Faculty of Electrical Engineering, Mathematics and Computer Science
Group of Microwave Technology and Systems for Radar (MTSR)

Design and evaluation of a simulation environment for evaluating departure scheduling algorithms

E. P. Stout

September 2011

Committee members:
Prof. Dr. Ir. E. Theunissen
Ir. G.J.M. Koeners
Ir. J.J.W. van Luipen
Dr. Ir. J.H. Weber

Abstract

To meet traffic demand predictions, the global air traffic management (ATM) system needs to be changed. Several visions on future ATM operations exist. A commonality between the different visions is 4D Trajectory management. This function enables plan-based operation as opposed to the state-based approach of the present system. Plan-based operation enables the optimization of traffic flows by generating 4D trajectories. A part of the traffic flow generation process is scheduling.

The research presented in this thesis focuses on these scheduling opportunities. In this research the scheduling opportunities for departure traffic at a runway are investigated. A study of the existing literature showed that the most common scheduling algorithms currently available can be divided into four categories: first come first served, brand-and-bound, greedy search and genetic algorithms. A simulation environment is designed for evaluation of the departure scheduling algorithms using various input parameters like traffic situation, airport map and algorithm. The four algorithm categories are evaluated on output aspects like delay and robustness of the schedule and are compared with the current method of traffic scheduling.

The evaluation of the scheduling algorithms shows that the performance of the current method of scheduling departure traffic performs well in comparison with the tested algorithms. In case of no disturbances the genetic algorithm performs slightly better than the current method, but the other algorithms do not have a better performance. When disturbances are taken into account, a bigger performance increase can be obtained by using scheduling algorithms.

Acknowledgements

The completion of this thesis has been one of the largest and most difficult challenges in my life. After months of reading literature and writing Matlab scripts, I have reached the end of my thesis work and with that my years as a student in Delft. I have spent my time in Delft with great pleasure. This was not possible without a lot of people. I thank everybody I have worked with, but I would like to name some people specifically.

First of all I would like to thank my supervisor, Ir. Joris Koeners for his patience and always being available for support and answering my questions. I also thank Prof. Dr. Ir. Theunissen for his excellence advice and reviews on my work.

I would like to thank my parents for their moral support, their good care and the facilities to relax to continue working with new energy. Last but not least, I want to thank the study association ETV for their constant supply of coffee and other social and valuable activities. Without them the time I spent studying in Delft would have been shorter, but much less pleasant.

Contents

List of figures

List of tables

List of abbreviations

ATM…………………………... Air Traffic Management
BIE……..……………………... Binary Implicit Enumeration
BNB……………………………Branch and Bound
CPS…………..…………………Constrained Position Shift
CTAS…………………………..Center/TRACON Automation System
CTOT…………………………. Calculated Take Off Time
DFW…………………………... Dallas Forth Worth International Airport
DSP…………………………….Departure Scheduling Problem
EASA…………………………. European Aviation Safety Agency
EIC……………………………. Belcher
ELD……………………………El Dorado
EOBT…………………………. Estimated Off-Block Time
ETA……………………………Estimated Time of Arrival
FAA……………………………Federal Aviation Agency
FAST………………………….. Final Approach Spacing Tool
FCFS…………………………...First Come, First Served
IE………………………………Implicit Enumeration
IMC…………………………… Instrument Meteorological Conditions
LIT……………………………..Little Rock
LP……..……………………….. Linear Programming
RTA……………………………Required Time of Arrival
SID……………………………. Standard Instrument Departure
STA…………………………… Scheduled Time of Arrival
TA……………………………...Time Advance
TMA…………………............... Traffic Management Advisor
TRACON……………………... Terminal Radar Approach Control
TXK……………………………Texarkana
4DT…………………………….Four-Dimensional Trajectory

XXII

1. Introduction

1.1 4DT air traffic management

One of the biggest challenges in air traffic is its growth. Although occurrences like the financial crisis had their impact on the growth of air traffic, it is expected that air traffic will still continue to grow with an annual rate of about 3% in the near future. This expected grow rate is shown in Figure 1.1. The grey lines show the MTF10b forecast, the blue lines the MTF11 forecast. For both forecasts the base scenario is shown as solid lines and the low/high scenario as dashed lines. [1]

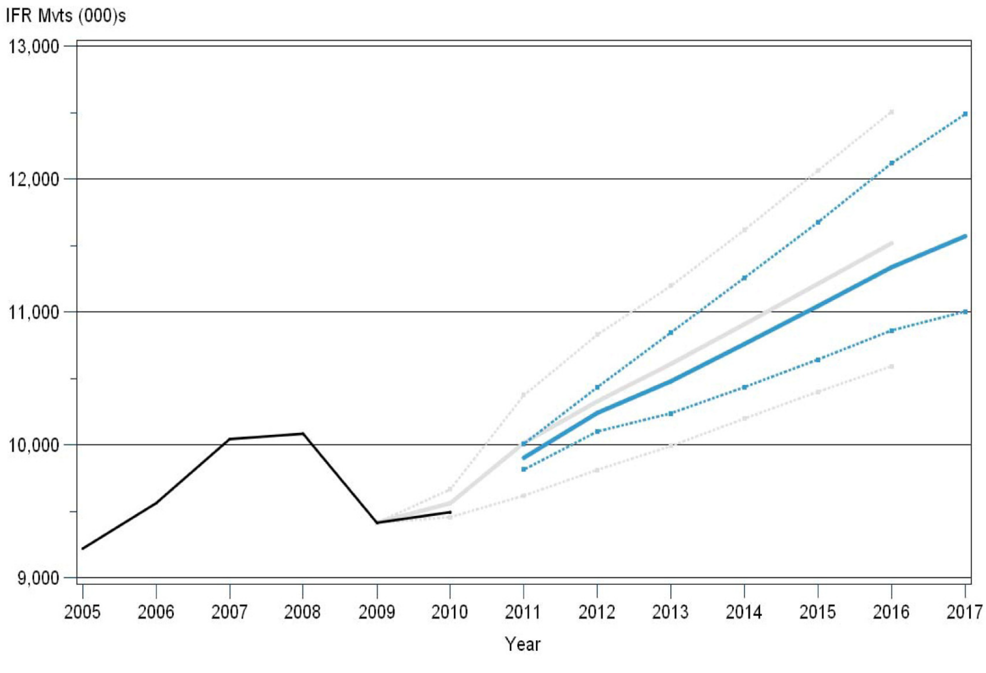

Figure 1.1: Expected growth of flight movements in the near future [1]

This growth is larger than the increase of the handling capacity at airports and air traffic controllers. Therefore, it is expected that the load factor of the air traffic will also continue to increase during the coming years, as is shown in Figure 1.2.

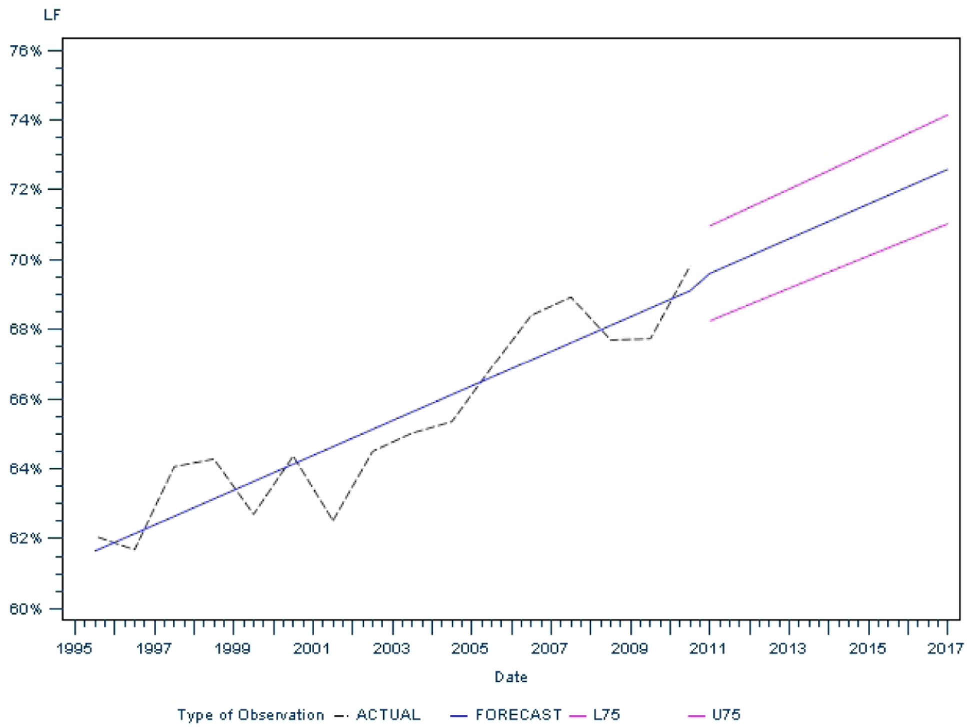

Figure 1.2: Forecast of load factor for the European air traffic [1]

When this growth continues, it is expected that there will be a lack of capacity in the future. To deal with this problem, the method for air traffic management which is currently used needs to be changed. The commonality of earlier research on this capacity problem is that, to deal with this growth, 4DT air traffic management is proposed to replace the state-based approach of the present system.

One of the most important differences between 4DT air traffic management and the current situation is that 4DT air traffic management uses plan-based operation instead of the currently used state-based operation. The biggest advantage of air traffic control via 4DT trajectories is that an estimate about the position of the aircraft at each future moment in time can be made. Instead of only defining the time at which the aircraft departs and arrives, its complete trajectory is defined in time. When, as in the current system, only the departure and arrival time are known, the position of the aircraft at each moment in time can also be estimated based on these times and the average aircraft speed, but this estimation is much less reliable compared to when 4DT trajectories are used, where the future position of the aircraft is not a rough estimation, but a part of the flight plan. In the new situation the aircraft are required to be at a certain position at a certain moment in time. This makes these positions much more accurate.

Plan-based air traffic control enables the opportunity to optimize traffic flows by generating 4DT trajectories. A part of this traffic flow optimization is scheduling. To optimize the traffic flow, a scheduling algorithm could generate a start time and speed profile for the whole trajectory of all aircraft. The ATM system provides time-slots at predetermined locations to every aircraft. These slots are based on the expected aircraft capabilities.

The runway capacity can be dependent on the order in which the aircraft depart. Big or heavy aircraft need a other separations than small aircraft [2]. At times when the amount of aircraft willing to depart approaches the maximum runway throughput, the runway capacity can be increased by changing the order of the departing aircraft. The possibility to do scheduling can also be used to change this order and thus to increase the runway capacity.

Research is needed to investigate if aircraft scheduling during the complete trajectory would improve the efficiency of the complete traffic flow. To be able to rate the possible efficiency improvement of the traffic flow, a method must be defined to compare the traffic flow in the different situations. Rating criteria must be defined to compare all the different situations. These are not only the situations with and without scheduling. There are a dozens of algorithms suitable for scheduling the air traffic in the 4DT trajectory-based situation. Because each algorithm implements a different strategy to optimize the departure sequence, it can be difficult to compare these scheduling algorithms. The optimal scheduling algorithm can change from minute to minute and depends on many parameters. Therefore a system to evaluate the algorithms and to be able to get insight in and evaluate their efficiency is required to be sure that using an algorithm for air traffic scheduling will be an improvement and not worsen the current situation.

1.2 Research goal

The goal of this research is to realize a simulation environment that allows an evaluation of traffic scheduling algorithms.

To structure this research, it will be divided into four steps:

1. A literature survey into existing scheduling algorithms. This literature survey gives more insight in the currently existing scheduling algorithms and helps to define the limitations of the current systems.

2. Classification of the algorithms, identification of the issues and defining rating criteria to compare the different algorithms. The issues and rating criteria are used to define a simulation scenario to evaluate the algorithms on as much aspects as possible to get a reliable result about the performance of the algorithms.

3. Selection of a concept, design and implementation of this concept. A simulation environment to evaluate the scheduling algorithms is designed and the algorithms are adapted to be used within this system.

4. Evaluation of the implementation using the defined rating criteria and discussion of the results. The system designed and the selected rating criteria and input scenarios are used to evaluate the selected algorithms. The outcome is used to draw a conclusion about the effectiveness of the algorithms.

1.3 Organization of the report

Chapter 2 defines the goal of this research. This chapter starts with an introduction on the departure scheduling problem. This problem is used to define the environmental factors the system should be able to take into account. The criteria to evaluate the system to be designed and to measure the effectiveness of the algorithms are defined here.

The next chapter, chapter 3, is used to discuss the results of the literature survey. More knowledge about the existing scheduling algorithms is needed to define the strong and weak points of these algorithms. This is used to define the rating criteria and input data to evaluate the system and the algorithms at a later stage. Without this knowledge it is impossible to be sure that the simulation is complete and covers all aspects of the scheduling process. Therefore more insight in the scheduling algorithms is essential before the system is designed. This chapter is concluded by a classification and an overview of the weak and strong points of the algorithms.

The design of the simulation environment is given in chapter 4. The implementation is split up into multiple subsystems. The function and implementation of each subsystem is discussed and the limitations and future applications of the design are given. The evaluation of the system and the selected scheduling algorithms is discussed in chapter 5. First, a standard situation is defined and evaluated. After that, all input parameters are varied and the effect of these variations on the efficiency of the algorithms is discussed. The chapter is ended with a conclusion about the efficiency of the algorithms. The last chapter, chapter 6, contains the conclusion of the complete research and recommendations for further research.

2. Scope of this research

This chapter describes the objective and scope of this research. First, the scheduling problem is investigated in more detail and also attention is paid on how to deal with disturbances. A possible solution to this problem is given in paragraph 2.2. This solution is worked out and explained in more detail further in this report. Finally, the design situation, design constraints and the rating criteria used to be able to deal with the problem are set out.

2.1 The scheduling problem

As explained in section 1.1, it is expected that the efficiency of airspace use can be increased by using scheduling. This research will investigate the real improvement of using scheduling algorithms and evaluates these scheduling algorithms. Before this can be done, it is important to define what scheduling and the scheduling problem are, and to define their applications in 4DT air traffic management.

A situation where no scheduling is needed is shown in Figure 2.1. In this figure the trajectories of two aircraft are shown in respectively red and blue. The trajectories have no common part, so it is impossible for the aircraft to be at the same place at the same time (a collision). The aircraft can move independently and thus no scheduling is needed.

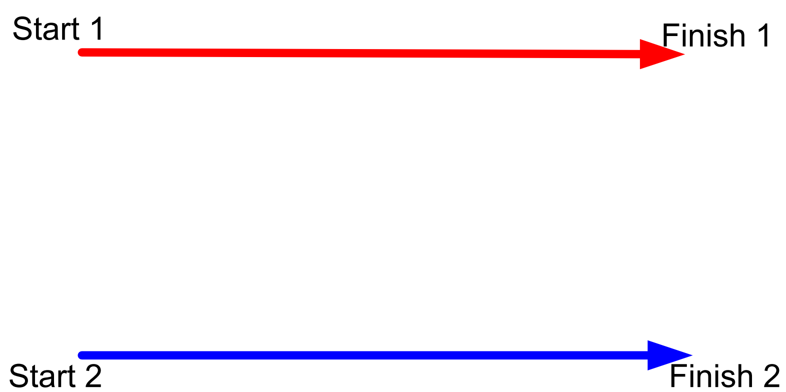

Figure 2.1: An example of a situation where no scheduling is needed

The most simple situation where scheduling is needed is shown in Figure 2.2. In this situation the trajectories of two aircraft have one segment in common (the purple part in the figure). This can be a shared segment or an intersection of both paths. It is possible that both aircraft planned to be at the shared segment at the same time. There is only space for one aircraft at each point on the trajectory, so a collision will occur. To prevent this collision scheduling is needed. This scheduling will prevent the aircraft to be at the same place at the same time by changing the departure times or the speed of one or both aircraft. These changes can have impact on the arrival time of the aircraft. This impact and the amount the real arrival time differs from the desired arrival time can decrease the efficiency of the algorithm and the final departure schedule.

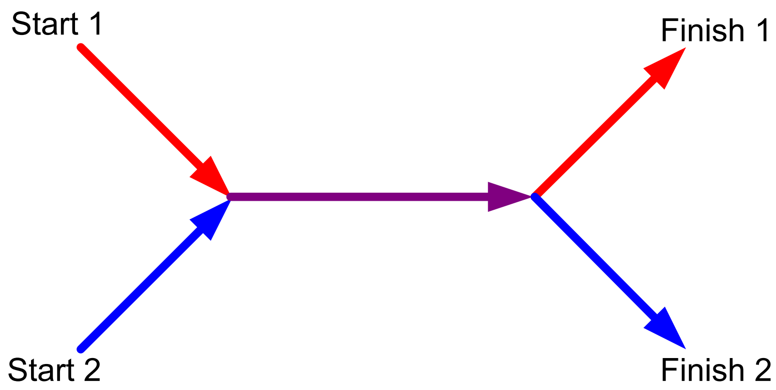

Figure 2.2: An example of a situation where scheduling is needed

The scheduling problem in Figure 2.3 is more difficult. In this situation both aircraft do not only share one part of their trajectory, but also their destination. Overtaking is assumed to be impossible, so the desired order of arrival at the finish defines the order at which the aircraft need to merge at the point where both paths come together. If aircraft 1 needs to arrive before aircraft 2, it must pass the point where both paths merge before aircraft 2. Otherwise it cannot arrive before aircraft 2 anymore. Also, if aircraft 1 gets delayed on the common path and aircraft 2 is behind this aircraft, the delay of aircraft 1 might cause aircraft 2 also to get delayed. The scheduling needed in this situation must be more intelligent. It must not only avoid both aircraft to be at the same place at the same time, but it should also take care about the delay that one aircraft can cause at the schedule of the other.

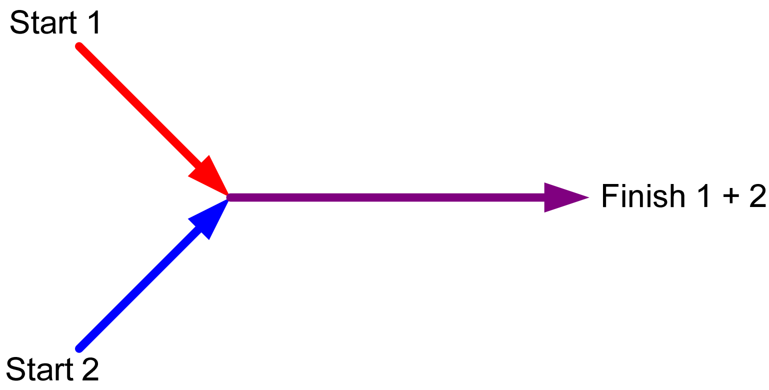

Figure 2.3: A more difficult scheduling problem

If more aircraft and paths are taken into account, the situation of Figure 2.3 can be extended to the situation shown in Figure 2.4. In this situation all aircraft have the same destination, but their departure location and the point where their paths merge is different. This situation also requires more intelligent scheduling. If aircraft 2 wants to reach the finish as the second aircraft, the scheduling algorithm must take care that it enters the common path before the third, and after the first aircraft. If the third aircraft enters the common path in front of the second aircraft, the second aircraft cannot reach the finish as second anymore. Also speed differences have their impact on the time when the aircraft need to enter the common path. Situations can occur where it is impossible to let all aircraft reach the finish at their desired time of arrival. Intelligent scheduling algorithms are needed to make the deviation as small as possible.

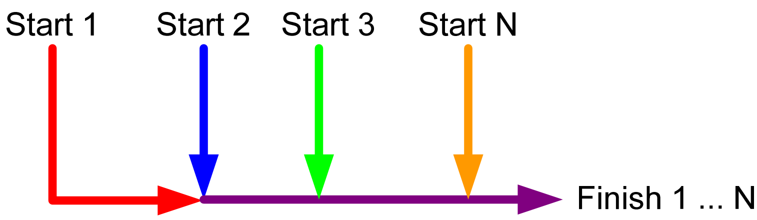

Figure 2.4: A scheduling situation with multiple limiting conditions

This scheduling scenario can also be used for the situation in aviation. At the departure airport, the aircraft needs to move from the location where it is parked to the runway. There are a fixed number of routes (taxiways) which the aircraft can follow to reach the runway. In case when one runway is used, the paths of all aircraft merge to one single path arriving at the runway. If there are multiple runways in use, the paths of all aircraft end at one of these runways, but as long there are less active runways as aircraft willing to depart, at least two of the desired paths have to merge at some point. Even if there are as much or more active runways as aircraft willing to depart, aircraft can have some part of the path in common, or can cross each others' path.

In some situations the beginning of the runway can be reached from two sides. For example when a part of the starting points is located at one side of the runway, and the other starting points are located at the other side of the runway. In this situation each aircraft uses one of both common paths. Both common paths are independent, but they end at the same point, the beginning of the runway. Coordination is needed to prevent aircraft from both common paths to arrive at the runway at the same time. This situation can be modeled as shown in Figure 2.5.

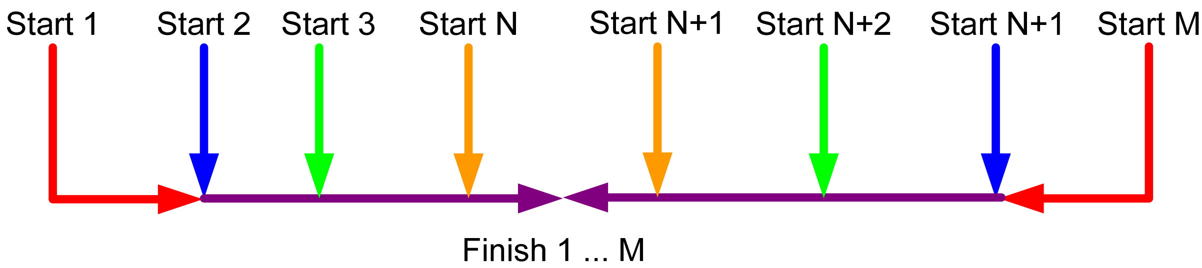

Figure 2.5: A scheduling situation with two common paths

The situation from departure runway to arrival runway is somewhat different. Except for parts of the trajectory like the final approach, aircraft are not required to follow a fixed trajectory. During the part without fixed trajectory, aircraft are assumed not to follow each others trajectory, so during these phase each aircraft has its own trajectory. This airborne phase can thus be modeled as a fixed trajectory with a private part (the free trajectory in the air) and a common part (the real fixed trajectory). Each airport is connected to a dozen (sometimes hundreds) of other airports. The complete situation can be modeled as a mesh network which is partially connected. If we assume that each airport has only one (active) runway, this network already consists of almost 44,000 nodes [3].

The last part of the trajectory is the part from runway to the parking place of the aircraft. It can be assumed that each runway leads to multiple parking places. The trajectories are fixed (taxiways). Therefore this part of the trajectory can be modeled like the trajectory in Figure 2.4 or Figure 2.5, with the directions reversed.

The complete trajectory of one aircraft from starting point to the final parking place at the destination airport is shown in Figure 2.6. The black arrows indicate merging or splitting paths.

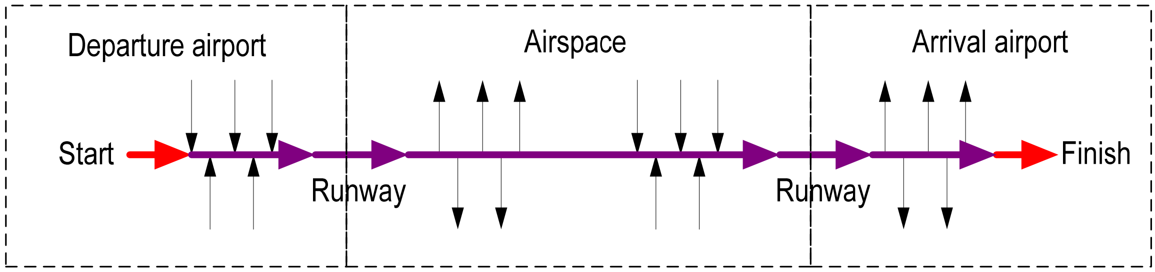

Figure 2.6: Overview of the trajectory of an aircraft

2.1.1 The departure scheduling problem

Figure 2.6 shows that an aircraft has to pass two critical points. These are the runways of the departure and arrival airports. On these runways all paths of the aircraft departing from or arriving at these airports overlap. These are the busiest points on the track and thus the points where congestion will occur first and where scheduling can be of most importance.

This research will focus on the departure scheduling problem (DSP). It is a big challenge to achieve an efficient departure schedule because constraints like safety, efficiency and equity need to be taken into account. These constraints are often competing. Besides this, the traffic situation includes the relatively short taxi distances, variance of aircraft characteristics and other aspects which can unexpectedly influence the situation on the airport. Due to this dynamic nature of the departure traffic the traffic situation is changing fast. Therefore a solution must be achieved in a short amount of time [4]. In existing literature, runways have been identified as the main source for system wide delay [5], so an important improvement can be made if this delay is reduced. In this paragraph only departure traffic is taken into account. It is assumed that the runway is exclusively used for departure traffic (single mode).

Despite the fact that for an airport with multiple runways the overall capacity is higher when all runways are used in mixed mode instead of single mode, this research focuses on runways used in single mode. This decision is made to make the results better comparable and less dependent on external factors like arrival traffic.

The requirement with highest impact on the runway throughput is the required separation between the aircraft. The minimum required wake vortex separation under IMC conditions depends on the weight class of the aircraft as shown in Table 2.1. These separation requirements are specified by the FAA and the EASA. Aircraft sizes are defined by weight ranges as: Small: 0 < Wt < 5,700kg; Large: 5,700kg < Wt < 136,000kg; Heavy: Wt > 136,000kg.

Table 2.1: Wake vortex separation requirements (NM/seconds) [6]

		Trailing aircraft			
		Small	**Large**	**B757**	**Heavy**
Leading aircraft	**Small**	2.5/80	2.5/68	2.5/66	2.5/64
	Large	4/164	2.5/73	2.5/66	2.5/64
	B757	5/201	4/115	4/102	4/101
	Heavy	6/239	5/148	5/136	4/104

The problem of assigning each aircraft a departure time taking into account the required separation minima is called the Departure Scheduling Problem (DSP). According to [7], the DSP can be defined as a total cost function as defined in equation (2.1), where

P the number of aircraft

E_i the earliest departure time for aircraft i

L_i the latest departure time for aircraft i

T_i the target departure time for aircraft i

g_i the penalty cost per unit of time for departure before target T_i for aircraft i

h_i the penalty cost per unit of time for departure after target T_i for aircraft i

S_{ij} the required separation time between aircraft i and aircraft j (where aircraft i lands before aircraft j)

x_i the departure time for aircraft i

δ_{ij} 1 if aircraft i lands before aircraft j, 0 otherwise

$$Z(\overline{x}) = \sum_{i=1}^{P} (g_i \max[0, T_i - x_i] + h_i \max[0, x_i - T_i]) \tag{2.1}$$

This cost-function is graphically shown in Figure 2.7.

It is assumed that for a single runway for each pair of aircraft one aircraft departs before the other (they do not depart simultaneously). This is modeled in (2.2).

$$\begin{gathered} C(\overline{x}) = \delta_{ij} + \delta_{ji} = 1 \\ i = 1, \ldots, P; j = i, \ldots, P; j \succ i \end{gathered} \tag{2.2}$$

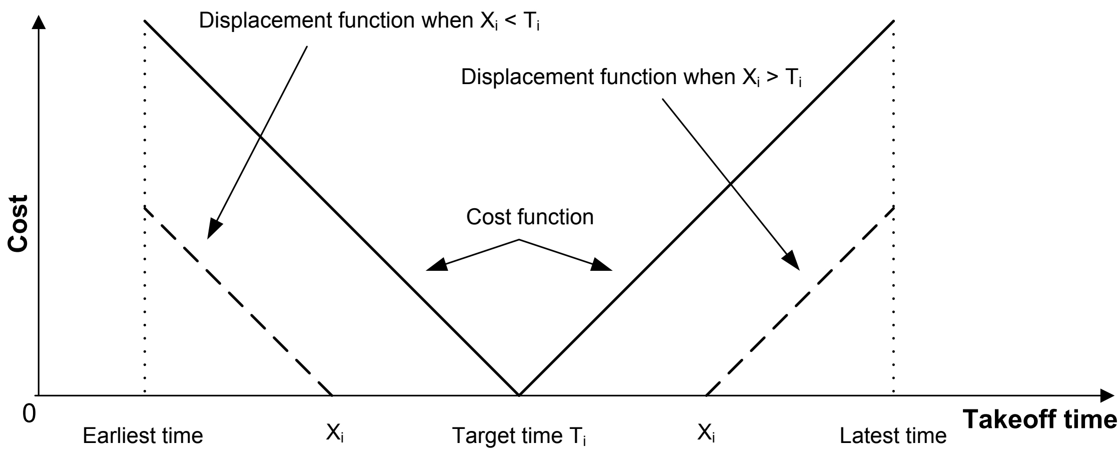

Figure 2.7: Cost function and displacement function [7]

Between each pair of aircraft, the minimum separation is maintained (2.3) and each aircraft departs within its time window (2.4).

$$x_j \geq x_i + S_{ij}\delta_{ij} - \left(L_i - E_j\right)\delta_{ji} \qquad (2.3)$$
$$i = 1,...,P;\ j = 1,...,P; i \neq j$$

$$E_i \leq x_i \leq L_i \qquad (2.4)$$
$$i = 1,...,P$$

The aim to maximize the runway throughput can be translated to the goal to minimize the total time needed to let all aircraft depart. According to [8] this can be mathematically represented as:

$$\min J = \sum_{i=1}^{N} (D_i - D_{i-1}) \qquad (2.5)$$

Where i indicates the i^{th} aircraft to depart, D_i is the departure time of the i^{th} aircraft and D_{i-1} is the departure time of the preceding aircraft. This equation can be simplified to:

$$\min J = D_N - D_0 \qquad (2.6)$$

Where D_0 is the departure time of the aircraft that is currently occupying the runway and D_N is the departure time of the last aircraft in the departure sequence. These separation equations are subject to constraints (2.7) and (2.8).

$$D_j - D_{j-1} \geq \max(\delta_{WV}^{j,j-1}, \delta_{MIT}^{j,j-1}) \qquad (2.7)$$

In this equations D_j is the departure time of the j^{th} aircraft, $\delta_{WV}{}^{j,j-1}$ is the required wake vortex separation between the j^{th} and the $j-1^{st}$ departure and $\delta_{MIT}{}^{j,j-1}$ is the required miles-in-trail separation between the j^{th} and the $j-1^{st}$ departure. The miles-in-trail separation requirements are dependent on the airport and the route of the aircraft. An example of nominal miles-in-trail separation requirements for aircraft departing from Dallas Forth Worth International airport (DFW) to four other airports is shown in Table 2.2.

The aircraft must depart in the order they are queued up. No overtaking of reordering is possible in the queues (2.8) [8] .

$$D_k^{Q_p} - D_{k-1}^{Q_p} \geq 0 \qquad (2.8)$$

Where Q_p is the p^{th} aircraft queue.

Table 2.2: Nominal miles-in-trail separation times (minutes)

		Trailing aircraft's destination			
		LIT	**TXK**	**EIC**	**ELD**
Leading aircraft's destination	**LIT**	2	1	1	1
	TXK	1	2	1	1
	EIC	1	1	2	1
	ELD	1	1	1	2

There are some specific situations where the queues are situated in such a way that reordering is possible within the queues. All possible options can be divided in three main groups (Figure 2.8). The situation in the left figure shows three queues. Each aircraft can be assigned to every queue, independent of the current location of the aircraft. In the middle situation, the aircraft are also lined up in three queues, but the queue in which a specific aircraft will be lined up is fixed and depends on which taxiway the aircraft uses to get to the queues. In the picture on the right there are only two queues. The center queue is left empty to make it possible to rearrange the aircraft within the queues. Every aircraft in both queues can leave the queue and taxi to the runway at every moment in time [9].

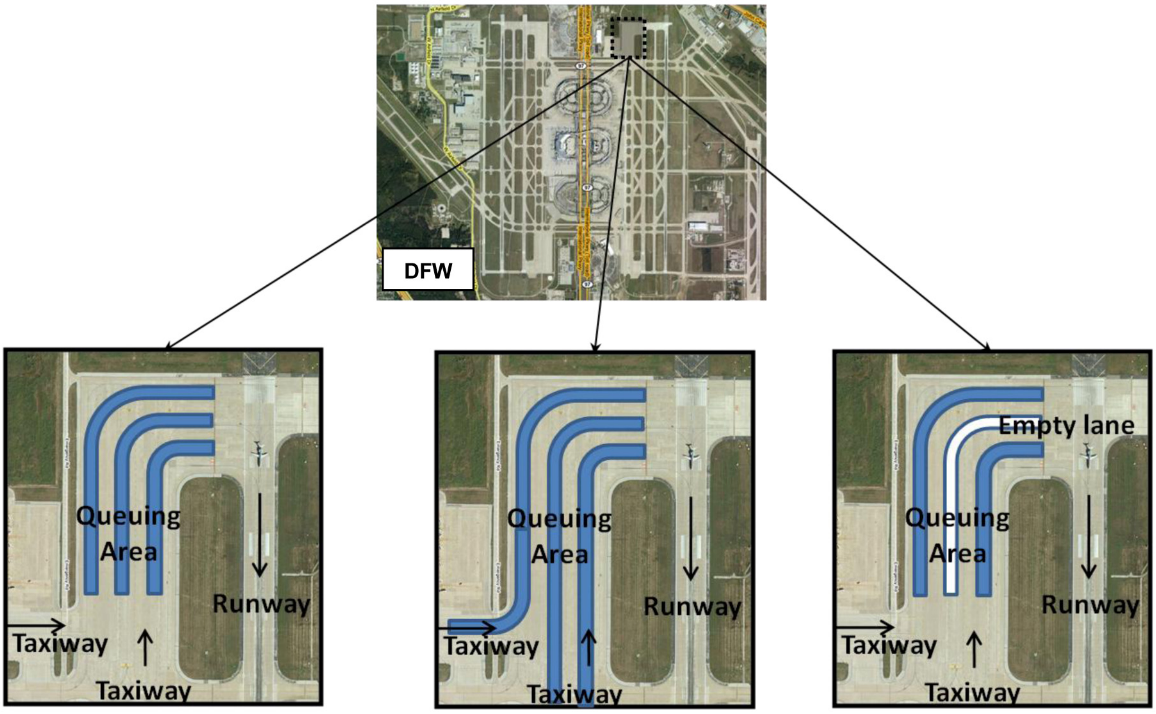

Figure 2.8: Queuing example for runway 17R at DFW [9]

It is assumed that the departure cost for an aircraft departing at its target time is zero. In reality these costs are not zero, but these costs are constant for each departure time, so taking these costs into account would only shift the cost function with a fixed cost and has no influence on the final result. According to [10], the linear cost function might not be the most realistic cost function, but the advantages by finding an optimal solution via a mathematical approach makes the linear cost function lead to better solutions than using a more accurate nonlinear cost function and solving it via a heuristic approach.

In accordance with the problem of scheduling aircraft landings, as discussed in [10], the departure scheduling problem described above can be solved using a LP-based tree search. To use this technique, the δ_{ij}, y_{ir}, and z_{ij} are stretched to continuous variables. In this situation, there are additional constraints which should be added to the problem. These constraints are redundant in the zero-one space, but improve the value of the LP relaxation in the continuous case.

Minimizing the departure cost as described is just one aspect of the evaluation of scheduling algorithms. There are more aspects which need to be taken into account when the algorithms are evaluated. The departure costs are a part of the total cost function which is used to evaluate the algorithms. The overall cost function is a tradeoff between all criteria. The weight factors of these criteria are flexible and can be adjusted by the user. The overall cost function is shown in equation (2.9), where

α_n the weight factor of the cost function f_n

f_n the cost function defining the cost of one of the aspects

x the aircraft involved in the simulation

$$Z(\bar{x}) = \alpha_1 f_1(\bar{x}) + \alpha_2 f_2(\bar{x}) + \ldots + \alpha_n f_n(\bar{x}) \tag{2.9}$$

It is important to keep in mind that presenting the user the outcome of this cost function is not the only option. Presenting only a single variable representing the efficiency of an algorithm can give the user the erroneous idea that an algorithm with a better efficiency parameter will always perform better than an algorithm with a worse efficiency parameter. Therefore it must be considered whether it is better not to present a single variable, but all individual cost functions of all aspects to the user. This gives the user the opportinity to compose its own cost function and will give him more insight the efficiency of the algorithms and how this is dependent on all rating criteria.

2.1.2 Disturbances

One of the biggest challenges in determining an optimal departure schedule is to deal with disturbances. Most disturbances are unexpected incidents which occur after the initial departure schedule is already computed. If no action is taken to deal with disturbances, they can grow and disturb more aircraft. In 2008, in the US, the average delay grew up to sixteen minutes per flight [11]. Looking at the complete aircraft trajectory, as shown in Figure 2.6, multiple type of disturbances can be distinguished. At each phase disturbances can occur. The total delay is distributed among all phases as shown in Figure 2.9 [11].

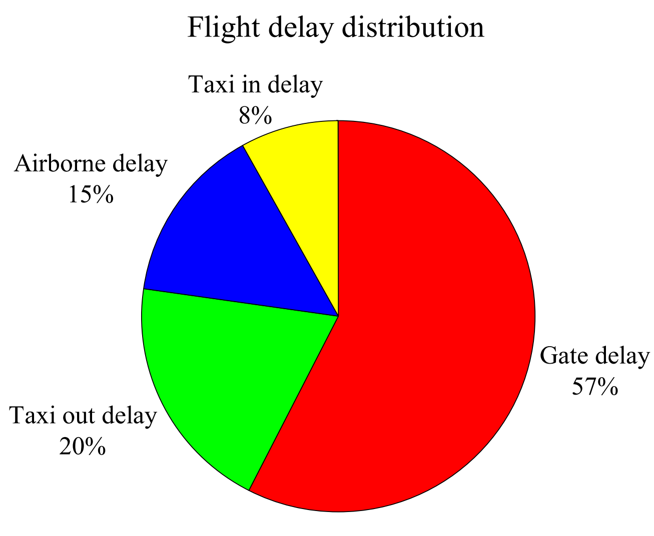

Figure 2.9: Flight delay distribution among all phases of flight

The total delay is not evenly spread over all aircraft. The departure delay of an individual aircraft consists of a part originating from a seasonal trend, a part from daily delay propagation and a random error. These factors causing the delay are shown in more detail in Figure 2.10 [12].

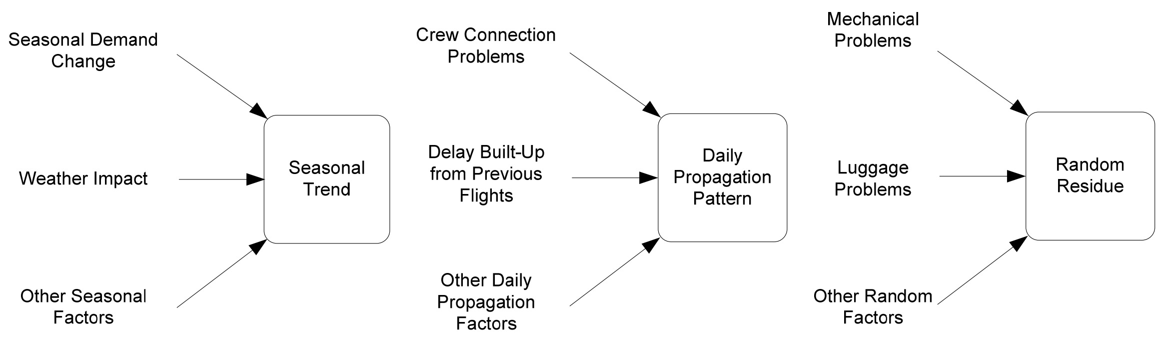

Figure 2.10: Factors influencing departure delay

When disturbance occurs, for example when an aircraft is not able to depart from the gate at its scheduled time or the speed of the aircraft during taxiing is different from the expected speed of the aircraft and thus the aircraft drifts away from its ideal position, actions need to be taken to limit the consequences of this disturbance.

Basically, there are two options to deal with this disturbance and to try to get the aircraft back at its desired position. The first option is to move the desired position of the aircraft towards the real position of the disturbed aircraft. This is done by changing the schedule of the aircraft. Rescheduling however, is a computational intensive process that might require a other aircraft to also change their plans. This might cause confusion and needs a lot of attention of the traffic controllers and the pilots of the aircraft. Sometimes rescheduling is the only option to correct for the arising disturbance, but it is preferred to correct the disturbance by only changing the plan of the disturbed aircraft. The other option is to try to correct the disturbance without rescheduling. This can save all non-disturbed aircraft from changing their schedule and thus it is assumed to be a less intensive operation.

Besides that, the disturbance is never exactly zero. Current aircraft are not equipped with systems to exactly maintain precisely defined taxi speeds and human pilots cannot follow the desired path with absolutely no deviation, so a certain deviation from the desired speed and position will always be present.

If the algorithm would try to correct the aircraft's position after every small disruption, the algorithm would continuously try to correct the aircraft. Most of the time, the pilots would not notify this because of the fact that in case of a very small disruption, the corrective operation is also very small and the result is almost equal to the situation before the correction. Nevertheless this is an unwanted situation because of the unnecessary increased workload it causes to the computer systems the algorithm is running on. Besides this, it would be annoying for the pilot if he would continuously be notified of negligible changes of the desired speed. This can be solved by hiding these small changes to the pilot, but it is better to prevent these rescheduling operations from taking place by defining a scheduling threshold (Figure 2.11).

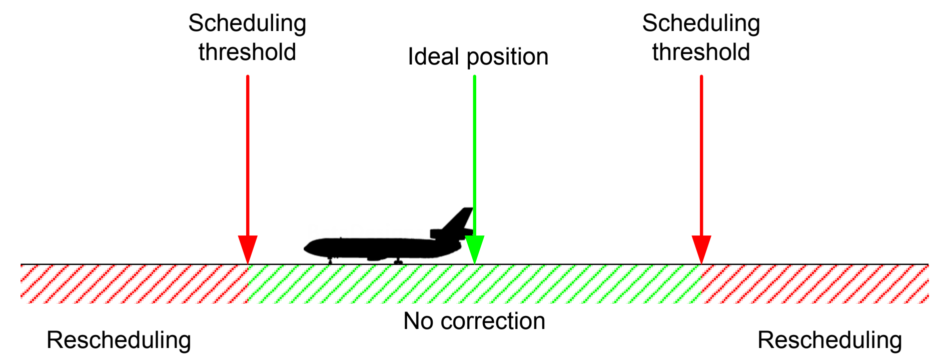

Figure 2.11: Thresholds are used to prevent unnecessary rescheduling

When the position of the aircraft is disturbed, but the disturbance is within the thresholds, no action is taken to correct the disturbance. This disturbance is bigger than a disturbance caused by an incidental speed deviation and thus a rescheduling operation is required to correct this disturbance. This system can be extended with a second threshold to further reduce the amount of rescheduling operations, as shown in Figure 2.12.

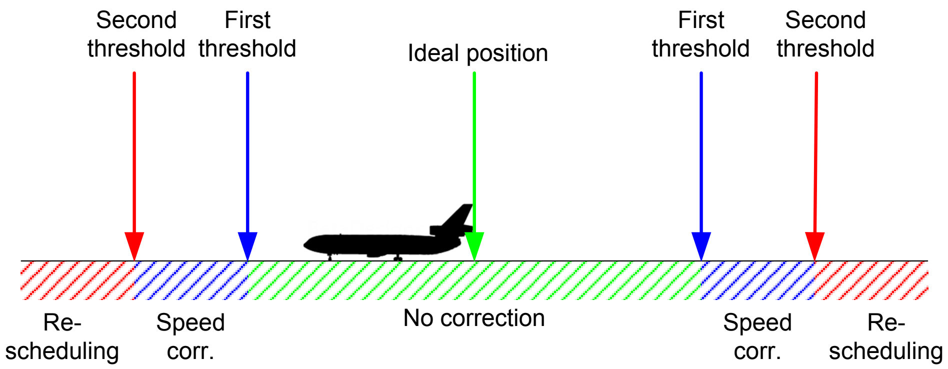

Figure 2.12: A second threshold is used to correct disturbances

When the position of the aircraft is disturbed, but this disturbance is within the first threshold, no action is taken to correct the disturbance. When the disturbance is between the first and second threshold, there is a significant disturbance. The deviation is big enough to take action, but the aircraft is still able to arrive at the runway at its scheduled time. In this case rescheduling is not needed (rescheduling is only needed when the aircraft is not able to reach the runway at its desired time anymore), but to prevent the disturbance getting bigger, a corrective action is taken. This action includes a change in speed and is done by the individual pilots without advice from the scheduling algorithm. As long as the pilots can correct the small disturbances by themselves, no rescheduling operations are needed. This margin makes the calculated schedule robust against small disturbances, so this has an influence on the total robustness of the calculated departure schedule.

When the aircraft exceeds the first threshold because it is taxiing too fast, the desired speed of the aircraft is decreased with an arbitrary value. If the aircraft exceeds the threshold on the other side of the window because it is taxiing too slowly, the desired speed of the aircraft is increased in the same way. This speed correction is taken by the pilot as a precaution to prevent the aircraft from drifting outside the second threshold and to try to get the aircraft back on its ideal schedule. When the aircraft is back at its ideal position, its speed is returned to the original desired speed. When, despite the speed corrections, the position disturbance of the aircraft keeps increasing and exceeds the second threshold, the aircraft is unable to reach the runway at the scheduled time anymore and a rescheduling operation is triggered.

It is important to keep in mind that, to be able to use this second threshold for speed corrections, the pilot must be aware of the deviation of the aircraft. Current aircraft are not equipped with systems to inform the pilot about this deviation. This second threshold can be added as an extra feature in the future, when 4DT air traffic management is used and aircraft are expected to be equipped with systems to detect these deviations from their ideal position. As long as these systems are not available, only one threshold as defined in Figure 2.11 can be used.

The rescheduling process can be done in multiple ways. Basically there are three possible options. The first option is to reschedule all aircraft when the position of one of them is disturbed (Figure 2.13). The current positions of all aircraft are taken as the new starting positions and the algorithm tries to find a new schedule best fitted to this situation. The advantages of this option include the fact that this option has the least restrictions. All aircraft are rescheduled and thus this is in fact nothing more than a complete new starting situation. However, we must keep in mind that when the aircraft are rescheduled in this way, it is required to be able to stop and halt the aircraft at each position on the taxiways if their new departure time is not reached yet. It should also be possible to overtake the aircraft when they are halted on the taxiway. Otherwise we should introduce restrictions like the possibility to lock the order of the aircraft if they are not able to change position anymore.

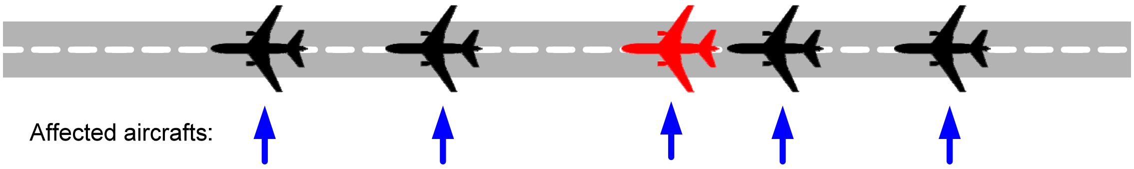

Figure 2.13: All aircraft are affected when an aircraft's position is disturbed. The disturbed aircraft is shown in red

Another possibility is to only reschedule the aircraft of which the position is disturbed (Figure 2.14). The biggest advantage of this option is that the schedule of the other aircraft is not influenced by this rescheduling operation, so the aircraft that keep their schedule do not have the risk of getting rescheduled on a later time (getting delayed) due to the delay of another aircraft. In practice, this rescheduling method is no more than holding the delayed aircraft until a free slot is found where it does not disturb other aircraft's positions. To be able to do this, it must be possible to hold the aircraft at every position on the taxiway. In practice this is often not possible. Existing taxiways are not designed to give the aircraft the possibility to overtake each other at every position. Often holding area's are defined at specific places along the taxiway where holding and overtaking is possible, but these are only located at a limited amount of positions. Holding at every position on or next to the taxiway will cause dangerous situations because the taxiway is not wide enough to overtake. Therefore this second rescheduling option is often not possible.

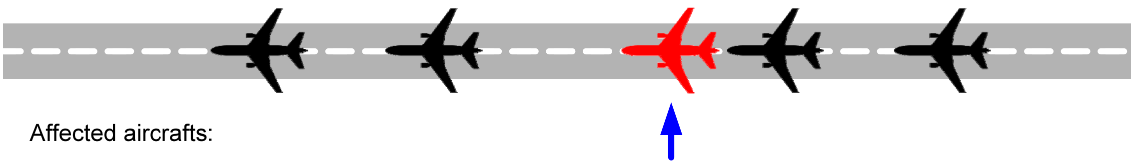

Figure 2.14: Only the aircraft of which its position is disturbed is affected. The disturbed aircraft is shown in red

The last option discussed here is to reschedule all aircraft except the aircraft with a disturbed position (Figure 2.15). Often it is quite difficult to predict the future behavior of this aircraft. Delays often occur as a result of an unforeseen incident. Because of this, it is difficult to predict how this disturbance will behave in the future. It could have been a once-only disturbance after which the aircraft will behave normal again, but it is also possible that the disturbance was caused by a problem which is still present and thus the disturbance can continue to grow in the future. Therefore it can be difficult to predict the future behavior of an aircraft of which its position is disturbed and thus it might be better to continue its current behavior instead of trying to change it. This might prevent future extra rescheduling operations due to the same aircraft.

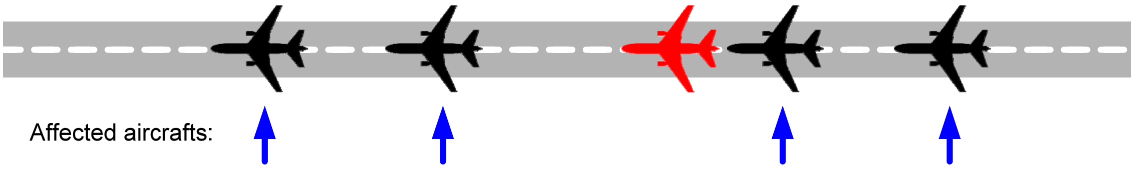

Figure 2.15: All aircraft except the aircraft with the disturbed position are affected. The disturbed aircraft is shown in red

In this option possible extra rescheduling operations are avoided by rescheduling all aircraft, except the one of which its position is disturbed. This aircraft is not scheduled with its original characteristics (speed, time of arrival at runway etc), but is expected to continue its path with its last known (disturbed) speed. The time of arrival of this aircraft is corrected for that speed. All other aircraft are rescheduled to try to minimize the effects of the aircraft with a disturbed position on the other aircraft as much as possible. During this rescheduling it is also taken into account that overtaking on the taxiways is not possible.

