Iterative Planning of Airport Ground Movements

Charles Lesire
ONERA - DCSD
Toulouse, France
Email: charles.lesire@onera.fr

***Abstract*—Ground traffic optimization is a major issue of air traffic management: optimal ground circulation could decrease flight delays and consequently decrease costs and increase passenger wellness. This paper proposes a planning algorithm for ground traffic based on contract reservation. This algorithm is iterative: it plans aircraft itinerary one after the other. A first version is described using the classical A^* algorithm. Then the model is extended to deals with time and speed uncertainty to ensure the feasibility of the planned trajectories while avoiding conflicts between aircrafts. Its efficiency is compared to optimal results on a small airport, regarding quality of the solution and computation times.**

I. Introduction

One of the major issues of Air Traffic Management concerns the optimization of airport traffic. Indeed, the air traffic growth is having a hard impact on airport congestion. Flight delays are obviously impacted leading to an economic interest on ground traffic optimization methods. This optimization may also take into account ecologic issues such as noise and pollution reduction.

Ground traffic optimization can hardly be performed by human controllers: managing several aircrafts moving on the airport during rush hours on quite complex taxiway networks may be difficult. It is especially the case when hard weather conditions occur (e.g. fog).

A lot of researches have tried to help ground controllers either by defining new visualization displays (DST [1], AMAN [2], DMAN, etc.) or by improving traffic predictability by sharing flight data between airports and controllers (CDM [3]). Currently, these methods help improving controller situation awareness or traffic predictability, but are not used to help planning the ground movements.

A lot of approaches manage flight departure scheduling from the airport, using constraint relaxation [4], cooperative/coordinated plannings [5], [6], or optimization algorithms [7]. However, they do not consider prediction or feasibility of the ground movements that correspond to these scheduling.

Some authors then tried to estimate taxiing time without planning or simulating the complete aircraft movements: [8] estimates this time using reinforcement learning; [9] stochastically computes flight delay based on airport congestion; [10] statistically estimates taxiing time from past data.

Some works are dedicated to simulation tools that compute aircraft ground trajectories from arrival/departure schedules and a ground movement map, based on genetic [11] and Branch & Bound algorithms [7]. However, these approaches are used for simulation purposes: they do not control aircrafts in real-time. Moreover, they do not provide a complete itinerary for the aircraft but are more "reactive" approaches, avoiding conflicts by planning in a limited time window.

This paper presents an iterative algorithm for real-time planning of ground movements. This algorithm is intended to be used on-line to plan itineraries for aircrafts moving on an airport. These itineraries (sequel of points with time intervals) have then to be used, either by controllers, pilots, or by an automatic control law to control the aircraft speed along the trajectory. This algorithm is currently used in a simulation infrastructure allowing to evaluate airport capacities, environmental impacts, or optimization of new airport infrastructure.

Section II presents the overall problem and notations and briefly describes the concepts. Section III details the A^*-based algorithm and some preliminary results. Then uncertainty management is addressed and experimented in section IV. Finally, section V discusses the benefits of the proposed approach, its limits, and the way it could be improved.

II. Problem description

A. Graph representation

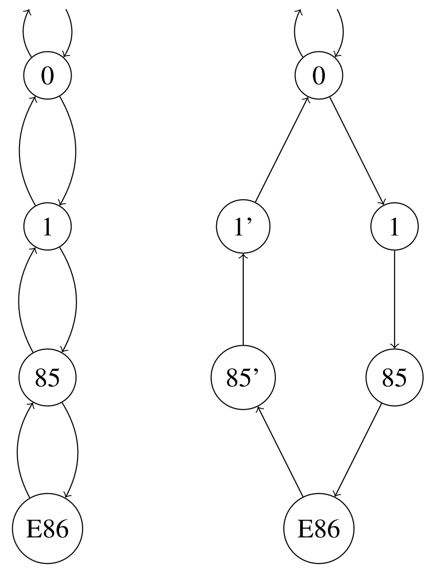
The airport infrastructure is modeled as an oriented graph $G = (V, E)$ where vertices V are located points of the airport (taxiway intersections, gates, runway access points), and edges E are the airport taxiways. Each edge (u, v) has a weight corresponding to the length of the edge, i.e. $dist(u, v)$.

A flight f is described by a starting vertex v_s (a gate for departures, or a runway for arrivals), a final vertex v_f, a starting time t_s (the departure time from gate for departures, or the estimated landing time for arrivals), and a type or category, that will constrain the maximal speed s_{max} of the aircraft. Moreover, aircraft separation must be ensured: two aircrafts must never be closer than a given distance D.

