 5min 18s per flight.

For medium flow simulations the flights were scheduled to arrive or depart between 12:00 and 12:30. For a group of 12 aircrafts the taxi time was 70 minutes, 5min 48s per aircraft, on average.

Finally, the simulations showed that for high aircraft flow, the results were very close to the average flow. Nineteen flights scheduled to arrive or departure between 9:30 a.m.

and 10:00 presented a total time of 113 minutes to complete taxiing, an average of 6 minutes per flight.

The airport website does not inform the reasons for the delays, but they may be caused, for instance, by weather conditions, congested airport or mechanical problems. In order to compare results, we considered that delays over 30 minutes cannot be decreased.

The comparison between simulated data and actual data is presented in Table II.

Increasing in average taxiing time verified along every flow increase is due to greater number of collision between aircraft, which have to be resolved. However, the average time increase from medium flow to high flow was relatively short, which shows that the algorithm maintains its efficiency when the aircraft flow increases, within a given period of time.

TABLE II: DELAY DECREASE IN DIFFERENT SITUATIONS

Situation	Difference	Decrease
Medium flow	26 min	27%
High flow	67 min	32%

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we applied the ant colony methodology to determine the optimal taxiing routes for a set of aircrafts at an airport, avoiding conflicts between them.

The simulations showed that the model may reduce up to 32% the aircraft taxiing time, efficiently avoiding aircraft collision. In addition, it is possible to use the same model for any airport, being only necessary to adapt the airport information.

The developed algorithm presents a linear increase regarding the quantity of planes to be allocated and depends on variables such as number of iterations and parameters. The code performance was considered satisfactory, taking about 4 to 11 seconds to find results for each flight. Therefore, it is concluded that the objective was successfully achieved.

Future work might include:

1) In order to optimize even more airport resources usage, we suggest considering another airport elements in the model, like hangars.
2) Modeling the problem so as conflicts between aircrafts are solved as solutions are constructed.
3) Applying parallel computing and multi-threads can increase significantly the algorithm efficiency, as well as allowing data usage in real time, which provides adaptation to changes (such as delays or flight cancellations).

It should be stressed that this system isn't intended to replace the traffic controller; instead, it should be used as a complementary tool, allowing the traffic controller to make eventual adjustments, if deemed necessary.

REFERENCES

[1] M. Dib, W. G. Li, and A. de Melo, "Approach of balancing of the negotiation among agents in traffic synchronization," *IEEE Latin America Transactions*, vol. 5, pp. 338-345, 2007.

[2] Departamento de Controle de Espaço Aéreo. Regras do ar e serviços de tráfego aéreo, *Technical Report*, Comando da aeronáutica, 2009.

[3] H. Balakrishnan and I. Deonandan, "Evaluation of strategies for Reducing taxi–out emissions at airports," *Aviation Technology, Integration, and Operations (ATIO)*, 2010.

[4] M. Dorigo. "Ant colony optimization," *Scholarpedia*, 2007.

[5] C. Liu and Y. Wang, "Aircraft taxi path optimization based on ant colony algorithm," in *Proc. 4th International Symposium Computational Intelligence and Design (ISCID)*, vol. 2, pp. 226-229, October 2011.

[6] L. P. Rosa, D. M. Ferreira, L. L. B. V. Cruciol, W. G. Li, and D. X. Jun, "Utilization of genetic algorithms for management of taxi scheduling," *in Proc. The 2013 International Conference on Artificial Intelligence - ICAI'13*, Las Vegas, USA.

[7] L. Changyou and K. Guo. "Airport taxi scheduling optimization based on genetic algorithm," in *Proc. 2010 International Conference on Computational Intelligence and Security (CIS)*, Nanning, China, pp. 205-208, 2010.

[8] J. Dreo and P. Siarry, "Hybrid continuous interacting ant colony Aimed at enhanced global optimization," *Algorithmic Operations Research*, vol. 2, no. 1, pp. 52-64, 2007.

[9] M. Dorigo, M. Birattari, and T. Stützle, "Ant colony optimization: Artificial ants as a computational intelligence technique," *IEEE Computational Intelligence Magazine*, vol. 1, no. 4, pp. 28-39. 2006.

Kamila Borges Nogueira was born in Patos de Minas, Minas Gerais, Brazil on April 12, 1990. She graduated in computer science in 2013, from University of Brasilia, Brazil. Since 2010, she works at the Ministry of Education. Her research interests include artificial intelligence and transportation engineering.

Paulo Henrique Cabral Aguiar was born in Valparaiso, Goiás, Brazil, on July 30, 1986. He graduated in computer science in 2013, from University of Brasilia, Brazil. His research interests include distributed system and cloud computer.

Li Weigang serves as an associate professor and coordinator of TransLab–Laboratory of Computational Model for Air Transportation in the Department of Computer Science of the University of Brasilia–UnB. He is also a researcher supported by Brazilian National Council for Scientific and Technological Development (CNPq).