It must be considered whether it is preferable to use a combination of the three possibilities mentioned above, instead of just one of the options. It might be more efficient not to immediately reschedule all aircraft but, for example, only to reschedule the aircraft in the neighborhood of the problem aircraft. If this is not sufficient to solve the problem, more aircraft can be rescheduled. This method might be preferable, because it reduces the number of rescheduling operations, and thus the workload of the controllers and pilots.

As mentioned before, overtaking on the taxiway is considered as impossible, so other methods must be used to reduce the delay when rescheduling the aircraft. Once an aircraft has already entered the main taxiway and it is located behind the aircraft with a disturbed position, it cannot arrive before this aircraft anymore. If its desired time of arrival at the runway is before the realistic time of arrival of the aircraft with the disturbed position, the only possibility is to reduce the delay as much as possible by arriving at the runway as close as possible behind this aircraft.

When an aircraft is not on the main taxiway yet or still at the gate, there are more possibilities to reduce or prevent delay. Sometimes it can be predicted that the aircraft with the disturbed position will cause a future delay at another aircraft. If this delay is caused because the other aircraft ends up waiting behind the delayed aircraft, this delay can be prevented by letting the other aircraft enter the main taxiway before the aircraft which is disturbed arrives at this entrance point. After entering the main taxiway the undisturbed aircraft can decrease its speed to arrive as close to its original time of arrival as possible.

Concluding this section it is important to keep in mind that the goal of the algorithms under investigation is to improve the efficiency and not the safety. An aircraft will be rescheduled if it is not able to arrive at the runway at its scheduled time anymore. This is checked by comparing the real position of the aircraft with the ideal position of this aircraft. So a deviation from the ideal position triggers a rescheduling operation because of a deviation from the ideal arrival time and not because of a separation violation. By the calculation of the departure schedule, the safety regulations are taken into account, but the algorithms focuses on efficiency, not on safety. So to be sure that the safety margins are not violated, another system is still needed next to the system developed during this research.

2.2 Design goals and requirements

There are dozens of algorithms for scheduling the runway departure traffic. The difference between these algorithms is that they all use another strategy to determine the optimal departure schedule. Furthermore, they don't use a uniform definition of the term 'optimal'. There are three criteria most used to define the term 'optimal' in case of air traffic scheduling. These criteria are:

- Minimize the total delay
- Minimize the average delay per aircraft
- Maximize the runway throughput

Besides these criteria it is also possible to define a maximum delay per route segment (waiting for pushback, taxiing, queuing before runway, etc) to prevent aircraft to get parked for hours. It is also possible to define different costs for the delay of each aircraft class [13]. There is no uniform definition of what can be defined as the optimal result of the algorithm. The definition of what is optimal depends on the algorithm, the situation and the demands of air traffic control.

Due to the introduction of 4DT air traffic management also new scheduling opportunities are created. As a result a demand for the ability to be able to evaluate and compare all the scheduling algorithms arose. As mentioned in section 1.1, currently there is no such a tool to evaluate and select the optimal algorithm for generating the aircraft departure schedule. A system should be designed which is able to rate and evaluate the scheduling algorithms or to help the user to select the optimal scheduling algorithm.

The system to be designed does not select the optimal algorithm by itself. The performance of the algorithms is tested by simulating a traffic scenario where the algorithms are used to schedule the traffic. The results of this simulation are presented to the user via a number of output parameters. The user can add its own weight factors to these parameters. These outputs are used as a measure to evaluate the efficiency of the algorithm.

Summarized, the system that is going to be designed needs to perform the following tasks: A traffic situation is fed into the system. This situation can be an expected situation in the future, or the current situation on the airport, including the aircraft on the airport, the time at which they are ready to depart and their desired takeoff time. The system lets the algorithm perform its scheduling operation on the simulated situation. Based on the rating criteria, a cost function can be composed and its outcome is a measure for the effectiveness of the algorithm. This result is presented to the user. A functional overview of the system is shown in Figure 2.16.

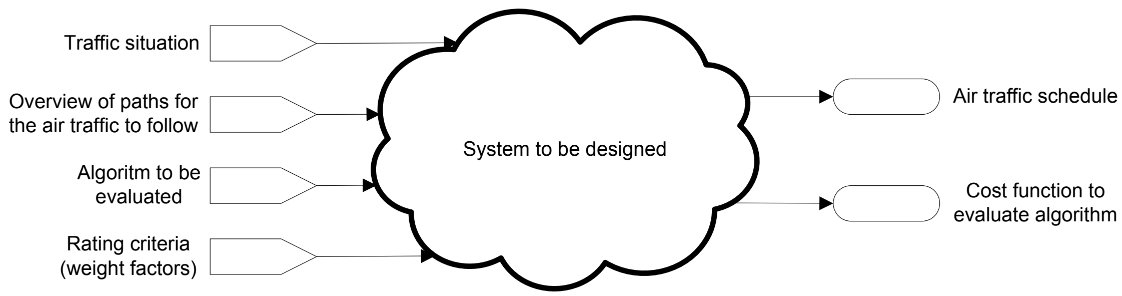

Figure 2.16: Functional overview of the system to be designed

The system does not only present the evaluation of the initial scheduling operation to the user, but will continue the simulation based on the results the algorithm provides. It will simulate the next time steps, and, if disturbances occur, also the impact of these disturbances and possible rescheduling operations. This will give the user the opportunity not only to evaluate the initial scheduling capabilities of the algorithm, but also the capability to handle disturbances and the robustness of the created departure schedules.

For the system to be universal usable, it must accept clearly defined inputs. This enables the possibility to use it for all airports, all traffic situations and all algorithms. This also makes the system future proof and makes it possible to use it to check the efficiency of new algorithms. Matlab is used to build the system to make it easy to add new features to the system or improve the existing possibilities.

The system must provide an output such that the user is able to make a decision about which algorithm is optimal for use in the simulated scenario. According to section 2.1.1, the definition of 'optimal' should be tunable, e.g. multiple output parameters are presented to the user. This gives the user the possibility to define a cost function to evaluate the algorithms.

The first things that are presented to the user to let the user be able to define a cost function are the parameters which show the performance on the aspects that are optimized by the algorithms. As mentioned before, these are the total delay, average delay and runway throughput. The throughput is presented as the amount of aircraft per hour that were able to depart in the current situation.

The other parameters that can be used to evaluate the algorithm are the variation of the time and speed windows as a function of time. The variation of these windows is an indication of the robustness of the algorithm. In the ideal situation these windows are large and do not vary much. The smaller the windows and the larger the variation, the more difficult it is for a pilot to keep its aircraft within the assigned window. To verify this robustness, the total number of rescheduling operations needed to let all aircraft take off is also presented to the user.

To approach the real world as much as possible during the simulation, extra inputs are needed to make the simulated situation less ideal and more realistic. The possibility to split up the traffic situation-input to simulate different types of traffic (in terms of allowed speed profile and separation) is already mentioned before. The second part is the ability to vary the amount of traffic, to create situations where only tight margins are available.

As explained in section 2.1.2, the simulation environment must be able to simulate disturbances. Therefore it should be possible to introduce disturbances into the system. This is done by introducing speed disturbances for the aircraft (changing the accuracy with which aircraft track the 4DT trajectory). There is also another way in which disturbances can be introduced into the system. For each aircraft there is a certain range over which the ground system (the algorithm) expects that its speed can be varied. This will determine the time window provided to the aircraft. The range over which the speed of the aircraft can be varied in reality can however be different from what the ground system expects. Both ranges are also inputs for the simulation. A difference in ranges from both inputs can also cause a disturbance if an aircraft is not able to move with the prescribed speed.

The last degree of freedom for the simulation is the update rate. To check the need for rescheduling, the simulation tracks all aircraft and determines if rescheduling is needed. The rate in which the time-constraint information is updated can be varied to investigate the influence of the update rate on the efficiency of the algorithm. Taking this into account, the functional overview of Figure 2.16 can be expanded to the new functional overview as shown in Figure 2.17.

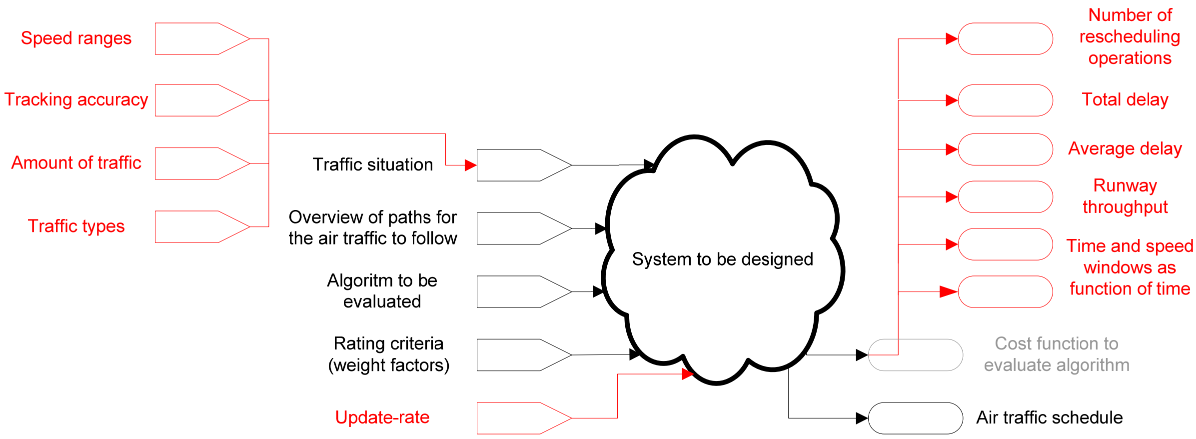

Figure 2.17: A more detailed functional overview of the system to be designed

The last requirement discussed here is the ability to keep evaluating the algorithm. The system must be able to continuously monitor the algorithm. To keep monitoring, also rescheduling must be taken into account. When an aircraft is disturbed in such a way that it is not possible to arrive at the runway at its scheduled time anymore, rescheduling or another corrective operation is needed. Each algorithm uses another principle of rescheduling and deciding when rescheduling is needed and thus this also has an impact on the efficiency of the algorithms. Therefore not only the initial scheduling, but also the possible rescheduling operations must be taken into account.

Summarized, the system must meet the following requirements:

- Able to deal with all 4DT departure scheduling algorithms
- Ability to take rescheduling operations into account

- Efficiency of the algorithms must be continuously evaluated
- Weight factors of output parameters can be adjusted by user

2.3 Design interaction

The algorithms involved in the simulation need external input data to perform their scheduling operations. The first required external input is the information about the paths which are available for the air traffic. These paths are the taxiways on the airports and the predefined paths in the air which the aircraft follow to reach their destination airport. The type and location of these paths are assumed not to be changing during the simulation, so this data is assumed to be constant. The way in which the paths are modeled is further investigated in section 2.3.1.

To create an optimal departure schedule, also information about the aircraft involved in the simulation is needed. First, the system needs to be informed about the time at which aircraft expect to be ready for leaving their spot. At that time the aircraft can start taxiing to the runway. The system uses this time to schedule a departure time for departure at the starting point. It is important to keep in mind that the aircraft can depart at a later time, but they cannot leave before they are ready to depart. It must also be known from which location the aircraft will depart.

To do accurate scheduling, more aircraft information is needed. The required separation is depending on the aircraft class. Details about the required separation are investigated in section 2.1.1. Besides separation constraints, the aircraft can also have different (minimum and maximum) speeds. During flight, the minimum and maximum speed of the aircraft is dependent on the aircraft characteristics. Also during taxiing, aircraft can have speed limitations. There is no minimum taxi speed, but to prevent overheating of tires and brakes, aircraft can have a maximum taxi speed [14]. The separation and speed requirements are defined by the class of the aircraft.

The inputs provided to the system are the times at which the aircraft expect to be ready for departure and the class of each aircraft. The airport map and the specification of the aircraft classes are assumed not to be changing during the simulation, so these variables are assumed to be static data and can be included in the system itself. To make the simulation more realistic, also disturbances are included.

The algorithm involved in the simulation performs the scheduling operation. Based on the input parameters, the algorithm determines a departure time and a speed for each aircraft. In principle this speed is constant over the entire path. This speed is equal to the ideal speed. However, due to disturbances, the algorithm can decide to give an aircraft another speed and departure time. This can also be a speed profile (not the same speed over the whole trajectory, but a speed varying from place to place). The algorithm also determines the time and speed windows for each aircraft.

In periods of heavy traffic, these windows can be so small that the aircraft must arrive exactly on time in order not to miss its departure slot. These departure slots have a fixed size and time and are assigned to the aircraft by the Air Traffic Management authorities. These slots and the presence of other aircraft can make it difficult to arrive at the runway on time. Therefore small departure windows can increase the runway capacity, but might reduce the robustness of the calculated schedule. Small windows result in small margins and thus an increased chance for rescheduling operations. A trade-off must be made between these aspects to find the optimal solution.

The main purpose of the system is to help the user to evaluate the scheduling algorithm. Therefore only providing the departure schedules computed by the algorithm is not sufficient. Additional outputs must be provided to be able to evaluate the algorithm and compare it with other algorithms. As mentioned before, this output consists of a set of output parameters such that the user can define its own definition of 'optimal'. This makes this output comparable with a kind of cost function, with tunable parameters. The goal is to make the total costs as low as possible. An overview of the required inputs and outputs is provided in Table 2.3.

Table 2.3: Inputs and outputs of the system to be designed

Inputs	Time at which aircraft are ready to depart Class of each aircraft Definitions of aircraft classes * Airport Layout * N Scheduling algorithms * Disturbances	**Outputs**	Speed profile and departure time of each aircraft Time and speed window of each aircraft Performance of each algorithm (multiple parameters)

** Static information*

2.3.1 Airport layout

The complete gate to gate scheduling process can be split up in multiple parts. Currently there are no algorithms used which perform the complete gate to gate scheduling. In practice, this trajectory is split up in multiple parts; a part from departure gate to departure runway, a part from departure runway to arrival runway and a part from arrival runway to arrival gate. In this research the phase from departure spot (gate) to departure runway will be investigated. The destination of the aircraft can be modeled as one single point (the runway). Within this phase of the trajectory, the inputs are not influenced by scheduling algorithms in a previous part of the trajectory, so algorithms can be evaluated in an objective and independent sense.

All the paths on the airport which the aircraft can take on their way from gate to runway are modeled as follows: Each starting point has its own taxiway. These taxiways are exclusively used by aircraft departing from that starting point. No other traffic is present on that taxiway. These taxiways feed in at one of the main taxiways. There can be more main taxiways on the airport. All taxiways from the staring points feed in at the main taxiways at different positions. The main taxiways lead to the start of the runway. Except for the points where the taxiways from the starting points feed in at the main taxiway and the point at the beginning of the runway where all main taxiways come together, there are no crossings or intersections. During the simulation, no other traffic is present on the taxiways. There is only one possible route from each starting point to the runway. This situation is schematically shown in Figure 2.18. In this figure two main taxiways are shown in red. The blue vertical lines are the taxiways from starting points to main taxiway. The lengths and number of taxiways are based on the real situation at the airport involved in the simulation.

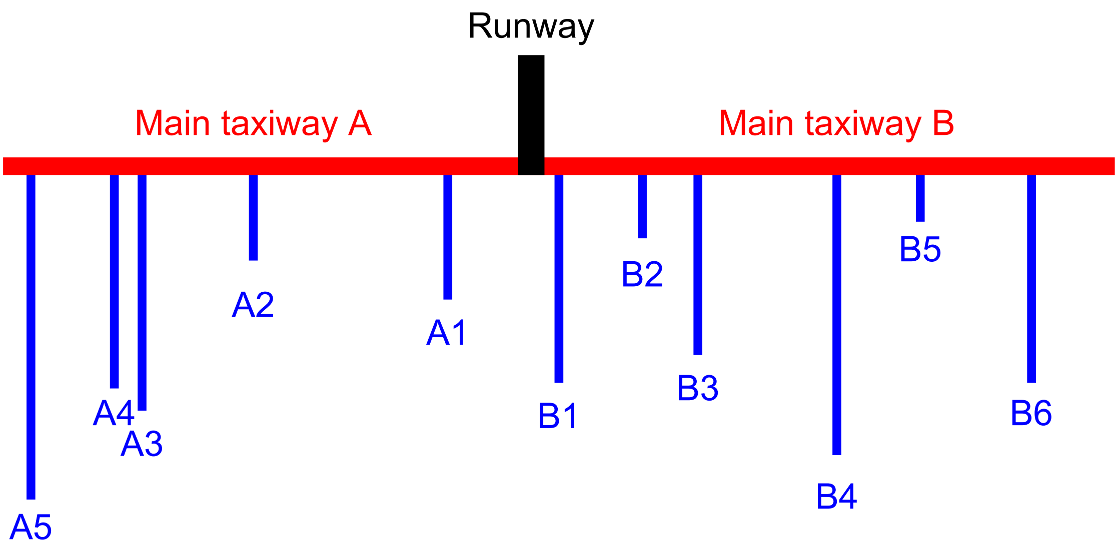

Figure 2.18: Schematic overview of the airport layout

According to [15], previous research shows that taxi speeds lower than eight knots do not occur in practice. Most of the time aircraft are taxiing at their maximum speed or, if they are close behind another aircraft, at the maximum speed of the preceding aircraft. This suggests that a minimum speed can be introduced for the taxiway without influencing the optimal aircraft schedule. This minimum speed does not influence the optimal aircraft schedule, but it reduces the computational power required to compute this schedule. Using a minimum and maximum speed, the earliest and latest time for which an aircraft is able to reach a certain point can be computed. Only if this time interval overlaps with the interval of another aircraft, a computation to calculate the optimal departure sequence is needed. If these windows do not overlap, there is only one sequence possible [15]. A summation of all constraints dealing with the airport layout is shown in Table 2.4.

The aircraft that wants to depart from the airport announces itself at ground control. It communicates the time from which it is available for pushback at the gate (pushback at a later time than the first possible moment is possible but not desired). Based on this earliest time of departure and the RTA at the start of the runway the algorithm calculates the scheduled departure time and the speed of the aircraft. Besides the scheduled time of departure, the algorithm also computes a departure window. In the case that an aircraft departs earlier or later than its desired time of departure, but still within its time window, it can still arrive at the runway at its scheduled takeoff time (the time delay or advance can be cancelled by in- or decreasing the aircraft speed).

Table 2.4: Design constraints dealing with the airport layout

Category	Constraint
Airport constraints	A single destination for all aircraft A single route from start to destination No other traffic on taxiways No crossings or intersections
Aircraft constraints	No influence of previous sections. Aircraft have minimum and maximum taxi speed

2.4 Rating criteria

The goal of this research is to build a simulation environment to evaluate (departure) scheduling algorithms. It is important to define rating criteria to be able to evaluate these algorithms. Currently there are no uniform rating criteria defined. In the current situation ground control tries to get the aircraft at the runway as fast as possible. They only try to optimize the situation for the individual aircraft instead of trying to optimize the overall situation [16].

One of the goals of departure scheduling algorithms is to maximize the runway throughput. In this case, the most optimal algorithm is the algorithm which achieves the highest throughput.

In specific cases, there are situations where multiple algorithms are able to deal with all the departing aircraft. In that case a second rating criterion must be defined [13].

Another rating criterion is based on the total delay of all aircraft involved in the simulation. It can be useful to use the delay as the rating criterion, because the aircraft need to depart at a specific time. When an aircraft is not able to depart at its assigned time, costs are involved to deal with this problem. Not only to prevent delay at arrival at the destination airport, but airports can also give fines to aircraft which do not depart in time, to encourage the airlines to keep their schedule and prevent congestion at the runway [17]. To really optimize the traffic flows at the airport, not the individual delays are important, but the total delay. The optimal algorithm is the algorithm which keeps the total delay of all aircraft the lowest. To prevent an algorithm to put aircraft aside for a very long time to prevent the other aircraft to be delayed, a maximum waiting time (delay) can be assigned to each individual aircraft.

A second drawback of using the total delay is that this delay is dependent on the amount of aircraft involved. During quiet periods it is much easier to keep the total delay at a low level than during busy periods. This can give a high degree of distortion to the result. This can be prevented by using the average delay per aircraft [13].

The robustness of the calculated schedules can also be of importance when evaluating the scheduling algorithms. A scheduling algorithm that generates a more robust departure schedule needs less rescheduling operations. This lowers the workload for the air traffic controllers and the pilots. Robustness is not a parameter which can be measured directly, so other parameters must be defined which are an indication for the robustness of the calculated departure schedules.

In this research the robustness is measured via the number of rescheduling operations and the number of aircraft departed out of their slots. An overview of all rating criteria and their advantages and disadvantages is given in Table 2.5.

As shown in Table 2.5, multiple rating criteria can be used to define the efficiency of the algorithms. There is no uniform definition of which rating criteria to use. This definition depends on the goal the user tries to achieve. Therefore the overall cost function should be a tradeoff between these criteria. The weight factor of these criteria and can be adjusted by the user. The individual parameters are presented to the user. The overall cost function is shown in equation (2.9).

Table 2.5: Summary of the proposed rating criteria

Parameter	Reason	Advantage	Disadvantage
Runway throughput	- Runway capacity is one of the bottlenecks within air traffic	- A higher throughput removes this bottleneck	- A drawback of higher throughput can be enormous delays
Total delay	- Higher delay means higher costs for airlines	- Directly related to accuracy of departure times	- Total delay is directly related to the number of aircraft involved - High delay of individual aircraft is invisible - A drawback of low delays can be a low runway capacity
Individual delay	- Higher delay means higher costs for airlines	- Directly related to accuracy of departure times - Not influenced by total number of aircraft	- A drawback of low delays can be a low runway capacity
Average delay	- Higher delay means higher costs for airlines	- Directly related to accuracy of departure times - Not influenced by total number of aircraft	- A drawback of low delays can be a low runway capacity - High delay of individual aircraft is invisible
Robustness	- A robust schedule lowers the air traffic controllers' workload	- Not influenced by individual aircraft	- Drawbacks of robust schedules can be low runway capacities or enormous delays

The focus of this research is on the design of the simulation environment for the evaluation of traffic scheduling algorithms and this evaluation. This evaluation is kept as general as possible. Therefore the choice of the weight factors is not a part of this research. For the evaluation in this research all rating criteria are taken into account on an equal value.

3. Scheduling algorithms

The goal of this literature survey is to get more insight into existing scheduling algorithms used for scheduling air traffic. The literature survey focuses on scheduling algorithms for departure traffic at airports. This literature survey starts with a discussion of twelve scheduling algorithms. The literature survey is concluded by dividing the scheduling algorithms into a number of categories.

3.1 Overview of algorithms

The problem of finding the optimal departure schedule is often a complex problem with millions of possible solutions. A situation where twenty aircraft are involved already results in $2.4 * 10^{18}$ possible sequences. However, the number of aircraft involved in the problem is finite, so the number of possible solutions is immense, but finite. Therefore it is, theoretically, possible to calculate the efficiency of all possible solutions and select the optimal solution from these results.

However, the time needed to calculate the efficiency of all possible solutions is often not available, so a method must be found to find the optimal departure schedule faster. Scheduling algorithms are designed to perform this task. Scheduling algorithms provide a method to find, or approach, the global optimum via local optima. These local optima are easier to calculate and require less computational power. The exact method used, and the efficiency of this method depends on the algorithm used.

The algorithms selected to discuss during this literature review are not randomly chosen. The first selection criterion is whether they are expected to produce reasonable good results. Algorithms for which it is known in advance that their performance is poor, would never be of use for improving the efficiency of the air traffic and therefore are not selected. Besides this, the algorithms are selected to represent all algorithm categories. There are four main principles on which all algorithms are based. At least one algorithm for each principle is selected for this literature review. All these algorithms are discussed in section 3.1.1 up to section 3.1.12. A quick overview of all algorithms discussed is shown in Table 3.1.

Table 3.1: Overview of all discussed algorithms

Section	Algorithm	Principle	Literature for more information
3.1.1	Implicit enumeration	Using a branch and bound methodology, an optimal schedule is calculated based on criteria like fuel or delay minimization.	[18], [19], [20]
3.1.2	pFAST / aFAST	Split trajectory of aircraft in segments. Do conflict resolution if aircraft share a segment.	[21], [22]
3.1.3	Modified first come first served	Schedule aircraft based on the order of their ETAs. Use scheduling horizons to calculate schedule and freeze schedule.	[23]
3.1.4	Heuristic time advance	Divide aircraft into groups and schedule tightly within group. Speed up all aircraft except first of each group.	[23]
3.1.5	Fuel saving time advance	Based on Heuristic time advance, but only speed up if aircraft benefit from a time advance.	[23]
3.1.6	Idealized fuel saving time advance	Optimize schedule by using time advance. Amount of time advance based on a cost function for the fuel use.	[23]
3.1.7	Pure time advance	Simplified version of idealized fuel saving time advance. Simplified by not removing unnecessary time advances.	[23]
3.1.8	Realizable fuel saving time advance	Based on idealized fuel saving time advance. Use a finite scheduling window.	[23]
3.1.9	Optimal constrained position shift	Reorder existing FCFS schedule by shifting aircraft at most one position.	[24]
3.1.10	Heuristic constrained position shift	Based on optimal constrained position shift. Try to achieve a number of fixed patterns for each group of aircraft.	[23]
3.1.11	Greedy search	Queue aircraft in (virtual) rows and pick optimal aircraft from aircraft at front of queues.	[8]
3.1.12	Genetic	Generate a number of random sequences by changing aircraft within sequence. Keep only most efficient sequence(s) and start again.	[25], [26]

The last aspect to check is whether the algorithms can be used for the situation discussed in this research. The algorithms can only be used if they are able to perform their scheduling task with the supplied inputs and don't need extra, unavailable, data. The outputs should also be in accordance with the expected output parameters. The algorithms must be able to provide not only a departure sequence, but also assign a departure time to each aircraft. This usability is discussed in section 3.2.

3.1.1 Implicit enumeration

The implicit enumeration (IE) scheduling, or Brinton's Branch and bound algorithm, is the algorithm used in the Traffic Management Advisor (TMA), which is part of the Center/TRACON Automation System (CTAS). This algorithm is designed to be used in a dynamic feedback environment. Therefore the optimization criteria can be changed while the simulation is already running (while the algorithm is in use). This algorithm was first used in 1992 at Denver Air Route Traffic Control Center for field evaluation of CTAS [18].

The IE algorithm uses optimization criteria like minimizing fuel use, keeping the delay as small as possible and maximizing the runway throughput to calculate the optimal landing schedule. The algorithm is based on the branch and bound principle [18]. The task performed by this algorithm can be divided in two parts: Sequencing and, after the sequence is determined, scheduling. The overall task of the algorithm is to provide a set of STAs which satisfies all requirements.

To continuously satisfy the separation requirements between all aircraft, the STAs of all aircraft must be updated every time when one of the ETAs changes. However, the ETAs can change as fast as the (sensor) input data changes, so this would trigger many rescheduling operations for minor variations in the ETAs. This results in very fast changing STAs which causes an unnecessary high controller workload. Therefore a trade-off between exact relative spacing at all times and a workable STA update frequency must be made. This trade-off is implemented via scheduling windows (instead of scheduling times) and freeze horizons. If an ETA falls below the freeze horizon, the STA of that aircraft is frozen and cannot be rescheduled anymore.

The scheduling problem can mathematically be represented as a minimization problem which is dependent on the ETA and STA of each aircraft. The cost function is represented in equation (2.10) [19].

This cost function must satisfy a number of requirements. It must be nonnegative and monotonically increasing. The non-negativity is the basis for the dynamic limiting. When an aircraft is added to the departure sequence, the cost will remain equal or increase. The costs will never decrease. Eventually, a bias term can be used to make the cost zero or larger. The cost function must be monotonically increasing to ensure that the earlier an aircraft is scheduled, the lower the costs are. Besides this minimization problem, other constraints can be taken into account, like a fixed sequence order for some of the aircraft.

$$\underset{STA_1,\ldots,STA_n}{MIN}\left(\sum_{i=1}^{n} Cost_i\left(STA_i, ETA_i\right)\right)$$

Such That:

$$STA_i \geq STA_j + ReqTimeSep(type_i, type_j) \tag{2.10}$$

or

$$STA_i \leq STA_j - ReqTimeSep(type_i, type_j) \forall i = 1,\ldots,n; j = 1,\ldots,n$$

$$STA_i \geq ETA_i \forall i = 1,\ldots,n$$

As mentioned before, one of the main challenges for the existing scheduling algorithms is that the computational time required to perform a complete optimal scheduling process is excessive. Reducing this computational time is one of the main goals of the design process of new scheduling algorithms [19]. Roger Dear developed a method to reduce this computational time by taking a subset of all possible sequences in which the aircraft are shifted a maximum number of positions from their original position [20].

The IE algorithm calculates the departure schedule by using this simplification. When a possible departure sequence is established, the schedule is developed as follows: The STA of the first aircraft of the sequence is set at its ETA. Next, the algorithm checks if setting the STA of the following aircraft at its ETA would violate the separation requirements. If not, the STA is set at this ETA, and if it does violate the separation requirements, the STA is set at the ETA of the previous aircraft plus the required separation time. This is done for all aircraft in the sequence.

The IE algorithm is based on the Binary Implicit Enumeration (BIE) algorithm [18]. This algorithm uses a special branch and bound technique for problems with binary input data. The basic idea behind the brand and bound technique is that requirements can be used to limit the search for an optimal solution. This can be used to search only a subset of solutions instead of all possible solutions, without a decrease in efficiency.

The branch used in the IE algorithm is branching out on the next aircraft to be added to the sequence in first to last order. The tree is searched in depth first to obtain a bound on the problem as fast as possible. For this reason, the first branch for each node is the next aircraft in FCFS order if the aircraft is not scheduled yet, or the first aircraft that was scheduled next if the scheduling process already took place before.

If a new aircraft is added to a given branch, the cost to that point can be computed. If this cost is higher than the existing best cost, this branch can be discarded. This is called dynamic limiting. Also depth limiting can be used, but this makes the algorithm less optimal. A third type of limiting which can be used is static limiting. This means that there is a maximum allowable number of shifts for each aircraft.

Besides the delay there are more aspects that can define the cost function. Examples are fuel consumption and runway throughput. A weighted combination of all these criteria is used to optimize over all aspects.

3.1.2 pFAST / aFAST

Research at NASA Ames Research Center on scheduling algorithms and decision support tools resulted in the development of the Center/TRACON Automation System (CTAS). One of the tools of CTAS is the Final Approach Spacing Tool (FAST). This tool helps traffic management controllers manage the arrival of air traffic [21]. There are two types of FAST. The Passive Final Approach Spacing Tool (pFAST) only gives passive advisories. These are landing sequences and runway advisories. The other type is the Active Final Approach Spacing Tool (aFAST). The advisories of this tool not only contain a more precise scheduling of aircraft on final approach [22], but it also provides the possibility to include other limiting factors such as wake vortex separation requirements. The aFAST tool generates heading and speed advisories with an algorithm which simultaneously sequences the aircraft and solves conflicts on their trajectories.

As the name of these algorithms already implies, these algorithms are designed for arrival traffic. However, their goal is to schedule traffic that approaches the runway from different flight paths. Therefore it is interesting to investigate if these algorithms can also be used to approach the runway from the other side, from the taxiways at the airport for departure traffic.

The main difference between pFAST and other optimization algorithms designed before is that pFAST is the first algorithm which uses trajectory-based spatial constraint satisfaction. To implement this, the sequencing problem is divided into multiple small local sequencing problems at intersections of upstream flight segments. By solving these local sequencing problems it is possible to generate a departure schedule at the runway threshold. This process is improved in the aFAST scheduling algorithm.

It is important to realize that the FAST algorithms aren't designed to produce the optimal schedule, but to produce an achievable one. Neumann and Krzeczowski showed that the benefits of optimal sequencing in the terminal area are small compared to the delay benefits of reduced in-trail separation buffers and optimal runway allocation [23, 27].

The pFAST algorithm is performing the sequencing and conflict resolution operations after each other (sequentially) [21]. First, a departure sequence is generated and afterwards the schedule is calculated by solving separation conflicts for each aircraft with the aircraft ahead. A disadvantage of this method is that sequencing decisions are based on the first flight segment that two aircraft share and not on a segment where a potential conflict can occur. Because of this, the total delay of all aircraft can be unnecessary high. To deal with this problem, aFAST uses trajectory-based techniques like first come first served (FCFS) to schedule the aircraft. These techniques can speed up the first airplane to lower the delay of the following aircraft or let a fast aircraft overtake a slower one before a conflict of the slower aircraft with a third one occurs and overtaking is no longer possible.

The scheduling process of the aFAST scheduling algorithm is done as follows: First, the path of an aircraft to its destination is determined. Next, its speed is taken into account to estimate its future position. The horizontal route, the altitude profile and the airspeed profile are now combined to construct a 4D trajectory. This trajectory is represented in a discrete way by a series of positions at a specific interval. This is done for all aircraft. For the aircraft that share at least one segment conflict resolution is needed. Because this conflict resolution or rescheduling operations are performed in parallel, these conflicts can best be represented as a matrix or a network of dependencies instead of a decision tree.

The main difference between the conflict resolution in aFAST and its predecessors is that aFAST immediately performs conflict resolution after each sequencing decision. Its predecessors perform the conflict resolution after all sequencing operations are finished. Because of this, aFAST is able to estimate the ETA of each aircraft in the system immediately. There is also a disadvantage of the method used by aFAST. Because of the parallel concurrent sequencing and conflict resolution procedure, the aFAST algorithm requires more computational power than the pFAST algorithm.

The theory behind the aFAST algorithm is based on making a reasonably first guess and then to improve that guess to achieve a better final scheduling order. This improvement is done in small steps. For example, an aircraft is only allowed to overtake another aircraft if there is enough free space available directly in front of the other aircraft. The first step in the sequencing process is to order the aircraft based on their geometrical position. Next, the algorithm looks for situations where improvements are needed. This is done by pair wise aircraft-to-aircraft analysis starting from the first aircraft to the last one in place.

3.1.3 Modified first come first served

The simple first come first served (FCFS) algorithm determines the aircraft departure sequence based on the order of the ETAs of each aircraft at the runway. The modified FCFS algorithm is also based in this principle, but it is extended with two scheduling horizons. The first horizon is called the initial scheduling horizon. At this point the initial schedule is computed. The second horizon is the final scheduling horizon. When an aircraft passes this second horizon its STA will be frozen and cannot be changed anymore [23].

When an aircraft passes the initial scheduling horizon its STA is computed. If there are no previously scheduled aircraft with a STA after the new aircraft's ETA, the new aircraft will not have any impact on the already scheduled ones, so the STA of the new aircraft will be equal to its ETA, or to the STA of the aircraft before plus the minimum separation time. If the ETA of the new aircraft lies before one or more STAs of already scheduled aircraft, the new aircraft will be inserted between the previously scheduled aircraft at the time of its ETA and at least the minimum required spacing after the aircraft before. The already scheduled aircraft behind the new aircraft will be rescheduled to ensure enough separation.

If there are aircraft with an STA later than the new aircraft's ETA which are already frozen, the algorithm first checks if the separation requirements will not be violated if the new aircraft is inserted. If there are aircraft with already frozen STAs and without enough separation when the new aircraft would be inserted, the new aircraft will be inserted after these aircraft, and the ones after the new aircraft will be rescheduled.

3.1.4 Heuristic time advance

The heuristic time advance algorithm is an algorithm designed to reduce the delay of the departing aircraft. At the start of the scheduling process, the algorithm creates groups of aircraft. These groups are tightly scheduled: the separation between the aircraft is just the minimum required separation. The first aircraft of each group is speed up to depart earlier than its original ETA. All other aircraft in the group are speed up with the same amount to reduce the delays of the delayed aircraft in the group. Because speeding up an aircraft is costly, this is only done when the aircraft immediately following the first aircraft in a group has a delay which can be reduced. The maximum allowed time advance is one minute. [23]

3.1.5 Fuel saving time advance

The fuel saving time advance algorithm is an improved version of the heuristic time advance algorithm. Just like the heuristic algorithm, it tries to reduce delays. To do this, first all aircraft in a group are speed up by an equal amount regardless of the future benefit. However, speeding up is costly, so after this, all aircraft that do not benefit from a time advance, will have their time advances removed or reduced [23].

3.1.6 Idealized fuel saving time advance

The idealized fuel saving time advance algorithm makes use of the ideal situation where the complete traffic sample is available on beforehand. Therefore this algorithm cannot be used in practice, where only the traffic within the scheduling window is known. Nevertheless, this algorithm is explained here because it is the basis of the realizable fuel saving time algorithm, which is discussed next.

The idealized fuel saving time advance algorithm optimizes the departure schedule bij time-advancing the aircraft. To determine how much time advance (TA) must be used, the cost of this time-advancing must be modelled. At first instance, the extra fuel use in case of a time advanced aircraft was only modelled as cost, but there are also fuel savings due to time reduction, so modelling the extra fuel use only as a cost proved to be not realistic. Table 3.2 shows examples of fuel expenditures and times for arriving flights [23] for a specific large aircraft as calculated by the descent advisor 250 miles from touchdown.

Table 3.2: Cost of time advance and gain for reduced delay. Maximize TA in cruise and descent [23]

Initial altitude (ft)	Initial speed (mach)	w_f (lb/min)	t_n (min)	f_n (lb)	t_f (min)	f_f (lb)	t_n-t_f (min)	f_n-f_f (lb)	$\frac{f_n - f_f}{t_n - t_f}$ (lb/min)
27,000	0.65	114.8	39.6	3,793	33.8	4,071	5.75	278	48
27,000	0.68	120.7	38.0	3,759	34.4	4,063	3.63	304	84
27,000	0.72	129.0	36.6	3,793	34.3	4,034	2.30	241	104
32,000	0.72	116.8	36.5	3,358	34.2	3,606	2.26	248	109
32,000	0.76	124.9	35.8	3,400	34.1	3,575	1.71	176	102
32,000	0.80	135.3	34.6	3,469	34.1	3,542	1.61	74	121
37,000	0.75	119.7	36.6	3,202	35.2	3,244	1.31	42	32
37,000	0.79	126.3	35.3	3,185	35.1	3,210	0.28	25	89

w_f = Fuel flow rate at initial altitude and speed

t_n = Time for nominal speed profile

f_n = Total fuel for nominal speed profile

t_f = Time for fast speed profile

f_f = Total fuel for fast speed profile

As seen in Table 3.2, slower aircraft have a higher TA capability and for higher entry speeds, the fuel costs for extra TA increases. When the real fuel costs would be taken into account for each aircraft, heavy aircraft would always be sequenced first because the fuel use of these aircraft is higher. Time advancing this aircraft would also reduce the fuel costs with a big amount. To prevent this aircraft to get a preferential treatment, the same performance function is used for all aircraft types.

Another reason for not minimising the overall fuel consumption of all aircraft is that the desired ETA is not the point of minimum fuel use of an aircraft. The desired ETA is chosen for several reasons, not only the fuel cost, but also time costs and the aim to achieve maximum time control without having to leave the desired (time)path. Therefore, a cost model is used which is equal for all aircraft and takes into account that it is desirable to maintain the nominal ETA as much as possible.

For cost estimation, the following steps are performed:

- Calculate the STA for each aircraft without TA using a FCFS method
- Obtain the time advance STAs by subtracting the desired t_a from each STA. This time shift is equal for all aircraft, so the sequence and spacing of the aircraft doesn't change by this operation.

$$STA\_TA_j = STA_j - t_a \tag{2.11}$$

- Calculate the delay after TA

$$Delay_j = STA\_TA_j - ETA_j \tag{2.12}$$

- When this delay is positive, the aircraft is still delayed, but because of the TA, this delay is decreased and thus the costs are decreased. The incremental costs are:

$$\Delta cost = -t_a \tag{2.13}$$

The reduced delay saves fuel and time, so these incremental costs are reasonable. When the delay after TA is negative, the aircraft is planned to arrive before its ETA.

This algorithm is designed to optimize arriving air traffic. However, all conditions, like more fuel use at a higher (taxi) speed, also hold for departure traffic. Therefore this algorithm can also be taken into account for scheduling the departure traffic.

3.1.7 Pure time advance

The pure time advance algorithm is a simplified version of the fuel saving time advance algorithm. This algorithm is reducing all STAs by the same amount of time without removing unnecessary time advances. In case of large demands, the optimum curve for time advance versus relative costs for pure time advance is almost equal to the optimum time advance situation. The reason is that in case of large demands, there are very few unnecessary time advances which can possibly be removed. The small increase in average cost caused by not removing these unnecessary time advances compensates for the large increase in costs when a time advance is removed first, but is finally found to be needed and must be placed back again [23].

3.1.8 Realizable fuel saving time advance

The realizable fuel saving time advance algorithm is based on the idealized fuel saving time advance algorithm, but, in contrary to the idealized algorithm, the realizable algorithm uses a finite time (scheduling) window. Therefore the complete traffic set does not have to be known in advance.

When the ETA of an aircraft falls within the scheduling window, the aircraft is scheduled based on FCFS and time advanced to achieve the average minimum cost. Next, all other aircraft are checked for a possible reduction of time advance. If there is a spacing window between two aircraft that is greater than the minimum required spacing, the time advance of the first of these two aircraft is removed to reduce this spacing window. The possible loss in efficiency compared to the idealized algorithm comes from the fact that aircraft with an already frozen STA could also have benefitted from the reduction of time advance, which is impossible now, because the STA is frozen.

The time by which all aircraft are advanced is the most important parameter of this algorithm. When a group of aircraft is leaving with minimal separation, the aircraft cannot be time advanced until a larger gap between two aircraft occurs. Even with an incorrect TA fuel is saved, so therefore it is recommended to apply a larger TA than currently needed.

The TA is dependent on both demand and acceptance rate. The optimal TA is dependent on the demand according to the schedule in Table 3.3 [23]. A TA of 1 stands for the maximum allowable TA.

Table 3.3: t_a for different demands

Demand	t_a
demand ≤ 20	$t_a = 0$
20 < demand ≤ 30	$t_a = 0.35+0.0325$ (demand - 20), remove unnecessary t_a
30 < demand ≤ 40	$t_a = 0.5 + 0.04$ (demand - 30), use pure t_a
40 < demand	$t_a = 1.0$

3.1.9 Optimal constrained position shift

The optimization method used in the constrained position shift (CPS) algorithm makes use of the different spacing requirements for different aircraft classes, as shown in Table 2.1. The optimal constrained position shift algorithm reorders the existing FCFS sequence by shifting the aircraft no more than a single position. It is required that the aircraft depart from different positions, so no overtakes are needed, but reordering can take place via speeding up or slowing down aircraft. In theory the reordering process is most effective when large groups of aircraft are used.

Due to the way the algorithm is written, the used TA can only be checked after the position switches are already proposed. This and the non-optimal methods of removing the violations on this rule make the algorithm non-optimal. Computational constraints on the algorithm and physical constraints of the aircraft involved make the use of shifts of more than 1 position not practical and therefore time shifts of more than 1 minute are not used. [24]

3.1.10 Heuristic constrained position shift

The heuristic CPS is a more realistic version of the optimal constrained position shift algorithm. The optimal constrained position shift algorithm uses the entire set of aircraft for determining the optimal sequence. In real life, this is not possible. Due to many traffic or freeze horizons a scheduling window of finite size has to be used. This finite scheduling window is used by the heuristic constrained position shift algorithm. If the group size is big enough (six or seven aircraft), the heavy aircraft in the group can be grouped in two subgroups. There is a fixed number of patterns (sequences) which the algorithm tries to achieve. These patterns can be acquired in one or two shifts, depending on the pattern and the group size.

When searching for the optimal algorithm, the first aircraft is never involved in a switch. Therefore this aircraft can be across the freezing horizon. When the sequencing pattern to be achieved is determined, the time advance of all aircraft is checked. If there is an aircraft with a TA of more than 1 minute, the old sequence is restored. [23]

3.1.11 Greedy search

The greedy search algorithm can be compared to a tree search algorithm. The sequence generated by this algorithm starts with the aircraft already at the runway. This is the first aircraft to take off and thus the first in sequence and the first node of the search tree. The other aircraft ready to take off are queued up in (virtual) rows. These rows represent the different paths to the runway. There must be at least two paths, otherwise there is only one possible sequence: The sequence the aircraft are already in. When the first aircraft is picked, the next one will be selected from all aircraft in front of the queues. The aircraft is selected by checking the required separation time between the last aircraft already in the final sequence and all candidates. The aircraft with the least required separation time is chosen as the next one to depart. This process will continue until all aircraft are scheduled.

When there are more aircraft with equal (shortest) separation time, a decision must be made based on other criteria. First, the algorithm looks one step further. It does not only look one step forward, but also takes the next step into account. The total cost of both scheduled aircraft is computed and the option with the least cost is chosen.

If there is still no unique solution, additional rules are used. The first rule searches for sequences with optimal heavy-large sequences. The next rule chooses the aircraft which is in the largest queue and the last rule chooses the aircraft which has the longest waiting time [8].

It can be concluded that the greedy algorithm tries to find the local optimum at each step. This makes the algorithm easy to implement but has also disadvantages. Because of the local search procedure, this algorithm does not always find the global optimum.

3.1.12 Genetic

A genetic algorithm calculates the best sequence via an iterative process. This process is based on the principle of ‘survival of the fittest’. A population of possible sequences is selected, and only the fittest will survive and reach the next step.

The algorithm starts with a number of possible sequences. This can be any number, but there should be at least two possibilities. The algorithm calculates the efficiency (fitness) of all sequences. Next, the least efficient sequences are discarded. After this, the next generation of sequences is created by interchanging some aircraft in each sequence. In this way, multiple new sequences are created from every 'old' sequence. Now, the least efficient sequences are discarded again, etc. The optimization process can be terminated in different ways. Mostly it is done after a fixed number of iterations. Another option which is used often is checking the improvement of each new generation. When the improvement is less than a certain threshold, the iteration is stopped. [25]

Due to the way the algorithm is written, it starts with a number of possible solutions, and these solutions converge to more and more optimal solutions, until the most optimal solution is found. In contrary to most other algorithms, there is always a complete solution available. This is the biggest advantage of this algorithm. Other algorithms often start with a small part of the final sequence, and let this sequence grow until all aircraft are scheduled. The genetic algorithm starts with a complete solution, and makes this more optimal during the scheduling process. So when it is needed to make the solution available earlier, this is always possible, but this solution is not as optimal as the final solution the algorithm would have produced.

Unfortunately, the genetic algorithm also has an important disadvantage. The algorithm tries to make the schedule more efficient every generation, so the algorithm converges to the optimal solution. However, this optimal solution is not always the global optimum. It is also possible that the algorithm converges to a local optimum [26]. Also, because of the random change of elements in the departure queue, the improvement per iteration step is unpredictable. It is impossible to predict how many iteration steps are needed to reach a certain efficiency level.

3.2 Usability of discussed algorithms

In section 3.1, a number of different scheduling algorithms are discussed. This is not a complete set of all scheduling algorithm suitable for this situation, but a selection of the algorithms most suitable for this task. A requirement for the algorithms to be used for this runway scheduling, is whether they are able to compute a runway schedule with the inputs which are available. These inputs are the aircraft characteristics like type, speed, location and the desired takeoff time. If an algorithm needs more input data which is not available, it cannot be used in this situation.

The output of the algorithm is also bound to restrictions. The algorithm must be able to perform the complete scheduling process. It must provide a complete departure schedule and not only a part of the process, like only a departure sequence, instead of a schedule, or a schedule for only a specific part of the aircraft. A check to what extend the algorithms comply with these requirements shown in Table 3.4. A summation of other characteristics of the algorithms is also given in this table.

3.3 Algorithm categories

Looking at the algorithms from section 3.1, it can be seen that some of the algorithms have many things in common. The algorithms do not differ completely, but are based on a limited number of basic principles. Looking at these basic principles, the algorithms can be divided into four main categories. These categories are the basic principles the algorithms are based on. The differences between the algorithms are aspects to improve the final result. The four algorithm categories are discussed in the sections below.

3.3.1 First come first served

A first come first served (FCFS) algorithm tries to schedule the algorithms in the order in which they announce themselves at the air traffic control as ready to depart. The algorithm attempts to schedule the aircraft at their ideal takeoff time. Before a takeoff time is assigned to the aircraft, the algorithm checks whether the separation requirements are not violated when the aircraft is scheduled at that time. Therefore the separations with the aircraft leading and following are checked. If no separation rules are violated, the aircraft is scheduled at its ideal takeoff time. Else, the aircraft is scheduled at the first next possibility where no separation rules are violated. A schematic representation of this algorithm category is shown in Figure 3.1.

The biggest advantage of this algorithm is that it is relatively simple. Therefore it requires little computational power compared to the other algorithms. A disadvantage of this easiness is that the optimization procedure is only one-sided. The algorithm only tries to schedule the aircraft at their desired takeoff time. This can reduce the delay of the aircraft currently scheduled, but can block other aircraft and therefore increase the total delay of all aircraft.