Departures usually follow a push-back procedure when leaving their gate. Such procedures are directly modeled in the graph structure by adding push-back nodes in the graph: the departure path from gates to push-back nodes are duplicated (Fig. 1).

B. Problem and constraints

The problem is then to find, for each flight $f_k \in F$, an itinerary, or contract, i.e. a set of points and associated times



(a) Initial graph. (b) Push-back duplicated.

Fig. 1. Push-back modeling: the departure path is duplicated.

$\sigma_k = (v_i, t_i)_{0 \leq i \leq l_k}$, such that:

- first and last points correspond to the flight characteristics

$$v_0 = v_s, \quad t_0 = t_s, \quad v_{l_k} = v_f \tag{1}$$

- consecutive points are reachable

$$\forall i, (v_i, v_{i+1}) \in E \tag{2}$$

- the aircraft speed is below its maximal speed

$$\forall i, t_{i+1} > t_i \text{ and } s_k = \frac{dist(v_i, v_{i+1})}{t_{i+1} - t_i} \leq s_{max} \tag{3}$$

- aircraft separation is ensured

$$\begin{gathered} \forall f_j \in F, j \neq k, \forall v \in V, \forall t, t' \,/\, (v,t) \in \sigma_k, (v, t') \in \sigma_j, \\ |t' - t| \geq \frac{D}{s_k} \end{gathered} \tag{4}$$

The overall objective is to minimize the travel time of all the aircrafts:

$$\min \Sigma_{f_k \in F} t_{l_k} \tag{5}$$

Computing a solution to this problem is quite complex. Although finding a path for a given aircraft f in the airport graph could be efficiently done in $\mathcal{O}(|V|^2)$ – Dijkstra algorithm complexity – computing a global optimum while managing time constraints (including separation) worsen the complexity to $\mathcal{O}(|F|!\,|V|^4)$. This is merely intractable without any appropriate resolution method.

The approach proposed in this paper decomposes the algorithm into iterative computations: each flight is planned one after the other. The contract of flight f_k is computed using the contracts of the flights already planned, without allowing to modify them. This solution is obviously not optimal regarding the global objective of equation (5). However, it is more realistic, as aircrafts start moving on the airport one after the other depending of their departure time. This approach is also robust to delays, as a flight starting δt after its initial starting time will not influence already planned flights but will try to be inserted in the current circulation.

III. Iterative planning algorithm

A. A^*-based modeling and planning

As discussed before, the approach proposed in this paper is iterative. Each flight will be announced and planned one after the other depending on its starting time. The flight itinerary is planned according to already reserved contracts in order to satisfy the separation constraint.

The algorithm is based on A^* ([12], algorithm 1): the itinerary is a path from an initial node v_0 to a final node v_f. It is a kind of best-first search algorithm, exploring nodes minimizing $f = g + h$ where g is the cost function and h the heuristic. If h is admissible (it must not overestimate the real cost to the goal), A^* returns a solution minimizing g.

Algorithm 1 A^* algorithm

```
O ← {v0}
∀v ∈ V, g(v) ← +∞,   g(v0) = 0
f(v0) ← h(v0, vf)
∀v ∈ V, p(v) ← v
while O ≠ ∅ do
    x ← argmax_{z ∈ argmin_{y∈O} f(y)} g(z)
    if x = vf then
        return path from v0 to vf
    end if
    O ← O − {x}
    for all (x, y) ∈ E do
        g'(y) = COST(x, y)
        if g'(y) < g(y) then
            g(y) ← g'(y)
            p(y) ← x
            O ← O ∪ {y}
        end if
    end for
end while
```

Constraints (3) and (4) are not managed by the algorithm itself – which is a standard A^* – but by defining an appropriate cost function. In standard shortest-path problems, g is defined as the weight matrix of graph G, and $COST$ function is given by equation (6).

$$\forall (u, v) \in E, \; COST(u, v) = g(u) + dist(u, v) \tag{6}$$

In the ground movements problem, the aim is to minimize the travel time of each flight. Hence, the cost function of a node v_i must be expressed according to the time taken by the aircraft to move from the initial point v_0 to v_i. Then $g(v_{i+1}) = t_{i+1}$. Constraint (3) corresponds to:

$$\frac{dist(v_i, v_{i+1})}{t_{i+1} - t_i} \leq s_{max} \Leftrightarrow g(v_i, v_{i+1}) \geq \frac{dist(v_i, v_{i+1})}{s_{max}} \quad (7)$$

providing a lower bound for t_{i+1}. Constraint (4) is satisfied by algorithm 2.