Table 3.4: Comparison of the scheduling algorithms

	Implicit enumeration	pFAST / aFAST	Modified first come first served	Heuristic time advance	Fuel saving time advance	Idealized fuel saving time advance
Does algorithm provide complete solution	Yes, sequencing and scheduling	pFast: No, only sequencing aFast: Yes, heading and Speed	Yes, sequencing and scheduling	No, only scheduling	No, only scheduling	No, only scheduling
Able to perform with provided inputs	Yes, set of ETAs	Yes, ETA and speed profile	Yes, ETA (FCFS)	Yes, ETA	Yes, ETA	Yes, ETA
Able to provide required outputs	Under conditions, set of STAs (no speed profile)	Yes, 4D trajectory	Under conditions, STA (no speed profile)	Under conditions, STA (no speed profile)	Under conditions, STA (no speed profile)	Under conditions, STA (no speed profile)
Able to deal with maximum runway throughput	Max number of shifts, so suboptimal	Profivides achievable plan, not optimal plan	No, schedule based on FCFS	No, no resequencing performed	No, no resequencing performed	No, no resequencing performed
How does algorithm place aircrafts in quiet periods	Aircraft scheduled on ETA	Aircraft scheduled on ETA	Aircraft scheduled on ETA	Aircrafts scheduled in small tight groups	Aircrafts scheduled in small tight groups	Aircrafts scheduled in small tight groups
Reaction on disturbance on one of the aircrafts	Rescheduling, and following aircrafts are updated (except when over freeze horizon)	Rescheduling and deconfliction	Update of STA, no rescheduling	Aircrafts behind delayed with same amount	Aircrafts behind delayed with same amount	Aircrafts behind delayed with same amount

	Pure time advance	Realizable fuel saving time advance	Optimal constrained position shift	Heuristic constrained position shift	Greedy search	Genetic
Does algorithm provide complete solution	No, only scheduling	No, only scheduling	Yes, sequencing and scheduling	Yes, sequencing and scheduling	Yes, sequencing and scheduling	Yes, sequencing and scheduling
Able to perform with provided inputs	Yes, ETA	Yes, ETA	Yes, ETA and aircraft class (speed profile)	Yes, ETA and aircraft class (speed profile)	Yes, ETA, aircraft class and path number	Yes, ETA and aircraft class
Able to provide required outputs	Under conditions, STA (no speed profile)	Under conditions, STA (no speed profile)	Yes, STA and speed profile	Yes, STA and speed profile	Under conditions, STA (no speed profile)	Under conditions, STA (no speed profile)
Able to deal with maximum runway throughput	No, no resequencing performed	No, no resequencing performed	No, max position shift 1 position	No, max position shift 1 position	Does not always converge to optimal plan	Does not always converge to optimal plan
How does algorithm place aircrafts in quiet periods	Aircrafts scheduled in small tight groups	Aircrafts scheduled in small tight groups	Aircrafts scheduled on STA	Aircrafts scheduled on STA	Aircrafts scheduled on STA	Aircrafts scheduled on STA
Reaction on disturbance on one of the aircrafts	Aircrafts behind delayed with same amount	Aircrafts behind delayed with same amount	Aircrafts behind delayed with same amount	Aircrafts behind delayed with same amount	Aircrafts behind delayed with same amount	Rescheduling

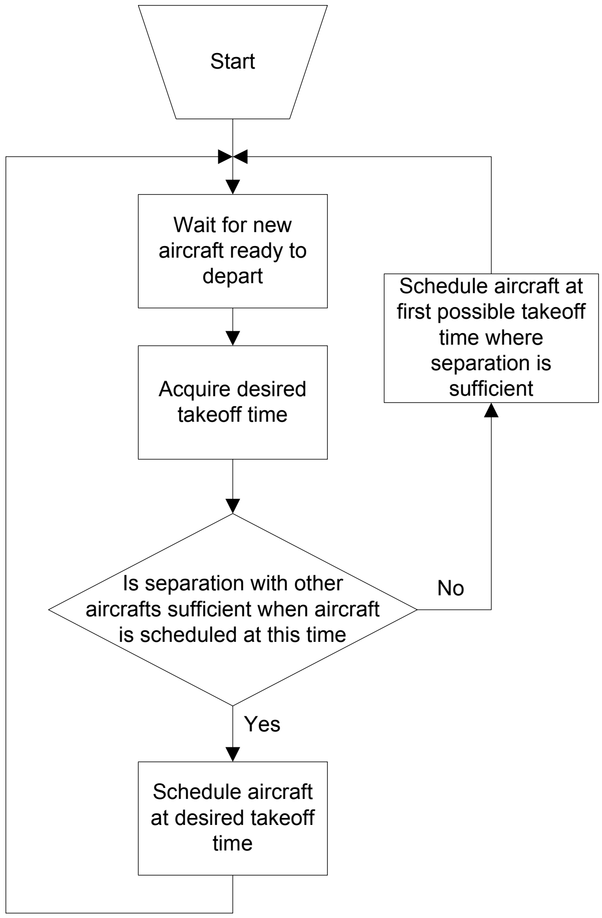

Figure 3.1: Flow diagram of a first come first served algorithm

3.3.2 Branch and bound

A branch and bound (bnb) algorithm is based on the tree search principle. When a new aircraft announces itself at the air traffic control as ready to depart, all possible departure sequences are calculated. All these sequences are put in a decision tree. Next, the optimal solution is found by searching the decision tree. The optimal solution is not found by performing a depth search, but via a breadth search. First the efficiency of all nodes which are located one step away from the origin is calculated, next the efficiency of all nodes two steps from the origin etc.

When the efficiency of all nodes at a certain step is calculated, all paths which contain the N least efficient nodes are discarded. This process continues until one path is left. The last step in the scheduling process is to convert this departure sequence into a departure schedule. This is done by assigning takeoff times to the aircraft in the order of which they are sequenced. The algorithm tries to schedule the aircraft at their desired time, when this does not violate the calculated departure sequence and the separation requirements. A flow diagram of this type of algorithms is shown in Figure 3.2. For the efficiency mentioned in this figure, the difference between the ideal takeoff times and the real takeoff times is used during this research. However, also other criteria for the efficiency can be used.

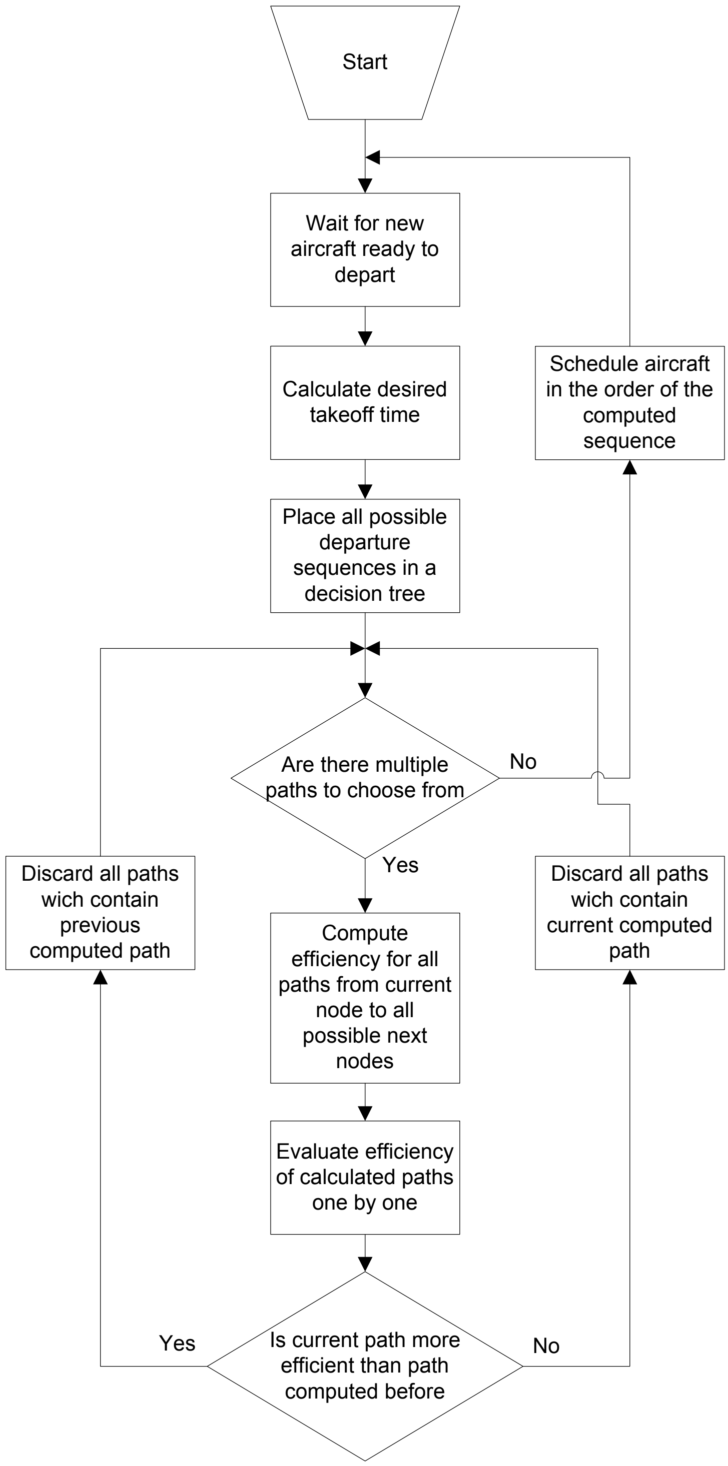

Figure 3.2: Flow diagram of a branch and bound algorithm

The main advantage of the branch and bound algorithm is the ability to handle big sets of aircraft. The optimal departure sequence is calculated node by node, and therefore the performance of the algorithm is not influenced by the size of the aircraft set. However, this aspect is also the main disadvantage of this algorithm. By calculating the optimal departure sequence, the algorithm only looks one node forward. The algorithm calculates a local optimum. This can result in a solution that is located far away from the global optimum.

3.3.3 Greedy search

A greedy search algorithm tries to achieve the global optimal solution by choosing the local optimum at each step. Each step number of possible options is presented and the algorithm chooses the optimal option.

The algorithm can be divided in two parts. The first part assigns the aircraft which are ready to depart to one of the queues. This part is shown on the left of Figure 3.3. The allocation of the aircraft to the queues can be done in multiple ways. The allocation can be based on the location on the airport where the aircraft is coming from. In this case all aircraft arriving from a specific direction are placed in a specific queue. Another option is to allocate the aircraft to a specific queue based on the aircraft type. This makes it easier for the algorithm to create an optimal runway departure schedule based on optimal heavy/large sequences. Other possible options are an allocation based on the number of aircraft already in the queue and an allocation based on the expected waiting time of aircraft in the queue.

The second part of the algorithm is the part where the actual scheduling is done. The algorithm chooses one of the aircraft at the front of the queues to depart next. The decision of which aircraft is chosen is based on trying to achieve a local optimum. If there is more than one possibility with which the local optimum can be achieved, other things are taken into account. First the algorithm looks at the optimal solution of the next aircraft. If there are more aircraft with the optimal solution, all combinations of the first two aircraft which are able to depart next are taken into account. In case that there are still multiple optimal solutions, the algorithm searches for optimal heavy/large sequences and if there are still multiple options, the aircraft with the highest waiting time is selected to depart (Figure 3.3).

The main advantage of the greedy search algorithm is that the complexity is independent on the number of aircraft involved. The complexity of this algorithm is only dependent on the number of queues which are used to place the aircraft in. The drawback of this fact is that the choice of how to divide the aircraft among the queues is an important factor defining the efficiency of the final departure schedule.

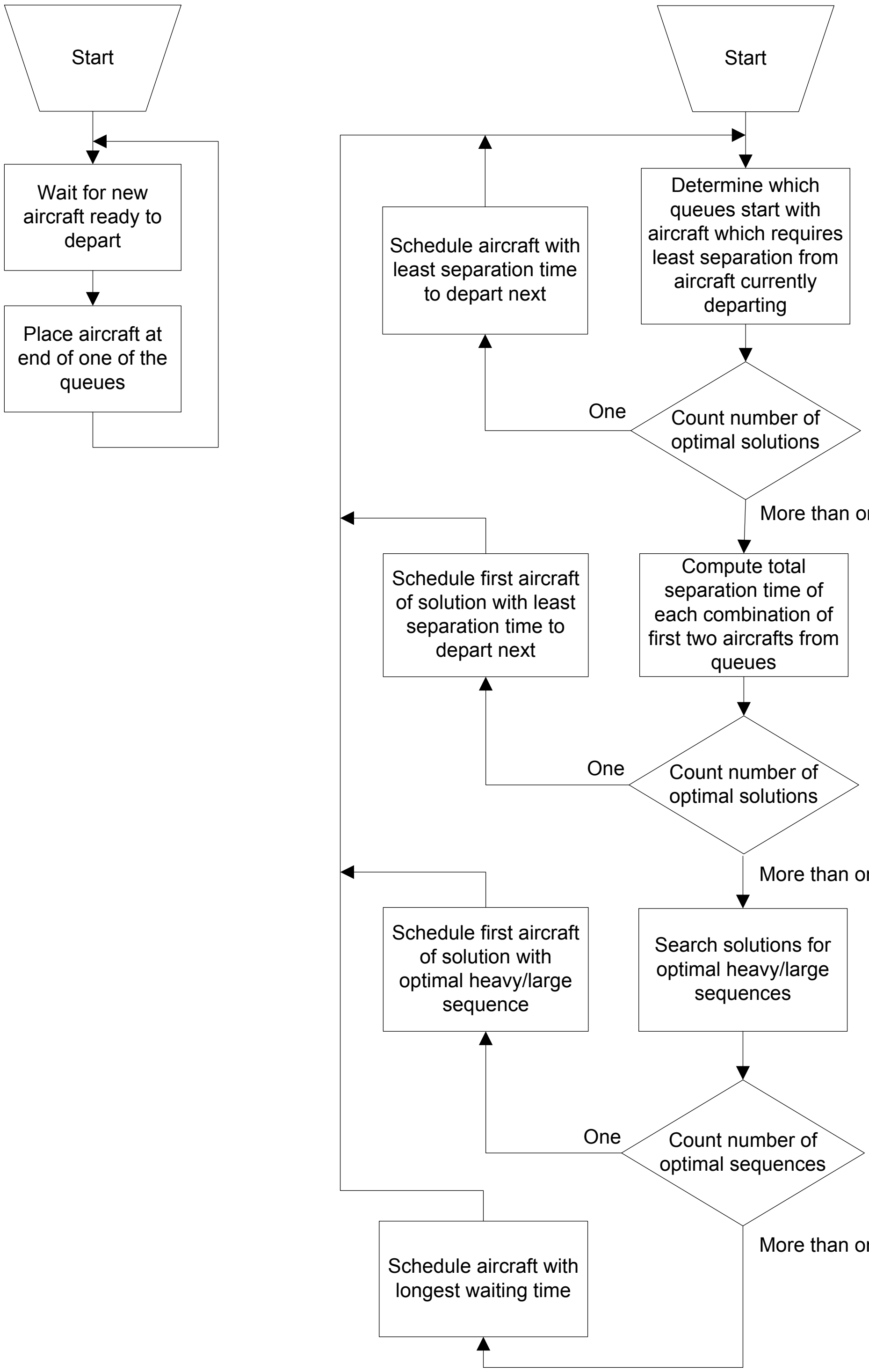

Figure 3.3: Flow diagram of a greedy search algorithm

3.3.4 Genetic

A genetic algorithm uses an iterative process to find the optimal departure schedule. When an aircraft announces itself as ready to depart, it is added to the (imaginary) departure queue. This is not a real queue, but a set of aircraft which are able to depart. This queue has no fixed order, but the order of the aircraft can change. When an aircraft enters this queue, the algorithm generates N possible departure sequences. This is not a complete set of all possible sequences, but a part of it. From all of these sequences, the efficiency is calculated. Next, the X least efficient sequences are discarded.

When no next iteration step is taken, the most efficient sequence is now taken as the departure sequence. When another optimization step is needed, the queues left are copied a number of times and M elements are interchanged in each copied sequence, so a new set of sequences is created. From these new sequences the efficiency is calculated. The original sequence is also taken into account, so during each iteration step, the efficiency improves, or stays equal, but will never deteriorate. (Figure 3.4) For the efficiency mentioned in Figure 3.4, during this research the difference between the ideal takeoff times and the real takeoff times is used. However, also other criteria for the efficiency can be used with this algorithm.

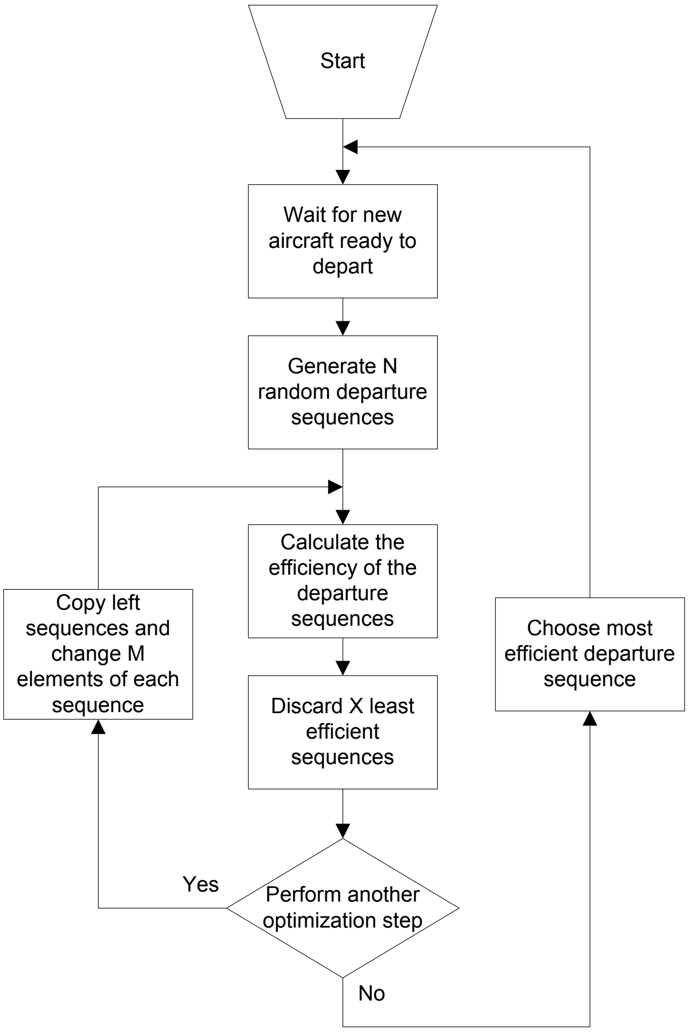

Figure 3.4: Flow diagram of a genetic algorithm

As already mentioned in section 3.1.12, the biggest advantage of a genetic algorithm is that there is always an optimal solution available. The genetic algorithm starts with a complete solution, and makes this more optimal during the scheduling process. When there is need to make the solution available earlier, this is always possible, but this solution is not as optimal as the final solution the algorithm would have produced.

Unfortunately, the genetic algorithm also has disadvantages. Because of the random change of elements in the departure queue, the improvement per iteration step is unpredictable. It is impossible to predict how many iteration steps are needed to reach a certain efficiency level.

3.4 Summary

All algorithms discussed in section 3.1 can be assigned to one of the categories discussed in section 3.3. The differences between the algorithms within a category are minor variations to improve the efficiency. An overview of the four categories and their differences is shown in Table 3.5. This table also shows how the algorithms are divided among these four categories. To get an overview on the performance of all discussed algorithms, this research continues with using these four categories. These global algorithms are used to test the system designed.

Table 3.5: Overview of the algorithm categories

	First come first served	Branch and bound	Greedy search	Genetic
Basic principle	Schedule aircraft in the order they are ready to depart	Put all sequences in a decision tree and perform a breadth-first search	Assign aircraft to queues, and choose optimal aircraft from front of queues	Generate sequence, change elements and keep most optimal sequence
Main advantage	Easy, requires few computational power	Complexity not influenced by size of aircraft set	Complexity only defined by number of queues	Complete solution available at any moment in time
Main disadvantage	Does not look at side effects. Optimizing current aircraft can delay integral schedule	Local optimum calculated can drift away from global optimum	Choice for how to assign aircraft to queues has high impact on efficiency	Improvement per iteration step is uncertain
Algorithms based on this principle	pFAST / aFAST Modified first come first served	Implicit enumeration Scheduling Optimal constrained position shift Heuristic constrained position shift	Heuristic time advance Fuel saving time advance Idealized fuel saving time advance Pure time advance Realizable fuel saving time advance Greedy search	Genetic

4. System overview and implementation

In this chapter the implementation of the simulation environment for the evaluation of traffic scheduling algorithms will be discussed. First the design of the simulation environment is discussed in paragraph 4.2. All inputs and outputs are explained in paragraph 4.3 and a functional overview of the simulation environment is discussed in paragraph 0.

The input variables of the design are subject to design constraints. These design constraints specify the format of the input variables and are discussed in paragraph 4.5. Together with these constraints, a solution is given to deal with this. This chapter is concluded with the future applications of the design. The considerations and actions taken into account to make the design future-proof are discussed in paragraph 4.6.

In chapter 2 the objective of this research is discussed. Within several years, 4DT air traffic control will be introduced. Controlling the air traffic in a plan-based approach instead of a state-based approach enables the opportunity to make the air traffic flows more efficient. A part of the complete air traffic system that might benefit a lot from 4DT air traffic control is the departure scheduling on airports. The existing scheduling algorithms each use their own strategy to optimize the traffic flow. These scheduling algorithms can be divided into four categories, as described in chapter 3. The goal of this research is to design a simulation environment to be able to evaluate all these scheduling algorithms.

4.1 Requirements

In chapter 2 the requirements which the system to be designed has to meet are defined. The first requirement is that it must be possible to change the possibilities of the simulation without having to redesign the complete system. This requirement can be met by building the system in a modular way. The complete simulation is not build as one integral system, but it is built up from separate modules. Each module has a specific task and together they form the complete system. This modularity makes it easy to make modifications to the system. The modularity is discussed in more detail in paragraph 4.2.

The subsystem for which it is most important to be able to be modified is the scheduling algorithm itself. Therefore this algorithm is implemented as a separate subsystem. When (re)scheduling is needed during the simulation, the system passes all necessary data to the scheduling subsystem. This subsystem calculates a new schedule which is handed back to the main system. After that, the simulation continues until a new scheduling operation is needed.

The inputs of the system are also subject to requirements. The input parameters to be used are the traffic situation, a map of the airport, the scheduling algorithm to be evaluated, the rate at which the simulation data is refreshed and aircraft and ground based speed constraints. The requirements for these inputs and why these inputs are essential for a reliable simulation are discussed in section 4.2.1.

To make the simulation more realistic, it is inevitable to take disturbances into account. To include these disturbances, a separate module is designed. The input and output variables of this module are discussed in section 4.3.6.1. The module itself is explained in 4.3.6. These disturbances must be included in the simulation to be able to investigate the effect of these disturbances on the efficiency of the scheduling algorithm. The effect of the disturbances can be minimized by making the schedule robust against disturbances, or by performing a rescheduling operation to deal with these disturbances. Rescheduling is an important part of the system. This part is discussed in section 4.3.2.

The outputs can be considered as the most important part of the system. The evaluation of traffic scheduling algorithms is the purpose for which this system is designed. The outputs of the system determine if the system fits the goal for which it is designed. Therefore six outputs are defined. The robustness, total delay, average delay, runway throughput and time and speed windows are presented to the user. These outputs are discussed in section 4.2.1.

The last requirement cannot be assigned to a specific part of the system. This requirement states that the system should be faster than reality. This cannot be satisfied by a single subsystem, but is a requirement that deals with the complete system. This requirement should be met via careful efficient programming and not creating too much overhead in the simulation.

All the requirements discussed before can be summarized as follows:

1. Easy to modify to change possibilities of simulation
2. Adjustable input parameters
    a. Traffic situation
    b. Airport map
    c. Scheduling algorithm
    d. Update rate
    e. Aircraft based speed constraints
    f. Ground based speed constraints
3. Take disturbances into account
4. Take rescheduling operations into account
5. Provide multiple output parameters to do evaluation of algorithm
    a. Robustness
    b. Total delay
    c. Average delay
    d. Runway throughput
    e. Time windows
    f. Speed windows
6. Simulation should be faster than reality

4.2 System design

Designing a simulation to evaluate traffic scheduling algorithms is a complex process. The simulation environment to be designed can be represented as the system shown in Figure 4.1.

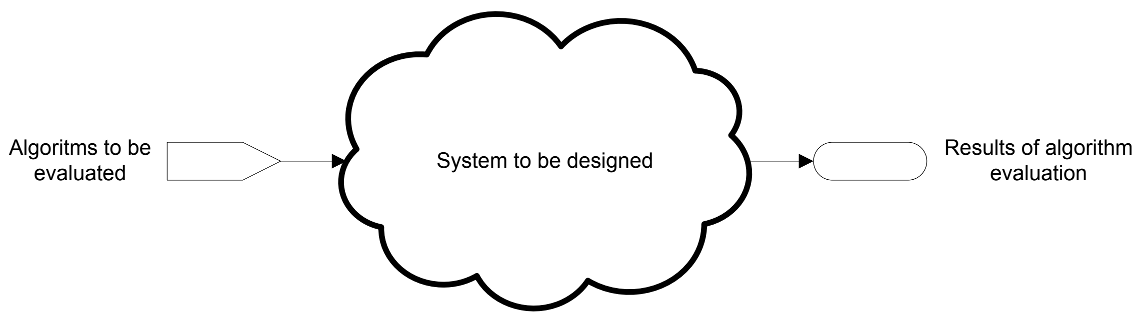

Figure 4.1: Functional overview of the system

The performance of a scheduling algorithm can depend on the type of situation it is used for. Algorithms that perform very well in one situation can, for example, perform much worse in another situation. To evaluate the algorithms presented to the system, it should be possible to vary the situation fed into the simulation, so more input data is needed than the algorithms only. The output of the system will also be more complex. There is no uniform output parameter to represent the evaluation of an algorithm.

As mentioned in chapter 2, the solution to this problem is to let the user perform the evaluation of the algorithms instead of letting the system evaluate the algorithms. The system does not perform the evaluation procedure, but provides information to the user to do the evaluation. As a result of this, the output of the system consists of multiple output parameters. This makes the functional overview of the system looks like Figure 4.2.

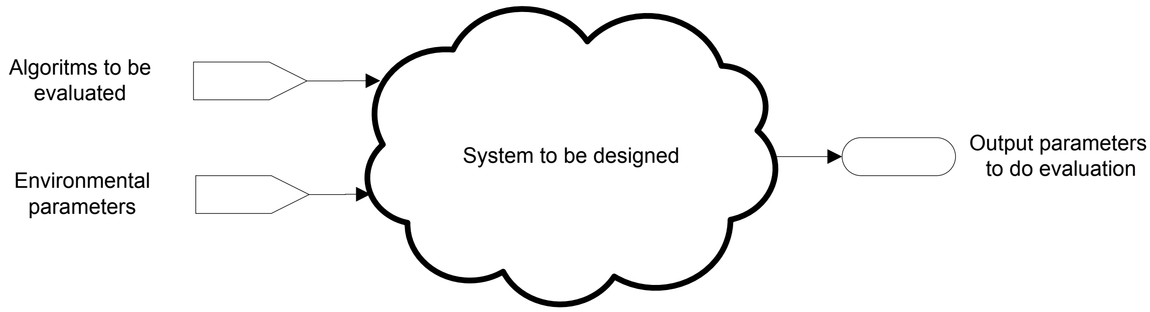

Figure 4.2: In reality multiple inputs and outputs are needed

4.2.1 Input and output parameters

Before expanding the functional design discussed in paragraph 4.2 to a fully functioning system, the input and output parameters need to be defined. This is an important aspect of the design phase, because these parameters define whether the system is applicable in specific situations. Also a consideration between flexibility (many parameters) and practical usability (less parameters) must be made.

The input parameters of the complete system can be divided into two types: fixed and variable parameters. Fixed parameters are parameters which cannot change during the simulation. These parameters include physical aspects of the aircraft like the class of an aircraft, the separation requirements, and the length and type of taxiways on the airport. Other parameters are variable and can still change during the simulation. These are parameters like the desired takeoff time of an aircraft and the time when an aircraft is ready for departure.

These input parameters allow the user to change the simulation environment to adapt the simulation to all kind of situations. The airport data can for example be modeled to represent an existing airport to simulate the effectiveness of an algorithm when it is used on that specific airport. By changing the aircraft data, different traffic situations can be simulated to test the effectiveness of an algorithm in specific traffic situations. These parameters also make it possible to copy a specific situation from reality, to check whether an (other) algorithm would have been more or less effective.

4.2.1.1 Simulation settings

The fixed and variable parameters can be divided into three categories. The first category is the simulation settings. These settings have nothing to do with the physical aspects of the airport or the aircraft involved in the simulation, but define the simulation conditions. The parameter *simtime* defines for how long the simulation will run. The simulation is built in a discrete way. The inputs, like the position of the aircraft will not be continuously updated. An update rate is used for which the position of the aircraft will be updated. The size of the time step used for this update rate is defined by the *step* parameter. In reality, not all aircraft that will ever depart from a specific airport are known in advance. If they would be known, the amount of aircraft would be so big that it becomes impossible to involve all aircraft in the calculation of the departure schedule. Therefore a scheduling window is used which defines for how far in the future the aircraft are involved in the calculation of the departure schedule. The size of this scheduling window is defined by the *schedwin* parameter. The *algorithm* parameter and the *disturbances* parameter define the algorithm and the disturbances used.

Airport data

The next parameter category is the airport data. The parameters *mainpath* and *path* define the position and length of the taxiways on the airport. All taxiways are modeled as straight lines which are connected to the main taxiway. The main taxiways will lead to the runway.

Aircraft data

The last parameter category in Table 4.1 is the aircraft data. The parameter *slotboundaries* is a margin which defines the departure slot of the aircraft. When a departure slot is assigned to an aircraft, this aircraft has to depart within the slot boundaries belonging to this departure slot. The *aircraft* parameter contains data about the individual aircraft like the desired takeoff time, the location from which the aircraft starts taxiing and the aircraft class. This class defines the separation requirements and taxi speed limitations. The parameter *abspeed* defines the aircraft individual speed limitations. These speed limitations are aircraft specific and can differ from the speed limitations the algorithm expects. *Rttaxi* defines when an aircraft is expected to be ready to start taxiing. This is an expected value. Disturbances can cause the aircraft to be delayed and not be able to really start taxiing at this time. The parameter *class* defines the overall speed limitations per aircraft class. The separation requirements are described by the parameters *separation* and *mitseparation*. An overview of all input parameters is shown in Table 4.1.

Table 4.1: Input parameters of the complete system

Category	Parameter	Description
Simulation settings	simtime	Simulation duration
	step	Simulation time step
	schedwin	Size of scheduling window
	algorithm	Scheduling algorithm used in the simulation
	disturbances	Disturbances which will occur during simulation - Disturbances in real taxi speed - Disturbances in time when aircraft start taxiing
Airport data	mainpath	Length of the main taxiways
	path	Position and length of other taxiways
Aircraft data	slotboundaries	Boundaries of aircraft's takeoff slot at runway
	aircraft	Aircraft data: - Desired takeoff time - Starting location at airport - Aircraft class
	abspeed	Aircraft based speed limitations
	rttaxi	Time when aircraft is ready to start taxiing
	class	Speed limitations per aircraft class
	separation	Required wake vortex separation per aircraft class
	mitseparation	Required miles-in-trail separation per aircraft class

4.2.1.2 Output parameters

As mentioned before, the output of the system also consists of multiple parameters. These parameters allow the user to evaluate the scheduling algorithm. Depending on the users' application of the algorithm, a specific parameter is more or less important as a rating criterion to evaluate the algorithm. The output parameters can also be divided into categories.

Aircraft data

The first category is the aircraft data. The parameters *timewin* and *speedwin* represent the time and speed windows of the aircraft. These windows indicate the margin in time and speed for the aircraft. If the aircraft stay within these margins they are able to arrive at the runway at the scheduled time. If an aircraft gets out of its window it cannot reach the runway in time anymore and rescheduling is needed. The cause for the aircraft to be unable to reach the runway in time anymore can be a conflict caused by another aircraft, or a violation of the speed limitations which would be needed to reach the runway in time. A smaller window means less margin and thus rescheduling is likely to occur more often.

The delay of an aircraft with respect to its desired takeoff time is stored in the parameter *delay*. During this research the delay is handled as a single parameter containing the total delay of an aircraft. The system is designed in such a way that it is possible to split this delay into a separate delay for each phase, such as gate delay, taxi delay and runway delay. The takeoff time scheduled by the algorithm can differ from the desired takeoff time of the aircraft, so even if all aircraft take off at their scheduled time, the *delay* parameter does not need to be zero. If this delay is bigger than the boundaries of the aircraft's takeoff slot at the runway, the aircraft is not able to depart within its assigned slot. The aircraft will not be allowed to enter the airspace and thus the aircraft will have to request a new departure slot. The aircraft will be dropped from the schedule. The amount of dropped aircraft is stored in the *dropouts* parameter. If a drop is only caused by the schedule generated by the algorithm, this is a severe disruption of the traffic. The final takeoff time of the aircraft is stored in the *arrival* parameter.

Airport data

The other category of output parameters is the airport data. The number of rescheduling operations is stored in the *numreschedule* parameter. This is an indication for the robustness of the schedule. The less rescheduling operations that are needed, the more robust the schedules generated by the algorithms are. The throughput parameters can be used as an indication for future problems. The *taxiway throughput* parameter defines the amount of aircraft fed into the system with respect to the amount of aircraft leaving the system over time. If the taxiway throughput is less than 100%, there are more aircraft fed into the system then there are aircraft which leave the system (aircraft taking off). The number of aircraft in the system will grow and thus the margins will decrease and the chance of a disruption will grow. The amount of aircraft that are able to take off per hour is stored in the *runway throughput* parameter. An overview of all output parameters is given in Table 4.2.

Table 4.2: Output parameters of the complete system

Category	Parameter	Description
Aircraft data	timewin	Time window of all aircraft
	speedwin	Speed window of all aircraft
	delay	Delay of all aircraft
	dropouts	Number of aircraft that missed their departure slot
	arrival	Takeoff time of all aircraft
Airport data	numreschedule	Number of performed rescheduling operations
	taxiway throughput	Airborne aircraft w.r.t. supplied aircraft
	runway throughput	Amount of aircraft taking off per hour

4.3 Subsystems

The complete system is split up into multiple subsystems. Besides that splitting up the system brings a reduction in complexity and thus makes the system easier to build and understand, there are also other advantages. Due to this modularity, it is possible to modify only a specific part of the whole system without having to rebuild the other parts of the system. An individual module of the system can easily be changed to improve the functionalities of the system. These modules can also be removed and replaced by other systems or modules. This makes it easy to combine the current system with other external systems.

4.3.1 Simulation of traffic scenario

The first subsystem discussed here is the simulation of the traffic scenario. This subsystem is considered as the core of the complete system. All other subsystems are connected to this subsystem.

When the simulation is started, an initial schedule is received from the scheduling subsystem. Next, this subsystem updates the position of the aircraft and determines whether rescheduling is needed. Rescheduling is needed when an aircraft drifts out of its window or when the scheduling horizon is reached. In these situations, the scheduling subsystem is used to provide a new schedule. All disturbances are also inserted into the simulation in this subsystem.

The simulation of traffic scenario subsystem is the system where the departure schedules calculated by the algorithm are tested. The positions of all aircraft involved in the simulation are tracked. The efficiency of the traffic movements in this phase is one of the most important benchmarks used for evaluating the algorithms. Specific tasks are put out to other subsystems. These tasks are the calculation of a new or improved schedule, the calculation of departure windows, updating the positions of the aircraft and the generation of the disturbances.

When the simulation is finished, the extraction of the evaluation parameters is also done by this subsystem. This block extracts the parameters which are useful for the evaluation of the algorithms from the simulation results. These parameters are presented to the user to let the user be able to do the evaluation of the scheduling algorithms.

4.3.1.1 Input and output parameters

As mentioned before, the modules that together form the designed simulation environment can all individually be replaced by new or updated versions of these modules. Therefore not only the inputs and outputs of the complete system should be clearly defined, but also the inputs and outputs of the individual modules.

The subsystem discussed here is the only subsystem which is connected to the outside world. Therefore the input and output parameters are equal to the parameters discussed in section 4.2.1. However, extra attention is given to the output parameters.

One of the functions of this subsystem is to present the data to the user in such a way that the user is able to do the evaluation of the algorithms. The outputs of this subsystem are the parameters used for the evaluation. The evaluation subsystem presents these parameters in a convenient way to the user. This can be done via a graph or by a single number. The best way to present the data is dependent on the type of parameter and the way the information should be interpreted. This will be explained in more detail in chapter 5. The output parameters are summed in Table 4.3.

Table 4.3: Output parameters of the evaluation subsystem

Parameter	Description
timewin	Time window of all aircraft over time
speedwin	Speed window of all aircraft over time
delay	Delay of all aircraft over time
dropouts	Number of dropped aircraft over time
arrival	Real takeoff times
numreschedule	Number of rescheduling operations
taxiway throughput	taxiway throughput over time
runway throughput	amount of aircraft taking off per hour

There are also parameters needed to communicate with the other subsystems. These are discussed at the next sections about the other subsystems.

4.3.2 Calculation of departure schedule

The next subsystem discussed here is the subsystem where the calculation of the initial departure schedule and the rescheduling operations take place. At the start of the simulation an initial schedule is calculated by the algorithm to be evaluated. This schedule is used as a starting point for the simulation of the traffic.

The rescheduling operations are closely related to the calculation of the initial schedule, so this process is also performed by this subsystem. Rescheduling is needed when an aircraft drifts out of its window or when aircraft are injected or retracted to and from the schedule. A rescheduling operation can have a high influence on the efficiency of the algorithm.

After a schedule is calculated, the separation times between the aircraft are verified whether they meet the separation requirements. This verification and possible correction of departure times is done in a separate subsystem. After this check, the results are handed back to the scheduling subsystem. The scheduling subsystem will present the results to the simulation subsystem.

4.3.2.1 Input and output parameters

To perform a rescheduling operation, more information is needed about the traffic scenario. Therefore the input parameters are divided into two categories. The input parameters in the category initial scheduling are needed for both the initial scheduling and the rescheduling operations. The parameters in the category rescheduling are only used by the rescheduling subsystem. The output parameters of both subsystems are exactly the same. An overview of the input and output parameters of this subsystem is given in Table 4.4 and Table 4.5.

Table 4.4: Input parameters of the scheduling subsystem

Category	Parameter	Description
Initial scheduling	aircraft	Aircraft data: - Desired takeoff time - Starting location at airport - Aircraft class
	path	Position and length of taxiways
	rttaxi	Time when aircraft are ready to start taxiing
	separation	Required wake vortex separation per aircraft class
	mitseparation	Required miles-in-trail separation per aircraft class
	slotboundaries	Boundaries of aircraft's takeoff slot at runway
Rescheduling	positions	Position of all aircraft
	departure	Scheduled departure time of all aircraft
	time	Time
	speed	Speed of all aircraft
	realetas	Scheduled takeoff time of all aircraft

Table 4.5: Output parameters of the scheduling subsystem

Parameter	Description
departure	Scheduled departure time of all aircraft
speed	Speed of all aircraft
realetas	Scheduled takeoff time of all aircraft

4.3.3 Verify and correct separations

After the calculation of a departure schedule, this schedule is forwarded to the subsystem discussed here to verify and, if needed, correct the separation of the aircraft. In the ideal situation this subsystem would be needless, because the scheduling subsystem would calculate a departure schedule that would not violate any separation constraints. This subsystem is considered as an extra check. If the scheduling algorithm is designed well, there is no need to verify and correct the separations.

However, by including this subsystem, it is possible to use also imperfect algorithms. When an algorithm would violate the separation requirements or does not take the separation requirements into account at all, this is corrected by this subsystem. A disadvantage of these corrections is that they do make the schedule less ideal and often introduce extra delay.

4.3.3.1 Input and output parameters

To perform a verification of the departure schedule, five parameters are required by this subsystem. The parameter *aircraft* consists of the aircraft scheduled and the aircraft data mentioned earlier when discussing this parameter at other subsystems. The parameters *departure* and *realetas* consist of output data of the scheduling algorithm. These are the times when the aircraft is scheduled to start taxiing and the time when the aircraft should arrive at the runway respectively. The *speed* parameter contains the calculated speeds of all aircraft. Finally, the *time* parameter is passed into this subsystem to be able to check whether scheduled takeoff times are realistic.

Data about the required separation times is not passed to this subsystem, but is considered as already known by this subsystem. An overview of all input parameters is given in Table 4.6. The outputs of this subsystem are the parameters which are verified by this subsystem; the parameters *departure*, *speed* and *realetas*.

Table 4.6: Input parameters of the verify subsystem

Parameter	Description	
aircraft	Aircraft data:	- Desired takeoff time - Starting location at airport - Aircraft class
departure	Scheduled departure time of all aircraft	
speed	Speed of all aircraft	
realetas	Scheduled takeoff time of all aircraft	
time	Time	

4.3.4 Calculation of departure windows

After a departure schedule is calculated and verified, the result is handed back to the simulation subsystem. This simulation subsystem simulates the air traffic with the given scenario and schedule. If needed, rescheduling is performed. To determine when rescheduling is needed, constraints need to be defined. When these constraints are defined, they can be checked and if an aircraft does not satisfies all the constraints anymore, rescheduling is needed. These constraints are modeled as departure windows. These windows define the minimum and maximum location and speed of all aircraft. If an aircraft stays within these borders, it is able to reach the runway in time. If it drifts out of these borders, the runway cannot be reached in time anymore and rescheduling is needed.

4.3.4.1 Input and output parameters

To be able to calculate these departure windows, seven input variables are needed. The *aircraft* parameter is used to get information about the scheduled aircraft. This information includes the class of the aircraft and the ideal arrival time. This information defines the earliest and latest arrival time allowed and the speed constraints of the aircraft. The *speed* and *departure* parameters define the speed and departure times assigned to the scheduled.

To calculate travel times and points where two taxiways merge and possible problems can occur, knowledge about the airport layout is necessary. This information is included in the *path* parameter. The *slotsize* parameter defines the minimal aircraft separation during taxiing.

The last two input variables are needed to calculate the departure windows in case of a rescheduling operation. These parameters, *time* and *positions*, define the current time and the current position of the aircraft. When rescheduling takes place, aircraft could already have started taxiing and thus are located somewhere on the taxiways. Their position should be known do determine whether they could form a possible bottleneck for other aircraft. An overview of these input parameters is given in Table 4.7.

Table 4.7: Input parameters of the departure window subsystem

Parameter	Description	
aircraft	Aircraft data:	- Desired takeoff time
		- Starting location at airport
		- Aircraft class
path	Position and length of all taxiways	
speed	Speed of all aircraft	
slotsize	Minimum required separation during taxiing	
departure	Scheduled departure time of all aircraft	
positions	Position of all aircraft	
time	Time	

The outputs of the departure window subsystem are the time and speed windows for all aircraft. These windows are split up in two parts: the departure window on the taxiway from spot to main taxiway (parameters *startpos* and *starttime*), and the departure window on the main taxiway (parameters *waytime*, *waypoints* and *wayspeed*).

The taxiway from spot to main taxiway does not contain any intersections, so it is sufficient to only define a starting time for each aircraft and no other waypoints. The starting position is also defined. After a rescheduling operation it is possible that the minimal and maximal time window start at another position than the aircraft itself.

In contrast to the taxiway from the spot, the main taxiway does contain intersections where other taxiways feed in to the main taxiway. These are points where other aircraft can enter the main taxiway and thus possible bottlenecks. Therefore the parameters which define the departure windows on the main taxiway (the parameters *waytime*, *waypoints* and *wayspeed*) does not only define an initial departure window, but can also contain a sequence of departure windows or bottlenecks for each aircraft.

The parameter *waypoints* defines for each aircraft the minimum, maximum and ideal position. The initial position of the departure window is defined, and the positions where de departure window changes. The parameter *waytime* defines at which time the boundaries of the departure window arrive at the corresponding positions as defined in *waypoints*. For example, if the position of the minimum departure window arrives at position x=4000 at time t=500, the aircraft should not arrive at position x=4000 before time t=500. The parameter *wayspeed* is used as an extra guide to get more insight into the departure window. If an aircraft obeys the speed limits given in the *wayspeed* parameter, it will never drift out of its departure window.

4.3.5 Update positions

The subsystem for updating the positions of the aircraft and windows is a very general subsystem. The only task of this subsystem is to update the position of the aircraft and windows during the simulation.

The reason why this is designed as a separate subsystem instead of an integral part of the simulation is because there are many aircraft and windows of which the position must be updated. When designing this functionality as a separate subsystem, this only needs to be designed once. Otherwise a function with this functionality will occur many times in multiple subsystems.

4.3.5.1 Input and output parameters

The first input parameter of the subsystem discussed here is the parameter *positions*. In this parameter all positions of the objects (windows or aircraft) of which the position must be updated are stored. The next parameter, the *speed* parameter, defines the speed of these objects. The parameter *step* defines the step size used in the simulation. If a step size of 5 seconds is used, and an object has a speed of 5 m/s, its position should be updated with 25 meter each time. The last two parameters, *departure* and *time*, define the departure time of the objects and the actual time. If the position vector contains objects of which the departure time is not reached jet, the position of these objects will not be updated, whether or not a speed is defined for these objects. An overview of all input parameters is given in Table 4.8.

This function only has one output parameter. The parameter *positions* contains all updated positions after the update performed by this subsystem.

Table 4.8: Input parameters of the update positions subsystem

Parameter	Description
positions	Position of all aircraft
speed	Speed of all aircraft
step	Simulation time step
departure	Scheduled departure time of all aircraft
time	Time

4.3.6 Insert disturbances

To make the evaluation of the algorithms a useful evaluation and not only an evaluation of the algorithms in a hypothetical ideal environment, the non-idealities of the real world must be taken into account as much as possible. The non-idealities which can occur during operation in the real world can be modeled like disturbances. In the ideal world all aircraft would exactly follow their schedule and thus the final schedule is exactly the same as the initial schedule. These disturbances are modeled as a separate subsystem.

The disturbance subsystem takes care for the insertion of the disturbances into the simulation. As mentioned in section 2.1.2, a distinction can be made between two types of disturbances. Both types of disturbances can be inserted into the system. The first type of disturbances is speed disturbances. These disturbances affect the speed of the taxiing aircraft. These disturbances can have multiple causes like bad weather conditions which cause the aircraft to taxi more slowly, technical failures or a pilot which is taxiing faster or slower than the system expects. The other type of disturbances is a disturbance of the time at which the aircraft starts taxiing. These disturbances can occur in two ways. If the disturbance can be foreseen before the aircraft is expected to start taxiing, the time when the aircraft is ready to taxi can be adjusted. In this case the disturbance is already known before it takes place. The algorithm can try to reduce the effect of this delay by recalculating the aircraft schedule taking into account this new information. The other possibility is that these disturbances are not known in advance. In this case the algorithm cannot reduce the effect in advance, but only when the aircraft drifts out of its window.

4.3.6.1 Input and output parameters

When the disturbance is not known in advance, it is not possible to change the time when the aircraft is ready to start taxiing. In this case the disturbance will not be noticed until the moment the aircraft was planned to leave the spot. This is implemented via a speed disturbance. The speed of the aircraft will stay zero and the aircraft will not leave the spot.

The input of the disturbance subsystem that is not discussed yet is the *time* parameter. This time information is needed to make the disturbances time variant. This gives the subsystem the possibility to trigger delays at a certain moment in time. Due to the structure of a Matlab simulation, all disturbances should be, just like all aircraft data, already known before the simulation starts. The disturbance subpart inserts these disturbances into the simulation at the time they should get known by the simulation. Although the disturbances are already known by the disturbance subsystem on beforehand, they are not published before the moment they would be known in real life. Besides these time variant disturbances, disturbances can also be dependent on other parameters like other disturbances or the speed of other aircraft. The inputs and outputs of the disturbance subsystem are given in Table 4.9 and Table 4.10.

Table 4.9: Input parameters of the disturbance subsystem

Parameter	Description
speed	Speed of all aircraft
time	Current time
rttaxi	Time when aircraft are ready to start taxiing

Table 4.10: Output parameters of the disturbance subsystem

Parameter	Description
disturbed_speed	Disturbed speed of all aircraft
disturbed_rttaxi	Disturbed time when aircraft are ready to start taxiing

4.4 Design flowchart

After all inputs and outputs are defined, the next step in the design process is the generation of the simulation itself. The individual subsystems can be designed one by one and connected to each other to complete the integral simulation. An overview of the system split up in all parts is shown in Figure 4.3. This overview can be more concretized to define all simulation steps.

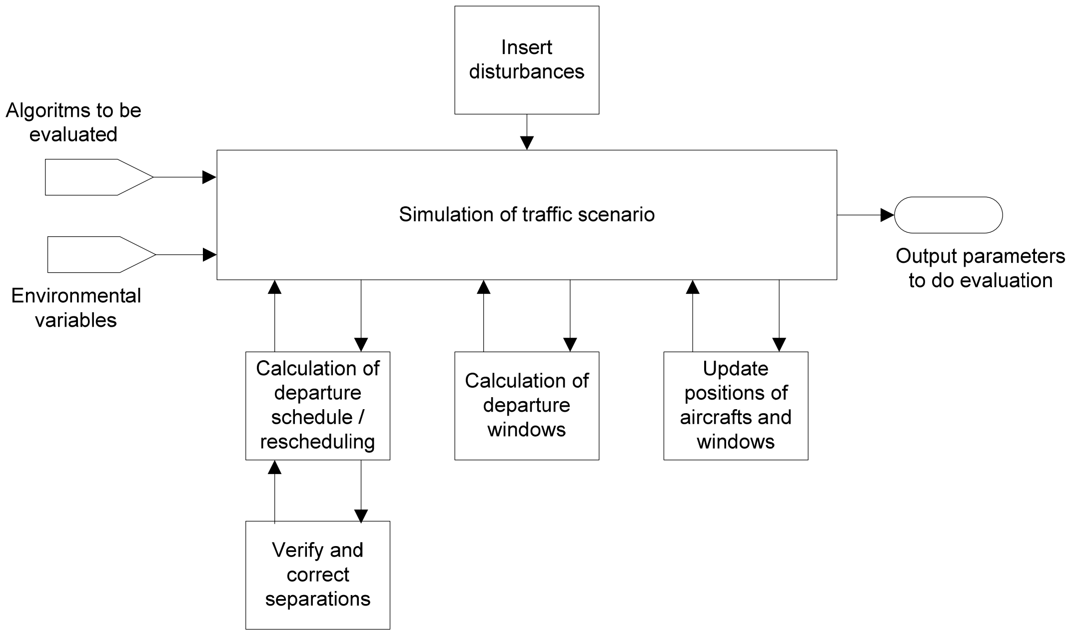

Figure 4.3: The system split up in functional blocks

The first block in the flow chart in Figure 4.4 is the block discussed in section 4.3.2. During this phase this block takes care for the initial departure schedule. All aircraft that are ready to leave their spot within this window will be scheduled here. For each aircraft a takeoff time and a time to leave its spot is defined. This schedule is calculated via an algorithm selected by the user. When an initial departure schedule is generated, the departure times are checked for separation violations. This is done in the next block, which was discussed in more detail in section 4.3.3.

The next step in the calculation of the initial schedule is the calculation of the time and speed windows. Based on the scheduled takeoff time and the time to start taxiing, time and speed windows are calculated for each aircraft, as discussed in section 4.3.4.

To make the simulation as realistic as possible, not only the ideal theoretical situation should be simulated, but the situation in real life. To estimate the real life situation as much as possible, a disturbance component is added to the simulation. Both types of disturbances can be applied to all aircraft at each possible moment. Most disturbances are time independent and not dependent on the current traffic situation. Detailed information about this subsystem can be found in section 4.3.6.

If an aircraft drifts out of its window, a rescheduling operation is triggered. This rescheduling operation is performed by the same three blocks as described by the calculation of the initial schedule.

The last system block discussed in this paragraph is the block where the evaluation parameters are presented to the user. As stated in chapter 2, the simulation does not perform the evaluation itself. The simulation is used as a tool to allow the user to do the evaluation. The outputs of the algorithm are graphs and vectors of specific parameters. The trend of this graphs and the value of the vectors enables the user to do this evaluation. This block is not a separate subsystem, but it is implemented as a part of the simulation of the traffic scenario subsystem. The more detailed flow chart discussed here is shown in Figure 4.4.

4.5 Limitations

The simulation performed by this system is an approximation of reality. To make this approximation as good as possible, constraints need to be taken into account. These are both constraints due to the fact that a simulation is an approximation of reality and constraints that also hold during real life. The constraints due to the fact that this simulation is an approximation of reality are discussed in this section.

The simulation is built using Matlab. All data that is needed during the entire simulation is loaded into the system at the beginning of this Matlab simulation. Therefore all data must be available already before the start of the simulation. Although the algorithm used in the simulation uses a scheduling window because in reality it is impossible to include all aircraft data directly from the start (see chapter 4), for this simulation all aircraft data should be known in advance. It is not possible to add new data when the simulation already started.