Algorithm 2 Cost function $COST(u, v)$.

```
t_v = t_u + dist(u,v)/s_max
for all f_j ∈ F, j < k do
    t' = contract(f_j, v)
    δ = D/s_uv = D/dist(u,v) (t_v − t_u)
    if |t_v − t'| < δ then
        t_v = t' + δ
    end if
end for
return t_v
```

This algorithm computes the shortest time t_v at which the aircraft will be able to arrive at v while satisfying the separation constraint. $contract(f_j)$ is the contract already planned for flight f_j, giving for each node v a time t' at which the aircraft will pass over v.
Algorithm 2 is executed at each step of the A^* algorithm. Hence the complexity of the contract computation for a flight is given by:

$$\mathcal{O}(|V|^2 \, |F|) \quad (8)$$

where $\mathcal{O}(|V|^2)$ is the complexity of A^* and $\mathcal{O}(|F|)$ the complexity of algorithm 2.

The heuristic function is given by equation (9).

$$h(v_i) = h(v_i, v_f) = \frac{dist(v_i, v_f)}{s_{max}} \quad (9)$$

This heuristic is admissible (see (10)) ensuring the optimality of A^*.

$$\begin{aligned} h(v_i) &= \frac{dist(v_i, v_f)}{s_{max}} \leq \Sigma_{j=i+1}^{f} \frac{dist(v_i, v_j)}{s_{max}} \quad (10) \\ &\leq \Sigma_{j=i+1}^{f} \frac{dist(v_i, v_j)}{s_{ij}} = \Sigma_{j=i+1}^{f} c(v_i, v_j) = g(v_i, v_f) \end{aligned}$$

B. Results

The previous algorithms have been implemented in C++, using the Boost Graph Library[1] structures and algorithms. Some experiments have been made based on the Toulouse-Blagnac airport, whose graph has 205 nodes and 361 edges (Fig. 11).

Figure 2 shows the number of delayed flights (in %) according to the number of flights planned on the airport during 100 hours. The flights start and final points are uniformly drawn from the set of gates and runways of the Blagnac graph. The flights starting time is uniformly drawn according to the number of flights managed during the 100h.

[1] www.boost.org/doc/libs/release/libs/graph

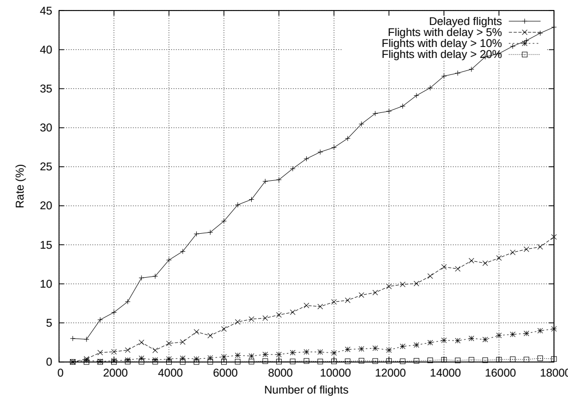


Fig. 2. Relative number of delayed flights.

The relative number of delayed flights (in % of the total number of flights) is linear, showing the complexity to manage a high number of aircrafts in such an airport. Results from an actual one-day traffic on Blagnac airport are shown in table I.

TABLE I
RESULTS ON A ONE-DAY TRAFFIC PLANNING.

Flights per hour	10
Delayed flights (%)	10.39
Flights w. delay $> 5\%$	1.3
Flights w. delay $> 10\%$	0.65
Flights w. delay $> 20\%$	0
Average delay (in %)	4.28
Worst delay (in %)	10.05

The number of delayed flights is not consistent between random simulation results and the real traffic data. This can be explained by the fact that the real traffic is not uniform over the day. Rush hours are nearer to 25 fl/h (2500 flights in 100 hours), giving more consistent results (around 10% of flights are delayed).

Figure 3 shows the resulting average and maximal delays for delayed flights according to the number of flights. The average and worst delays are consistent with those of the real Blagnac traffic results.

Globally, the results for the Blagnac airport give some acceptable delays. Managing around 20 flights per hour leads to 8% delayed flights, with an average delay less than 5% of their travel time.

Moreover, the computation time associated to the itinerary planning is less than 1 second per flight on a Core2 2.16GHz, 2Go RAM standard laptop, which makes the process fully usable on-line.