Although all data needed during the simulation must be defined in advance, this data does not necessary have to be fixed during the whole simulation. It is possible to change the simulation parameters. This is important to be able to update parameters like the estimated time when an aircraft is ready to leave the spot (if an aircraft knows that it is getting delayed in the future, it can update its time schedule to prevent further delays). Other examples are the possibility to insert or remove aircraft to and from the system and to introduce new disturbances during the simulation. These parameter changes are introduced via the disturbance part of the system, so the parameter changes are not known by the algorithm to be evaluated during the simulation, but they are already known in the disturbance part of the simulation.

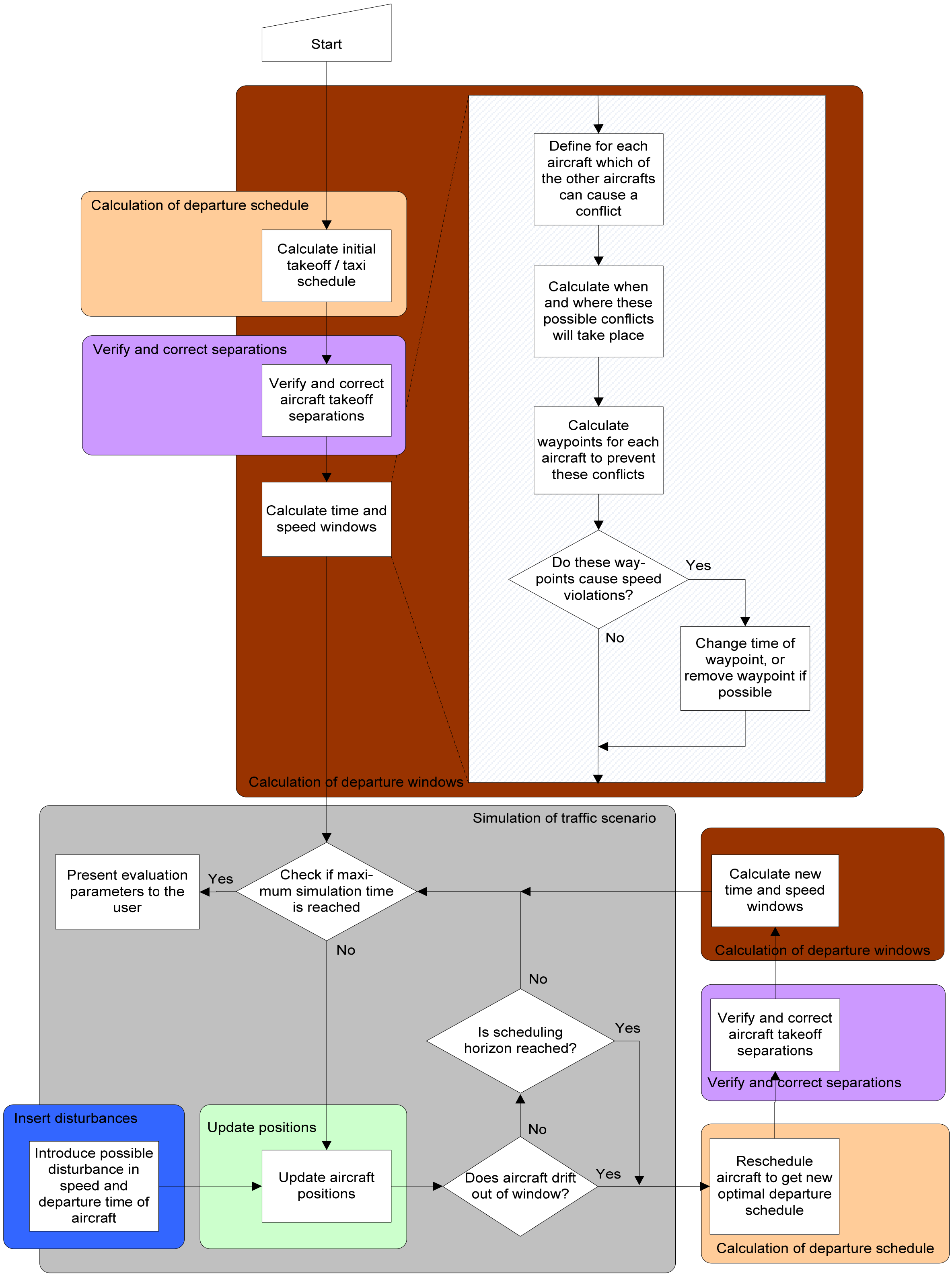

Figure 4.4: More detailed functional overview of the system to be designed

This distinction is important to maintain the restrictions to make the simulation as realistic as possible. The algorithm uses a scheduling window to schedule the aircraft. It must be impossible for the algorithm to know what will happen after this scheduling window (in the future) to prevent the algorithm making an unrealistic efficient schedule, because in reality these future events are also not known in advance. All data is available to the simulation at the moment when the simulation is started. The simulation takes care for passing through this data to the algorithm part by part, just like the way how it reaches at the algorithm in real life.

The last constraint discussed here has to do with the discrete nature of the simulation. Due to the way the simulation is built, the data is processed by the simulation in a discrete way. This does not necessary have to be a negative aspect for the simulation. In reality, input parameters like the position information of the aircraft, will also be available in a discrete way. As long as the update rate of the input variables is lower than or as high as the internal update rate of the algorithm, this discreteness does not have a negative impact on the simulation results.

4.6 Future applications of the design

As mentioned before, the simulation environment is built in a modular way. A number of separate modules is coupled together to create the functionality described in chapter 2. The modularity of de design makes the design flexible. It is easy to make modifications to the simulation without having to rebuild the complete system. This makes the design future proof.

The modules the simulation is built with, as described in section 4.2, can individually be modified or replaced by another module or system. This creates the ability to easily upgrade specific parts of the system. Besides updating, these modules can also be modified to change the use of the simulation. Instead of evaluating scheduling algorithms, the simulation can be modified to evaluate for example a method to reduce disturbances. Other possibilities include evaluating the efficiency of multiple taxiway layouts or routes to reach the runway, research to determine optimal taxi speed or expand the simulation to multiple runways.

Another, less fundamental modification of the simulation is the option to add new algorithms or modify the currently used scheduling algorithms. The only action needed for this modification is a replacement of the algorithm block. The ability to do this modification is crucial for the use of the simulation. The five algorithms included in the simulation are the general algorithm categories, but for practical use it is useful to be able not only to check the general categories, but also specific scheduling algorithms.

The possibility to use specific, more detailed algorithms in the simulation is closely related to the last possibility for future application discussed in this paragraph. This possibility uses the system not for testing and evaluation of algorithms, but during actual practice. If the system is used to simulate a real airport in such a way that the simulation is close to reality, it is possible to use the simulation as a guide for the ground controller.

The ground controller's workload can be reduced by using the system to simulate a copy (direct mirror) of the current airport situation. In this situation the system can be used to verify the decisions of the ground controller, or to suggest an (optimal) solution to the ground controller. This can optimize the ground traffic handling and reduce the ground controller's workload. However, more research is needed before this suggestion can be put into practice.

5. Design evaluation

In this chapter the algorithms discussed in chapter 3 are evaluated using the system designed in chapter 4. The goal of this evaluation is two-sided. The system designed is evaluated using the determined rating criteria and the selected algorithms are tested for their efficiency.

Before the efficiency of the selected algorithms can be tested and the system itself can be evaluated, test cases should be defined. The simulation procedure itself is explained in section 5.1. The test cases define the aspect to be investigated, the input data to do this investigation and the expected output. These test cases are discussed in section 5.2. The results of these experiments are discussed in section 5.3. In that section the algorithms are evaluated and the weak and strong points of the system are discussed.

5.1 Simulation procedure

To be able to perform the evaluation of the system and the scheduling algorithms, a value for all input parameters of the system should be defined. These values can be defined in a scenario-based or in a parameter-based approach. When the scenario-based approach is chosen, a traffic situation (scenario) is defined and translated to values for all input parameters.

Specific scenarios, like a busy scenario, a quiet scenario and scenarios with few or many disturbances are fed into the system. The outputs are used to investigate how the system reacts on these situations. A disadvantage of this approach is that even though the input scenarios can be realistic scenarios, it is difficult to determine the exact cause of a change of the output results. There is no direct relation between the change of the output parameters and an individual input parameter. To deal with this problem, the parameter-based approach is chosen to test the system and the algorithms.

This parameter-based approach makes use of a standard situation. For this situation a default parameter value is defined for each input parameter. When this default situation is fed into the system, this will result in an output result for each algorithm which is evaluated. This output is assumed as the default output (the so called benchmark). The next step is to change only one of the input parameters. All other inputs are not changed. The simulation is executed again with this new input data. The change of the output parameters in this new situation is caused by the variation of the changed input parameter. Using this concept, the impact of changing each single input parameter can be investigated.

5.2 Test cases to test the algorithm

In this section the variations of the input parameters to test the system and the algorithms as explained in section 5.1 are discussed. Each input parameter is discussed, the chosen variations are explained and the expected change of the output parameters is discussed.

5.2.1 Aircraft

The set of aircraft is the most important input parameter of the system. The amount of aircraft and their characteristics like their desired takeoff time, their departure route and the aircraft type determine whether it is possible to optimize the departure schedule and to what extent the algorithms under investigation will succeed in this.

To be able to find the differences between the algorithms, a number of test scenarios is defined. These scenarios consist of a number of aircraft and their characteristics. The scenarios are defined in such a way that the differences between the scheduling algorithms will become visible as much as possible. In some of the situations, the aircraft have an ideal takeoff time with that less separation such that runway capacity is insufficient to let all aircraft take off without delay. The scheduling algorithms are tested with these scenarios to investigate whether the algorithms are able to remove or reduce the disturbances.

The first scenario consists of heavy and large aircraft willing to depart alternating. Their standard instrument departure routes are also alternating such that the minimum required miles-in-trail separation will not be a limiting factor. The required gaps between the desired takeoff times of the aircraft are too short to let all aircraft take off at their desired time without violating the separation requirements. After 54 minutes and 1:34 hour a gap is scheduled to be able to let all aircraft take off which are delayed in the period before.

The second scenario also consists of heavy and large aircraft willing to depart alternating. In this scenario each two successive aircraft have the same standard instrument departure route, so the miles-in-trail separation can also be a limiting factor. The separation between the desired takeoff times is smaller than the required wake vortex separation.

Scenario three consists of bursts of aircraft. During these bursts the aircraft wants to take off with a separation of one minute. This separation is insufficient to satisfy the separation requirements so a change of departure time is required. During each burst the aircraft are grouped in groups of heavy and groups of large aircraft.

The fourth scenario contains groups of a specific aircraft class, which are willing to take off at exactly the same moment. After each group a gap is included and then a new group of aircraft wants to take off. This next group contains aircraft of another class. The SID-routes of each group are divided among two SIDs to prevent the SID to be the limiting factor. An overview with the characteristics of all four scenarios is given in Table 5.1.

Table 5.1: Characteristics of the test scenarios

	Scenarios			
	1	**2**	**3**	**4**
Number of aircraft	60	60	60	60
Timespan takeoff of all aircraft	1:41:00	1:51:00	1:24:00	1:20:00
Departure locations	Random alternating between all options	Random alternating between all options	Random alternating between all options	Random alternating between all options
Aircraft classes	Heavy and large alternately	Heavy and large alternately	5x heavy and 5x large alternately	5x heavy and 5x large alternately
SIDs	SID 1 and 2 alternately	2x SID 1 and 2x SID 2 alternately	SID 1 and 2 alternately	SID 1 and 2 alternately
Desired takeoff times	- Separations slightly smaller than required separation - 12 minutes break after 0:54 hour - 5 minutes break after 1:34 hour	- Separations slightly smaller than required separation - 10 minutes break after 0:30, 1:00 and 1:30 hour	- 1 minute separation between aircraft - 10 minutes break after each 15 aircraft	- Each 5 successive aircraft scheduled at same time - Alternately 5 and 10 minutes break between groups

The departure routes used by the aircraft are based on the situation at Schiphol airport. These departure routes from runway 24 (Kaagbaan) are simplified to three departure routes. These routes are shown in Figure 5.1. The departure routes used in this simulation can be different from the real departure route used by the aircraft. This also holds for the starting positions.

The starting positions of the aircraft are based on their departure gate, but this assumption can differ from the real situation. An overview of the scenarios used is given in Appendix A.

5.2.2 Simulation time

The parameter *simtime* defines the time span to be simulated. The scenarios used in the simulation use a time span up to three hours. To be able to also take the behaviour of delayed aircraft into account, the simulation will run for an extra half hour, so the total simulation time is 3.5 hours. This parameter will be kept constant for all simulations. Varying this parameter will enlarge or shorten the duration of the simulation, but it will not change the outcome at each moment in time.

5.2.3 Step

The next input parameter is the *step* parameter. This parameter defines the step size used during the simulation. A smaller step size will increase the resolution, but also increases the required computational power or the time needed to complete the simulation. Therefore the step size should be chosen as large as possible. However, if the step size is chosen too large, potential disturbances might not be detected by the algorithm, or not picked up in time. Meanwhile, the effects of the disturbance will grow and affect more aircraft compared to the situation when it was detected and corrected immediately. Two step sizes are used during the simulations. The default step size is five seconds, to be sure that the system always will have an accurate overview of the situation and a step size of sixty seconds, to investigate the effect of a low update rate on the efficiency of the algorithms.

5.2.4 Scheduling window

The *schedwin* parameter defines the size of the scheduling window. An aircraft will receive its departure slot at the earliest two hours before estimated off-block time (EOBT). Therefore it is useless to calculate a departure schedule more than two hours in advance. So the maximum scheduling window used during simulation will be two hours. The shortest scheduling window which is possible is the step size which is used. In this situation the departure times of the aircraft are not calculated in advance, but at the moment the aircraft are ready to start taxiing. When this window is made larger, the algorithm looks more forward and thus is expected to be able to prepare for possible conflicts in the future, but the uncertainties of aircraft more far in the future are also bigger, thus rescheduling might be needed more often. When the window is made smaller, it is more difficult to predict the traffic situation in the future, so the schedule is expected to become less efficient. By default, a scheduling window of fifteen minutes is used.

5.2.5 Algorithm

The next input of the system is the *algorithm* input. Via this input the algorithm used to calculate the schedule is selected. Besides the four types discussed in section 3.3, a reference algorithm can be selected. All combinations of the other input variables are tested with all five algorithms implemented in the system.

5.2.6 Disturbances

The *disturbances* parameter is implemented in a different way. This input is not modeled as a single scalar or vector, but as one of the subsystems of the design. By default, the simulation is ideal and no disturbances are taken into account. After simulating this ideal situation, the disturbances are increased. It is expected that the runway capacity will decrease when the amount of disturbances increases. Also the number of rescheduling operations is expected to increase. The type of disturbances used is based on [11]. The delay is gradually increased from no delay to twice the values stated in this reference. This will mean a maximum average delay of two times sixteen minutes per flight, of which 20% is taxi delay, and 58% gate delay. The rest of this delay occurs during the other parts of the trajectory, which are not under investigation in this research. So the average delay used using this simulation is two times 78% of sixteen minutes, which is 25 minutes per aircraft. The cause and type of disturbances are discussed in more detail in section 2.1.2.

5.2.7 Airport layout

The parameters *mainpath* and *path* define the position and length of the taxiways of the airport which is used during the simulation. The scope of this research is to design a system to be able to evaluate the scheduling algorithms and not to design an optimal airport layout. Therefore the airport layout does not change during this research. The airport layout used during this evaluation is based on the layout of the taxiways of Schiphol airport, using runway 24 (Kaagbaan). An overview of the layout used and the length and location of the taxiways are shown in Figure 5.2 and Table 5.2. The two main taxiways are drawn in red, and the taxiways from spot to the main taxiways are shown in blue in Figure 5.2.

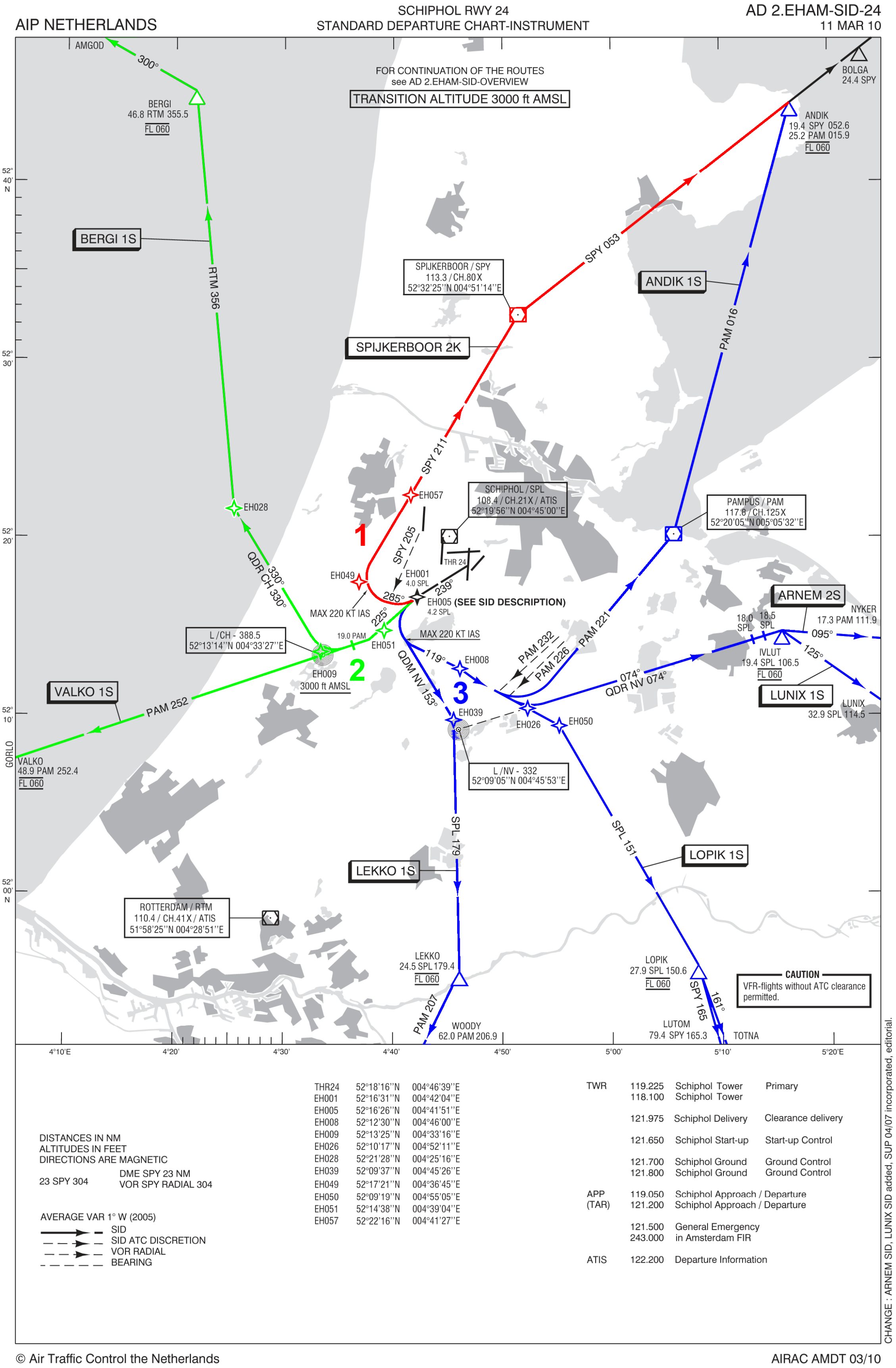

Figure 5.1: Departure routes used during simulation

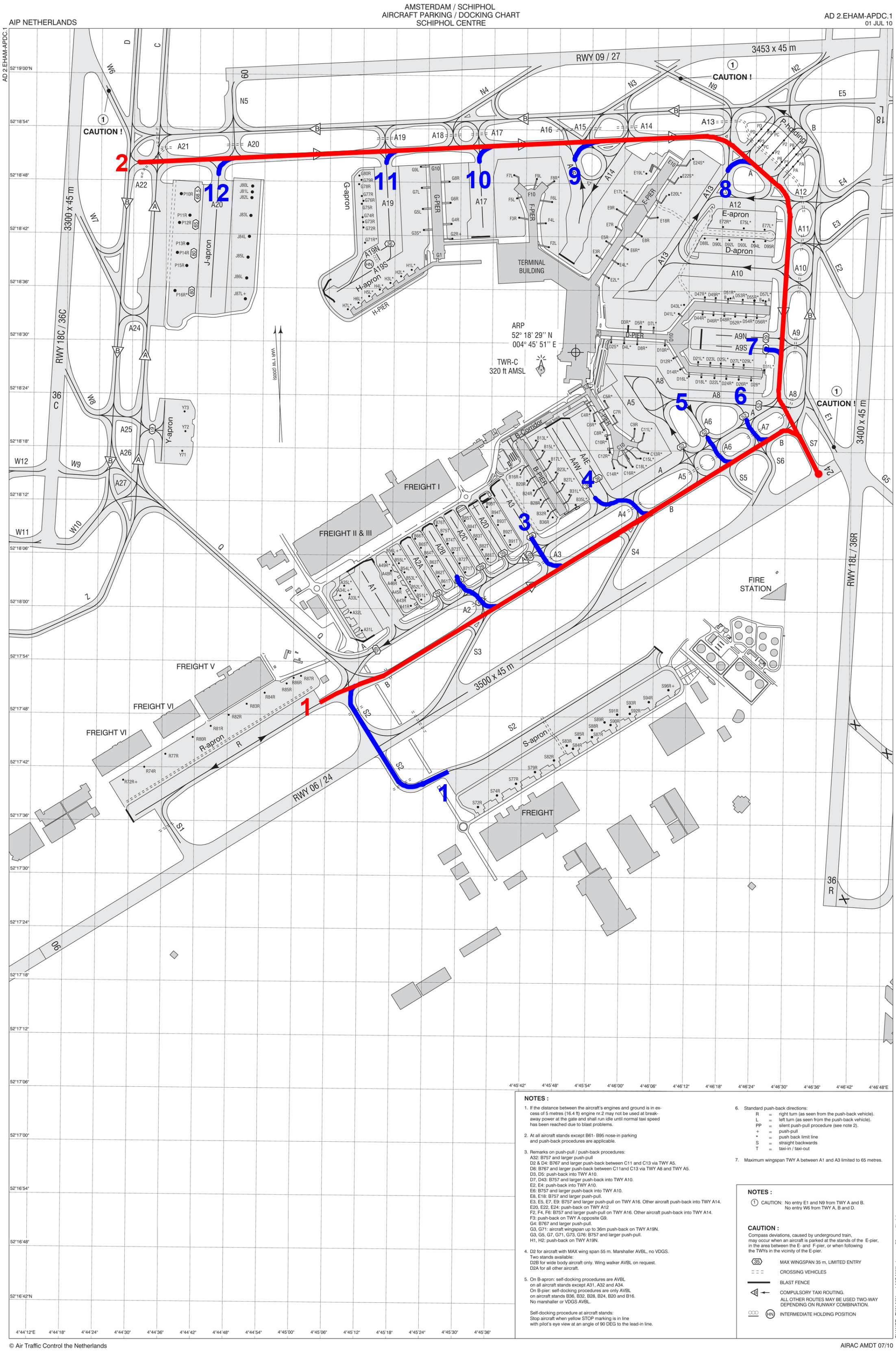

Figure 5.2: Location of the taxiways

Table 5.2: Location and length of taxiways

Taxiway number	Length (m)	Position where connected to main taxiway (m)	Number of main taxiway connected to
1	566	1953	1
2	152	1398	1
3	145	1128	1
4	178	768	1
5	145	422	1
6	145	280	1
7	65	458	2
8	100	1177	2
9	100	1774	2
10	65	2122	2
11	65	2450	2
12	65	3020	2

5.2.8 Slotboundaries

The boundaries of the departure slots assigned to the aircraft are defined by the parameter *slotboundaries*. As defined by Eurocontrol, the slot boundaries are -5 and +10 minutes from calculated take-off time (CTOT). This will be the default value used for the simulation. A smaller departure slot reduces the uncertainties. Therefore these slots will be gradually tightened to a size of -1 and +1 minute, and widened to -10 and +15. Both symmetrical and asymmetrical margins are used, to check whether there is a difference in efficiency between using both types. It is expected that smaller slots will cause more rescheduling operations and aircraft which are unable to take off within their assigned departure slots.

5.2.9 Aircraft based speed constraints

The aircraft based speed constraints, as defined by the parameter *abspeed*, will by default not be different than the speed constraints defined by the class of the aircraft. However, in reality, situations might occur where the aircraft based speed constraints do differ from the speed constraints defined by the class of the aircraft. To simulate these situations, the aircraft based maximum speed is gradually lowered to 10% of the maximum speed based on the aircraft class. This speed constraint is unknown for the scheduling algorithm. It is expected that a lower maximum speed will cause more disturbances and thus more rescheduling operations.

5.2.10 Time when ready to taxi

The parameter *rttaxi* defines the time when the aircraft are ready to start taxiing. The default time when an aircraft announces itself as ready to taxi is taken as the scheduled takeoff time of the aircraft minus the time it takes to taxi to the runway and ten extra minutes margin. For example, if an aircraft's ideal takeoff time is at t=500 and the taxi time to the runway is 200, the aircraft should start taxiing at t=300. If the ready to taxi margin is 100, the time when the aircraft is ready to start taxiing is at t=200.

The ten minutes margin provides more possibilities for the scheduling algorithm to create an efficient departure schedule. When no margin is used, the aircraft should immediately start taxiing when it is ready to taxi. Otherwise the aircraft will always be delayed. Due to this margin, the departure time can be varied within this margin without being delayed immediately. So the aircraft is able to take off ahead of its ideal departure time due to this margin.

The minimal margin used during the simulation is no margin, and the maximum margin is thirty minutes. No negative margins are used, because in that case it is impossible for an aircraft to reach the runway without being delayed. It is expected that a bigger margin will lead to a more efficient departure schedule.

5.2.11 Aircraft classes

The data about the aircraft class is directly linked to the aircraft used in the simulation. Al aircraft can be divided into three classes, as explained in Table 2.1. The class of which an aircraft belongs to depends on the aircraft type. In practice, only heavy and large aircraft visit Schiphol airport, so only these classes are used during the simulation.

5.2.12 Separation times

The parameters *separation* and *mitseparation*, representing the minimum required wake vortex and miles-in-trail separation times are also fixed parameters. These separation times are fixed, as explained in chapter 2.

5.2.13 Parameter overview

As mentioned before, all parameters, except the scenario and the algorithm used, will be varied one by one. An overview of all parameters which are varied during the simulation, their default and their other values is given in Table 5.3.

Table 5.3: All simulation parameters and their variations

Parameter	Default value	Variations	Unit
Step	5	5 / 60	Seconds
Schedwin	15	0 / 15 / 30 / 60 / 90 / 120	Minutes
Algorithm	All	1 / 2 / 3 / 4 / 5	
Disturbances	No delay	No delay / gate delay / taxi delay / gate and taxi delay 6.25 / 12.5 / 25 minutes average delay	Minutes
Mainpath		See Table 5.2	
Path		See Table 5.2	
Slotboundaries	[-5 10]	[-1 1] / [-1 2] / [-5 5] / [-5 10] / [-10 10] / [-10 15]	Minutes
Aircraft		See Appendix B	
Abspeed	Class speed	0.10 / 0.50 / 0.75 / 1.00 * Class speed 10% / 20% of aircraft have deviated speed	m/s
Rttaxi	Taxitime + 10	Taxitime / Taxitime + 10 / Taxitime + 20 / Taxitime + 30	Minutes
Class		See Appendix B	
Separation		See Table 2.1	
Mitseparation		See Table 2.2	

5.3 Evaluation of the algorithms

After having defined the test cases, it is time to start with the evaluation of the departure scheduling algorithms. First, the standard situation defined in section 5.2 will be simulated. The four air traffic scenarios are used to evaluate the algorithms using this standard situation. Next, all input parameters are changed one by one in the way as described in Table 5.3. These variations are used to get more insight in the effectiveness of the scheduling algorithms and to determine an optimal value for all input parameters.

For each algorithm and scenario, the outputs are evaluated on eight aspects. These aspects are:

- Total aircraft delay
- Average aircraft delay
- Maximum individual aircraft delay
- Number of aircraft departing out of their slots
- Number of rescheduling operations
- Arrival times
- Taxiway throughput
- Runway throughput

These aspects are selected based on the rating criteria defined in section 2.4.

5.3.1 Standard situation

5.3.1.1 Delay

After performing a simulation with the five different algorithms and four aircraft scenarios, the first output to be analyzed is the *delay* parameter. The most remarkable result is the bad performance of the branch and bound and the greedy algorithms. Looking at the total delay in Figure 5.3, the delay of these algorithms is in all scenarios at least three times the total delay of the other algorithms. Also the maximum individual and the average delay of the branch and bound and greedy algorithms are higher than the delay of all other algorithms in all scenarios, as shown in Figure 5.4.

This difference can be explained by the fact that the branch and bound and greedy algorithms are not designed to schedule the aircraft at their desired takeoff time as good as possible, but to minimize the required separation between all aircraft and to optimize the runway throughput. Therefore aircraft can be delayed to improve the performance on these aspects.

The results of the other three algorithms do not show a clear difference. However, it is worth to mention that in scenario 1 and 2 the reference algorithm starts with a negative delay. This is caused by the principle that the reference algorithm schedules the aircraft at their first possible takeoff time. This can be before their ideal takeoff time, causing a negative delay. Scenario 4 shows that as soon as no initial order is given for the desired takeoff times and multiple aircraft have the same desired takeoff time, the advantage of the reference algorithm disappears.

The plots with the individual and maximum delays in Figure 5.4 show high peaks in the maximum delay of the reference and genetic algorithm in scenario 3 and 4. The maximal individual aircraft delay of these algorithms is five times as high as the average delay of these algorithms. This means that high delays can be assigned to individual aircraft to reduce the total delay of the complete system.

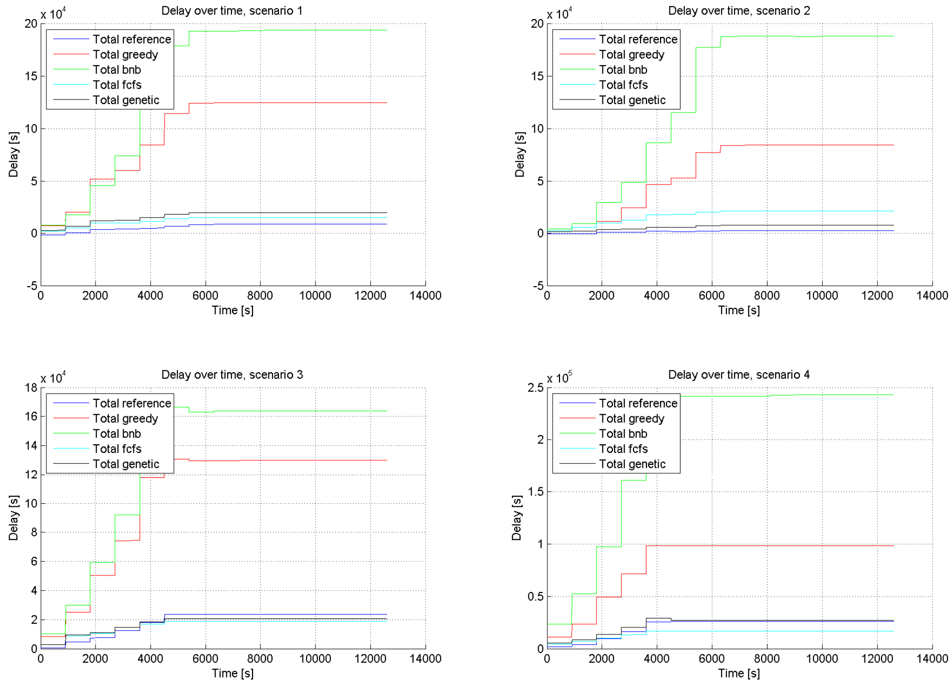

Figure 5.3: Total delay in the standard situation

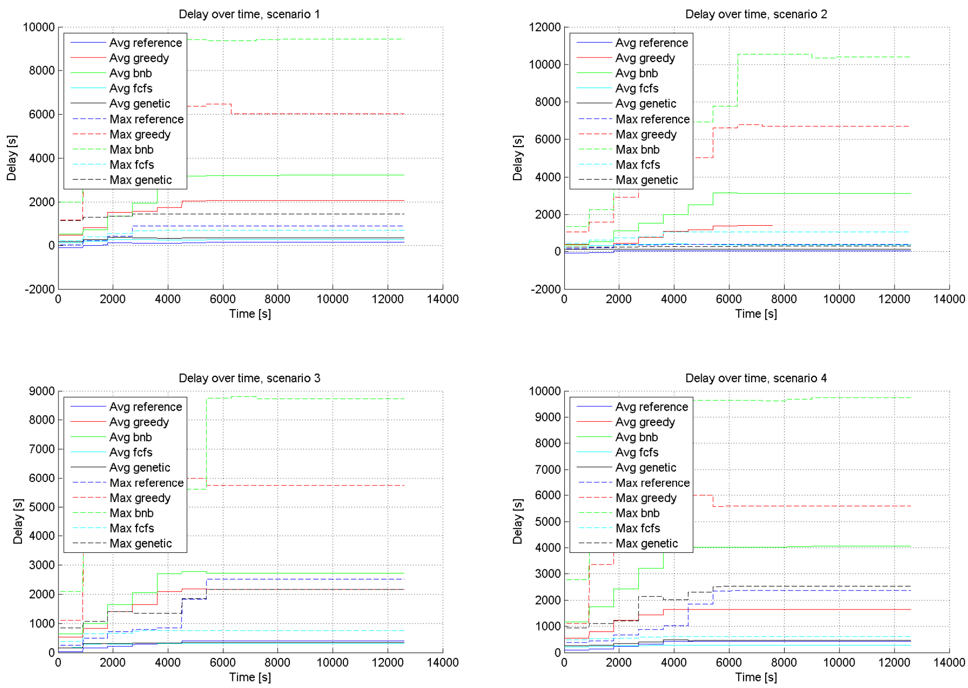

Figure 5.4: Individual and maximum delay in the standard situation

5.3.1.2 Rescheduling operations

The next output discussed here is the number of times a rescheduling operation is performed by the algorithms. It is expected that, in case of no disturbances, the initial schedule would be sufficient and thus no rescheduling operations will be needed, except for the rescheduling each time a new time window is reached. However, Figure 5.5 shows a different number of rescheduling operations, depending on the scenario and the algorithm used. Some of the algorithms will not perform new rescheduling operations after t = 6300 seconds and other algorithms will continue with new rescheduling operations at the start of each new time window. This can be explained by the differences in scheduled departure times for all algorithms. In, for example, scenario 2, the greedy and branch and bound algorithm need ten and twelve periodic rescheduling operations, while the other algorithms only need eight. This is caused by the fact that the greedy and branch and bound algorithms need more time to let all aircraft take off. At t = 7200 seconds, all aircraft have already taken off for the genetic, first come first served and reference algorithm, so for these algorithms new rescheduling operations are not needed anymore. The greedy and branch and bound algorithms still have aircraft on the ground, so new rescheduling operations are still needed for the new scheduling windows.

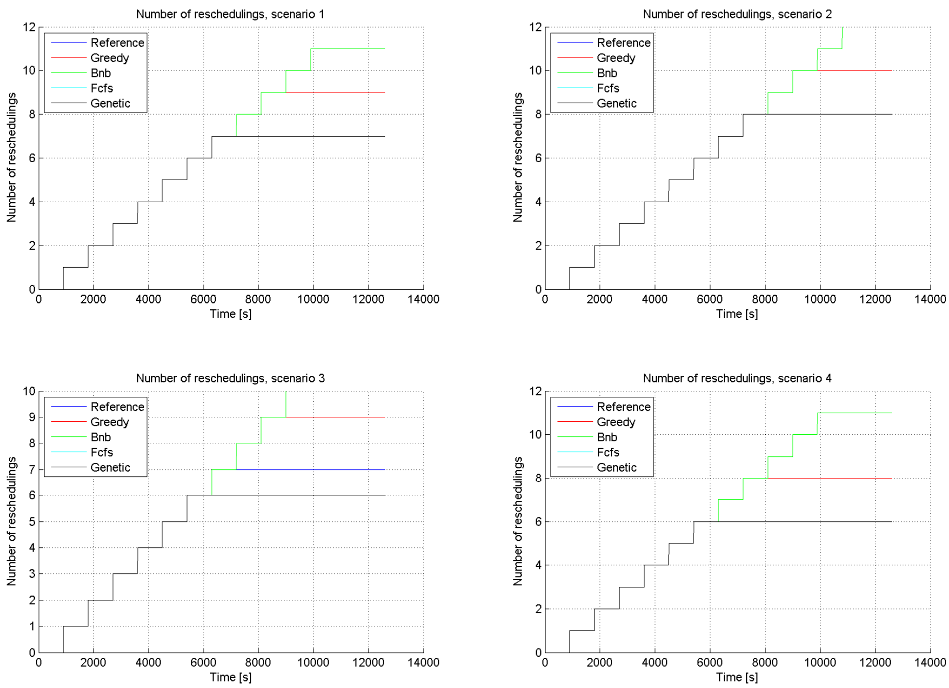

Figure 5.5: Number of rescheduling operations in the standard situation

5.3.1.3 Arrival times

In Figure 5.6, the arrival times of all aircraft are shown. Each moment an aircraft arrives at the runway is marked with a cross. The ideal arrival times are shown in magenta as algorithm 0. Algorithm 1 up 5 five are respectively the reference, greedy, branch and bound, first come first served and genetic algorithm. This figure gives insight in how the arrival times are spread over the while simulation timespan. Scenario 1, 2 and 3 show that the reference algorithm schedules some aircraft before their ideal arrival time to reduce the total delay of all aircraft. The other algorithms do not schedule aircraft before their ideal arrival time. Because of the too small separation margins between the ideal takeoff times, all algorithms fill the gaps in the ideal takeoff sequence to let the remaining aircraft take off.

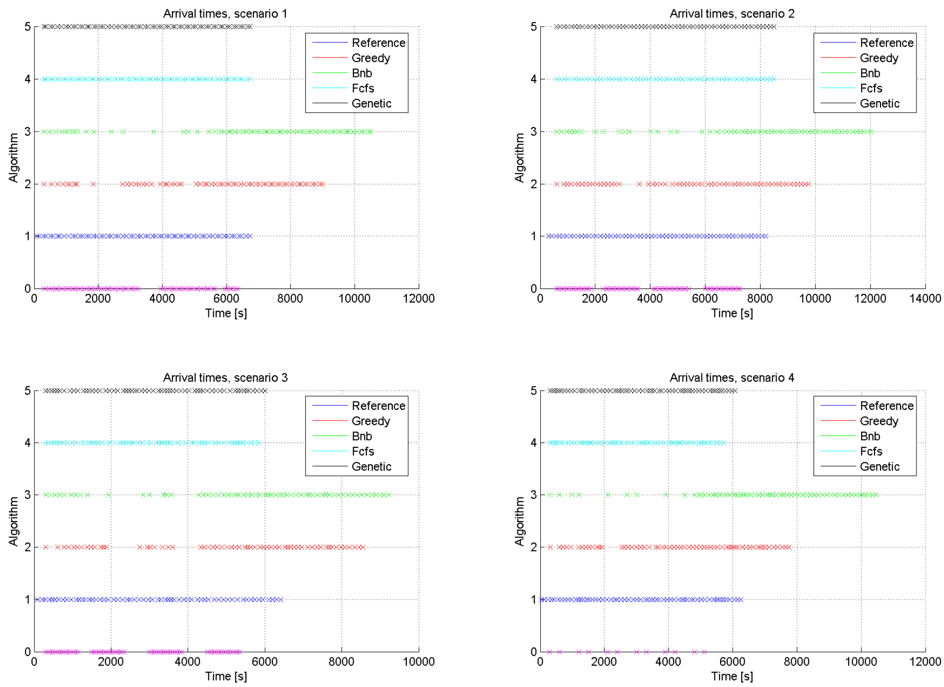

Figure 5.6: Arrival times in the standard situation

The behavior of the branch and bound algorithm is in all scenarios different from the behavior of the other algorithms. In particular in scenario 2 and 4 this algorithm shows that until the last ideal takeoff time of all aircraft is reached, only a few aircraft are scheduled by the branch and bound algorithm. After this time, all remaining aircraft are scheduled and are spaced more closely than before this moment. This can be explained by the fact that this algorithm tries to minimize the runway occupancy. This algorithm searches for optimal heavy/large sequences. The drawback of this optimization is a significant delay increase for most aircraft.

5.3.1.4 Taxiway throughput

The throughput graphs in Figure 5.7 show the throughput of the complete system of taxiways. A throughput of 1 means that during the sixty minutes before the number of aircraft that ware ready to start taxiing is equal to the number of aircraft taking off at the runway. Throughputs below 1 mean that more aircraft are entering the taxiways than that there are leaving the taxiways (taking off). In this situation the number of aircraft that are involved in the simulation is increasing and thus the possibility of disruptions will increase. These graphs underline the conclusion drawn in section 5.3.1.3. In scenario 4 it is most clear that for the greedy and branch and bound algorithm, there is a low throughput until approximately t=9000. After this moment, all aircraft are sent away and no new aircraft are entering the system anymore. This explains the fact that the throughput will increase to values above 1.

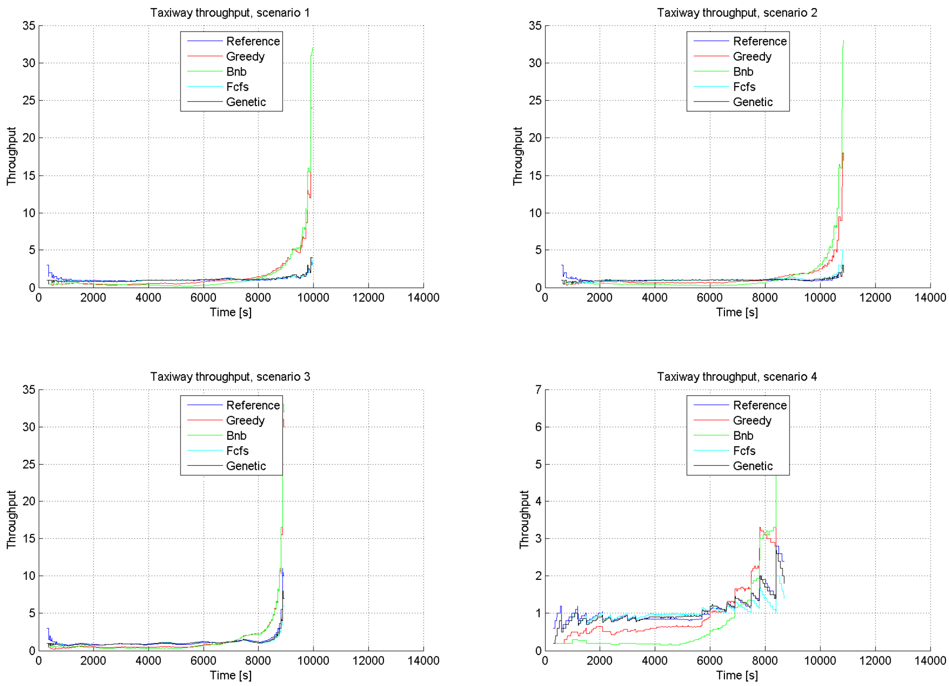

Figure 5.7: Taxiway throughput in the standard situation

5.3.1.5 Runway throughput

The runway throughput as shown in Figure 5.8 shows the amount of aircraft that are departing from the runway per hour. This throughput is limited by the required wake vortex separation and the miles-in-trail separation times between the departed aircraft.

According to Table 2.1, the biggest required separation between any combination of aircraft used in these scenarios is 148 seconds. The least required separation is 64 seconds. Therefore, when no gaps occur in the runway takeoff schedule, the runway throughput should vary between 24 and 56 aircraft per hour. This holds for scenario 1, 3 and 4 in Figure 5.8. In these scenarios the throughput for the greedy and branch and bound algorithms are lower. This is in line with the extra delay of these algorithms as discussed before. In scenario 2, the runway throughput does not get above thirty aircraft per hour. This limit is caused by the minimum required wake vortex separation of 120 seconds.

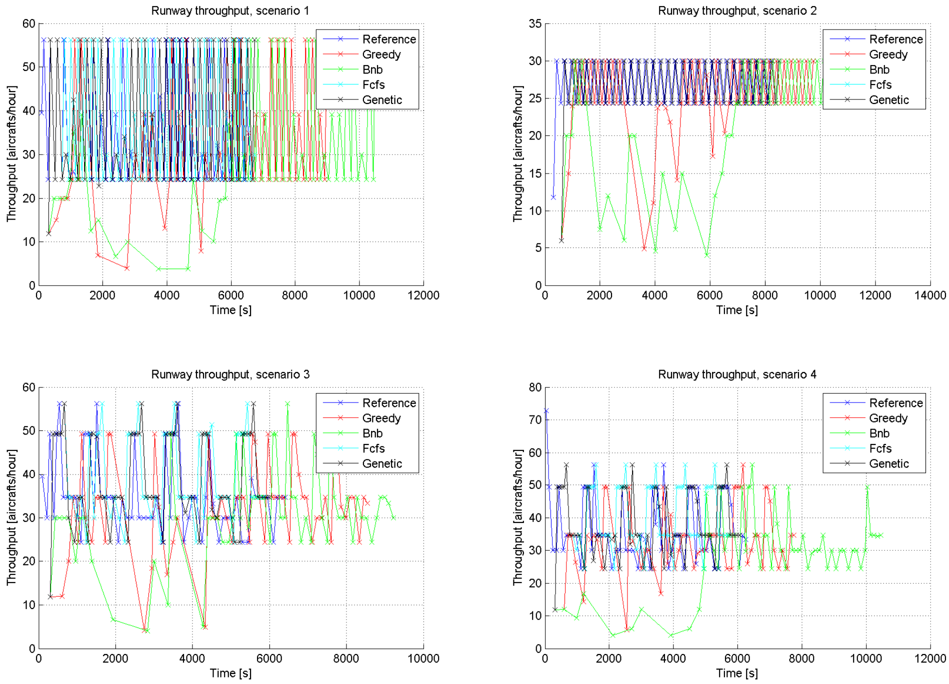

Figure 5.8: Runway throughput in the standard situation

5.3.1.6 Aircraft departed outside their slots

Figure 5.9 shows the amount of aircraft that departed outside their departure slots. At some time instances the amount of aircraft departing outside their slots is decreasing. This is caused by the fact that the amount of aircraft departing outside their departure slot is based on the estimated departure time of the aircraft. This estimated departure time is included in the schedule generated by the algorithms. If an aircraft is still taxiing, but its estimated departure time is outside its slot, it is counted as departed outside its slot. A rescheduling operation can change this schedule and thus it can occur that an aircraft that was initially scheduled outside of its departure slot is now scheduled within its slot again and thus the amount of aircraft departing outside their departure slot in Figure 5.9 is decreasing.

The performance of the reference and the genetic algorithm is best in scenario 2. In this scenario both algorithms manage to schedule all aircraft within their slot boundaries. The performance of the other algorithms in this situation is comparable to the performance of these algorithms in the other situations, or even worse, like the performance of the first come first served algorithm. So in situations where the margins are tight and both miles-in-trail and wake vortex separation are limiting factors, the genetic algorithm is a good option to use, in contrary to the first come first served algorithm, for which its performance is worst in these situations. The bad performance of the greedy and branch and bound algorithm is caused by the fact that these algorithms only try to minimize the required separation and do not take care of the desired departure times, as already explained in section 5.3.1.1.

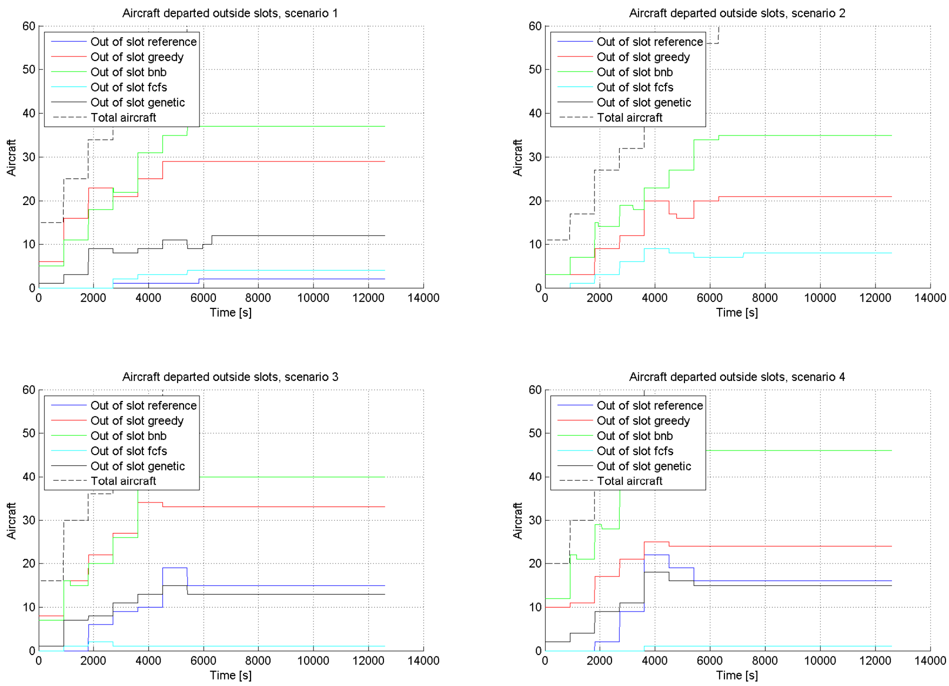

Figure 5.9: Amount of aircraft departed outside slots in the standard situation

5.3.2 Change of step size parameter

The next step in the evaluation process is to change the input parameters one by one to evaluate the effect of these changes on the performance of the algorithms. The first parameter to be changed is the parameter that defines the step size of the simulation. Besides the default step size of five seconds, a step size of sixty seconds is used.

Looking at the delay outputs, it is shown that the delay of the aircraft increases with approximately 55 seconds per aircraft when the step size is increased from five to sixty seconds. This is shown in Figure 5.10 and Figure 5.11. The solid lines represent a scheduling window of five seconds and the dashed lines a scheduling window of sixty seconds. This increase is, as expected, caused by the reduction of sample moments. This causes a reduction of moments where the behavior of the aircraft can be corrected. Therefore the aircraft is on average corrected at a later moment in time. This also holds for the moment where the aircraft should change its speed. If an aircraft should change its speed at t, but the first sample moment after t is at t + 55, the speed of an aircraft will not be corrected before t + 55, which is 55 seconds too late.