However, the resulting itineraries, that correspond to sequels of timed nodes, are not realistic. The hypothesis is that the aircraft speed is constant on each edge, leading to a

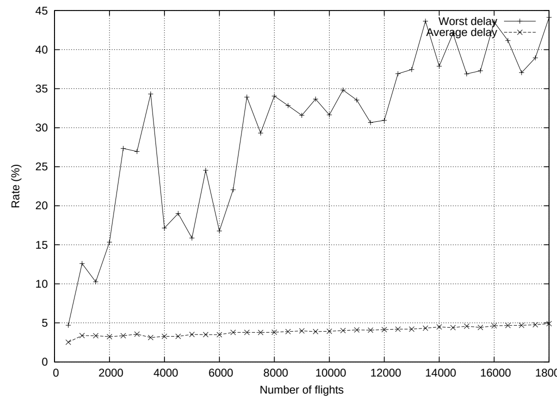


Fig. 3. Relative average and worst delays.

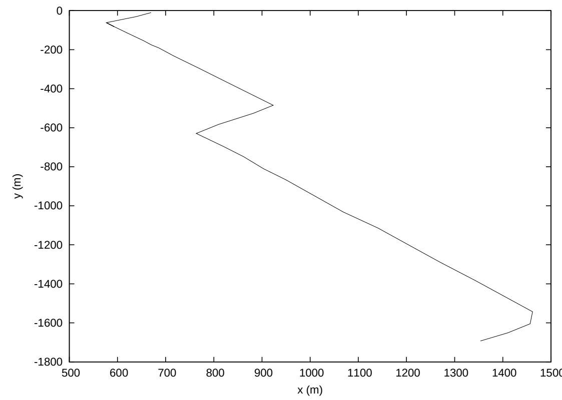


Fig. 5. Flight A trajectory.

discontinuous speed evolution of the aircraft (figure 4) along its trajectory (figure 5).

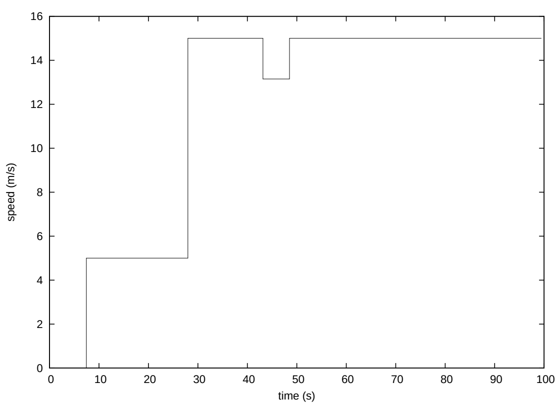


Fig. 4. Flight A speed profile.

The second drawback concerns the accuracy of starting time. To be sure an itinerary will be ready for an arriving flight as soon as it goes out of its runway, the planning process must compute its itinerary around a couple of seconds before it lands. However, the "starting time" (i.e. the time at which the aircraft will join the first taxiway) cannot be known precisely.

The following section deals with these two drawbacks and the way their associated uncertainties are managed in the planning algorithm.

IV. Managing uncertainty

Improving the realism of the planned itineraries means that the strong time constraint (a unique date associated to a node) must be relaxed. The itinerary must be represented as a sequel of nodes associated to time intervals. These intervals may be due to: (1) the uncertainty on the flight starting time (that will be propagated over the itinerary), or (2) the uncertainty on the aircraft speed, leading to an uncertainty on the time taken to cover a taxiway.

A. *Propagating time uncertainty*

To represent time uncertainty, the itinerary of flight f_k is now a set $\sigma_k = (v_i, T_i)_{0 \leq i \leq l_k}$, where T_i is an interval $[t_i^-, t_i^+]$. The cost function for the A^* algorithm must be defined to provide, for each node v_{i+1} the time interval T_{i+1} during which[2] the aircraft can go over node v_{i+1} while satisfying separation constraint (4).

As done in algorithm 2, T_{i+1} is iteratively computed by comparing the sooner possible interval T to already planned contracts T'. This comparison is based on the Allen's algebra [13]. Allen defines 13 relations to compare two intervals, summarized in Tab. II.

TABLE II
ALLEN'S ALGEBRA RELATIONS.