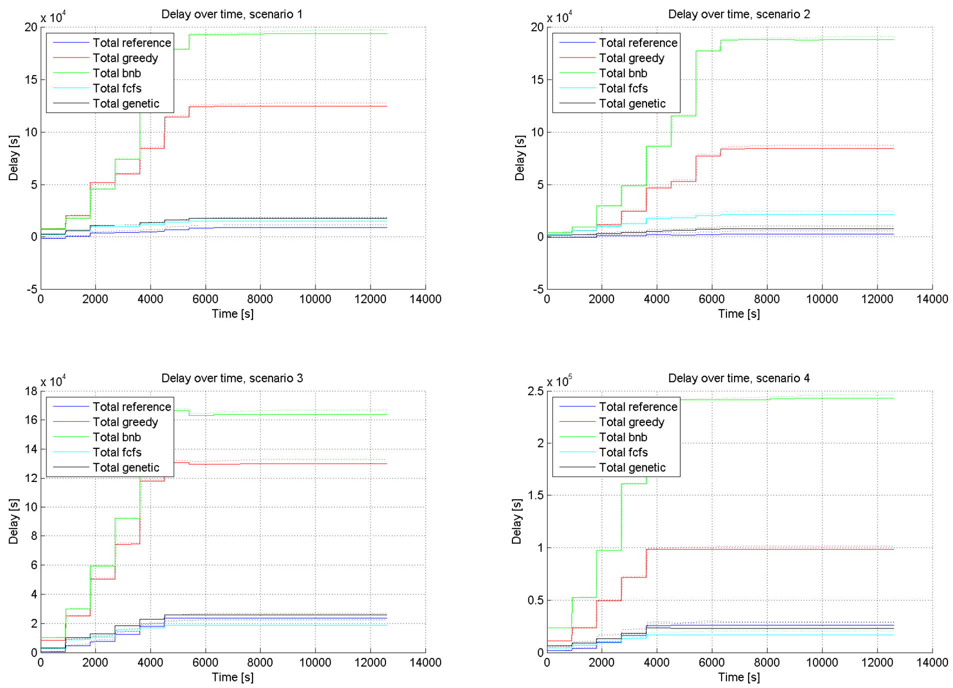

Figure 5.10: Total aircraft delay for varying step size

The step size is not an input for the scheduling algorithms. Therefore changing this parameter does not directly influence the scheduled arrival times, runway throughput and the runway occupancy. However, changes in these parameters can still occur because disturbances are corrected at a later moment in time due to a bigger step size. The increase in delay, as discussed before, also increases the number of aircraft departed outside their slot, as can be seen in scenario 1 and 3 in Figure 5.12.

Having discussed this output parameter, it can be concluded that a change of simulation step size does not have a significant influence on the output results. However, it is expected that in case when disturbances occur, a smaller step size makes the simulation detect and correct disturbances earlier and thus reduces the total delay and improves the efficiency of the algorithms. On the other hand, the time needed to complete the simulation will decrease significantly when the step size is increased. The reduction in required simulation time is almost directly proportional to the increase of the step size.

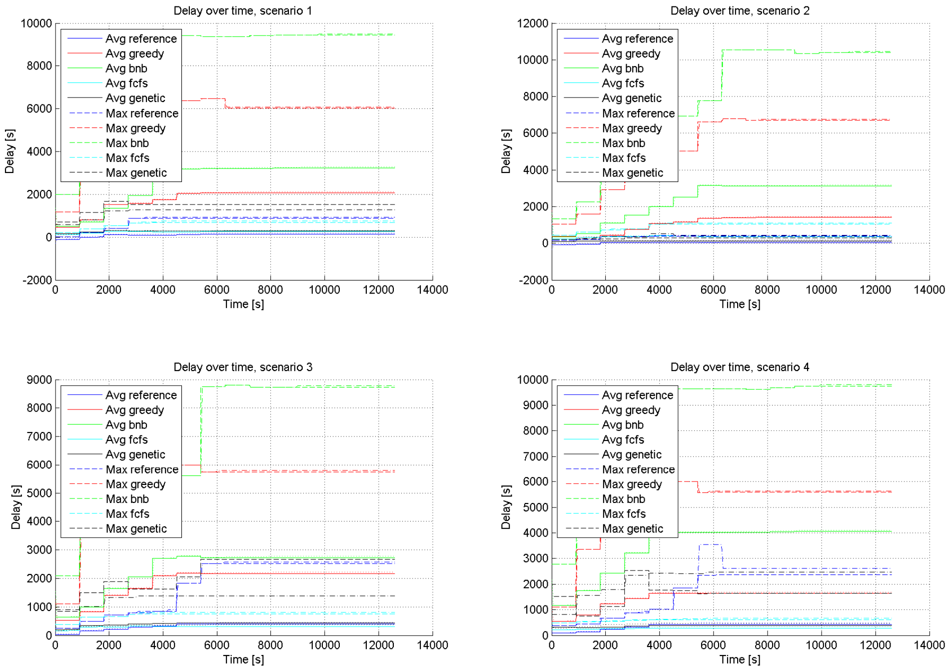

Figure 5.11: Individual aircraft delay for varying step size

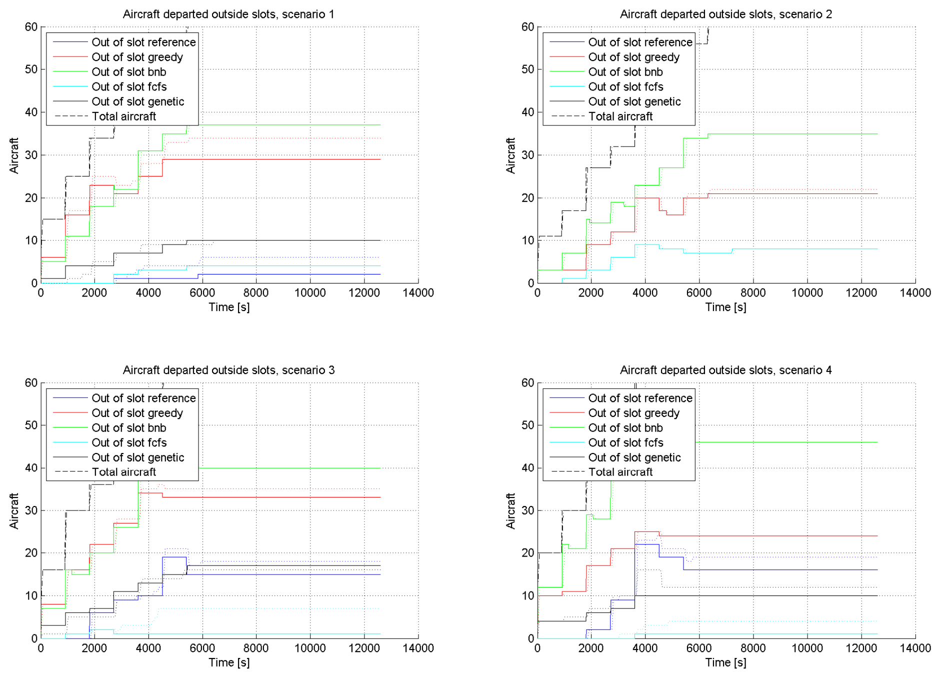

Figure 5.12: Aircraft departed out of their slots for varying step size

5.3.3 Change of scheduling window

The next parameter under investigation is the parameter which defines the scheduling window. As mentioned before, the scheduling window will be varied from five seconds (the step size used during this simulation) to 120 minutes. A scheduling window of five seconds means that a new schedule is calculated every time step. As a result the algorithm does not look ahead more than five seconds and does not schedule an aircraft before it is ready to start taxiing.

When looking at the delay graphs in Figure 5.13, a scheduling window of five seconds is shown as a solid line, a window of fifteen minutes as a dotted line and a window of two hours as a dashed line. It is expected that a bigger scheduling window would make the algorithms look more forward and thus would enable the algorithm to better deal with future problematic situations. The algorithm would be able to reduce the disturbances.

However, this statement does not hold for all situations. Looking at the reference and first come first served algorithm, for scenarios 1, 3 and 4 the largest scheduling window reduces the delay the most, but for scenario 2 the largest scheduling window causes the most delay. For the greedy and branch and bound algorithms, the delay in case of a scheduling window of 120 minutes is lower than the delay for a fifteen minute window, but the delay of the five seconds window is even lower in all scenarios. The result of the simulation with the genetic algorithm is completely opposite to the expectations. The 120 minute window causes the largest delay and the five seconds window the least delay.

The behavior of the genetic algorithm can be explained by the way this algorithm calculates the optimal departure schedule. When the departure window is made larger, more aircraft are involved in the scheduling process. When this amount of aircraft is too large, there are so many possible schedules that the algorithm is not able to find the global optimum anymore. Instead of that, a local optimal departure schedule is generated which can be totally different than the global optimum solution. The good performance of the five seconds window in case of the greedy and branch and bound algorithm can be explained by the fact that because of a small scheduling window, only a single new aircraft needs to be added to the schedule during each rescheduling operation. This limits the amount of possible solutions and therefore it is easier to calculate the optimal solution.

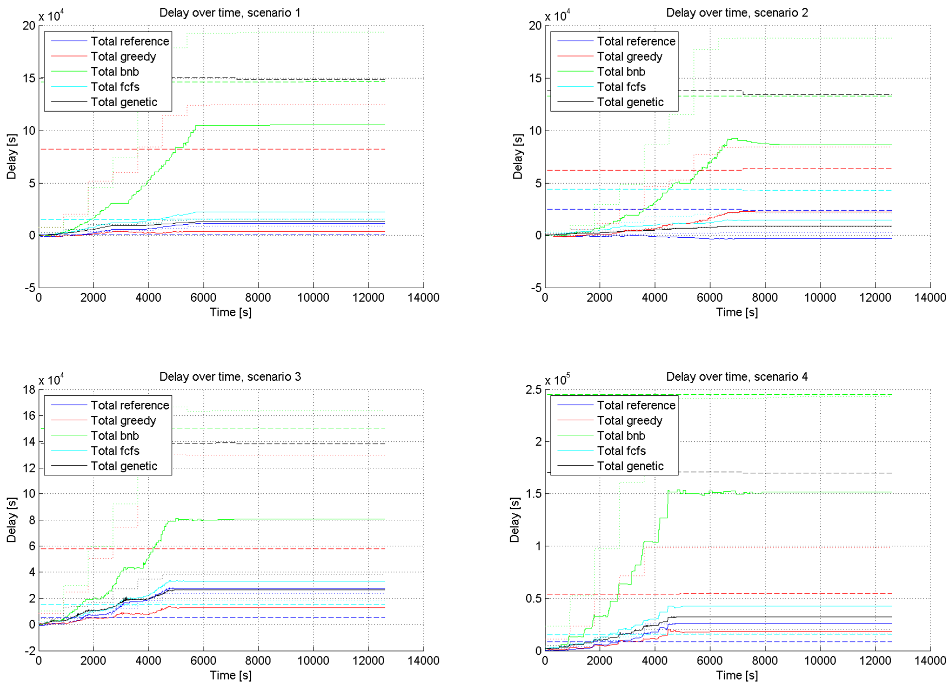

Figure 5.13: Total aircraft delay for varying scheduling windows

The number of rescheduling operations for a scheduling window of fifteen minutes (the solid lines) and 120 minutes (dotted lines) are shown in Figure 5.14. The only rescheduling operations shown in this figure are the periodic rescheduling operations at the start of each new scheduling window. This means that the number of extra rescheduling operations, and thus the robustness of the calculated departure schedule is independent of the size of the scheduling window.

Figure 5.15 shows the arrival times of all aircraft. From bottom to top the scheduling windows of 5 seconds, 15, 30, 60 90 and 120 minutes are shown for each algorithm. These arrival times underline the statement that using large scheduling windows, the genetic algorithm is not able to determine an optimal departure schedule anymore. Figure 5.15 shows that the arrival times of the aircraft using the reference and first come first served algorithm does hardly change when the scheduling window is increased. Using the other algorithms, more gaps occur between the arrival times of the aircraft, causing more delay and a decrease in the efficiency of the algorithms.

It can be concluded that a bigger scheduling window will have a negative result on the effectiveness of the algorithms, except for the reference and first come first served algorithms. These algorithms are not influenced by the size of the scheduling window. A smaller scheduling window implicates more rescheduling operations. This makes the algorithm computationally more intensive and the schedules of the aircraft should be updated more often. In this sense, the scheduling window should be taken as large as possible. A scheduling window of fifteen minutes will be the best compromise between speed and efficiency.

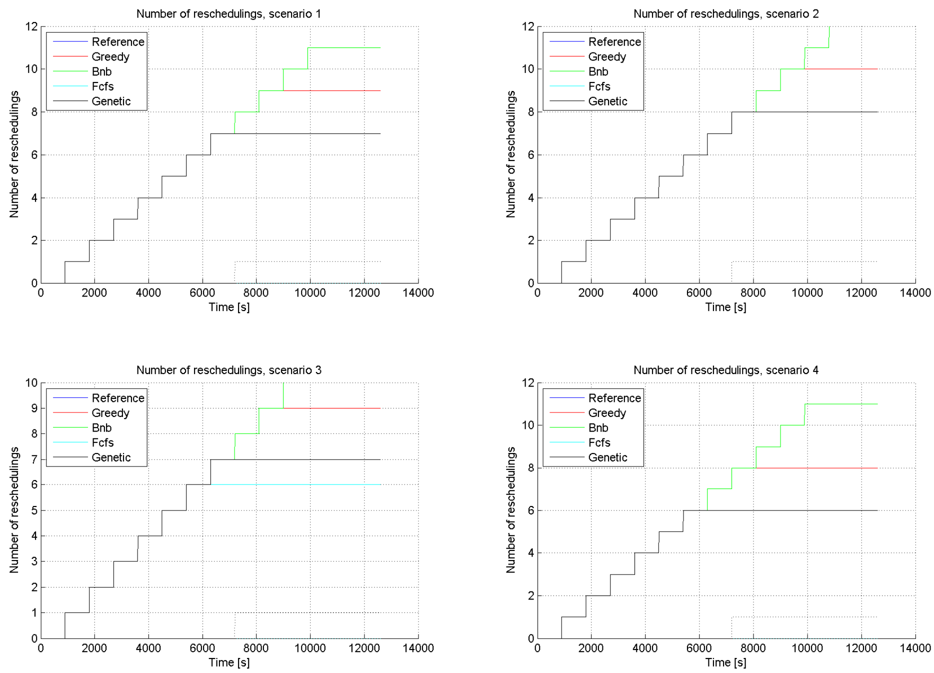

Figure 5.14: Number of rescheduling operations for varying scheduling windows

5.3.4 Change of disturbances parameter

Up until now, disturbances were not taken into account. It is expected that the added value of a scheduling algorithm will be the highest when aircraft are disturbed. Therefore the disturbance input can be seen as the most important input parameter to evaluate. Four disturbance severities are investigated; no disturbance, and average disturbances of 6:15, 12:30 and 25:00 minutes. All combinations of gate delay and taxi delay are simulated.

The total delay of all algorithms using an average disturbance of 6:15, 12:30 and 25:00 minutes is shown in Figure 5.16, Figure 5.17 and Figure 5.18. No disturbance is plotted as a solid line, only gate disturbance a dotted line, only taxi disturbance as a combination of dashes and dots, and both gate and taxi delay is plotted as a dashed line.

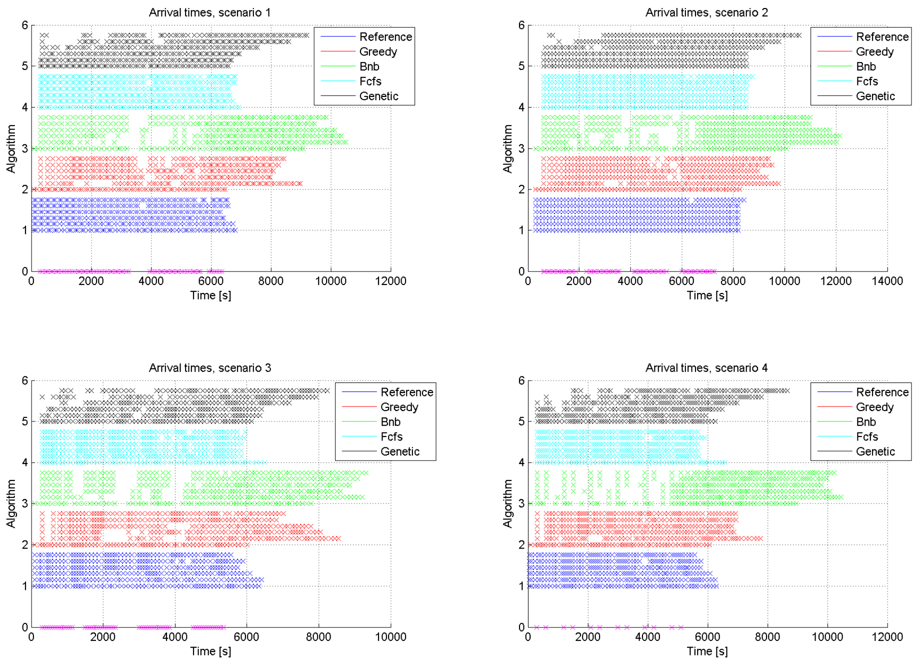

Figure 5.15: Arrival times for varying scheduling windows. The scheduling windows from bottom to top are 5 seconds, 15, 30, 60 90 and 120 minutes

The first remarkable detail when looking at the delay graphs is the contribution of the individual disturbance components to the total aircraft delay. For all algorithms and all scenarios it holds that the delay graph for the case where taxi delay is taken into account is almost identical to the delay graph for the case where no delays were taken into account. The contribution of the taxi delay to the total delay is less than 10%, even when an average disturbance of 25 minutes is used. Only for the greedy algorithm the contribution of the taxi delay can be more than 10%, but still in the order of 10 to 20%. The small contribution of the taxi delay can be explained because of the relative short taxi distances. Because of these short distances, a speed disturbance will last for a short time and therefore the effect of this disturbance stays small.

Looking at the gate delay, it is clear that this delay has the highest contribution to the total delay of the aircraft. Because of the small contribution of the taxi delay, the total delay when only gate delay is taken into account, is almost identical to the total delay when both gate and taxi delay are taken into account. Therefore trying to reduce the gate delay will have more influence on the performance of the algorithms than trying to reduce the taxi delay.

The effect of reducing the disturbances is not directly proportional to the reduction of the delay. Figure 5.16, Figure 5.17 and Figure 5.18 show us that when the amount of disturbance is doubled, the amount of delay only increases with 50%. So to reduce the delay by a factor of two, the disturbances must be reduced with a factor of four. This roughly holds for all algorithms, except the reference and the first come first served algorithm. For these algorithms the total delay is directly proportional with the amount of disturbance.

Looking at Figure 5.18 and taking into account the relations mentioned above, the genetic algorithm is the most suitable algorithm for scheduling aircraft in case of high disturbances. The performance of the greedy algorithm is in this situation also better than the reference algorithm.

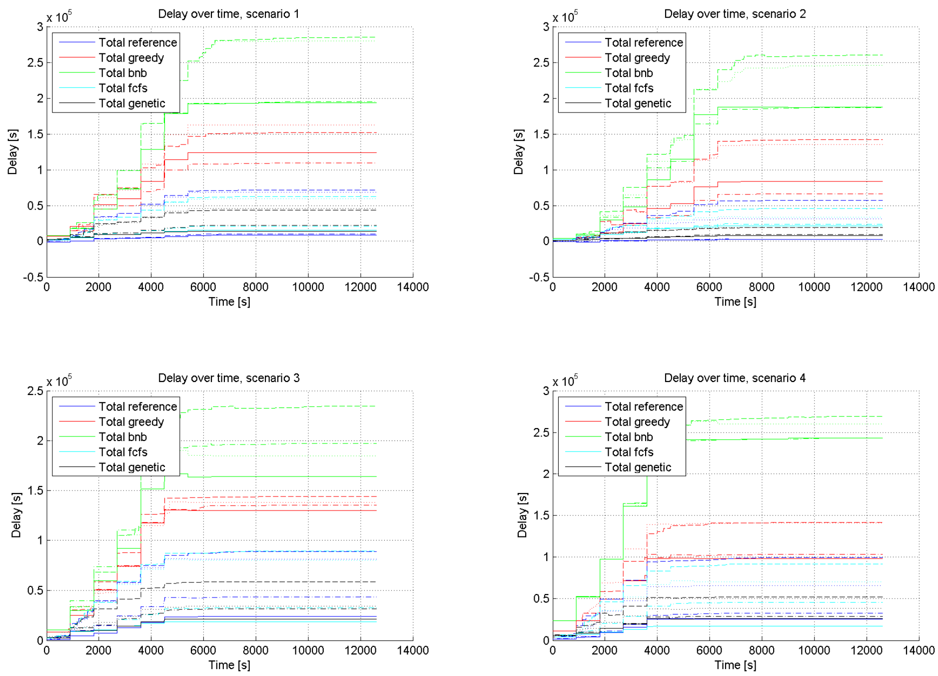

Figure 5.16: Total delay of all aircraft using an average delay of 6:15 minutes

Looking at the moments where rescheduling operations are triggered (Figure 5.19, Figure 5.20 and Figure 5.21, where the situation of no disturbance is plotted as a solid line and both gate and taxi disturbance as a dotted line), no uniform relation can be made up between the disturbances and the number of rescheduling operations. This relation is highly dependent on the algorithm which is used. This can be explained by the different scheduling methods used by the algorithms. Algorithms which schedule bigger margins between the aircraft will make the schedule more robust and thus will need less rescheduling operations.

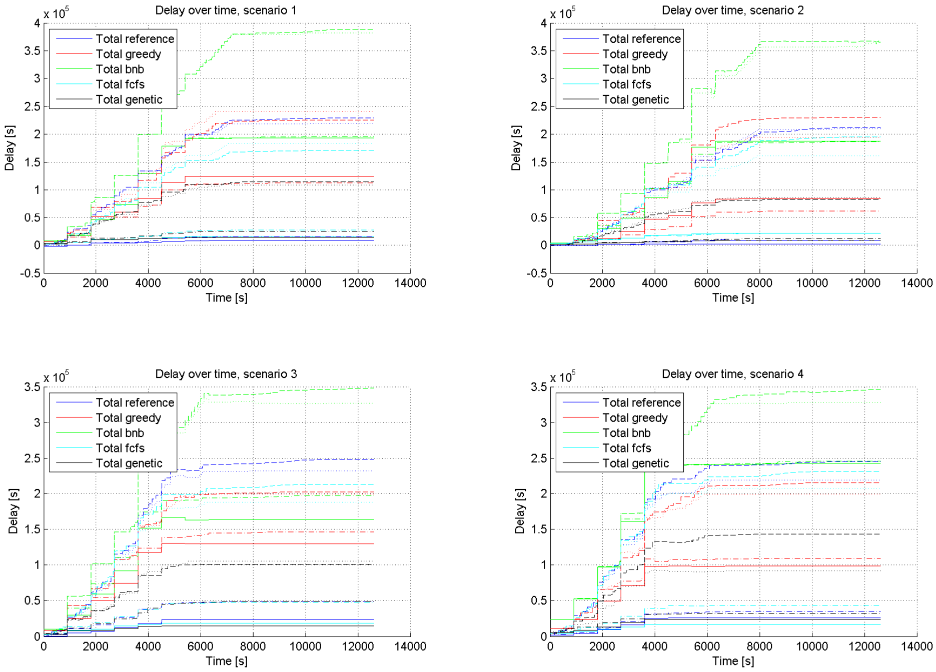

Figure 5.17: Total delay of all aircraft using an average delay of 12:30 minutes

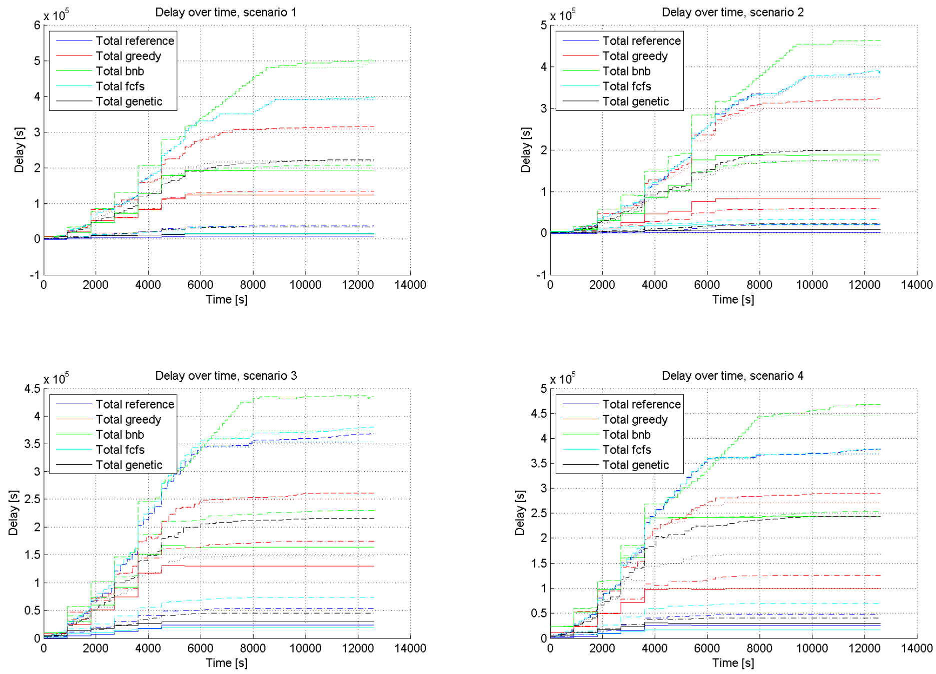

Figure 5.18: Total delay of all aircraft using an average delay of 25:00 minutes

When the number of rescheduling operations is an important factor for determining the efficiency of the algorithms, all algorithms, except the first come first served algorithm, perform better than the reference algorithm. So the robustness of a departure schedule calculated by one of the algorithms is higher than the robustness of the reference algorithm.

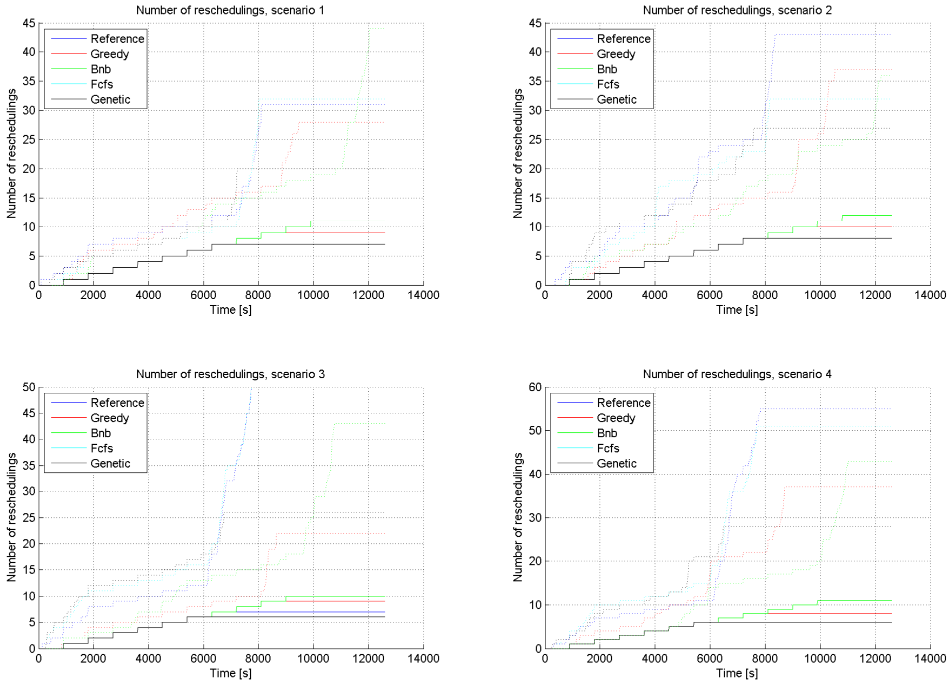

Figure 5.19: Number of rescheduling operations for an average disturbance of 6:15 minutes

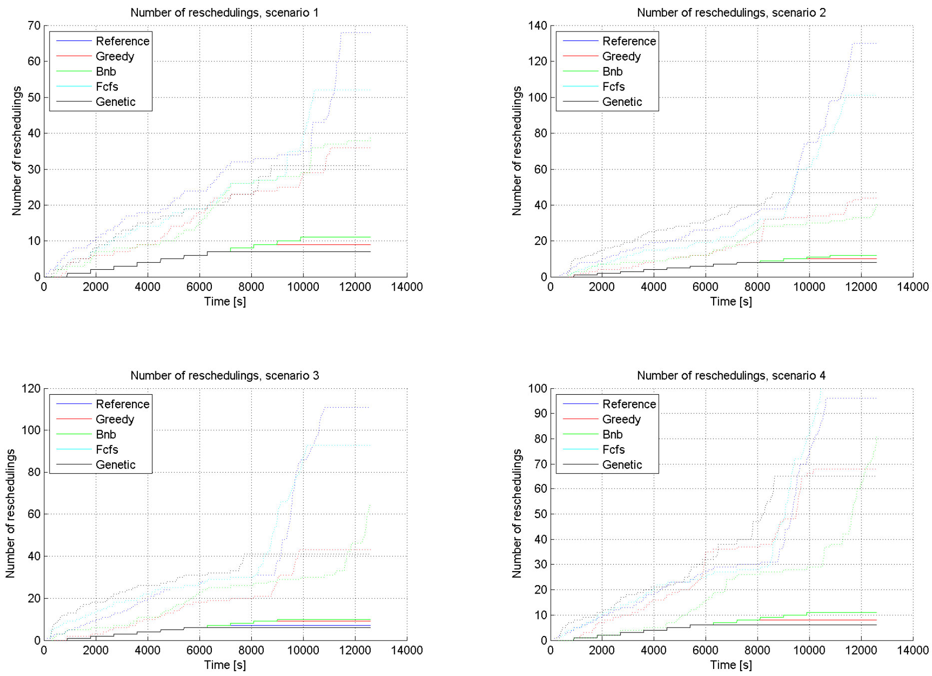

Figure 5.20: Number of rescheduling operations for an average disturbance of 12:30 minutes

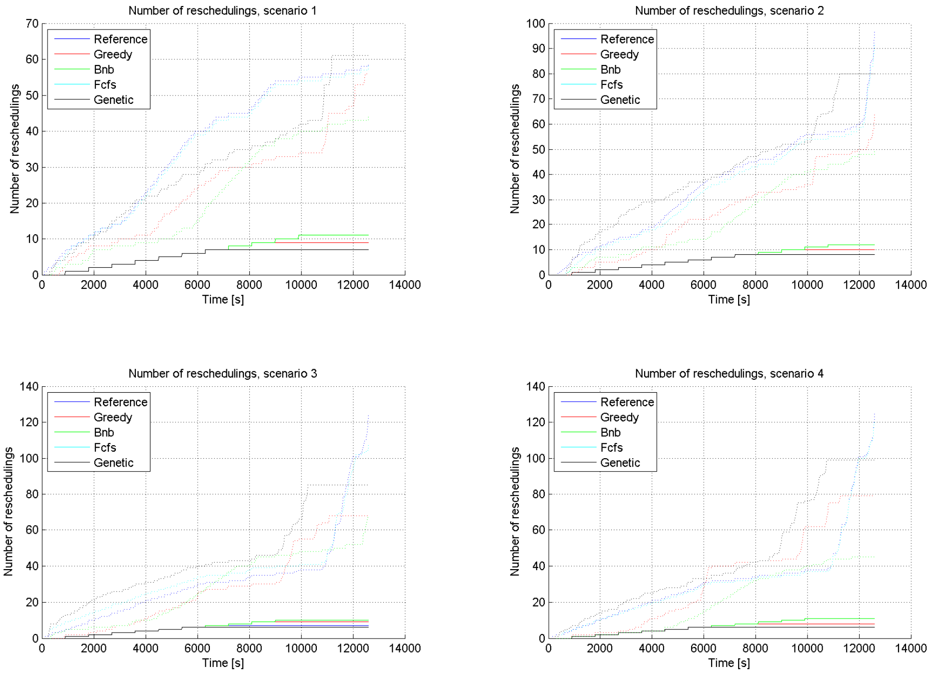

Figure 5.21: Number of rescheduling operations for an average disturbance of 25:00 minutes

The overview of the arrival times of all aircraft confirms the conclusion that the gate delay has the highest contribution to the total delay. Looking at Figure 5.22, Figure 5.23 and Figure 5.24, where no delay, only gate delay, only taxi delay and gate and taxi delay are shown for each algorithm from bottom to top, it is clear that the arrival times hardly change when only taxi delay is taken into account. The plots of the total delay are almost equal to the plots of the simulation where only gate delay is taken into account. The advantage of the genetic algorithm is also made clear by these figures. Especially in the case where big disturbances are used, these graphs show that the genetic algorithm is able to let all aircraft depart in a shorter amount of time than needed by the other algorithms.

Comparing all scenarios and variations in the amount and type of disturbances, it can be concluded that gate delays have the highest impact on the departure schedule. The impact of taxi delays is much less and can be corrected by all algorithms. Taking into account the complete delay, the genetic algorithm provides the best solution for scheduling the aircraft. This algorithm is able to provide the best departure schedule with the least delay and the least aircraft departing out of their slots. The efficiency improvement when using this algorithm is best for large disturbances.

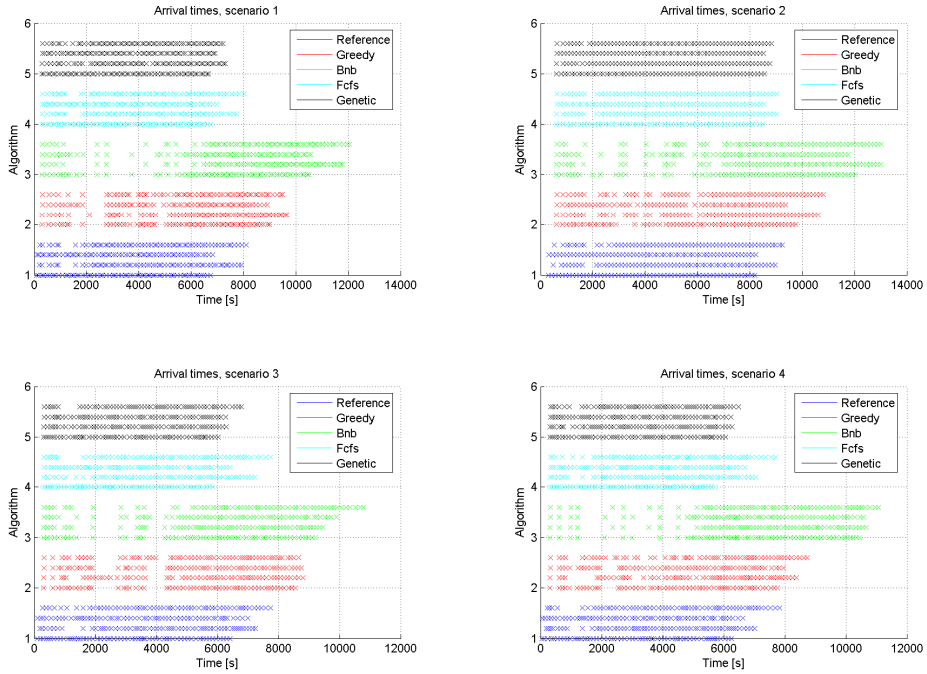

Figure 5.22: Arrival times using an average delay of 6:15 minutes. The delay scenarios from bottom to top are no delay, gate delay, taxi delay and gate and taxi delay

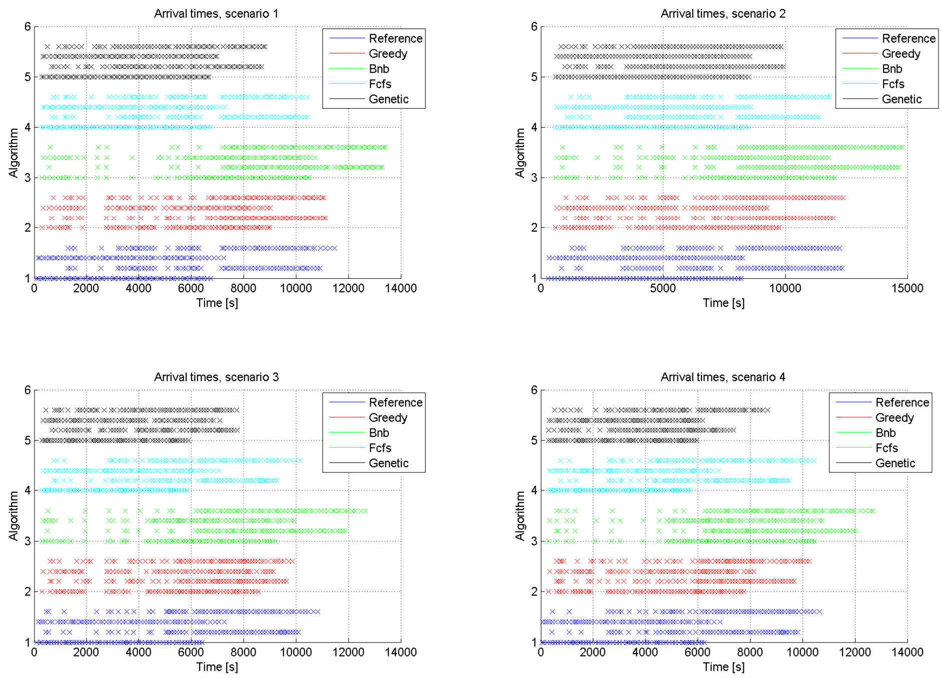

Figure 5.23: Arrival times using an average delay of 12:30 minutes. The delay scenarios from bottom to top are no delay, gate delay, taxi delay and gate and taxi delay

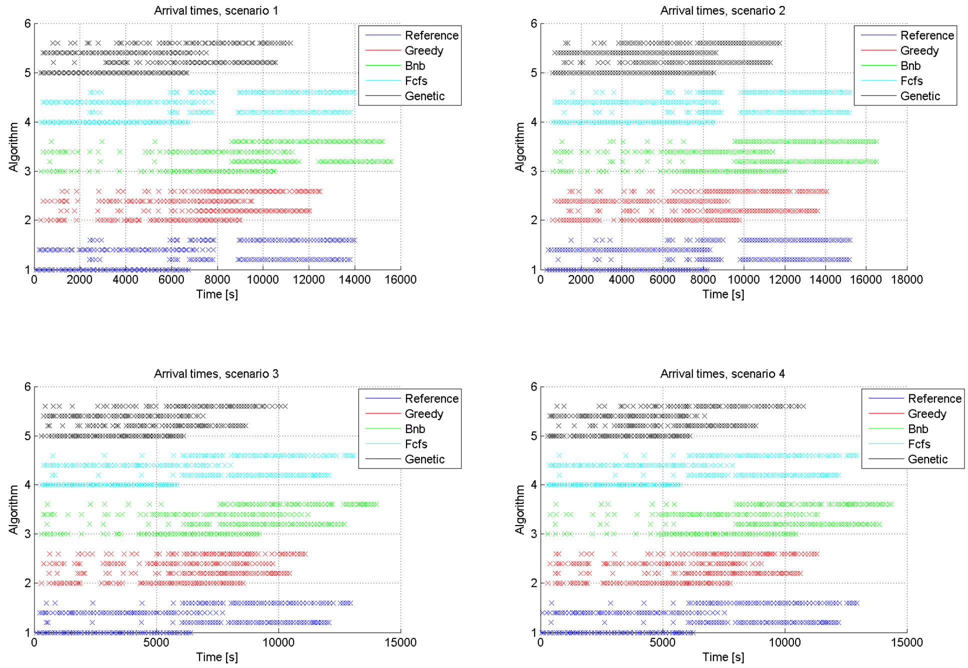

Figure 5.24: Arrival times using an average delay of 25:00 minutes. The delay scenarios from bottom to top are no delay, gate delay, taxi delay and gate and taxi delay

5.3.5 Change of slotboundaries parameter

Changing the size of the departure slot at the runway does not directly affect the departure schedule. A change of the size of the departure slot changes the freedom for the algorithm to change the takeoff time of the aircraft. Nowadays a slot size of -5 and +10 minutes is used at Schiphol airport. A bigger slot size increases the freedom for the algorithm to change the departure sequence and therefore it is expected that a bigger slot size will increase the efficiency of the departure schedule. However, if this slot size could be narrowed without negative consequences, the uncertainty in the departure times of the aircraft can be reduced. Reducing this uncertainty can help to improve other parts of the flight schedule or other processes where the aircraft are involved in.

In Figure 5.25 the total aircraft delay is given for a departure window of -1 +1 minute (dotted line), -5 +10 minute (solid line) and -10 +15 minute (dashed line). Looking at Figure 5.25, it is clear that a smaller departure window will increase the aircraft delay. This effect is the biggest for the branch and bound and greedy algorithms. A larger departure window increases the possibilities for the algorithms to change the aircraft schedule without letting aircraft depart out of their window. Scenario 1 and 2 show that these increased possibilities make the algorithms able to reduce the total delay.

Scenario 3 and 4 show a different behavior. Using a -5 +10 window and the branch and bound algorithm results with these scenarios in a higher delay than when the -1 +1 window is used. For the genetic and reference algorithm the smallest window provides the least delay and the biggest window the most delay. This effect can be explained by the fact that the primary goal of these algorithms is to let all aircraft depart within their departure window. The ideal departure times of the aircraft in scenario 3 and 4 are defined in such a way that the chosen departure windows do not limit the theoretical maximum number of aircraft that is able to depart within their departure slot. Tightening the departure slot limits the number of possible departure sequences. Therefore a more narrow departure window will make it easier for these algorithms to calculate a good departure schedule.

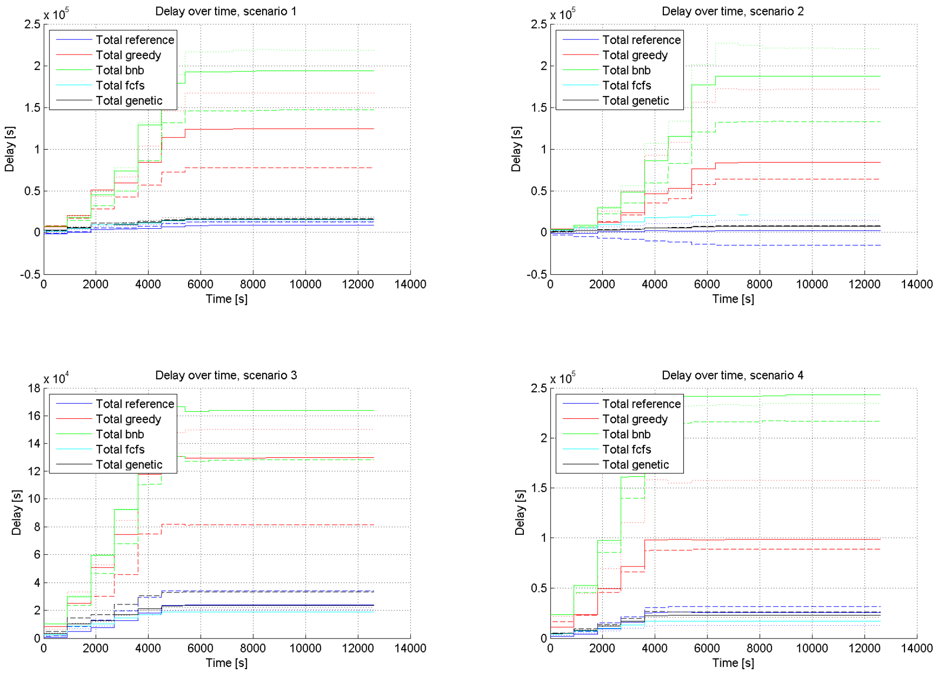

Figure 5.25: Total aircraft delay using varying slot boundaries

The difficulty for the algorithms to schedule all aircraft within their departure slot when a small departure slot is used can be made up via the number of aircraft that departed out of their departure slot, as shown in Figure 5.26. This figure shows that a small departure slot has the disadvantage that many aircraft are unable to depart within their departure slot. A remarkable fact is the good performance of the reference, first come first served and genetic algorithm when large departure slots are used. In case of a departure slot of -10 +15 minutes, not only the total delay of these algorithms is lower than compared to smaller departure slots, but the algorithms also manage to schedule the most aircraft within their departure slots. This can be explained by the way these algorithms determine the departure schedule. For these algorithms the ideal departure time is leading and therefore they will always try to schedule the aircraft within their slot instead of trying to optimize other things like the runway throughput. For narrow departure slots the total delay of these algorithms is still the lowest, but almost no aircraft are able to depart within their slot.

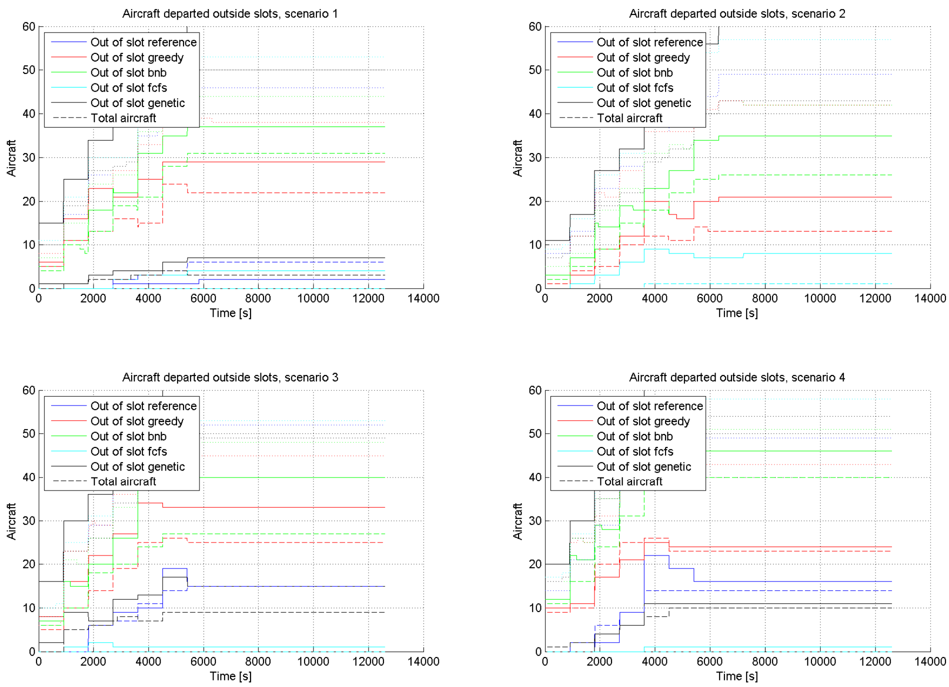

Figure 5.26: Number of aircraft departed out of their departure slot using varying slot boundaries

The number of rescheduling operations is independent of the size of the departure slots, as shown in Figure 5.27. This can be explained by the fact that the size of the departure slots limits the freedom of the algorithms to determine the departure schedule. Once this departure schedule is computed, the size of the departure slot does not have any influence on parameters that can trigger a rescheduling operation. The difference in the amount of rescheduling operations that is shown in Figure 5.27 is caused by periodic rescheduling operations. If all aircraft are already airborne, there is no need for new rescheduling operations anymore.

The plot of the aircraft arrival times in Figure 5.28 validates the conclusions drawn from the delay graphs. The first come first served and genetic algorithms are not influenced by a change of the slot size. The biggest change takes place when the branch and bound or greedy algorithms are used. The reference algorithm tries to schedule an aircraft as early as possible. The figure shows that when the front size of the departure slot is moved to a point earlier in time, all aircraft are scheduled more early. The schedule of the greedy algorithm is changing most due to a change in departure slot size. This algorithm searches among all possibilities for the aircraft which requires the least separation and departs within its departure slot. If the departure slots of these aircraft are made larger, there are more possibilities and thus this algorithm will be able to generate a more efficient departure schedule. Figure 5.28 shows that the bigger the slot sizes, the earlier all aircraft are departed.

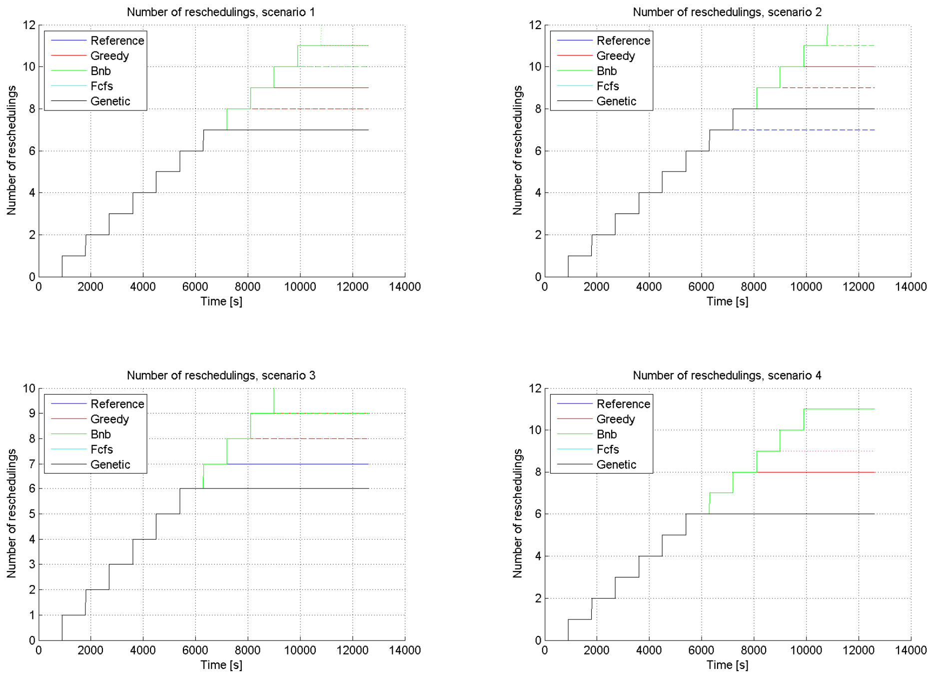

Figure 5.27: Number of rescheduling operations using varying slot boundaries

Summarizing the discussion above, it can be concluded that the optimal size of the departure slots is heavily depending on the algorithm used. The first come first served and genetic algorithms are independent on the size of the departure window. Therefore it is possible to use a very small departure window. The branch and bound and greedy algorithms are much more dependent on the size of the departure slot. If a narrow slot is used, these algorithms are less efficient than the other algorithms. In case of wide departure slots, these algorithms are more efficient.

5.3.6 Change of aircraft based speed parameter

Changing the aircraft based speed parameter introduces another disturbance into the system. This speed restriction is not known by the algorithm and thus not used when calculating the departure schedule. An aircraft based restriction of the maximum speed which is higher than the speed of the class the aircraft belongs to does not introduce problems, but a lower one does. To test the impact of these speed restrictions on the efficiency of the algorithms, multiple disturbances and amount of aircraft that are disturbed are tested. It is expected that a change of the aircraft based speed parameter will decrease the efficiency of the departure schedules and that the algorithms will be able to reduce this effect.