Timeline	Relation	Notation[3]
X	X before Y	$X < Y$
Y	Y after X	$Y > X$
X	X meets Y	XmY
Y	Y is met by X	$YmiX$
X	X overlaps Y	XoY
Y	Y is overlapped by X	$YoiX$
X	X starts Y	XsY
Y	Y is started by X	$YsiX$
X	X finishes Y	XfY
Y	Y is finished by X	$YfiX$
X	X during Y	XdY
Y	Y contains X	$YdiX$
X Y	X equals Y	$X = Y$

The fact that X is either (for instance) before or overlaps Y is noted $X\{b, o\}Y$.

[2] Actually f_k can be on v_{i+1} at any time $t \in T_{i+1}$.
[3] i stands for *inverse*.

Interval time computation is ensured by algorithm 3. Some specific points have to be detailed:

- The computation of the separation time is over-estimated to guarantee the separation constraint (line 4);
- If T_v has an intersection with $T' + \Delta$, and finishes later (line 6), then T_v is truncated: as the aircraft may arrive on v at any time between t_v^- and t_v^+, it can obviously move slower to arrive between $(t'^+ + \delta_T)$ and t_v^+;
- Line 8 is an extreme case of the previous one.

Algorithm 3 Interval cost function $COST(u, v)$.

1: $T_v \leftarrow T_u + \frac{dist(u,v)}{s_{max}}$
2: **for all** $f_j \in F, j < k$ **do**
3: $\quad T' \leftarrow contract(f_j, v)$
4: $\quad \delta_T \leftarrow \frac{D}{s_{min}} = \frac{D}{dist(u,v)}(\max(t_v^+, t'^+) - t_u^-)$
5: $\quad \Delta \leftarrow [-\delta_T, +\delta_T]$
6: $\quad$ **if** $T_v \,\{si, oi, di\}\, T' + \Delta$ **then**
7: $\quad\quad T_v \leftarrow [t'^+ + \delta_T, t_v^+]$
8: $\quad$ **else if** $T_v \,\{s, f, fi, o, d, =\}\, T' + \Delta$ **then**
9: $\quad\quad T_v \leftarrow t'^+ + \delta_T$
10: $\quad$ **end if**
11: **end for**
12: **return** T_v

In the special case where $T_u = [t_u, t_u]$ (i.e., is reduced to a single time) algorithm 3 is similar to algorithm 2.

B. Speed uncertainty

Managing starting time uncertainty gives some flexibility to the flight trajectories: arriving at a given node v must be done between t_v^- and t_v^+, allowing the aircraft to manage its speed. However, it is not sufficient: T_v intervals may be reduced to a singleton (algorithm 3, line 9), forcing a discontinuous speed profile.

Hence a speed uncertainty must be introduced in the $COST$ function to have a more realistic speed profile. This uncertainty is given by a δ_S parameter representing the tolerance over the nominal speed s_k. Typically, $\delta_S = 3\text{m/s}$ in the following experiments.

Then algorithm 3 is slightly modified to introduce the speed uncertainty, leading to algorithm 4.

The overall complexity has not changed ($\mathcal{O}(|V|^2\,|F|)$, equation (8)), but the computation time should be slightly higher as interval operations are more expensive than float operations.

C. Results

Figure 6 shows the speed profile bounds (min and max speeds) for Flight A (see Fig. 5 for flight trajectory and Fig. 4 for its initial speed profile). While there still is a discontinuity around $y = 40$, the provided profile allows the aircraft speed to be more smoothly controlled. The itinerary is now more realistic and executable.

Figures 7 and 8 present the evolution of the number of delayed flights and their delays according to the width of the starting time interval $|T_0|$. The number of delayed flights is

Algorithm 4 Interval cost function $COST(u, v)$ with speed uncertainty.

1: $T_v \leftarrow T_u + \frac{dist(u,v)}{[s_{max}-\delta_S, s_{max}+\delta_S]} = T_u + [\frac{dist(u,v)}{s_{max}+\delta_S}, \frac{dist(u,v)}{s_{max}-\delta_S}]$
2: **for all** $f_j \in F, j < k$ **do**
3: $\quad T' \leftarrow contract(f_j, v)$
4: $\quad \delta_T \leftarrow \frac{D}{s_{min}} = \frac{D}{dist(u,v)}(\max(t_v^+, t'^+) - t_u^-)$
5: $\quad \Delta \leftarrow [-\delta_T, +\delta_T]$
6: $\quad$ **if** $T_v \,\{si, oi, di\}\, T' + \Delta$ **then**
7: $\quad\quad T_v \leftarrow [t'^+ + \delta_T, t_v^+]$
8: $\quad$ **else if** $T_v \,\{s, f, fi, o, d, =\}\, T' + \Delta$ **then**
9: $\quad\quad T_v \leftarrow t'^+ + \delta_T$
10: $\quad$ **end if**
11: **end for**
12: **return** T_v