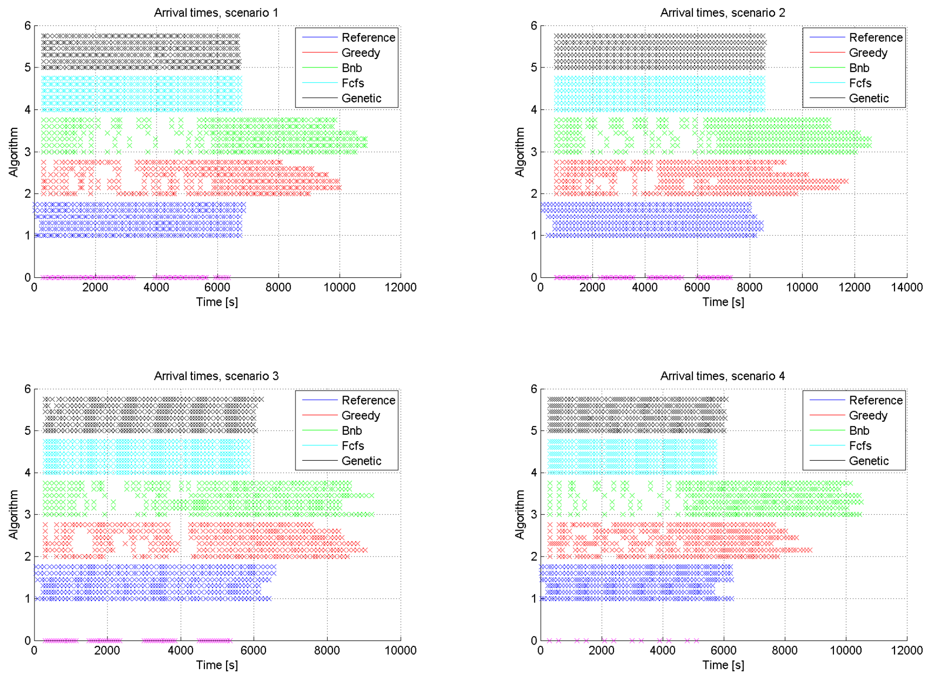

Figure 5.28: Arrival times using varying slot boundaries. The slot boundaries from bottom to top are -5 +10 minutes, -1 +1 minutes, -1 +2 minutes, -5 +5 minutes, -10 +10 minutes and -10 + 15 minutes

The first scenario under investigation is a disturbance of the aircraft based speed restriction at 10% of the aircraft (six aircraft). The delay in this scenario is shown in Figure 5.29, where the disturbance of 90% of the original speed (the aircraft based speed is reduced to 10% of its original speed) is shown as a dotted line, a disturbance to 50% as a dashed line, and no disturbance as a solid line. A remarkable result is that only very severe disruptions (an aircraft based speed that is only 10% of the class based speed) have a noticeable influence on the aircraft delay. For scenario 1, 2 and 4 the simulations with disturbances up to 50% are almost as efficient as the simulation without disturbances. The disturbances of the aircraft based speed parameter of 90% of the class speed cause an increase of the disturbance varying from 30 up to 500%, except for the branch and bound algorithm. For this algorithm increase of the disturbance is much lower. This algorithm has a high delay when no aircraft are disturbed, but this delay includes a margin against unforeseen disturbances (a margin to make the schedule more robust).

A remark must be placed by the conclusion that disturbances up to 50% of the aircraft based speed constraints almost cause no extra delay. This is not completely achieved by the buffer the algorithms create against disturbances. The taxi distances and thus the taxi times used in this simulation are relatively short. A disturbance of 50% sounds like a big disturbance, but if the original taxi time is only sixty seconds, the delay introduced by this disturbance is no more than thirty seconds. The effect would increase when longer taxi times and distances are used.

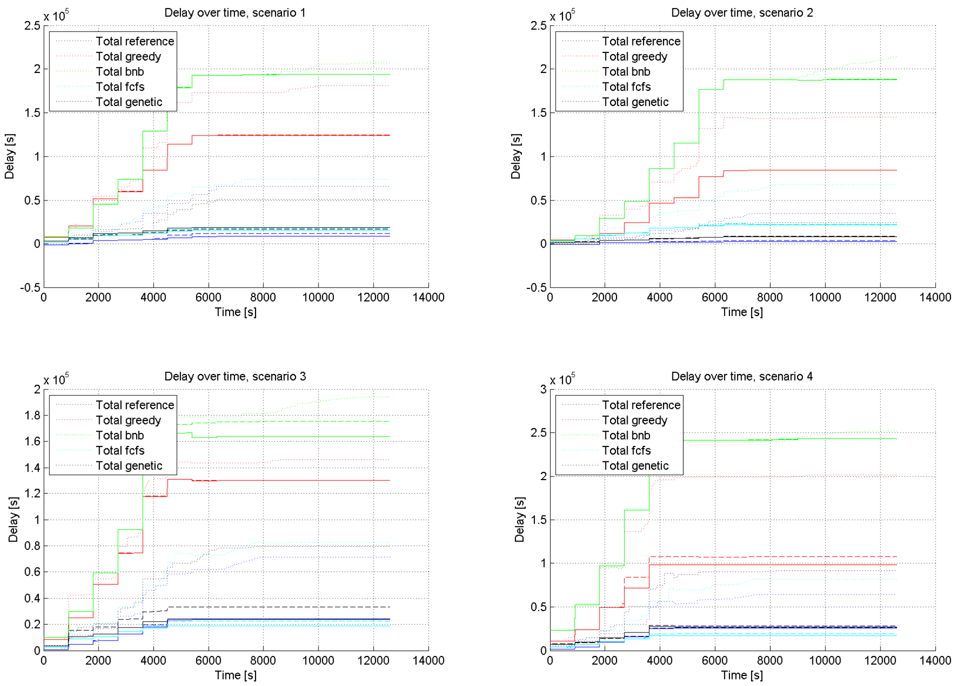

Figure 5.29: Total delay when the aircraft based speed constraints of 10% of the aircraft are disturbed

When 20% of the aircraft are influenced by the disturbance (Figure 5.30), the result is similar. Disturbances up to 50% of the aircraft based speed only have a small influence on the total delay of the schedule. The delay when aircraft based speeds of 10% of the original speed (a disturbance of 90%) are used is bigger when more aircraft are disturbed. This is as expected because more severe disturbed aircraft will cause more extra delay and will influence more other (undisturbed) aircraft.

Looking at the number of rescheduling operations in Figure 5.31, the most remarkable aspects are the rescheduling bursts which occur at some time instances. As expected when looking at the delay graphs, these rescheduling bursts occur mainly when the speed disturbance is 90%. These extreme speed disturbances and the fact that these disturbances cannot be predicted by the algorithms result in many rescheduling operations after each other. In this situation, a disturbed aircraft is rescheduled, but immediately disturbed again, causing a new rescheduling operation, etc. There is no direct relation between the number of rescheduling operations and the amount of aircraft disturbed, as can be seen by comparing Figure 5.31 and Figure 5.32. This can be explained by the fact that the disturbance for an individual aircraft is equal in both situations, so an aircraft will not trigger a rescheduling operation earlier when more aircraft are disturbed. However, when there are more aircraft disturbed, the chance that one of the aircraft will trigger a rescheduling operation will increase. Therefore more rescheduling bursts occur when 20% of the aircraft are affected.

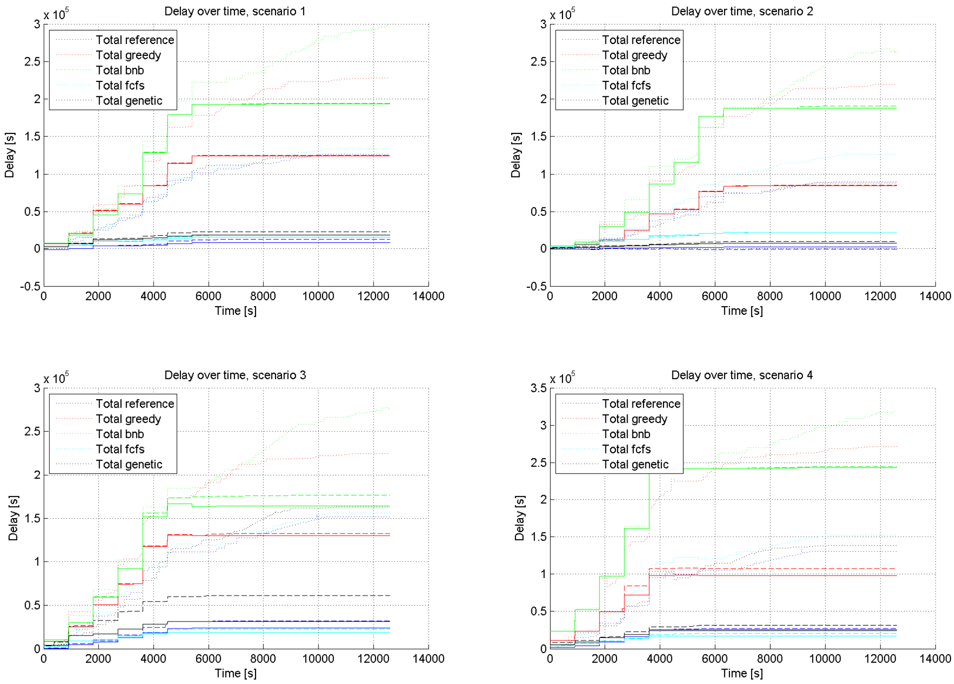

Figure 5.30: Total delay when the aircraft based speed constraints of 20% of the aircraft are disturbed

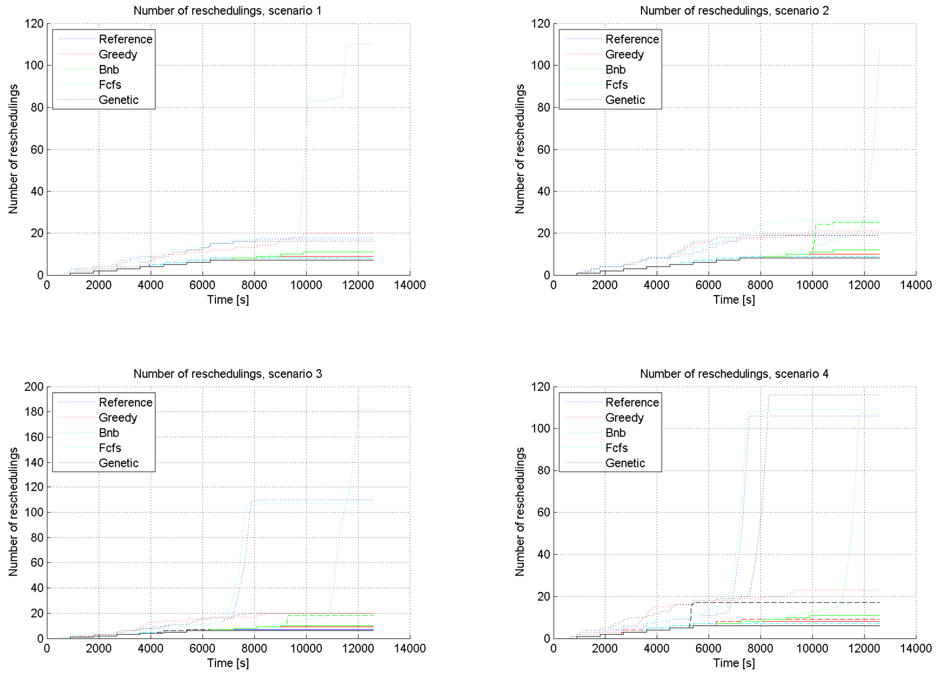

Figure 5.31: Number of rescheduling operations when the aircraft based speed constraints of 10% of the aircraft are disturbed

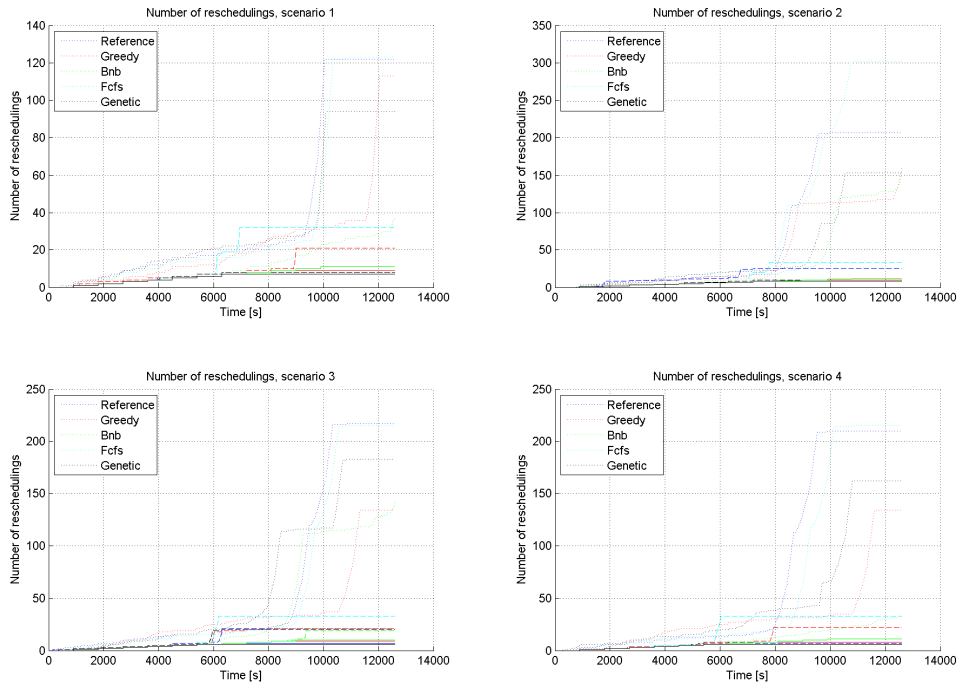

Figure 5.32: Number of rescheduling operations when the aircraft based speed constraints of 20% of the aircraft are disturbed

The amount of aircraft that were unable to depart within their departure slot is shown in Figure 5.33 and Figure 5.34. Looking at these figures, it can be concluded that the difference in the number of aircraft that were unable to depart within their departure slots in the case of no disturbance and 90% disturbance is the highest for the reference and the first come first served algorithm. Especially in scenario 1 and 3 these algorithms do not succeed in letting the undisturbed aircraft continue their way without disturbances. This can be explained by the fact that these algorithms try to maintain the original departure sequence as much as possible, so when an aircraft is delayed, the aircraft behind will have an increased chance to also get delayed than compared to the other algorithms. In every situation more aircraft are affected than the disturbed aircraft only, but the genetic algorithm succeeds best in trying to reduce the effect as much as possible.

Summarizing all the results of the simulation of varying the aircraft based speed parameter, it can be concluded that when the taxi times are relatively short, all algorithms are robust against disturbances of the aircraft based speed parameters. Only very severe disturbances of 90% have an influence on the departure schedule. There are almost no differences between all algorithms, but the genetic algorithm has a slightly better performance.

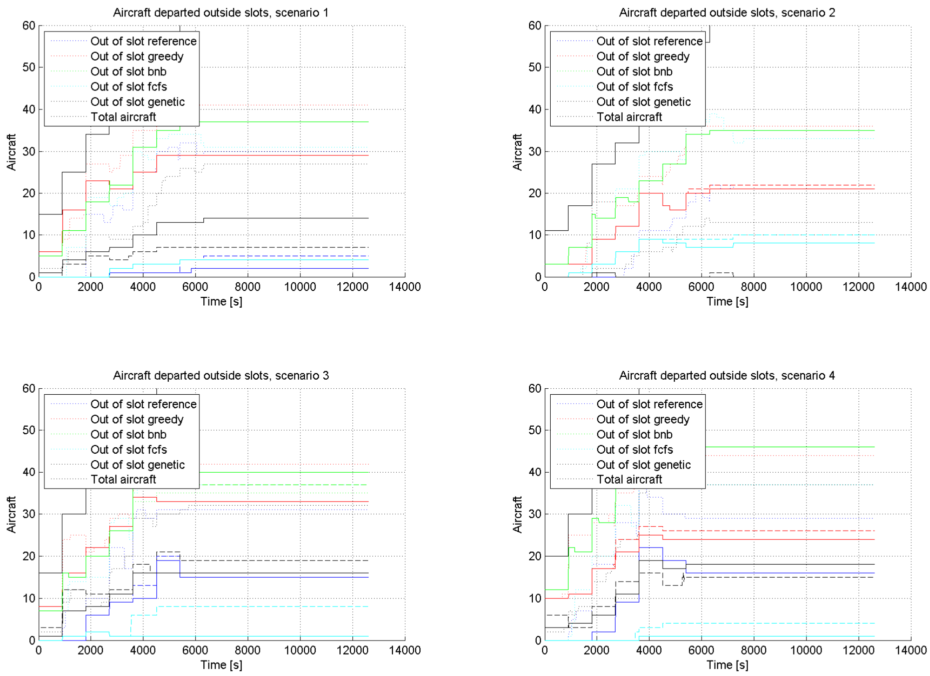

Figure 5.33: Number of aircraft departed out of their slots when the aircraft based speed constraints of 10% of the aircraft are disturbed

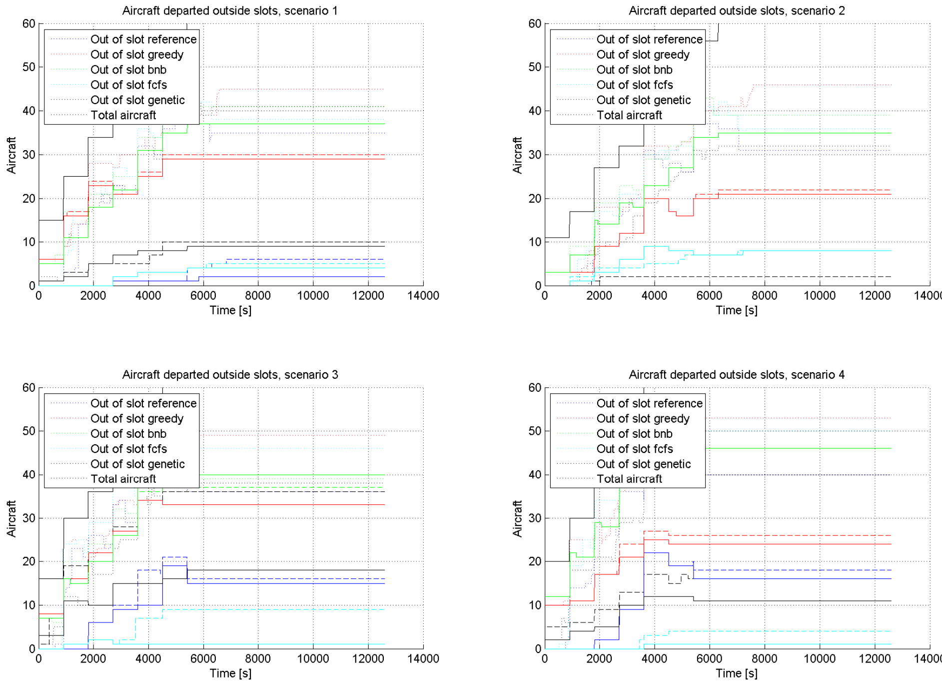

Figure 5.34: Number of aircraft departed out of their slots when the aircraft based speed constraints of 20% of the aircraft are disturbed

5.3.7 Change of rttaxi parameter

A smaller ready to taxi margin reduces the freedom for the algorithm to schedule the departure of the aircraft. In the extreme situation, the time between the moment when the aircraft is ready to taxi and the desired takeoff time is equal to the required taxi time. In this situation the aircraft are unable to depart before their desired takeoff time, so the only options are no delay or extra delay.

Looking at the delay graph of this simulation (Figure 5.35, where ready to taxi margins of zero, ten, twenty and thirty minutes are shown as respectively a solid, dotted, combination of dashes and dots and a dashed line), the result is remarkable. It was expected that when the ready to taxi margin is made larger, the delay would decrease, because the algorithm would be able to calculate a more efficient departure schedule due to the increased scheduling freedom. This is true for the reference, genetic and first come first served algorithms. The greedy and branch and bound algorithms show a different behavior. Looking at the greedy algorithm, when the ready to taxi margin is made larger, the delay for this algorithm first decreases, but then increases when the margin continues to grow. This also holds for the branch and bound algorithm in scenario 3 and 4, but for the other scenarios the delay immediately increases when the ready to taxi margin is increased. So the more freedom for these algorithms to schedule the aircraft, the less efficient the output will be. The explanation for this phenomenon is that these algorithms use a recursive loop to reach a local optimum. A bigger scheduling freedom increases the possibility that these algorithms iterate to the wrong local optimum. This is what happened in these situations.

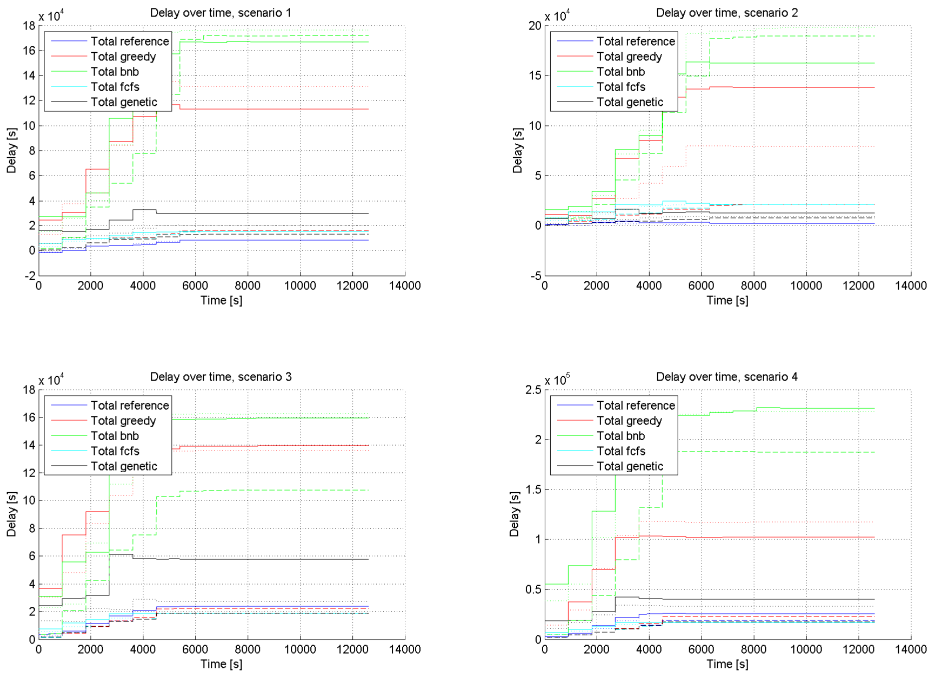

Figure 5.35: Total aircraft delay when the ready to taxi margin is varied

Figure 5.36, which shows the arrival times of the aircraft, where a ready to taxi margin of zero, ten, twenty and thirty minutes is shown respectively, confirms this theorem. This figure shows that the arrival times of the aircraft using the reference, first come first served and genetic algorithm hardly change when the ready to taxi margin is changed. For a margin of twenty and thirty minutes the aircraft are not departing more early, because that would let the aircraft depart outside their departure slots. Figure 5.36 also shows that the departure times using the greedy and branch and bound algorithms become more chaotic when the ready-to-taxi margin is increased. This can also be assigned to the increased scheduling freedom for the algorithms.

Taking all these results into account, it can be concluded that a small margin for the aircraft between the moment that it is ready to taxi and the last moment to start taxiing without taking off with delay is useful to improve the efficiency of the algorithms. It provides extra freedom to the algorithms, enabling them to optimize the departure schedule. However, a bigger margin does not lead to a better schedule by definition. A margin which is too big can cause the algorithm to iterate to a worse, or even the worst possible solution. Therefore, among all evaluated margins, a margin of ten minutes is chosen as the optimal situation.

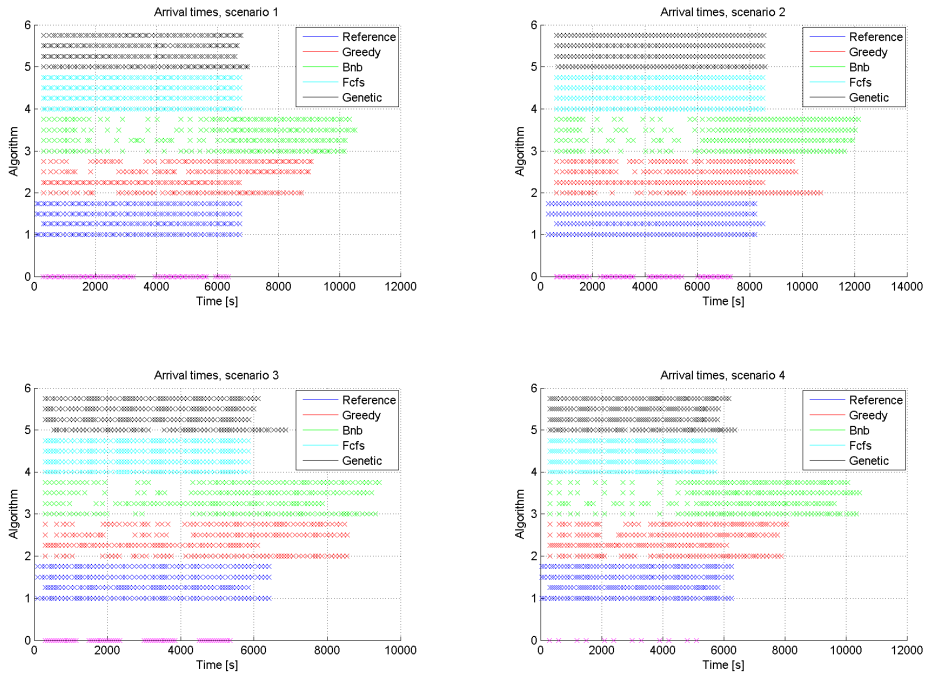

Figure 5.36: Arrival times when the ready to taxi margin is varied. The ready to taxi margins from bottom to top are zero, ten, twenty and thirty minutes

5.4 Optimal algorithm and parameters

Looking at the simulation results from section 5.3, an optimal value for all input parameters can be determined. If these optimal input values are used, the output parameters are optimized without changing the incoming traffic flow.

All input parameters are shown in Table 5.3. The first input parameter to be varied was the step size. The optimal value for the step size parameter is highly dependent on the amount of disturbances. In case of no disturbances, the step size should be taken as high as possible. This reduces the required computational power and almost has no negative influence on the efficiency of the algorithms. In case when disturbances are present, a smaller step size is advised. This allows the algorithm to react on disturbances more quickly. Therefore, in practice, a step size of five seconds is defined as the optimal step size.

There is no unique optimal value for the parameter which defines the size of the scheduling window. A bigger scheduling window makes the algorithm look more forward and thus increases the amount of possible departure sequences. This increases the chance that the scheduling algorithm iterates to a less optimal solution. However, a smaller scheduling window increases the number of rescheduling operations and thus the workload of the algorithm. The simulations in section 5.3.3 show that a scheduling window of fifteen minutes can be assumed as the best tradeoff between the number of rescheduling operations and the overall efficiency of the departure schedule.

In the situation without disturbances, the efficiency of the evaluated algorithms is almost independent on the slot size, as explained in section 5.3.5. The only exception are the greedy and branch and bound algorithms. These algorithms are more efficient when a smaller slot size is used. For all algorithms the number of aircraft departing out of their slots is bigger for a smaller slot size, but the real departure times of the aircraft are not affected by a change in the size of the departure slot, except for the greedy and branch and bound algorithm. A bigger slot size will make these algorithms less efficient.

It must be taken into account that this result was acquired using a simulation without disturbances. If the smallest slot size used during this research will be used in practice, disturbances can disturb the aircraft and decrease the efficiency drastically. Therefore it is advised to keep the slot boundaries at the size currently used and to perform further research on changing these slot boundaries.

At fist sight, it is expected that a larger ready to taxi margin will provide more possibilities to the algorithm to improve the departure schedule. However, a bigger margin can let the algorithms iterate to a wrong local optimum. This especially holds for the greedy and branch and bound algorithm. No ready to taxi margin reduces the degrees of freedom in such a way that it is also not possible to calculate an optimal departure schedule. Therefore a ready to taxi margin of ten minutes seems to be the best option among all the tested margins.

Taking all tests discussed in section 5.3 into account, the branch and bound algorithm is not usable for scheduling the departure traffic. This algorithm can be used for reducing the workload of the air traffic controllers, but the performance of this algorithm is in all tests worse or as good as the reference algorithm, but never better than the performance of the reference algorithm.

These tests show that the performance of the reference algorithm is in almost all situations comparable to the other algorithms under investigation. This is a remarkable result because of the ease of this algorithm. Looking at the other algorithms, the genetic algorithm is a good option to use for runway scheduling. The evaluation in section 5.3 shows that this algorithm has a good performance in tests where disturbances are taken into account.

Summarizing the above it can be stated that the currently used reference algorithm still is a good option for quiet and ideal traffic situations. The performance of the reference algorithm in these situations is good in comparison with the other algorithms. Only in case of changing input parameters which cause unexpected effects like disturbances and variations in the aircraft based speed constraints, other algorithms can be useful to improve the performance. The genetic algorithm provides the most performance increase in these situations.

6. Conclusion and recommendations

After having performed all the simulations with the system designed and having evaluated the results of these simulations, a conclusion about the total system and the performance of the departure scheduling algorithms can be drawn. This conclusion is discussed in section 6.1. During this research, aspects came across that could be further improved or might be interesting for further research. These improvements and further research could not be part of this research because of reasons like the amount of time needed for this tasks or the fact that the scope of these improvements laid too far from the original scope of this research. These recommendations for further research are discussed in section 6.2.

6.1 Conclusion

The goal of this research was to realize a simulation environment for an evaluation of traffic scheduling algorithms. This traffic scheduling algorithms might be used to improve the efficiency of the air traffic handling in the future. This report focused on the departure traffic at airports Using algorithms in this context might improve this efficiency as discussed in chapter 1.

The first step in the process to design a simulation environment for evaluation of the algorithms is to get more insight into the existing scheduling algorithms. There are dozens of different algorithms to improve the efficiency of air traffic scheduling (chapter 2). Not all of the algorithms are suitable for the goal under investigation in this research, because they do not provide a complete solution (only sequencing or only scheduling, instead of both). The algorithms that fit all requirements can be divided into four categories: first come first served, branch and bound, greedy and genetic algorithms. A quick overview of the characteristics of these algorithms is given in Table 3.5.

The requirements the system to be designed has to meet are (chapter 2):

1. Easy to modify to change possibilities of simulation
2. Adjustable input parameters
    a. Traffic situation
    b. Airport map
    c. Scheduling algorithm
    d. Update rate
    e. Aircraft based speed constraints
    f. Ground based speed constraints
3. Take disturbances into account
4. Take rescheduling operations into account
5. Provide multiple output parameters to do evaluation of algorithm
    a. Robustness
    b. Total delay
    c. Average delay
    d. Runway throughput
    e. Time windows
    f. Speed windows
6. Simulation must be faster than reality

The input and output parameters of the complete system are given in Table 4.1 and Table 4.2. Not all input parameters are fixed. Some of them can change while the simulation is running. These are the parameters *disturbance*, *abspeed* and *rttaxi*. The other parameters define simulation settings, physical aircraft constraints or regulations and therefore will not change during operation.

Due to the complexity of the complete system and the demand to comply with requirements like the possibility to easily modify the possibilities of the simulation, the system is built in a modular way and consists of six subsystems. The first subsystem is the simulation of the traffic scenario (section 4.3.1). This subsystem can be considered as the core of the system and within this subsystem the actual simulation is performed. Other tasks like updating the positions of the aircraft, performing a (re)scheduling operation and inserting disturbances are done by other subsystems, but the action is initiated by this subsystem. All other subsystems are connected to this subsystem.

The second subsystem is the calculation of the departure schedule (section 4.3.2). The task of this subsystem is not only to calculate the initial departure schedule, but also the rescheduling operations are performed by this subsystem. After a (new) schedule is calculated, the separation times are verified by another subsystem (section 4.3.3). In the ideal situation the algorithm would take care of maintaining the separation requirements and thus this subsystem would be useless. This system is used as an extra check to prevent violations of the separation requirements in case of improper functioning algorithms.

After the departure schedule is calculated and verified, the departure windows are calculated (section 4.3.4). These windows indicate a minimum and maximum allowed speed and time window for the aircraft. These windows indicate a disturbance margin of the aircraft. If the aircraft stay within these windows no rescheduling operations are needed.

Notwithstanding the ease of the operation to update the position of the aircraft, this operation is designed as a separate subsystem (section 4.3.5). This decision is made to be able to reuse this module and not having to implement this operation many times within multiple other subsystems.

The last subsystem discussed is the subsystem that inserts the disturbances into the simulation (section 4.3.6). In this subsystem disturbances are added to the position and speed of the aircraft to make the simulation more realistic. An overview of the complete system is given in Figure 4.3 and Figure 4.4.

The system designed and the algorithms involved are evaluated via four test scenarios. For each test scenario, the parameters *step*, *schedwin*, *algorithm*, *disturbances*, *slotboundaries*, *abspeed* and *rttaxi* are varied according to the variations in Table 5.3. The other parameters are kept constant.

Having simulated the default scenario, the delay caused by the greedy and branch and bound algorithm is four to eight times as high as the delay caused by the other algorithms. This causes 30 to 60% of the aircraft to be unable to depart within their assigned slots (section 5.3.1). When the step size parameter is changed from five to sixty seconds, the influence on the performance of the algorithms is negligible. A bigger step size reduces the computational power required, but it is expected that when disturbances are taken into account, a bigger step size would have a negative impact on the performance of the algorithms.

Decreasing the scheduling window increases the workload for the system, because more scheduling operations are needed. The efficiency change or this decreased window is dependent on the algorithm which is used. The greedy and branch and bound algorithms benefit from a smaller scheduling window. A smaller window decreases the chance that these algorithms converge to a less optimal solution. The performance of the genetic algorithm increases when a bigger scheduling window is used. The reference and first come first served algorithms are not influenced by a change of the scheduling window. It is expected that in case of more disturbances, the effectiveness of a larger scheduling window decreases (section 5.3.3).

Adding disturbances to the system makes clear that the impact of gate delay is much higher than the impact of taxi delay (section 5.3.4). All algorithms can deal with average delays of 6:15 minutes without problems. When the disturbance is increased to average delays up to 25:00 minutes, the genetic algorithm has the best performance. The larger the delays, the higher the efficiency improvement of this algorithm compared to the reference algorithm.

Changing the aircraft based speed parameter has little influence on the efficiency of the algorithms (5.3.6). This is caused due to the short taxi times at the airport used in this simulation. Only very high disturbances of 90% have a noticeable influence. This leads to the result that when this type of disturbances is used the genetic algorithm also has the best performance.

Changing the size of the departure slot changes the degree of freedom for the algorithms to schedule the aircraft. The first come first served and reference algorithm schedule the aircraft based on the time they are ready to start taxiing and therefore are not directly influenced by a change of the size of the departure slot. The other algorithms are able to generate a more efficient departure schedule when a bigger slot size is used (section 5.3.5).

When the ready to taxi margin is changed, this leads to the conclusion that some ready to taxi margin is required to give the algorithms the freedom to compose an optimal departure schedule (section 5.3.7). If the margin is made too large, the efficiency decreases because the algorithms are unable to find the optimal solution anymore. Therefore a ready to taxi margin of ten minutes is the best option. This holds for all tested algorithms.

Taking all simulations into account, the genetic algorithm has the best performance. The advantage of using a scheduling algorithm is most significant when disturbances are taken into account. In case of no disturbances, the performance of the reference algorithm is in almost all situations comparable to the performance of the genetic algorithm. The branch and bound algorithm has the least advantages when used for air traffic control. This algorithm can reduce the workload for the air traffic controllers, but the performance on aspects as delay and robustness is in all investigated situations as good as or worse than the reference algorithm.

6.2 Recommendations

As was stated before, the result of this research is a simulation environment for the evaluation of departure traffic scheduling algorithms and an evaluation of these algorithms. It is expected that this research will be continued to improve the simulation and gather more information about the performance of the algorithms and how this performance can be improved. To help future research, the promising fields of interest which became clear during this work are listed here.

6.2.1 Speed profile

The current simulation environment does not use speed profiles. The aircraft do get a minimum and maximum speed to define the boundaries of their window, but the ideal speed of the scheduled aircraft is equal to the ideal speed based on the aircraft class and independent on the situation. One of the disadvantages of this principle is that an aircraft with a slower ideal speed can never take off directly after an aircraft with a higher ideal speed if they share the last segment of the taxiway. It is expected that the departure schedules can be further optimized if aircraft are given a speed profile.

6.2.2 Departure slots

This research made clear that using scheduling algorithms can improve the efficiency of departure schedules. Research on the impact of a change of the departure slot size is done in section 5.3.5. This shows that for some of the algorithms, a smaller slot size did not have a negative impact on the efficiency, but a smaller slot size does have advantages. A smaller slot size results in takeoff times being known more exact in advance and thus planning aircraft trajectories in the air can also be done more accurate. To be able to draw a more detailed conclusion about the effect of a reduction of the slot size on the efficiency of the complete air traffic system, more research is needed on this subject.

6.2.3 Disturbances

In section 5.3, the algorithms are evaluated by changing the input parameters. Due to practical limitations, not all combinations of input parameters and disturbances are investigated. The results form this section show that the added value of using an algorithm for scheduling is the highest when disturbance is taken into account. Therefore it is advised to perform further research on the algorithms when more disturbances are taken into account. This might further increase the added value of algorithms for calculating the departure schedule.

6.2.4 Mixed mode

The airport situation used in the simulations in this work contained a runway that was only used for departure traffic. In reality, many airports use a runway in mixed mode. Besides practical reasons (some airports only have one runway), is the sum of the capacity of two runways that are both used in mixed mode higher than the sum of a runway exclusively used for landing and a runway exclusively used for departure traffic. Expanding the simulation with the possibility of using runways in mixed mode increases the use of the simulation.

6.2.5 Other traffic

There are situations where, besides departure traffic, also other traffic is present on the airport. This can be (taxiing) arriving aircraft and blockings of runways and taxiways due to crossings, other vehicles and inspections. This other traffic and the blockings also have their impact on the efficiency of the departure traffic. It is expected that in these situations the use of algorithms can also improve the handling of departure traffic. More research is needed to confirm this statement.

Bibliography

1. Eurocontrol, *Eurocontrol Medium-Term Forecast: IFR Flight Movements 2010-2016*. 2010.
2. Anagnostakis, I.D. and J.-P. Clarke, *A Multi-Objective, Decomposition-Based Algorithm Design Methodology and its Application to Runway Operations Planning*. 2004, MIT International Center for Air Transportation.
3. Agency, C.I. *World Factbook 2009*. 2009 March 18, 2011 [cited 2011 March 20]; Available from: https://www.cia.gov/library/publications/the-world-factbook/index.html.
4. Balakrishnan, H. and B. Chandran, *Efficient and Equitable Departure Scheduling in Real-Time: New Approaches to Old Problems*, in *USA/Europe Air Traffic Management R&D Seminar*. 2007: Barcelona, Spain.
5. Idris, H.R., et al., *Identification of Flow Constraint and Control Points in Departure Operations at Airport Systems*, in *AIAA Guidance, Navigation, and Control Conference and Exhibit*. 1998: Boston, MA. p. 947-956.
6. Le, L., G. Donohue, and C.-H. Chen, *Using Auction-Based Slot Allocation for Traffic Demand Management at Hartsfield Atlanta International Airport: A Case Study*, in *National Research Council Transportation Research Board Meeting January 2004*. 2003.
7. Beasley, J.E., et al., *Displacement Problem and Dynamically Scheduling Aircraft Landings.* Journal of the Operational Research Society, 2004. **55**(1): p. 54-64.
8. Bolender, M.A. and G.L. Slater, *Analysis and Optimization of Departure Sequences*, in *AIAA Guidance, Navigation and Control Conference and Exhibit*. 2000, American Institute of Aeronautics & Astronautics: Denver, CD. p. 12.
9. Gupta, G., W. Malik, and Y.C. Jung, *A Mixed Integer Linear Program for Airport Departure Scheduling*, in *9th AIAA Aviation Technology, Integration, and Operations conference (ATIO)*. 2009, American Institute of Aeronautics and Astronautics, Inc.: Hilton Head, South Carolina.
10. Beasley, J.E., et al., *Scheduling Aircraft Landings - The Static Case.* Transportation science., 2000. **34**(2): p. 180-197.
11. Schumer, C.E. and C.B. Maloney, *Your Flight Has Been Delayed Again*. 2008, Joint Economic Committee Majority Staff.
12. Tu, Y., B. Michael, and W. Jank, *Estimating Flight Departure Delay Distributions - A Statistical Approach with Long-term Trend and Short-term Pattern*. 2005.
13. Trivizas, D.A., *Optimal Scheduling with Maximum Position Shift (MPS) Constraints: A Runway Scheduling Application.* Journal of navigation., 1998. **51**(2): p. 250-266.

14. Harmon, P. *MD-11 Limitations*. 2007 [cited 2010 June 29th]; Available from: http://nhbiz.net/md11/Videos/limitations.htm.
15. Smeltink, J.W., et al., *An Optimisation Model for Airport Taxi Scheduling.* Proceedings of the INFORMS Annual Meeting, Denver, Colo, USA, 2004.
16. Wichman, K.D., et al., *Four-Dimensional Trajectory Based Air Traffic Management*, in *AIAA Guidance, Navigation and Control Conference and Exhibit*. 2004: Providence, Rhode Island.
17. Mahadev, N.V.R., A. Pekec, and F.S. Roberts, *On An Aircraft Scheduling Problem with Priorities and Earliness/Tardiness Penalties*. 1995, Rutgers Center for Operations Research.
18. Bradley, S.P., A.C. Hax, and T.L. Magnanti, *Applied Mathematical Programming*. 1977, Reading, Mass.: Addison-Wesley.
19. Brinton, C.R. *An Implicit Enumeration Algorithm for Arrival Aircraft Scheduling*. in *Digital Avionics Systems Conference, 1992. Proceedings., IEEE/AIAA 11th*. 1992. Seattle, WA, USA.
20. Dear, R.G., *The Dynamic Scheduling of Aircraft in the Near Terminal Area*. 1976, Massachusetts Institute of Technology, Flight Transportation Laboratory: Cambridge.
21. Davis, T.J., K.J. Krzeczowski, and C. Bergh, *The Final Approach Spacing Tool*, in *13th IFAC Symposium on Automatic Control in Aerospace*. 1994: Palo Alto, California.
22. Credeur, L., et al., *Final-Approach Spacing Aids (FASA) Evaluation for Terminal-Area, Time-Based Air Traffic Control*. NASA technical paper, 3399. 1993, Washington, D.C.; Springfield, Va.: National Aeronautics and Space Administration, Office of Management, Scientific and Technical Information Program ; National Technical Information Service.
23. Neuman, F., H. Erzberger, and C. Ames Research, *Analysis of Delay Reducing and Fuel Saving Sequencing and Spacing Algorithms for Arrival Traffic*. 1991, National Aeronautics and Space Administration, Ames Research Center: Moffett Field, Calif.; Springfield, Va.
24. Balakrishnan, H. and B. Chandran, *Scheduling Aircraft Landings Under Constrained Position Shifting*, in *AIAA Guidance, Navigation and Control Conference and Exhibit*. 2006, American Institute of Aeronautics and Astronautics: Keystone, Colorado.
25. Houck, C.R., J.A. Joines, and M.G. Kay, *A Genetic Algorithm for Function Optimization: A Matlab Implementation.* 1996.
26. Durand, N., J.-M. Alliot, and J. Noailles. *Automatic Aircraft Conflict Resolution Using Genetic Algorithms*. in *Proceedings of the 1996 ACM symposium on Applied Computing - SAC '96*. 1996.
27. Krzeczowski, K.J., et al., *Knowledge-Based Scheduling of Arrival Aircraft*, in *AIAA Guidance, Navigation and Control Conference*. 1995, American Institute of Aeronautics and Astronautics: Baltimore, MD. p. 1758-1768.

Appendix A. System inputs and outputs

The tables in this appendix list all inputs and outputs of the functions used for the simulation environment and the dimensions of these variables.

Table A.1: Input and output variables of the main function

main.m	This function connects the other functions and coordinates the simulation.		
	Input variables		
Variable	**Dimension**	**Unit**	**Description**
simtime	1 x 1	[s]	Time for how long the simulation will run
step	1 x 1	[s]	Step size used during simulation
schedwin	1 x 1	[s]	Size of the scheduling window
mainpath	1 x #mainpaths	[m]	Length of each main taxiway
path	#paths x 3	[m m -]	Position where connected to main taxiway, length of taxiway, main taxiway connected to
slotboundaries	1 x 2	[s]	Negative and positive departure slot boundaries
aircrafts	#aircraft x 4	[s - - -]	Aircraft RTA, departure path, class, SID
abspeed	#aircraft x 2	[m/s m/s]	Aircraft based min and max speed constraints
rttaxi	#aircraft x 1	[s]	Time when ready to taxi
class	#classes x 2	[-]	Class based min and max speed constraints
separation	#classes x #classes	[s]	Wake vortex separation requirements
mitseparation	1 x 1	[s]	Miles-in-trail separation requirements
algorithm	1 x 1	[-]	Algorithm to use for simulation
slotsize	1 x 1	[m]	Minimum required separation during taxiing
	Output variables		
Variable	**Dimension**	**Unit**	**Description**
timewin	simtime/step+1 x #aircraft x 2	[s]	Min and max time window for each aircraft at each time step
speedwin	simtime/step+1 x #aircraft x 2	[m/s]	Min and max speed window for each aircraft at each time step
delay	simtime/step+1 x #aircraft	[s]	Delay for each aircraft at each time step

dropouts	simtime/step+1 x 2	[-]	Number of dropped aircraft and total aircraft at each time step
arrival	#aircraft x 1	[s]	Arrival times for all aircraft
numreschedule	simtime/step+1 x 1	[-]	Number of rescheduling operations at each time step
throughput	simtime/step+1 x 2	[-]	Number of aircraft left and entered taxiway during last hour
Global variables			
Variable	**Dimension**	**Unit**	**Description**
separation	#classes x #classes	[s]	Wake vortex separation requirements
mitseparation	1 x 1	[s]	Miles-in-trail separation requirements
slotboundaries	1 x 2	[s]	Negative and positive departure slot boundaries
class	#classes x 2	[-]	Class based min and max speed constraints
prevdeparted	1 x 4 cell	[-]	Lists aircraft, speed, departure and realetas data of last two departed aircraft

Table A.2: Input and output variables of the algorithm functions

reference.m greedy.m bnb.m fcfs.m genetic.m	These functions compute the takeoff and departure schedules for all aircraft		
Input variables			
Variable	**Dimension**	**Unit**	**Description**
aircrafts	#aircraftnowscheduled x 4	[s - - -]	Aircraft RTA, departure path, class, SID. Only for aircraft currently scheduled
path	#paths x 3	[m m -]	Position where connected to main taxiway, length of taxiway, main taxiway connected to
rttaxi	#aircraftnowscheduled x 1	[s]	Time when ready to taxi. Only for aircraft currently scheduled
positions	#aircraftnowscheduled x 2	[m m]	Position on sub and main taxiway. Only for aircraft currently scheduled
reschedule	1 x n	[-]	Id's of aircraft that triggered rescheduling
departure	#aircraftnowscheduled x 1	[s]	Previously scheduled departure times. Only for aircraft currently scheduled
time	1 x 1	[s]	Current time
speed	#aircraftnowscheduled x 3	[m/s m/s m/s]	Max speed on sub, max on main and minimum speed of aircraft
realetas	#aircraftnowscheduled x 1	[s]	Previously scheduled arrival time. Only for aircraft currently scheduled
simtime	1 x 1	[s]	Time for how long the simulation will run
step	1 x 1	[s]	Step size used during simulation

Output variables			
Variable	**Dimension**	**Unit**	**Description**
departure	#aircraftnowscheduled x 1	[s]	Scheduled departure times. Only for aircraft currently scheduled
speed	#aircraftnowscheduled x 3	[m/s m/s m/s]	Scheduled max speed on sub, max on main and minimum speed of aircraft
realetas	#aircraftnowscheduled x 1	[s]	Scheduled arrival time. Only for aircraft currently scheduled
Global variables			
Variable	**Dimension**	**Unit**	**Description**
separation	#classes x #classes	[s]	Wake vortex separation requirements
mitseparation	1 x 1	[s]	Miles-in-trail separation requirements
slotboundaries	1 x 2	[s]	Negative and positive departure slot boundaries
class	#classes x 2	[-]	Class based min and max speed constraints

Table A.3: Input and output variables of the slot_separation function

slot_separation.m	This function checks if the scheduled departure times do not violate the separation requirements and corrects them if necessary		
Input variables			
Variable	**Dimension**	**Unit**	**Description**
aircrafts	#aircraftnowscheduled x 4	[s - - -]	Aircraft RTA, departure path, class, SID. Only for aircraft currently scheduled
departure	#aircraftnowscheduled x 1	[s]	Previously scheduled departure times. Only for aircraft currently scheduled
speed	#aircraftnowscheduled x 3	[m/s m/s m/s]	Max speed on sub, max on main and minimum speed of aircraft
realetas	#aircraftnowscheduled x 1	[s]	Previously scheduled arrival time. Only for aircraft currently scheduled
time	1 x 1	[s]	Current time
Output variables			
Variable	**Dimension**	**Unit**	**Description**
departure	#aircraftnowscheduled x 1	[s]	Scheduled departure times. Only for aircraft currently scheduled
speed	#aircraftnowscheduled x 3	[m/s m/s m/s]	Scheduled max speed on sub, max on main and minimum speed of aircraft
realetas	#aircraftnowscheduled x 1	[s]	Scheduled arrival time. Only for aircraft currently scheduled
Global variables			
Variable	**Dimension**	**Unit**	**Description**
separation	#classes x #classes	[s]	Wake vortex separation requirements
mitseparation	1 x 1	[s]	Miles-in-trail separation requirements
slotboundaries	1 x 2	[s]	Negative and positive departure slot boundaries
prevdeparted	1 x 4 cell	[-]	Lists aircraft, speed, departure and realetas data of last two departed aircraft

Table A.4: Input and output variables of the depwindow function

depwindow.m	This function calculates the time and speed windows of all aircraft.		
Input variables			
Variable	**Dimension**	**Unit**	**Description**
allaircrafts	#aircraftnowscheduled x 4	[s - - -]	Aircraft RTA, departure path, class, SID. Only for aircraft currently scheduled
path	#paths x 3	[m m -]	Position where connected to main taxiway, length of taxiway, main taxiway connected to
allspeed	#aircraftnowscheduled x 3	[m/s m/s m/s]	Max speed on sub, max on main and minimum speed of aircraft
slotsize	1 x 1	[m]	Minimum required separation during taxiing
alldeparture	#aircraftnowscheduled x 1	[s]	Scheduled departure times. Only for aircraft currently scheduled
allpositions	#aircraftnowscheduled x 2	[m m]	Position on sub and main taxiway. Only for aircraft currently scheduled
time	1 x 1	[s]	Current time
Output variables			
Variable	**Dimension**	**Unit**	**Description**
allwaypoints	#aircraftnowscheduled x #aircraftnowscheduled x 3	[m]	Positions of min max and ideal waypoints. Each row contains all waypoints for corresponding aircraft
allwaytime	#aircraftnowscheduled x #aircraftnowscheduled x 3	[s]	Times of min max and ideal waypoints. Each row contains all waypoints for corresponding aircraft
allwayspeed	#aircraftnowscheduled x #aircraftnowscheduled x 3	[m/s]	Speeds of min max and ideal waypoints. Each row contains all waypoints for corresponding aircraft
allstartpos	#aircraftnowscheduled x 3	[m]	Starting positions of min max and ideal window of aircraft at sub taxiway
allstarttime	#aircraftnowscheduled x 3	[s]	Starting times of min max and ideal window of aircraft at sub taxiway
Global variables			
Variable	**Dimension**	**Unit**	**Description**
none			

Table A.5: Input and output variables of the updatepos function

updatepos.m	This function updates the position of the aircraft and waypoints in the simulation		
	Input variables		
Variable	**Dimension**	**Unit**	**Description**
positions	#aircraft x 2	[m m]	Positions on sub and main taxiway
speed	#aircraft x 2	[m/s m/s]	Scheduled speed on sub and main taxiway
step	1 x 1	[s]	Step size used during simulation
departure	#aircraft x 1	[s]	Scheduled departure times
time	1 x 1	[s]	Current time
	Output variables		
Variable	**Dimension**	**Unit**	**Description**
positions	#aircraft x 2	[m m]	New position on sub and main taxiway
	Global variables		
Variable	**Dimension**	**Unit**	**Description**
	none		

Table A.6: Input and output variables of the disturbance function

disturbance.m	This function inserts the disturbance to the speed and departure time of the aircraft		
	Input variables		
Variable	**Dimension**	**Unit**	**Description**
speed	#aircraft x 3	[m/s m/s m/s]	Max speed on sub, max on main and minimum speed of aircraft
time	1 x 1	[s]	Current time
rttaxi	#aircraft x 1	[s]	Time when ready to taxi
	Output variables		
Variable	**Dimension**	**Unit**	**Description**
disturbed_speed	#aircraft x 3	[m/s m/s m/s]	Disturbed max speed on sub, max on main and minimum speed of aircraft
disturbed_rttaxi	#aircraft x 1	[s]	Disturbed time when ready to taxi
	Global variables		
Variable	**Dimension**	**Unit**	**Description**
disturbances	1 x 2	[-]	Binary vector to enable gate and taxi disturbance
rttaxidisturbance	#aircraft x 1	[s]	Gate disturbance for each aircraft
speeddisturbance	#aircraft x 3 x simtime/step+1	[m/s]	max on sub, max on main and minimum speed disturbance for each aircraft
step	1 x 1	[s]	Step size used during simulation

Appendix B. Scenario data

The tables in this appendix list the data of all aircraft of the four scenarios used during the evaluation of the traffic scheduling algorithms.