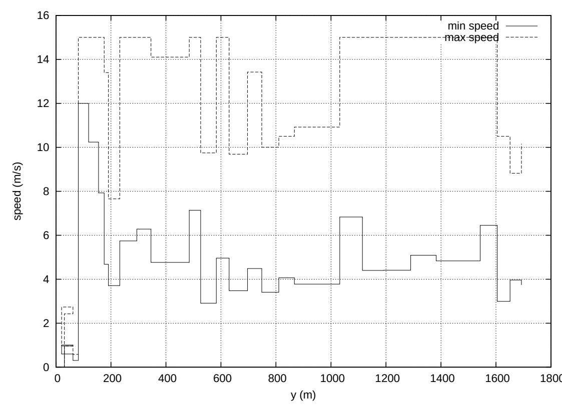


Fig. 6. Flight A speed bounds.

near constant (Fig. 7), meaning that $|T_0|$ has only a local effect on ”already delayed” flights. Moreover, although the maximal delay is linear according to $|T_0|$ – which is reasonable – the average delay is always under 20% (Fig. 8).

Figures 9 and 10 clearly show that speed uncertainty as very few influence on the number of delayed flights and their delays.

Table III shows results on the Blagnac airport actual traffic using a time interval uncertainty of 20 seconds and a speed uncertainty of 3 m/s. These results seem realistic compared to the actual situation.

TABLE III
RESULTS WITH $|T_0| = 20$ AND $\delta_S = 3$.

Flights per hour	10
Delayed flights (%)	29.2
Flights w. delay $> 5\%$	19.5
Flights w. delay $> 10\%$	18.2
Flights w. delay $> 20\%$	14.9
Average delay (in %)	17.8
Worst delay (in %)	447.4

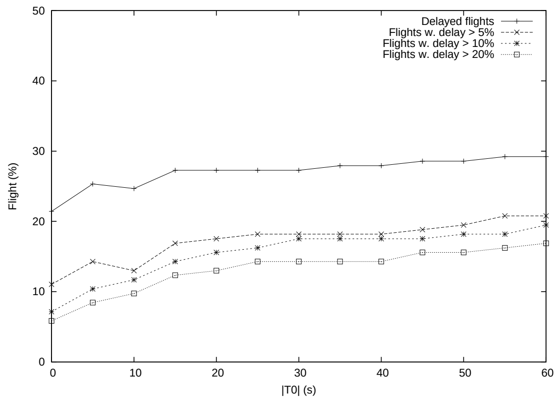


Fig. 7. Number of delayed flights according to the initial time uncertainty.

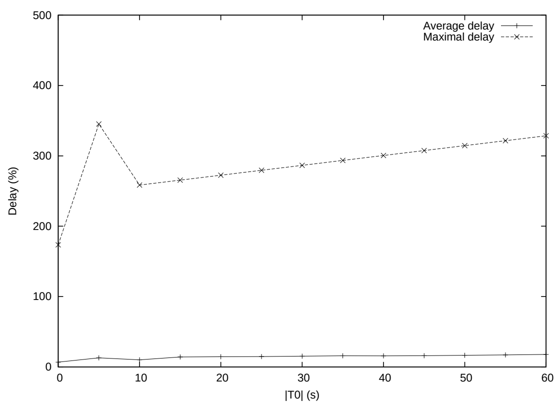


Fig. 8. Average and maximal delays according to the initial time uncertainty.

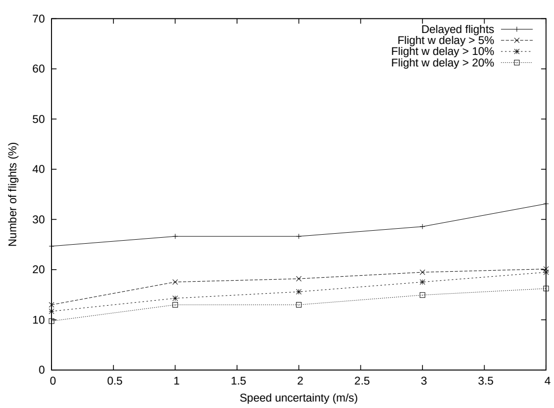


Fig. 9. Number of delayed flights according to speed uncertainty.

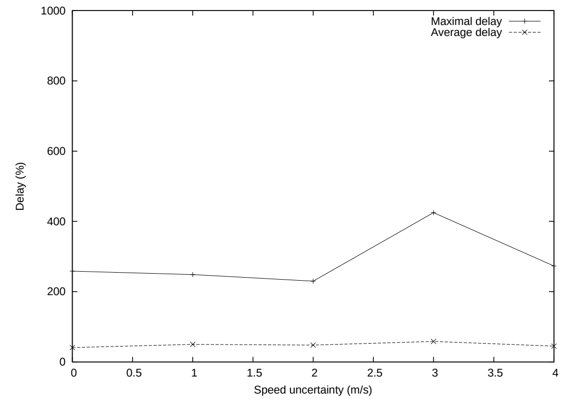


Fig. 10. Average and maximal delays according to speed uncertainty.

V. Conclusion

The approach proposed in this paper is dedicated to automatically plan flights ground movements. The planning algorithm is iterative, i.e. it plans flights one after the other, ensuring speed and separation constraints. The standard A^* algorithm has been modified to manage time and speed uncertainties as time intervals.

The results have shown the realism of the provided itineraries (in term of delays, speed profile and airport capacity), and proved the efficiency of the algorithm in term of computation time (less than 1 second per flight).

However, some drawbacks must be pointed out:

1) Controlling the aircraft speed to ensure separation may lead to unexpected situations where aircraft speed is very small; As separation constraint is only verified on nodes (and not on edges), a situation where several aircrafts are slowly moving on a busy taxiway is possible.
2) The planned trajectory are over-constrained: during execution, the aircraft will have a specific trajectory, arriving on each node at a unique time; next flights will not reconsider their itinerary and will then use a "worst-time" assumption.

These two issues will be addressed by adopting a real-time behavior, each flight planning (and modifying) its itinerary as it (and others) is moving on the airport taxiways. These could moreover allow to deal with runway crossing (which is dependent on the actual situation), and on-line controller clearances. These developments will then include a simulation of the aircraft trajectory intimately connected to the planning algorithm.

Finally, the proposed approach is to be used not only to plan and simulate ground movements, but also to evaluate airports capacities, or give accurate estimation of "gate to runway" travel time to the departure management team or runway control.

ACKNOWLEDGEMENTS

This work is part of the IESTA program, funded by a set of European (ERDF - European Regional Development Fund) and national French public credits.

The activities described in this paper are included in a collective work carried out by the whole team of the IESTA program. Moreover, the IESTA program owes its existence to a federative and close collaboration between several Onera scientific departments that gather a multi-disciplinary team of scientific experts of the wide range following domains: Long-Term Design & Systems Integration, Systems Control and Flight Dynamics, Computational Dynamics & Aeroacoustics, Physics, Instrumentation & Sensing, Aerodynamics & Energetics Modeling.

REFERENCES

[1] S. Swierstra and S. Green, "Common trajectory prediction capability for decision support tools," in *ATM R&D Seminar 2003*, Budapest, Hungary, 2003.

[2] H. Oberheid and D. Soffker, "Designing for cooperation mechanisms and procedures for air-ground integrated arrival management," in *IEEE Conf. SMC 2007*, Montréal, Canada, 2007.

[3] P. Martin, O. Delain, and F. Fakhoury, "Collaborative decision making: results of experiments to identify limitations of information exchanges in stand and gate operations," in *ATM R&D Seminar 2001*, Santa Fe, NM, USA, 2001.

[4] P. van Leeuwen and B. van Hanxleden, "Scheduling aircraft using constraint relaxation," in *UK Planning and Scheduling Meeting*, Glasgow, UK, 2003.

[5] H. de Jonge, E. Tuinstra, and R. Seljée, "Outbound punctuality sequencing by collaborative planning," NLR - National Aerospace Laboratory, The Netherlands, Tech. Rep., 2005.

[6] D. Bohme, R. Brucherseifer, and L. Christoffels, "Coordinated arrival departure management," in *ATM R&D Seminar*, Barcelona, Spain, 2007.

[7] R. Deau, J. Gotteland, and N. Durand, "Runways sequences and ground traffic optimisation," in *Int. Conf. on Research in Air Transportation (ICRAT'08)*, Fairfax, VA, USA, 2008.

[8] P. Balakrishna, R. Ganesan, and L. Sherry, "Application of reinforcement learning algorithms for predicting taxi-out times," in *ATM R&D Seminars*, Napa, CA, USA, 2009.

[9] R. Hoffman, M. Ball, R. Smith, and A. Mukherjee, "Ration-by-distance with equity garantees: a new approach to ground delay program planning and control," in *ATM R&D Seminar*, Barcelona, Spain, 2007.