Table B.1: Aircraft data for scenario 1

Aircraft	RTA [s]	Path / Departure location	Class	Airroute / SID	Aircraft based min speed [m/s]	Aircraft based max speed [m/s]	Rttaxi [s]
1	300	8	Heavy	1	0	16	214
2	360	10	Large	2	0	25	270
3	480	8	Heavy	1	0	16	394
4	540	5	Large	2	0	25	512
5	660	8	Heavy	1	0	16	574
6	720	3	Large	2	0	25	663
7	840	5	Heavy	1	0	16	796
8	900	3	Large	2	0	25	843
9	1020	4	Heavy	1	0	16	950
10	1080	7	Large	2	0	25	1056
11	1200	8	Heavy	1	0	16	1114
12	1260	4	Large	2	0	25	1215
13	1380	4	Heavy	1	0	16	1310
14	1440	4	Large	2	0	25	1395
15	1560	3	Heavy	1	0	16	1471
16	1620	3	Large	2	0	25	1563
17	1740	4	Heavy	1	0	16	1670
18	1800	8	Large	2	0	25	1745
19	1920	5	Heavy	1	0	16	1876
20	1980	8	Large	2	0	25	1925
21	2100	8	Heavy	1	0	16	2014
22	2160	3	Large	2	0	25	2103
23	2280	8	Heavy	1	0	16	2194
24	2340	3	Large	2	0	25	2283
25	2460	8	Heavy	1	0	16	2374
26	2520	10	Large	2	0	25	2430
27	2640	4	Heavy	1	0	16	2570
28	2700	5	Large	2	0	25	2672
29	2820	8	Heavy	1	0	16	2734
30	2880	4	Large	2	0	25	2835
31	3000	9	Heavy	1	0	16	2877
32	3060	4	Large	2	0	25	3015

Aircraft	RTA [s]	Path / Departure location	Class	Airroute / SID	Aircraft based min speed [m/s]	Aircraft based max speed [m/s]	Rttaxi [s]
33	3180	5	Heavy	1	0	16	3136
34	3240	7	Large	2	0	25	3216
35	3960	3	Heavy	1	0	16	3871
36	4020	5	Large	2	0	25	3992
37	4140	3	Heavy	1	0	16	4051
38	4200	4	Large	2	0	25	4155
39	4320	4	Heavy	1	0	16	4250
40	4380	8	Large	2	0	25	4325
41	4500	11	Heavy	1	0	16	4339
42	4560	7	Large	2	0	25	4536
43	4680	11	Heavy	1	0	16	4519
44	4740	3	Large	2	0	25	4683
45	4860	11	Heavy	1	0	16	4699
46	4920	11	Large	2	0	25	4817
47	5040	4	Heavy	1	0	16	4970
48	5100	5	Large	2	0	25	5072
49	5220	3	Heavy	1	0	16	5131
50	5280	3	Large	2	0	25	5223
51	5400	5	Heavy	1	0	16	5356
52	5460	7	Large	2	0	25	5436
53	5580	8	Heavy	1	0	16	5494
54	5640	8	Large	2	0	25	5585
55	5940	3	Heavy	1	0	16	5851
56	6000	4	Large	2	0	25	5955
57	6120	4	Heavy	1	0	16	6050
58	6180	4	Large	2	0	25	6135
59	6300	4	Heavy	1	0	16	6230
60	6360	4	Large	2	0	25	6315

Table B.2: Aircraft data for scenario 2

Aircraft	RTA [s]	Path / Departure location	Class	Airroute / SID	Aircraft based min speed [m/s]	Aircraft based max speed [m/s]	Rttaxi [s]
1	600	8	Heavy	1	0	16	514
2	660	10	Large	1	0	25	570
3	780	8	Heavy	2	0	16	694
4	840	5	Large	2	0	25	812
5	960	8	Heavy	1	0	16	874
6	1020	3	Large	1	0	25	963
7	1140	5	Heavy	2	0	16	1096
8	1200	3	Large	2	0	25	1143
9	1320	4	Heavy	1	0	16	1250
10	1380	7	Large	1	0	25	1356
11	1500	8	Heavy	2	0	16	1414
12	1560	4	Large	2	0	25	1515
13	1680	4	Heavy	1	0	16	1610
14	1740	4	Large	1	0	25	1695
15	1860	3	Heavy	2	0	16	1771

Aircraft	RTA [s]	Path / Departure location	Class	Airroute / SID	Aircraft based min speed [m/s]	Aircraft based max speed [m/s]	Rttaxi [s]
16	2280	3	Large	2	0	25	2223
17	2400	4	Heavy	1	0	16	2330
18	2460	8	Large	1	0	25	2405
19	2580	5	Heavy	2	0	16	2536
20	2640	8	Large	2	0	25	2585
21	2760	8	Heavy	1	0	16	2674
22	2820	3	Large	1	0	25	2763
23	2940	8	Heavy	2	0	16	2854
24	3000	3	Large	2	0	25	2943
25	3120	8	Heavy	1	0	16	3034
26	3180	10	Large	1	0	25	3090
27	3300	4	Heavy	2	0	16	3230
28	3360	5	Large	2	0	25	3332
29	3480	8	Heavy	1	0	16	3394
30	3540	4	Large	1	0	25	3495
31	4140	9	Heavy	2	0	16	4017
32	4200	4	Large	2	0	25	4155
33	4320	5	Heavy	1	0	16	4276
34	4380	7	Large	1	0	25	4356
35	4500	3	Heavy	2	0	16	4411
36	4560	5	Large	2	0	25	4532
37	4680	3	Heavy	1	0	16	4591
38	4740	4	Large	1	0	25	4695
39	4860	4	Heavy	2	0	16	4790
40	4920	8	Large	2	0	25	4865
41	5040	11	Heavy	1	0	16	4879
42	5100	7	Large	1	0	25	5076
43	5220	11	Heavy	2	0	16	5059
44	5280	3	Large	2	0	25	5223
45	5400	11	Heavy	1	0	16	5239
46	6000	11	Large	1	0	25	5897
47	6120	4	Heavy	2	0	16	6050
48	6180	5	Large	2	0	25	6152
49	6300	3	Heavy	1	0	16	6211
50	6360	3	Large	1	0	25	6303
51	6480	5	Heavy	2	0	16	6436
52	6540	7	Large	2	0	25	6516
53	6660	8	Heavy	1	0	16	6574
54	6720	8	Large	1	0	25	6665
55	6840	3	Heavy	2	0	16	6751
56	6900	4	Large	2	0	25	6855
57	7020	4	Heavy	1	0	16	6950
58	7080	4	Large	1	0	25	7035
59	7200	4	Heavy	2	0	16	7130
60	7260	4	Large	2	0	25	7215

Table B.3: Aircraft data for scenario 3

Aircraft	RTA [s]	Path / Departure location	Class	Airroute / SID	Aircraft based min speed [m/s]	Aircraft based max speed [m/s]	Rttaxi [s]
1	300	8	Heavy	1	0	16	214
2	360	10	Heavy	2	0	16	219
3	420	8	Heavy	1	0	16	334
4	480	5	Heavy	2	0	16	436
5	540	8	Heavy	1	0	16	454
6	600	3	Large	2	0	25	543
7	660	5	Large	1	0	25	632
8	720	3	Large	2	0	25	663
9	780	4	Large	1	0	25	735
10	840	7	Large	2	0	25	816
11	900	8	Heavy	1	0	16	814
12	960	4	Heavy	2	0	16	890
13	1020	4	Heavy	1	0	16	950
14	1080	4	Heavy	2	0	16	1010
15	1140	3	Heavy	1	0	16	1051
16	1500	3	Large	2	0	25	1443
17	1560	4	Large	1	0	25	1515
18	1620	8	Large	2	0	25	1565
19	1680	5	Large	1	0	25	1652
20	1740	8	Large	2	0	25	1685
21	1800	8	Heavy	1	0	16	1714
22	1860	3	Heavy	2	0	16	1771
23	1920	8	Heavy	1	0	16	1834
24	1980	3	Heavy	2	0	16	1891
25	2040	8	Heavy	1	0	16	1954
26	2100	10	Large	2	0	25	2010
27	2160	4	Large	1	0	25	2115
28	2220	5	Large	2	0	25	2192
29	2280	8	Large	1	0	25	2225
30	2340	4	Large	2	0	25	2295
31	3000	9	Heavy	1	0	16	2877
32	3060	4	Heavy	2	0	16	2990
33	3120	5	Heavy	1	0	16	3076
34	3180	7	Heavy	2	0	16	3143
35	3240	3	Heavy	1	0	16	3151
36	3300	5	Large	2	0	25	3272
37	3360	3	Large	1	0	25	3303
38	3420	4	Large	2	0	25	3375
39	3480	4	Large	1	0	25	3435
40	3540	8	Large	2	0	25	3485
41	3600	11	Heavy	1	0	16	3439
42	3660	7	Heavy	2	0	16	3623
43	3720	11	Heavy	1	0	16	3559
44	3780	3	Heavy	2	0	16	3691
45	3840	11	Heavy	1	0	16	3679
46	4500	11	Large	2	0	25	4397
47	4560	4	Large	1	0	25	4515

Aircraft	RTA [s]	Path / Departure location	Class	Airroute / SID	Aircraft based min speed [m/s]	Aircraft based max speed [m/s]	Rttaxi [s]
48	4620	5	Large	2	0	25	4592
49	4680	3	Large	1	0	25	4623
50	4740	3	Large	2	0	25	4683
51	4800	5	Heavy	1	0	16	4756
52	4860	7	Heavy	2	0	16	4823
53	4920	8	Heavy	1	0	16	4834
54	4980	8	Heavy	2	0	16	4894
55	5040	3	Heavy	1	0	16	4951
56	5100	4	Large	2	0	25	5055
57	5160	4	Large	1	0	25	5115
58	5220	4	Large	2	0	25	5175
59	5280	4	Large	1	0	25	5235
60	5340	4	Large	2	0	25	5295

Table B.4: Aircraft data for scenario 4

Aircraft	RTA [s]	Path / Departure location	Class	Airroute / SID	Aircraft based min speed [m/s]	Aircraft based max speed [m/s]	Rttaxi [s]
1	300	8	Heavy	1	0	16	214
2	300	10	Heavy	2	0	16	159
3	300	8	Heavy	1	0	16	214
4	300	5	Heavy	2	0	16	256
5	300	8	Heavy	1	0	16	214
6	600	3	Large	2	0	25	543
7	600	5	Large	1	0	25	572
8	600	3	Large	2	0	25	543
9	600	4	Large	1	0	25	555
10	600	7	Large	2	0	25	576
11	1200	8	Heavy	1	0	16	1114
12	1200	4	Heavy	2	0	16	1130
13	1200	4	Heavy	1	0	16	1130
14	1200	4	Heavy	2	0	16	1130
15	1200	3	Heavy	1	0	16	1111
16	1500	3	Large	2	0	25	1443
17	1500	4	Large	1	0	25	1455
18	1500	8	Large	2	0	25	1445
19	1500	5	Large	1	0	25	1472
20	1500	8	Large	2	0	25	1445
21	2100	8	Heavy	1	0	16	2014
22	2100	3	Heavy	2	0	16	2011
23	2100	8	Heavy	1	0	16	2014
24	2100	3	Heavy	2	0	16	2011
25	2100	8	Heavy	1	0	16	2014
26	2400	10	Large	2	0	25	2310
27	2400	4	Large	1	0	25	2355
28	2400	5	Large	2	0	25	2372
29	2400	8	Large	1	0	25	2345
30	2400	4	Large	2	0	25	2355

Aircraft	RTA [s]	Path / Departure location	Class	Airroute / SID	Aircraft based min speed [m/s]	Aircraft based max speed [m/s]	Rttaxi [s]
31	3000	9	Heavy	1	0	16	2877
32	3000	4	Heavy	2	0	16	2930
33	3000	5	Heavy	1	0	16	2956
34	3000	7	Heavy	2	0	16	2963
35	3000	3	Heavy	1	0	16	2911
36	3300	5	Large	2	0	25	3272
37	3300	3	Large	1	0	25	3243
38	3300	4	Large	2	0	25	3255
39	3300	4	Large	1	0	25	3255
40	3300	8	Large	2	0	25	3245
41	3900	11	Heavy	1	0	16	3739
42	3900	7	Heavy	2	0	16	3863
43	3900	11	Heavy	1	0	16	3739
44	3900	3	Heavy	2	0	16	3811
45	3900	11	Heavy	1	0	16	3739
46	4200	11	Large	2	0	25	4097
47	4200	4	Large	1	0	25	4155
48	4200	5	Large	2	0	25	4172
49	4200	3	Large	1	0	25	4143
50	4200	3	Large	2	0	25	4143
51	4800	5	Heavy	1	0	16	4756
52	4800	7	Heavy	2	0	16	4763
53	4800	8	Heavy	1	0	16	4714
54	4800	8	Heavy	2	0	16	4714
55	4800	3	Heavy	1	0	16	4711
56	5100	4	Large	2	0	25	5055
57	5100	4	Large	1	0	25	5055
58	5100	4	Large	2	0	25	5055
59	5100	4	Large	1	0	25	5055
60	5100	4	Large	2	0	25	5055

Appendix C. Simulation results

The figures in this appendix show all results of the simulations performed during this research. All output parameters from the simulations discussed in section 5.3 are shown here.

Figure C.1 up to Figure C.7 show the output parameters using the standard situation as described in Table 5.3. In Figure C.4, the ideal arrival times as desired by the aircraft are shown as algorithm 0.

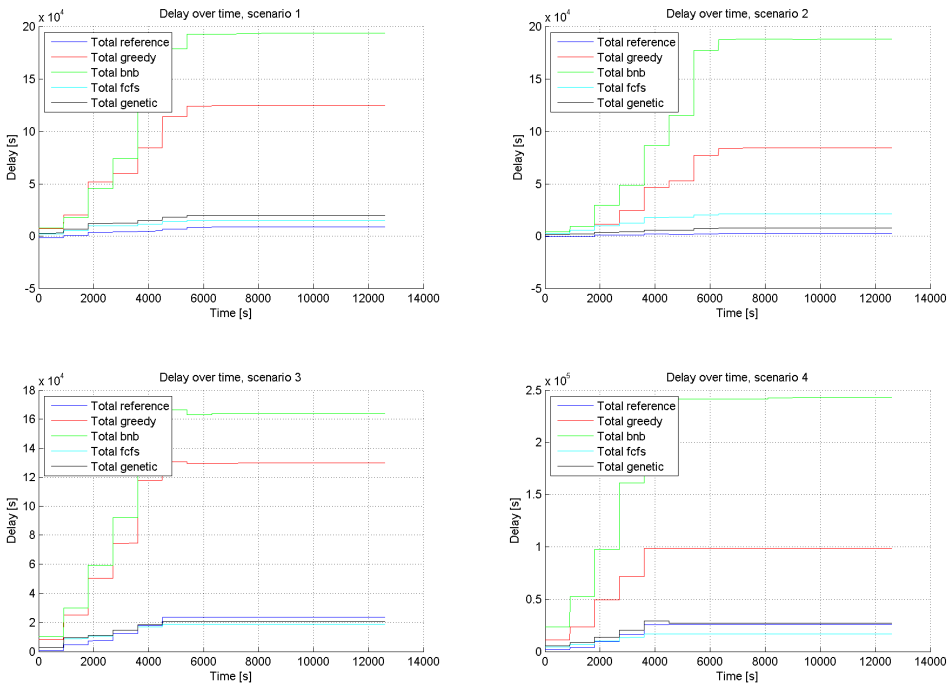

Figure C.1: Total delay over time in the standard situation

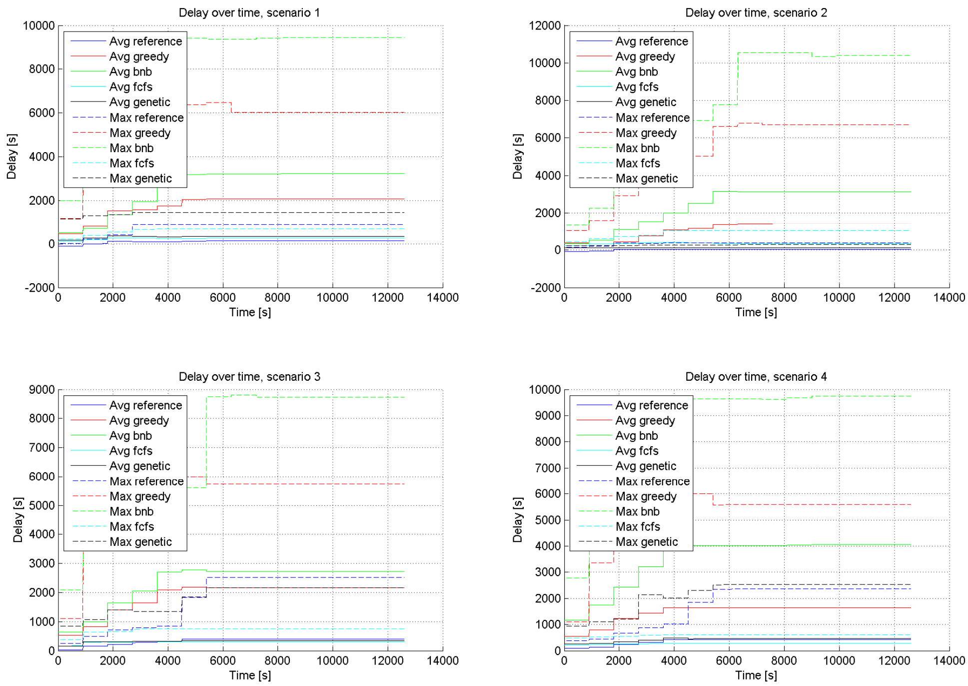

Figure C.2: Maximum and average individual delay in the standard situation

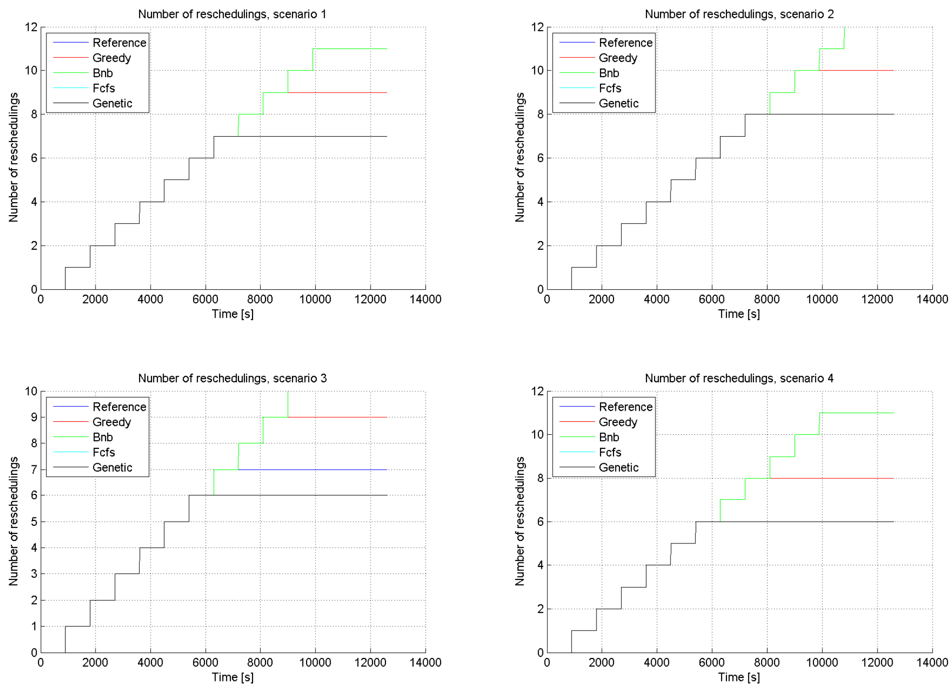

Figure C.3: Number of rescheduling operations in the standard situation

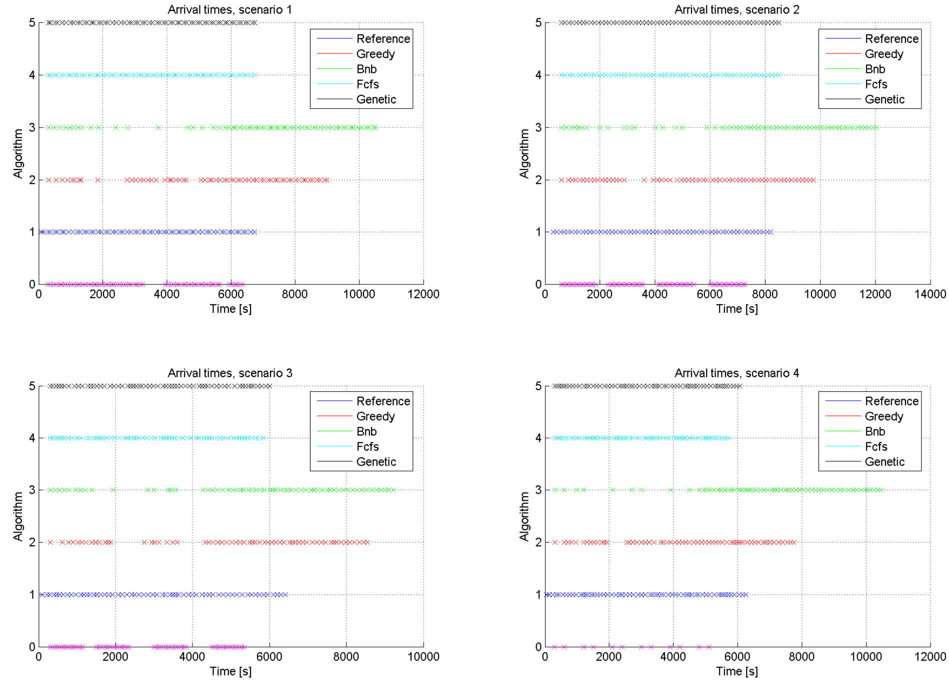

Figure C.4: Arrival times in the standard situation

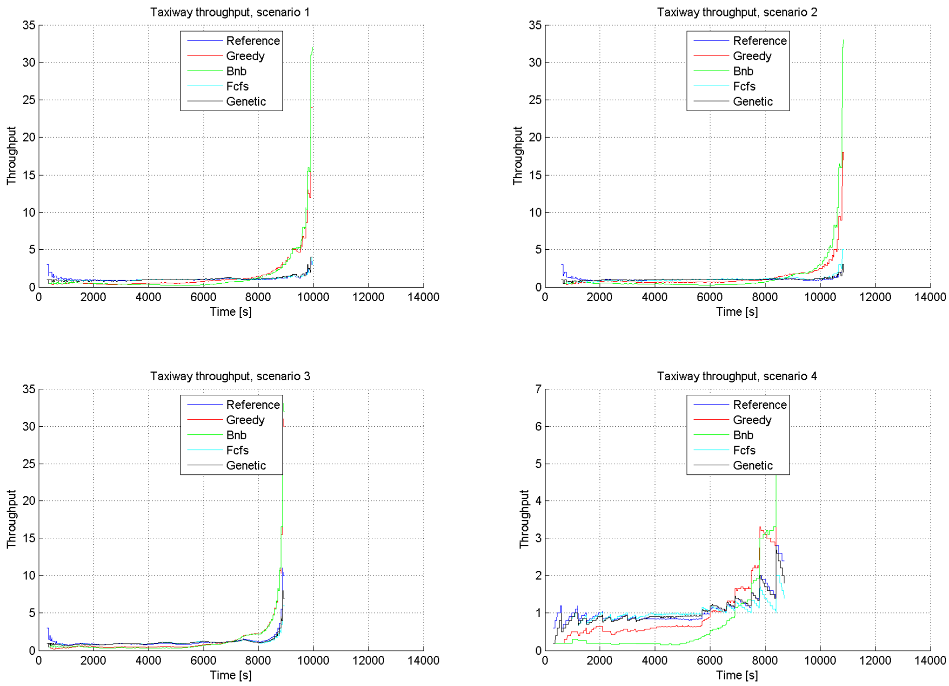

Figure C.5: Taxiway throughput in the standard situation

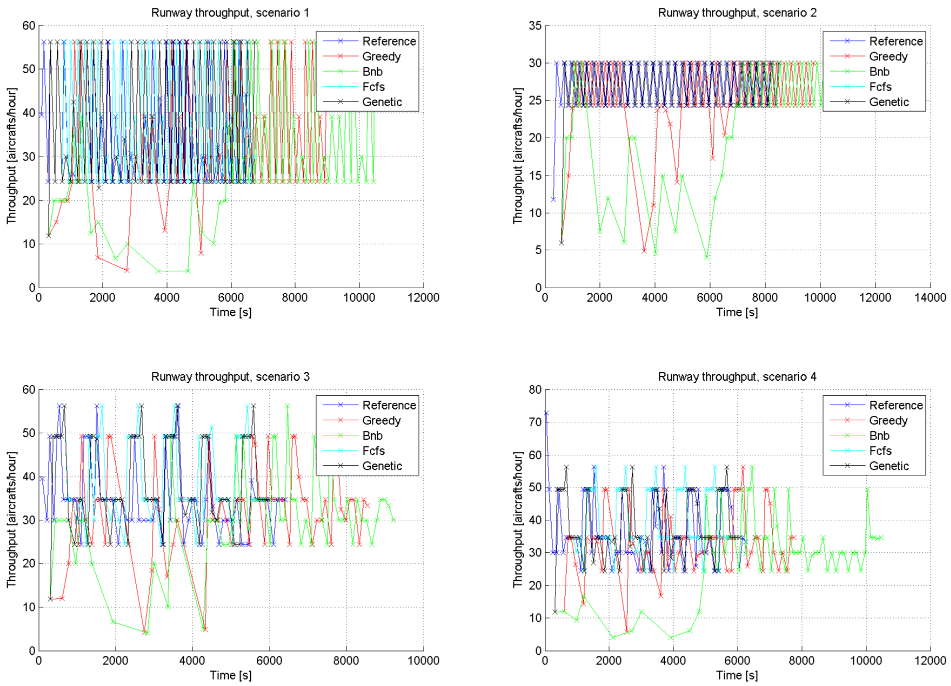

Figure C.6: Runway throughput in the standard situation

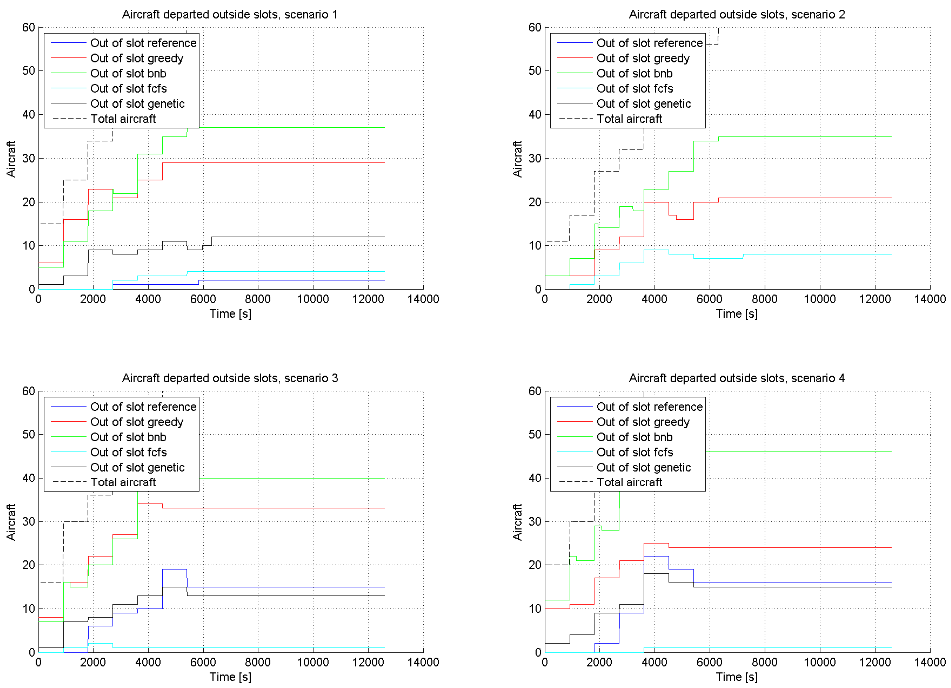

Figure C.7: Aircraft departed out of their slots in the standard situation

Figure C.8 up to Figure C.14 show the output parameters when the step size is varied as described in Table 5.3. A step size of five seconds is shown as a solid line and a step size of sixty seconds as a dotted line. In Figure C.11, the ideal arrival times as desired by the aircraft are shown as algorithm 0. A step size of 5 seconds is shown as the lower line and the step size of sixty seconds is represented by the upper line for each algorithm.

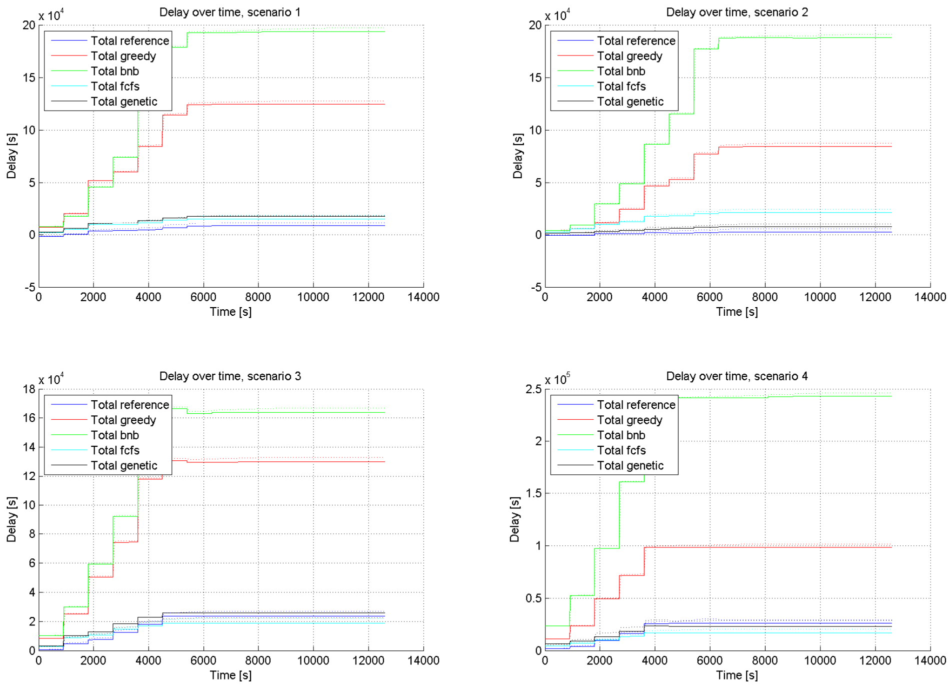

Figure C.8: Total delay over time when the step size is varied

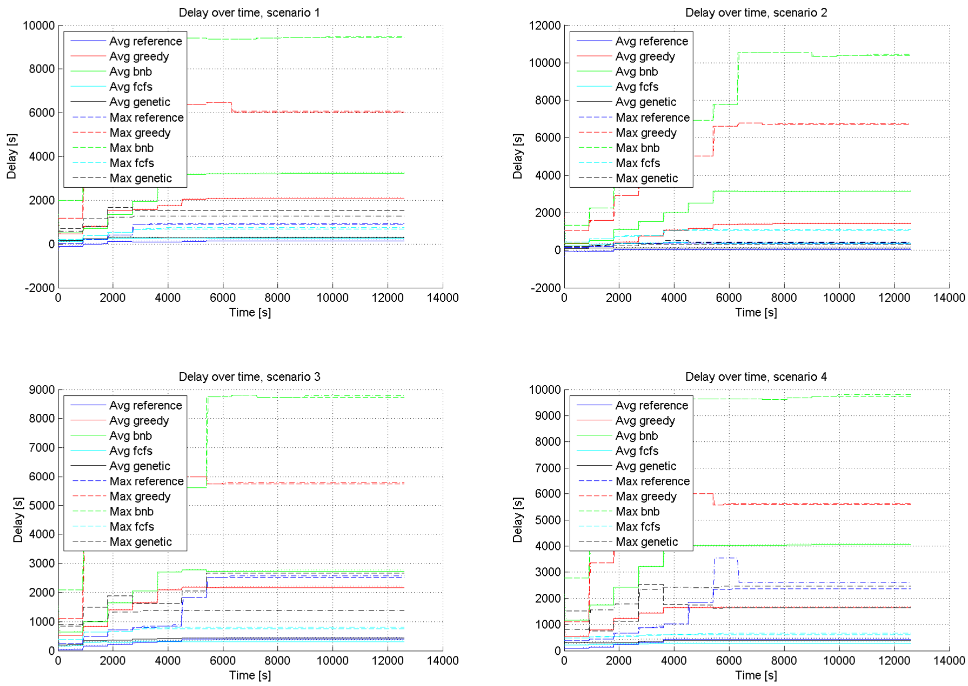

Figure C.9: Maximum and average individual delay when the step size is varied

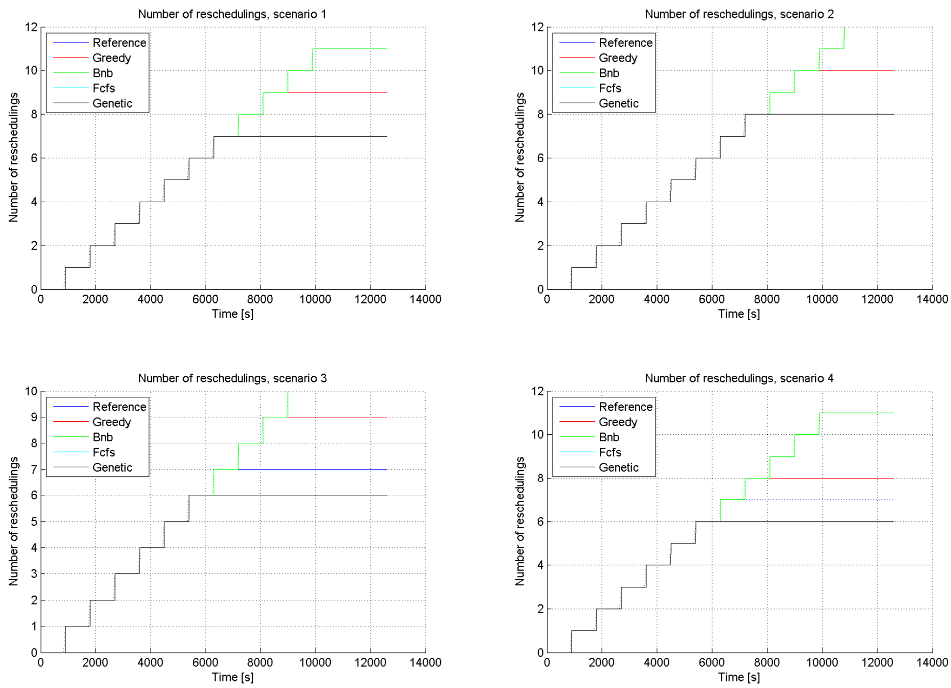

Figure C.10: Number of rescheduling operations when the step size is varied

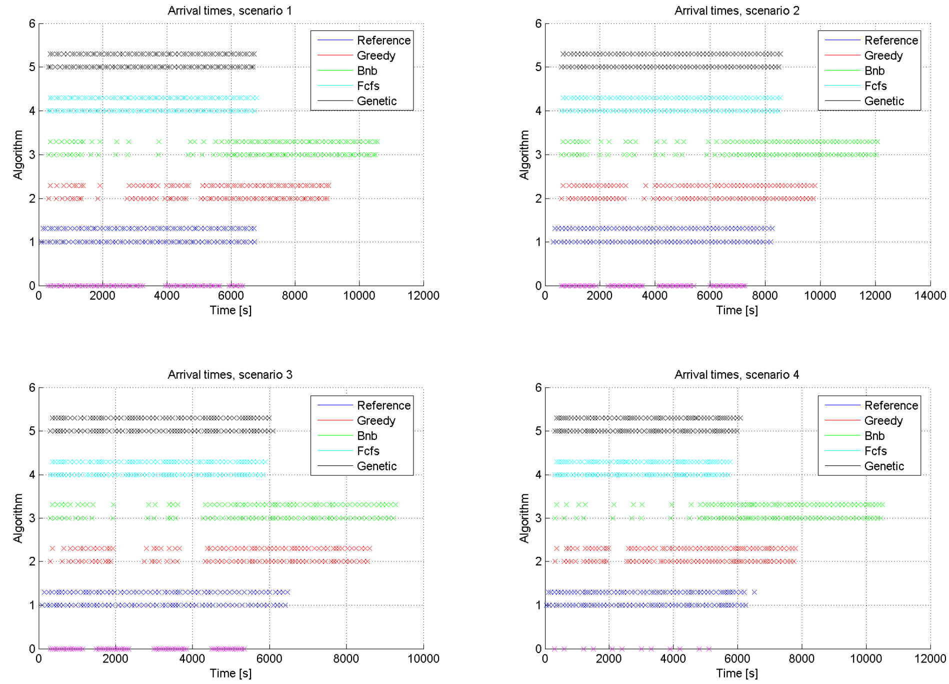

Figure C.11: Arrival times when the step size is varied

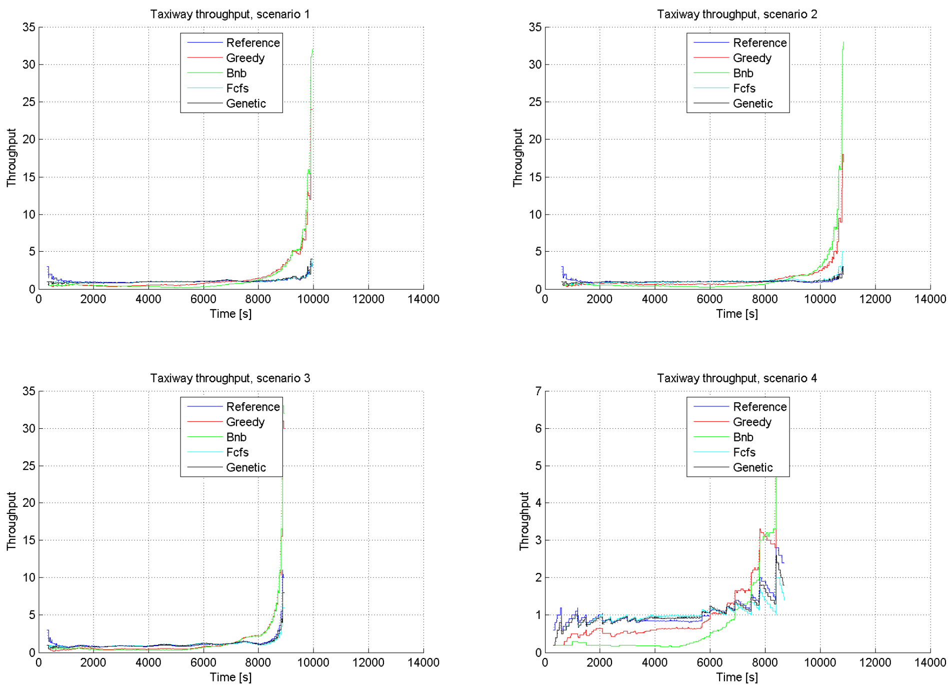

Figure C.12: Taxiway throughput when the step size is varied

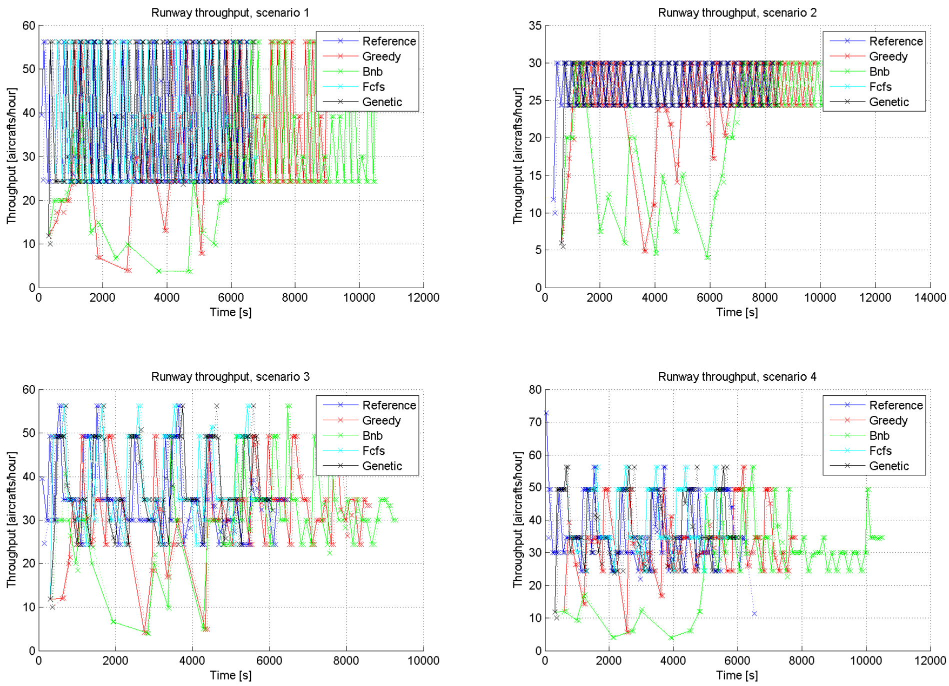

Figure C.13: Runway throughput when the step size is varied

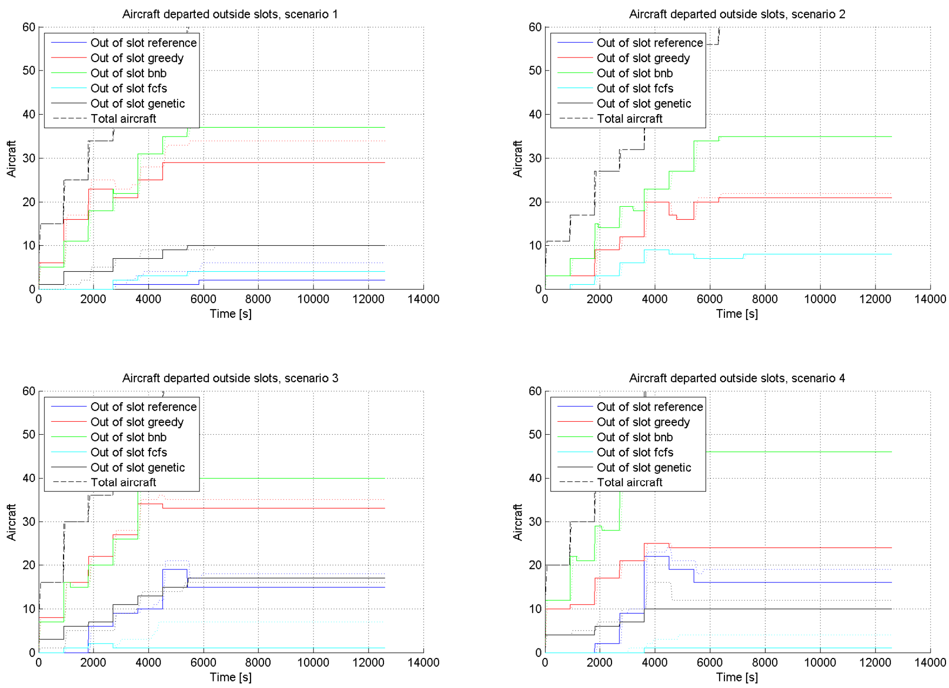

Figure C.14: Aircraft departed out of their slots when the step size is varied

Figure C.15 up to Figure C.21 show the output parameters when the scheduling window is varied as described in Table 5.3. A scheduling window of five seconds is shown as a solid line, fifteen minutes as a dotted line, thirty minutes as a combination of dashes and dots, sixty minutes as a dashed line and ninety and 120 minutes as a solid line again. In Figure C.18, the ideal arrival times as desired by the aircraft are shown as algorithm 0. For each algorithm the lines from bottom to top represent scheduling windows of five seconds, fifteen, thirty, sixty, ninety and 120 minutes respectively.