[10] P. Pina and J. De Pablo, "Benefits obtained from the estimation and distribution of realistic taxi times," in *ATM R&D Seminar*, Baltimore, MD, USA, 2005.

[11] J. Gotteland, N. Durant, and J. Alliot, "Genetic algorithms applied to airport ground traffic optimization," in *Congress of Evolutionnary Computing*, Canberra, Australia, 2003.

[12] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, 1968.

[13] J. Allen, "Maintaining knowledge about temporal intervals," *Communications of the ACM*, vol. 26, no. 11, 1983.

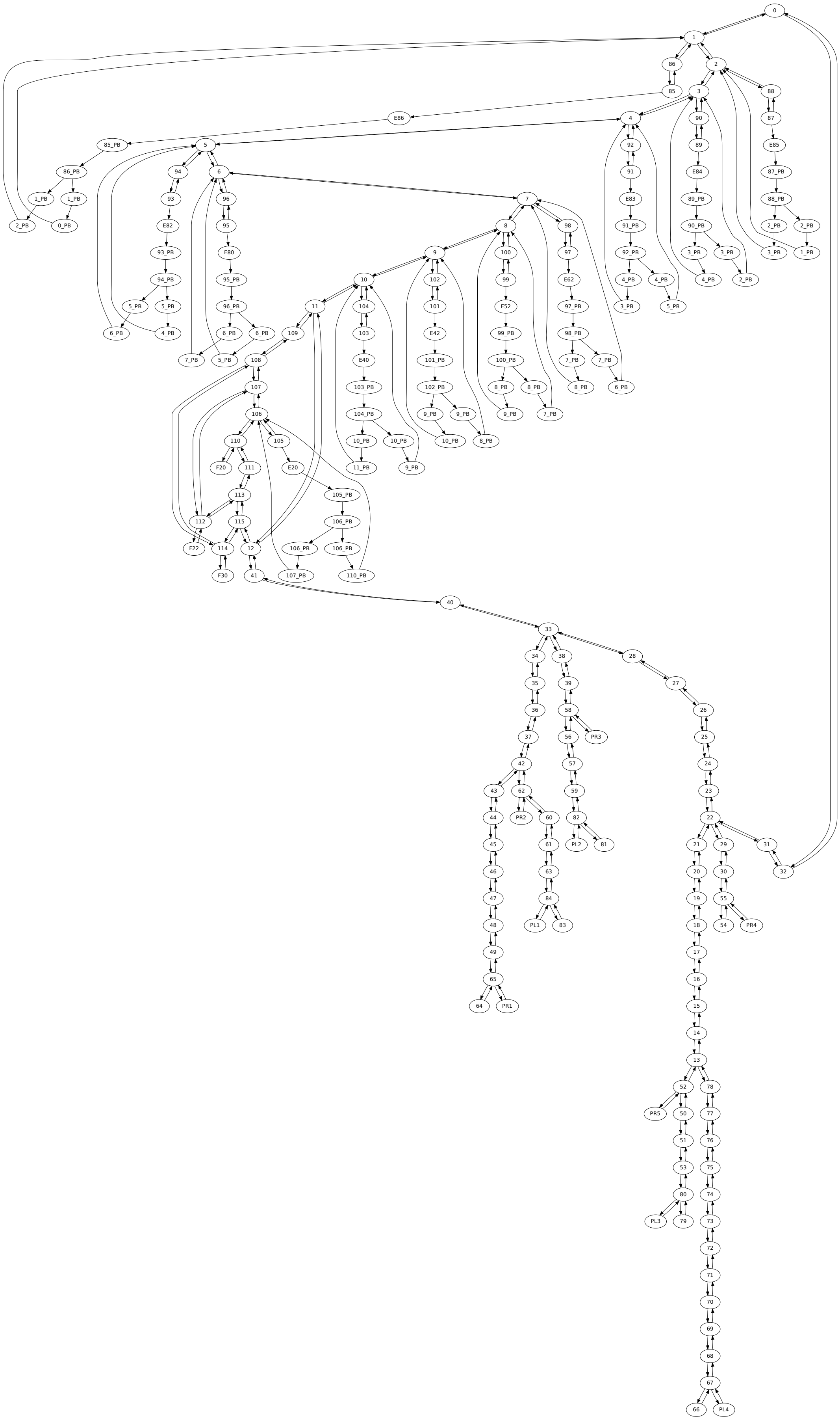


Fig. 11. The Toulouse-Blagnac airport graph.