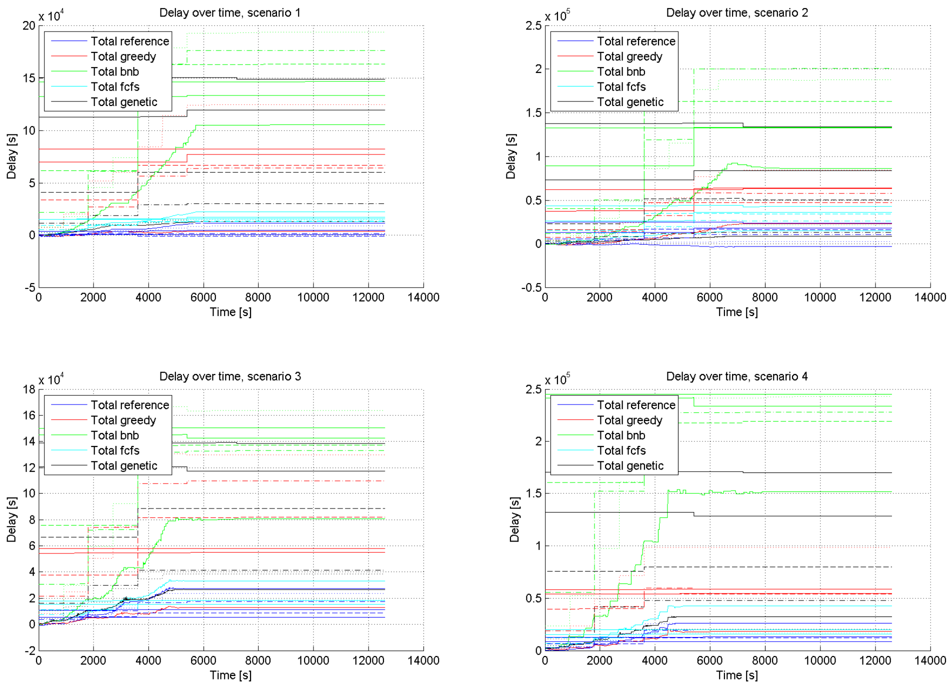

Figure C.15: Total delay over time when the scheduling window is varied

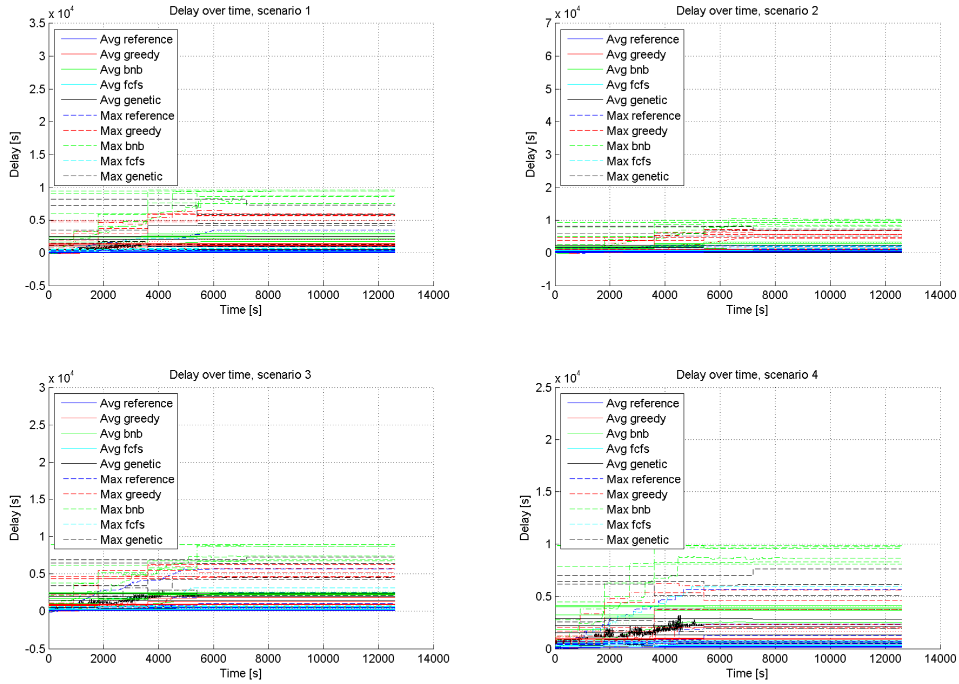

Figure C.16: Maximum and average individual delay when the scheduling window is varied

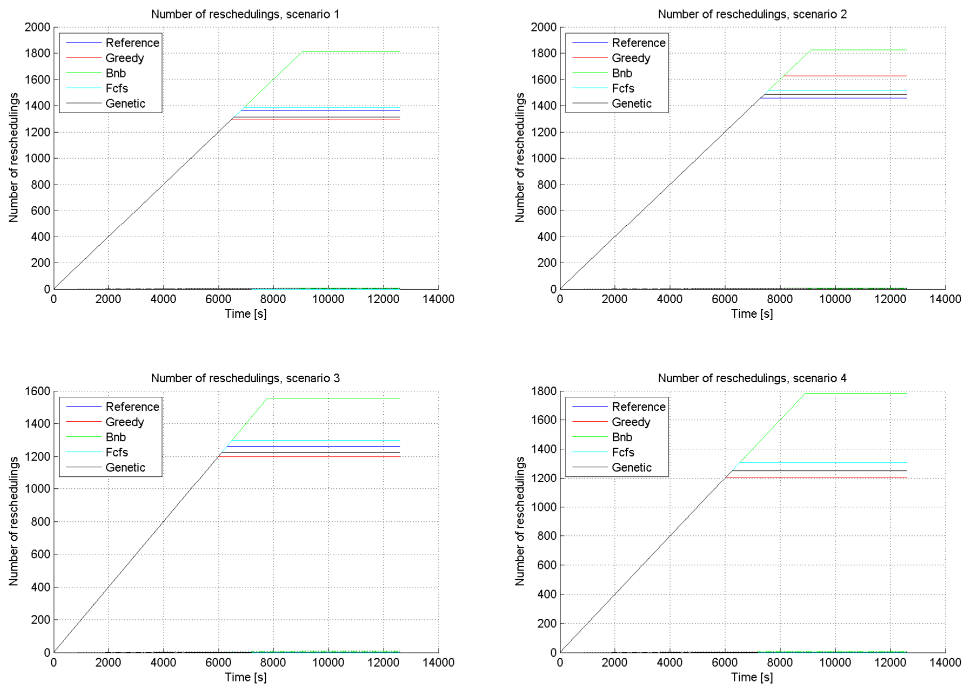

Figure C.17: Number of rescheduling operations when the scheduling window is varied

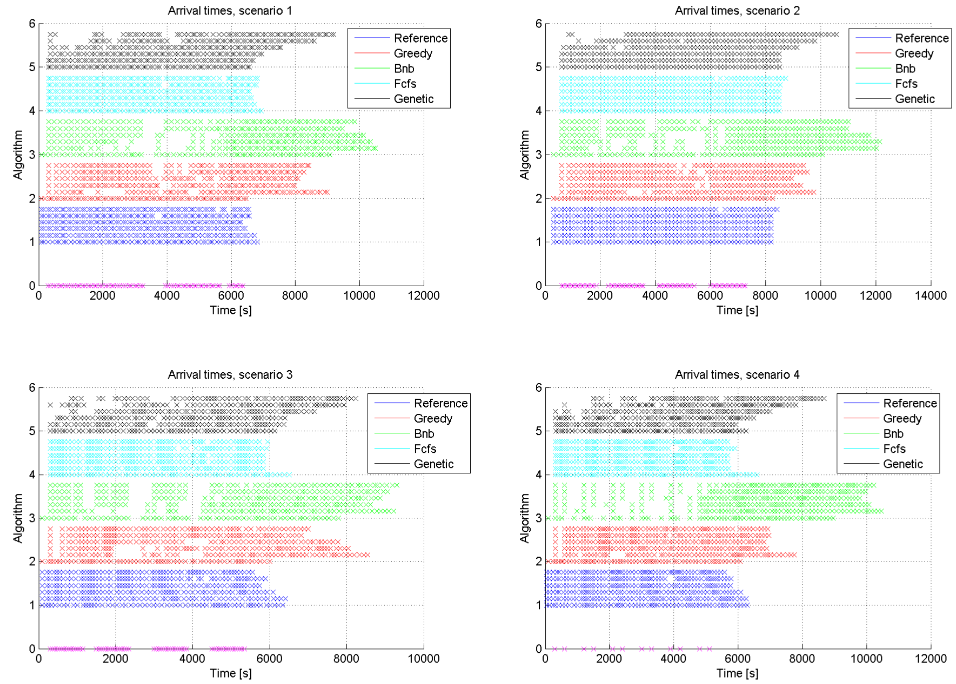

Figure C.18: Arrival times when the scheduling window is varied

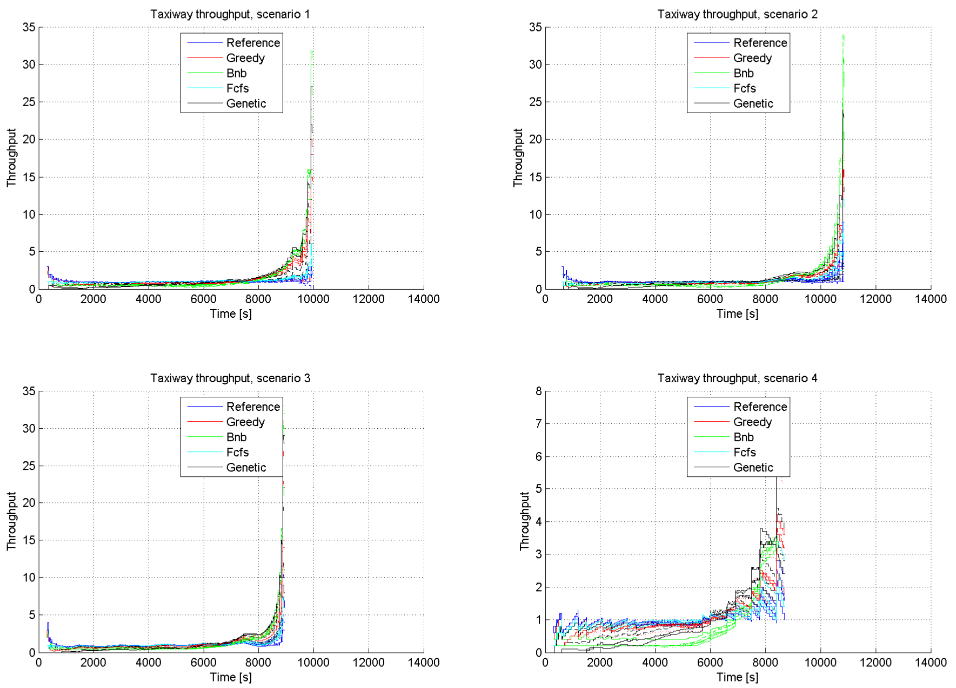

Figure C.19: Taxiway throughput when the scheduling window is varied

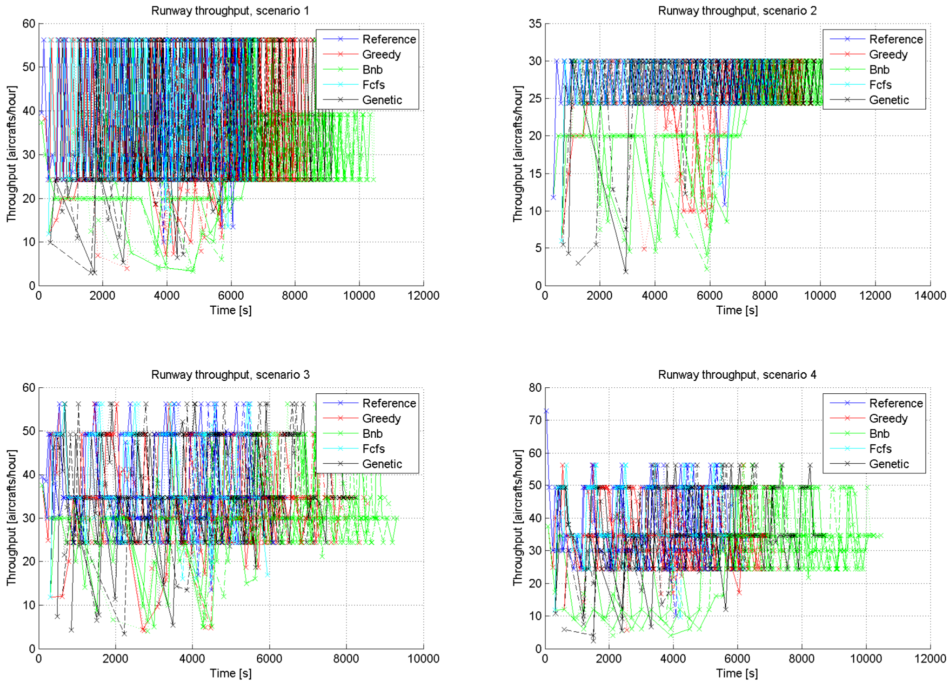

Figure C.20: Runway throughput when the scheduling window is varied

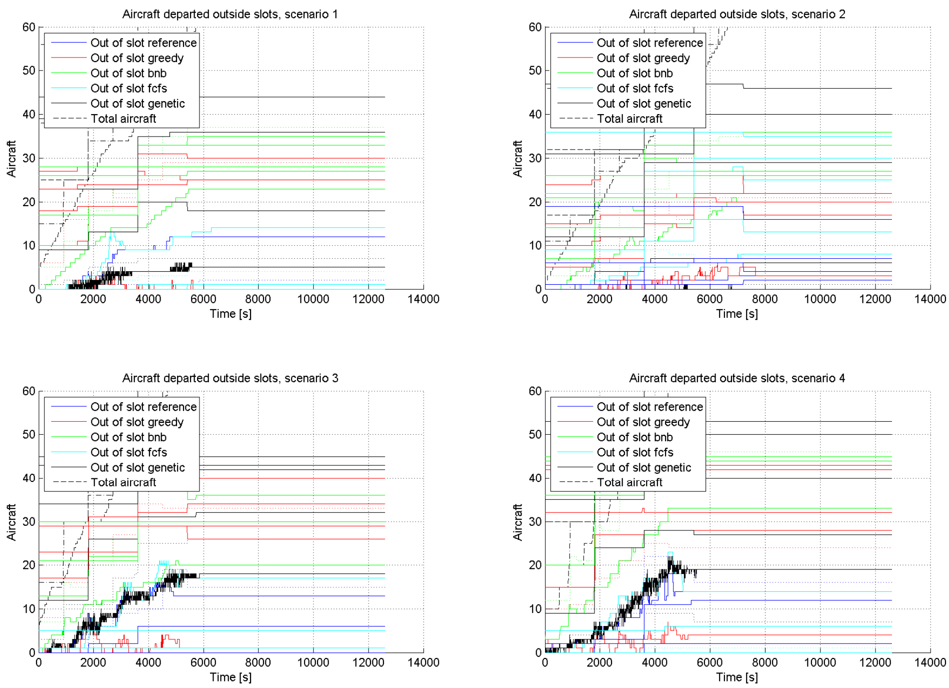

Figure C.21: Aircraft departed out of their slots when the scheduling window is varied

Figure C.22 up to Figure C.28 show the output parameters when an average delay of 6:15 minutes is added to the system as described in Table 5.3. The original situation without delay is shown as a solid line. Only gate delay as a dotted line, only taxi delay as a combination of dashes and dots and both gate and taxi delay as a dashed line. In Figure C.25, for each algorithm the lines from bottom to top represent no delay, only gate delay, only taxi delay and both gate and taxi delay respectively.

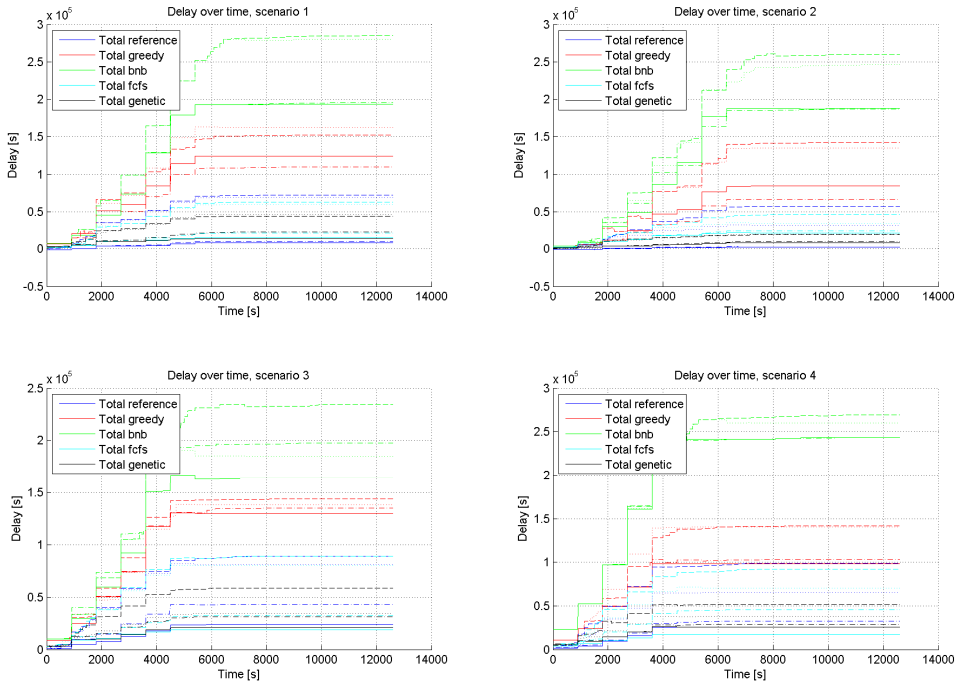

Figure C.22: Total delay over time when an average disturbance of 6:15 minutes is added to the system

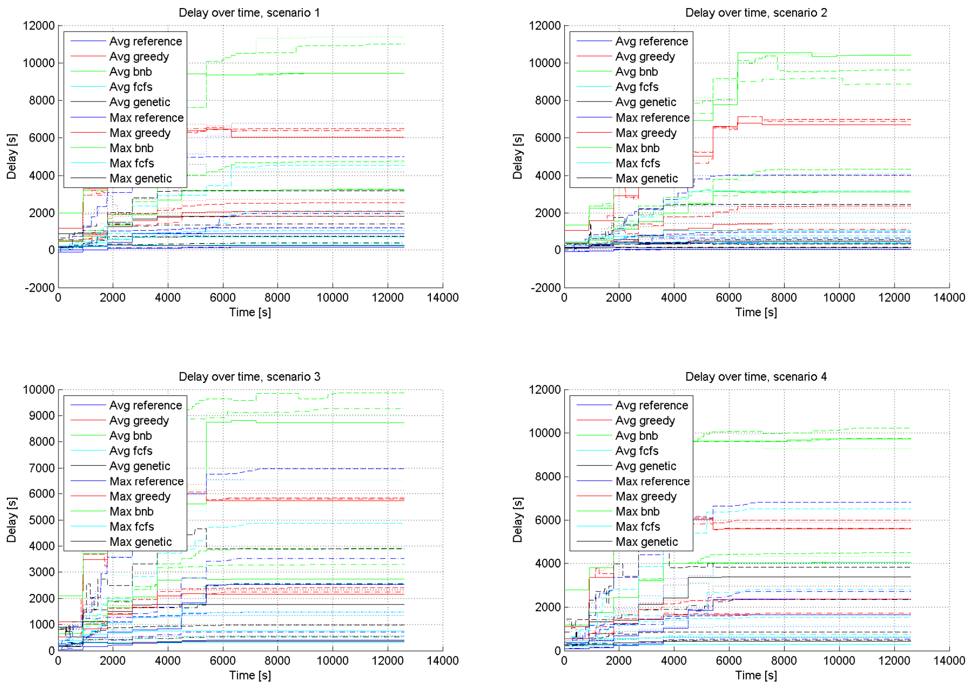

Figure C.23: Maximum and average individual delay when an average disturbance of 6:15 minutes is added to the system

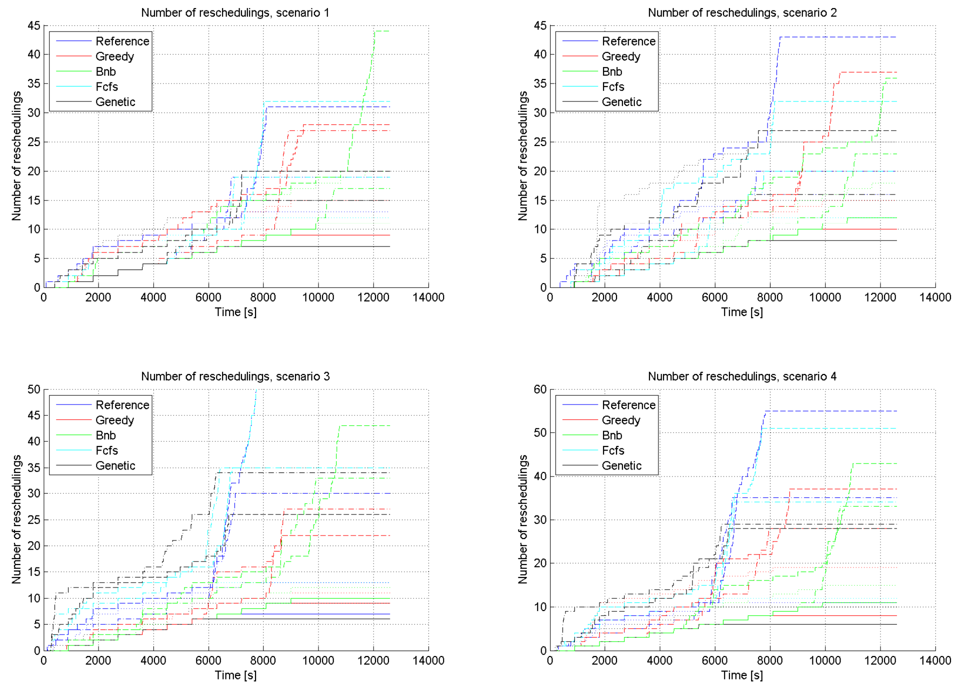

Figure C.24: Number of rescheduling operations when an average disturbance of 6:15 minutes is added to the system

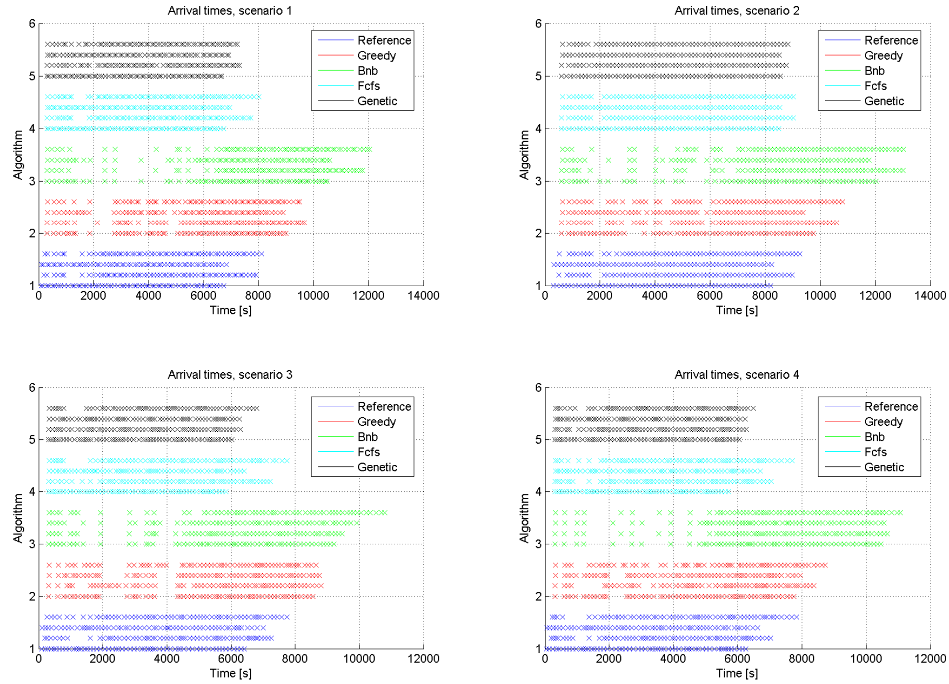

Figure C.25: Arrival times when an average disturbance of 6:15 minutes is added to the system

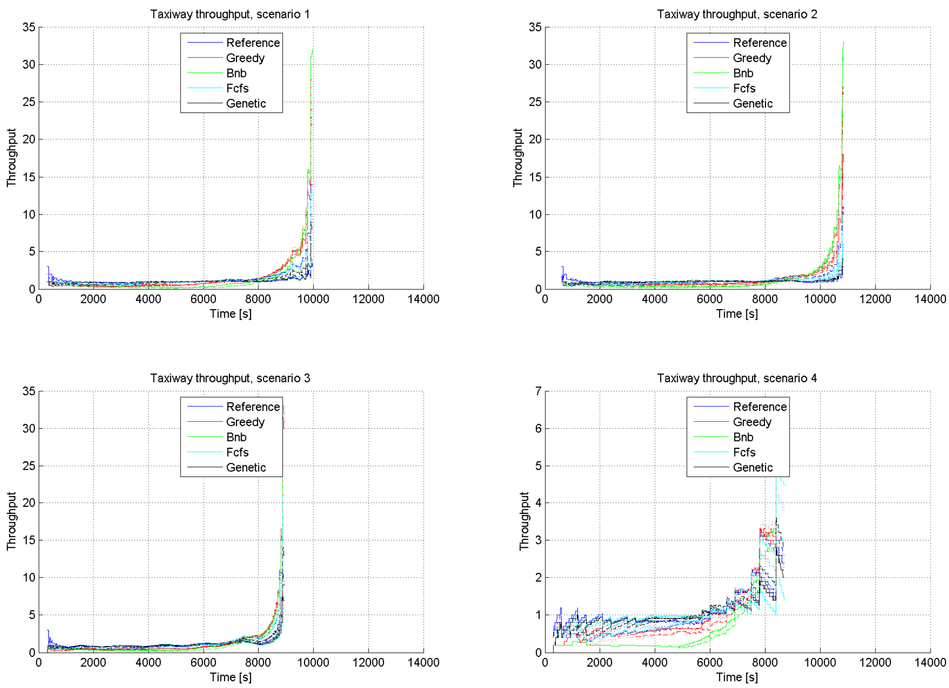

Figure C.26: Taxiway throughput when an average disturbance of 6:15 minutes is added to the system

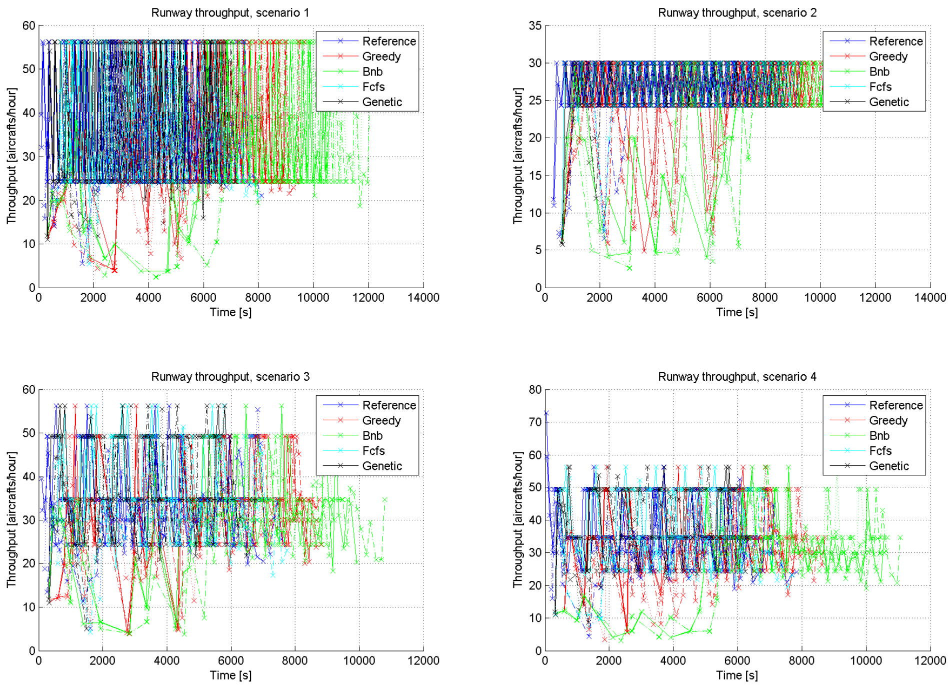

Figure C.27: Runway throughput when an average disturbance of 6:15 minutes is added to the system

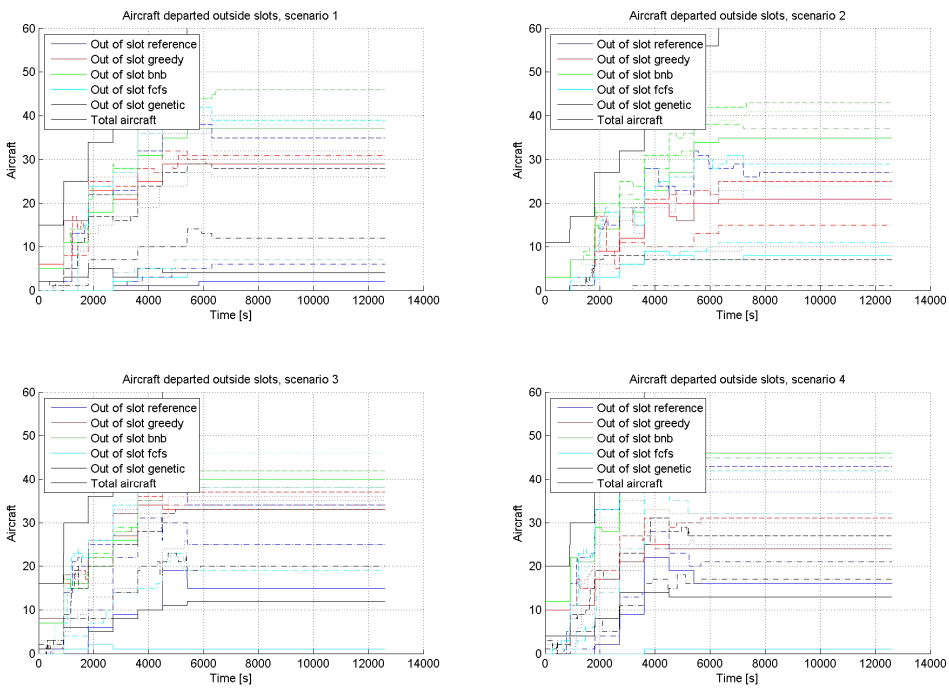

Figure C.28: Aircraft departed out of their slots when an average disturbance of 6:15 minutes is added to the system

Figure C.29 up to Figure C.35 show the output parameters when an average delay of 12:30 minutes is added to the system as described in Table 5.3. The original situation without delay is shown as a solid line. Only gate delay as a dotted line, only taxi delay as a combination of dashes and dots and both gate and taxi delay as a dashed line. In Figure C.32, for each algorithm the lines from bottom to top represent no delay, only gate delay, only taxi delay and both gate and taxi delay respectively.

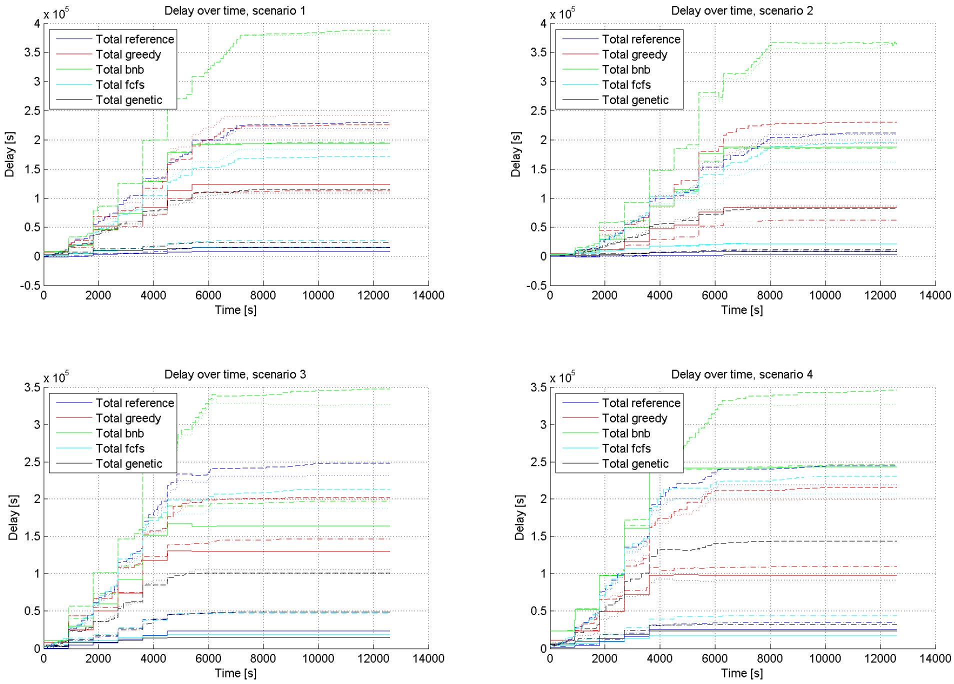

Figure C.29: Total delay over time when an average disturbance of 12:30 minutes is added to the system

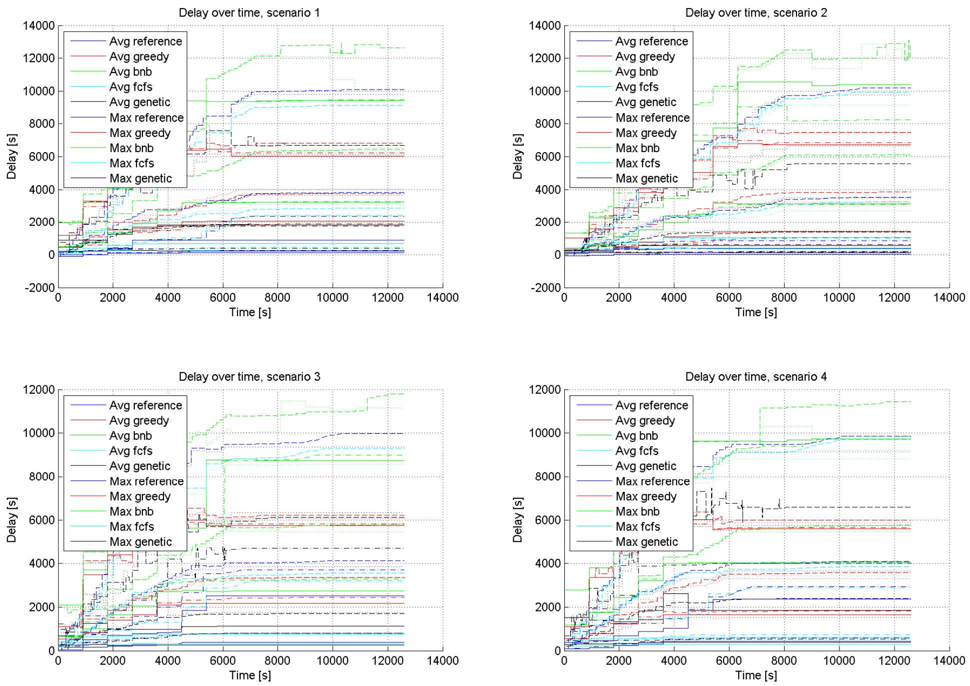

Figure C.30: Maximum and average individual delay when an average disturbance of 12:30 minutes is added to the system

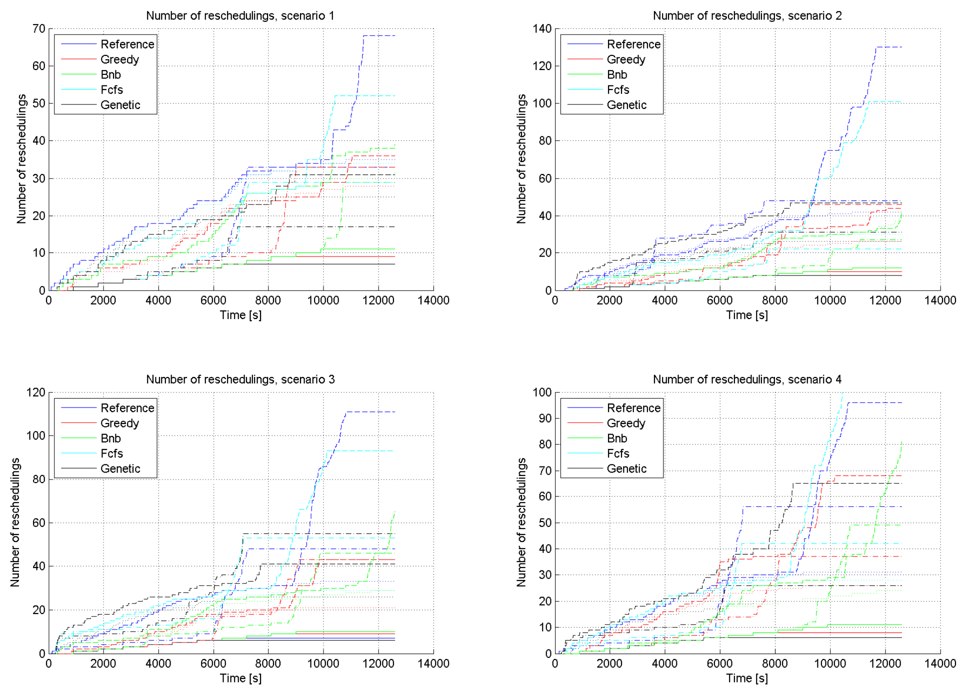

Figure C.31: Number of rescheduling operations when an average disturbance of 12:30 minutes is added to the system

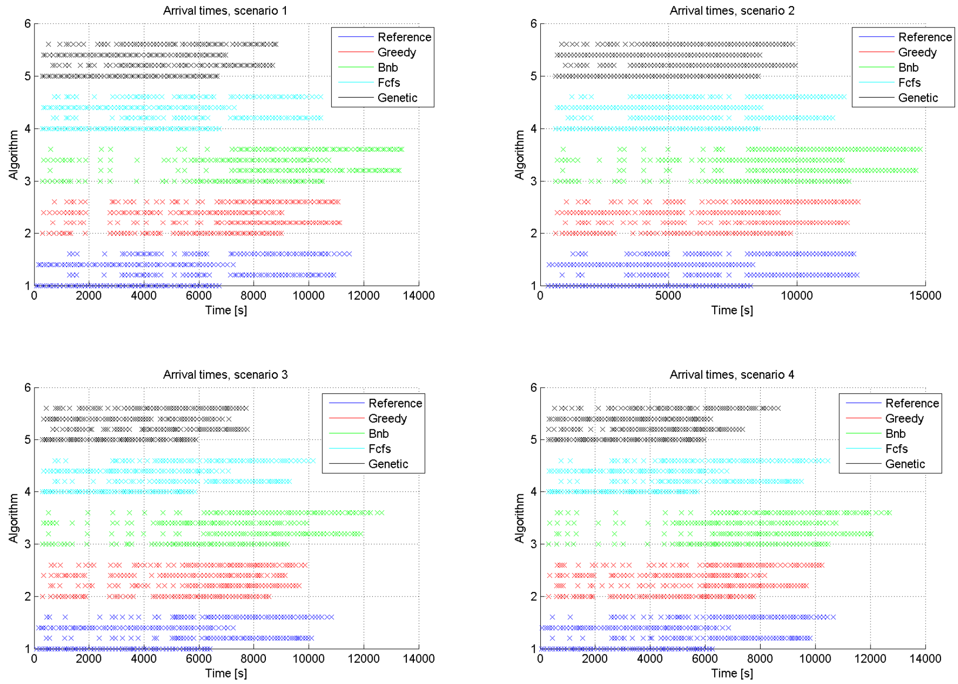

Figure C.32: Arrival times when an average disturbance of 12:30 minutes is added to the system

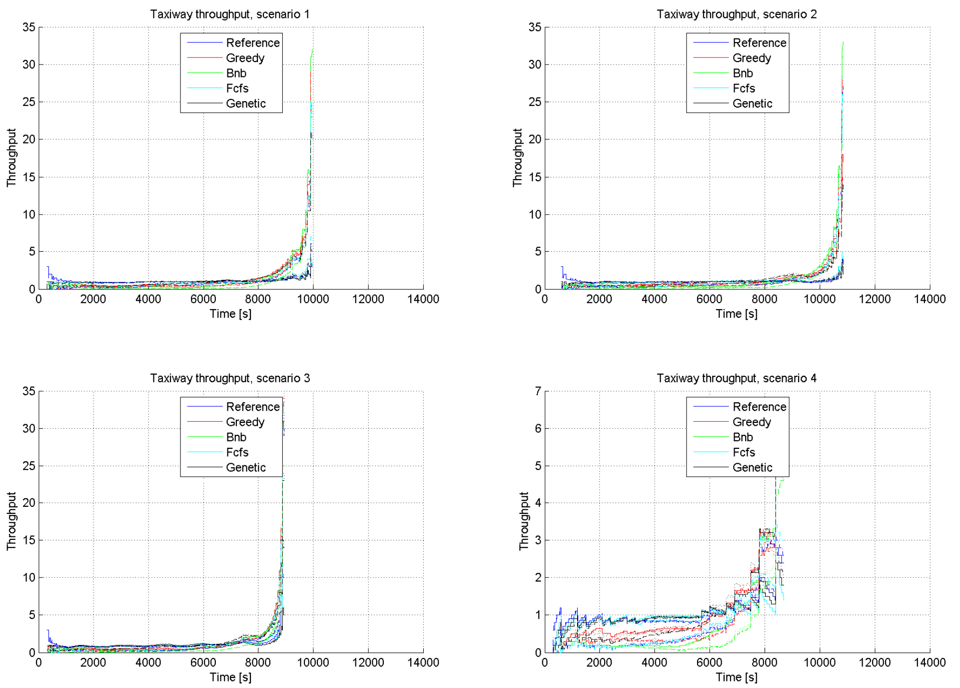

Figure C.33: Taxiway throughput when an average disturbance of 12:30 minutes is added to the system

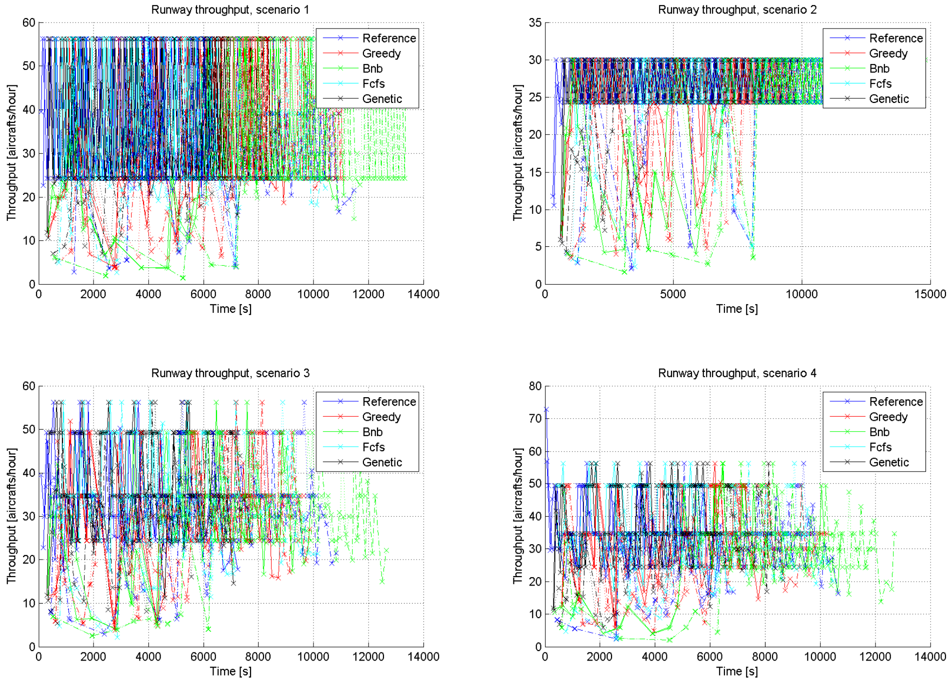

Figure C.34: Runway throughput when an average disturbance of 12:30 minutes is added to the system

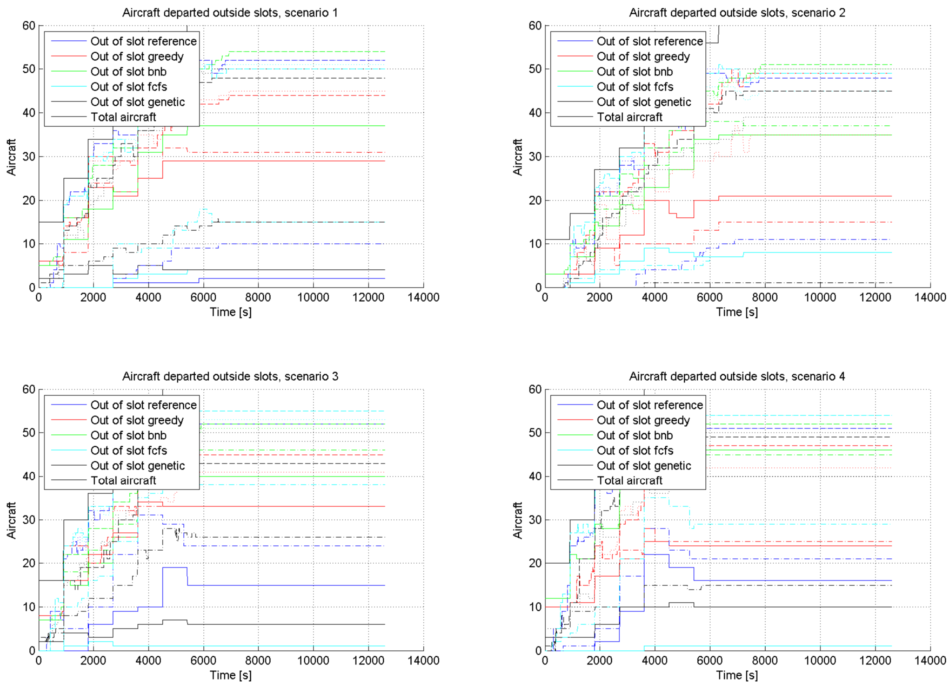

Figure C.35: Aircraft departed out of their slots when an average disturbance of 12:30 minutes is added to the system

Figure C.36 up to Figure C.42 show the output parameters when an average delay of 25:00 minutes is added to the system as described in Table 5.3. The original situation without delay is shown as a solid line. Only gate delay as a dotted line, only taxi delay as a combination of dashes and dots and both gate and taxi delay as a dashed line. In Figure C.39, for each algorithm the lines from bottom to top represent no delay, only gate delay, only taxi delay and both gate and taxi delay respectively.

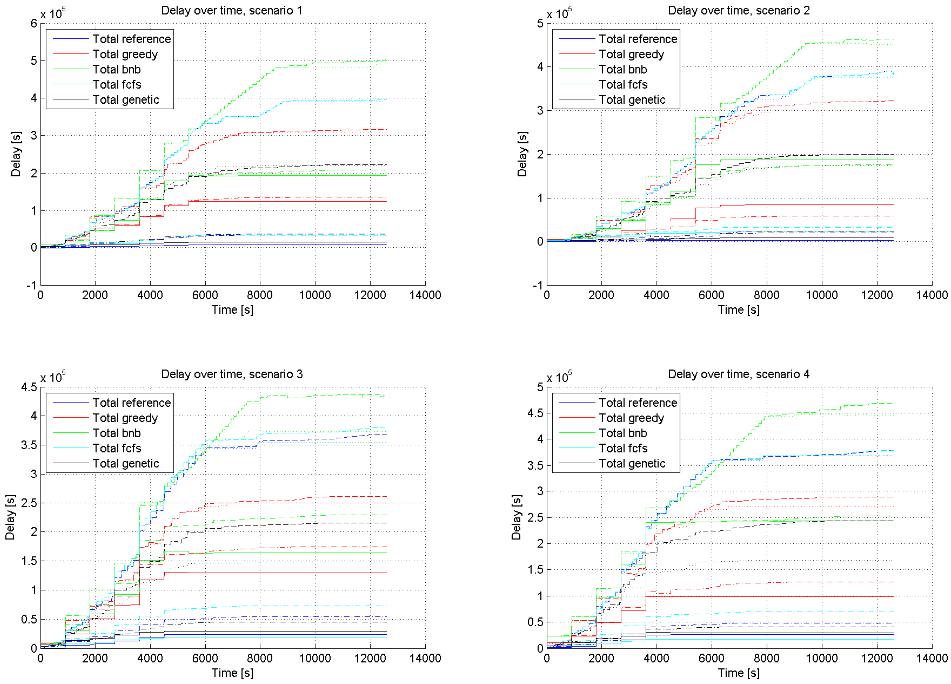

Figure C.36: Total delay over time when an average disturbance of 25:00 minutes is added to the system

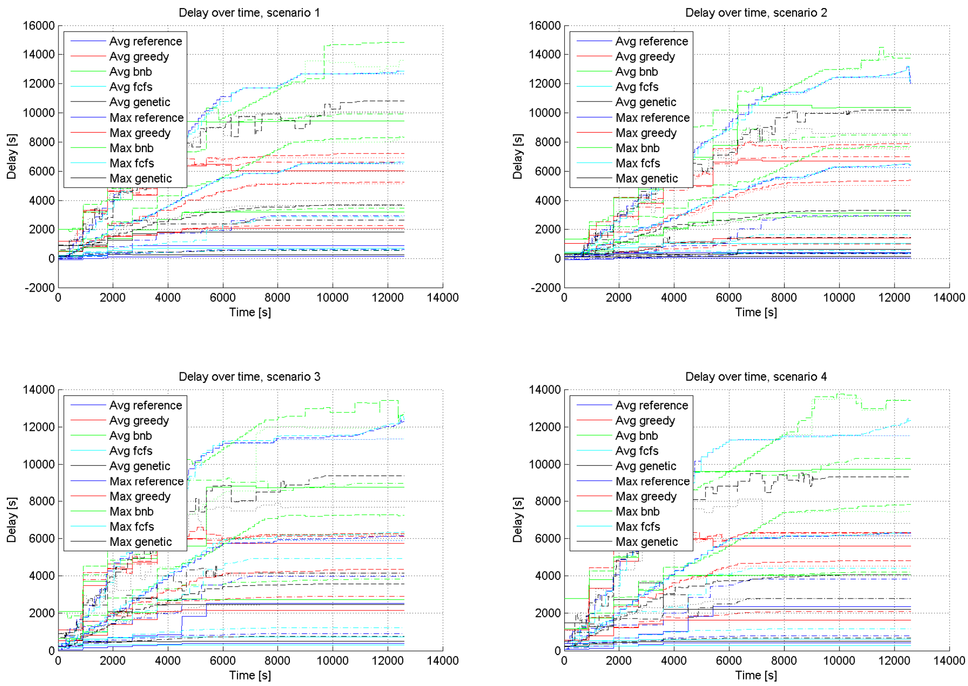

Figure C.37: Maximum and average individual delay when an average disturbance of 25:00 minutes is added to the system

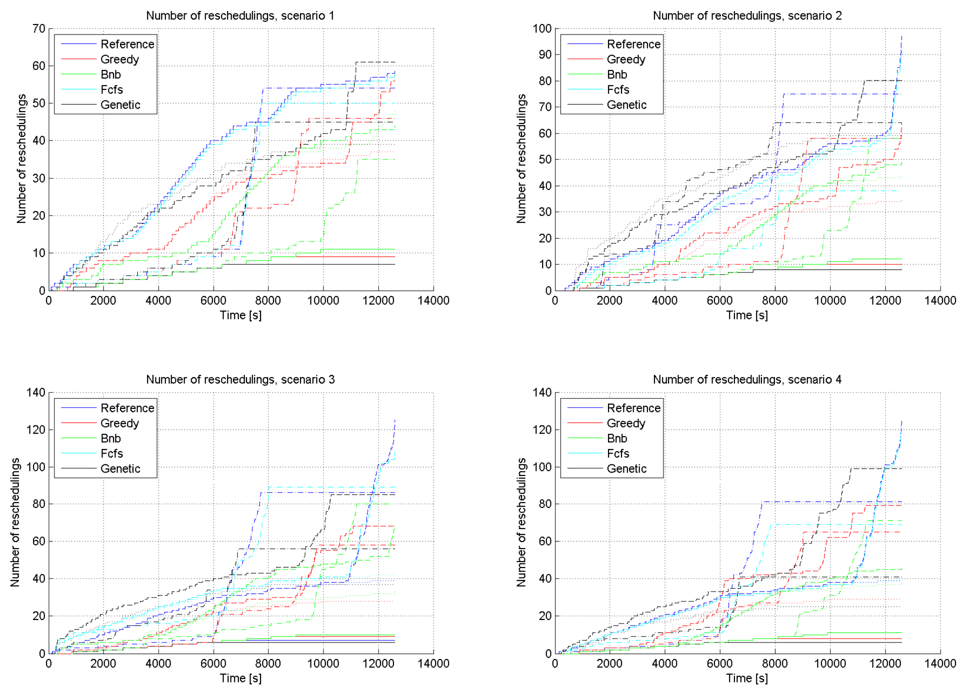

Figure C.38: Number of rescheduling operations when an average disturbance of 25:00 minutes is added to the system

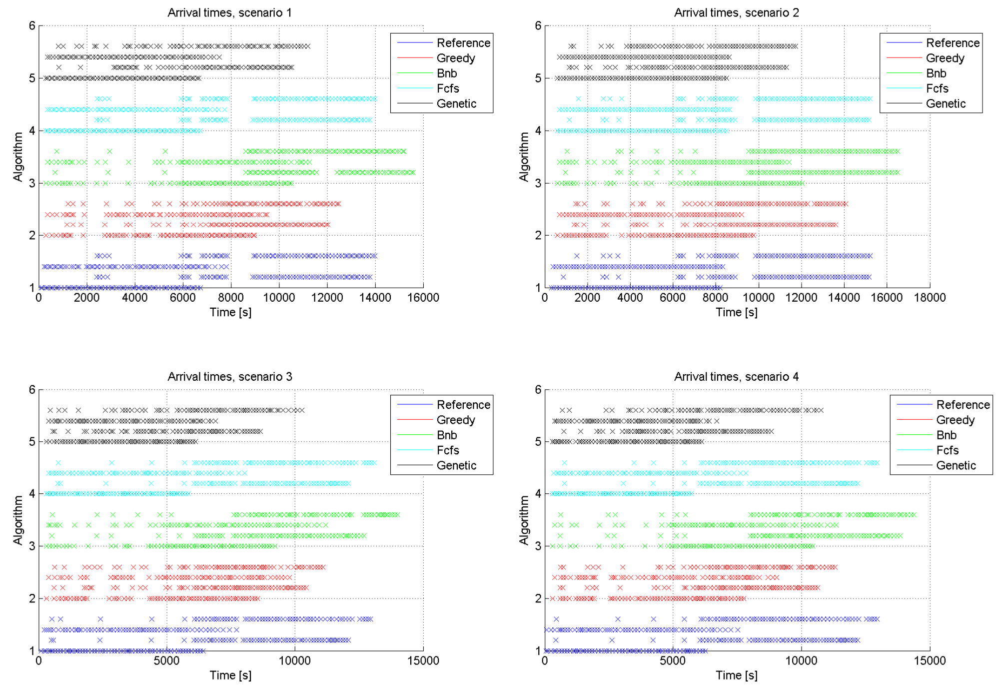

Figure C.39: Arrival times when an average disturbance of 25:00 minutes is added to the system

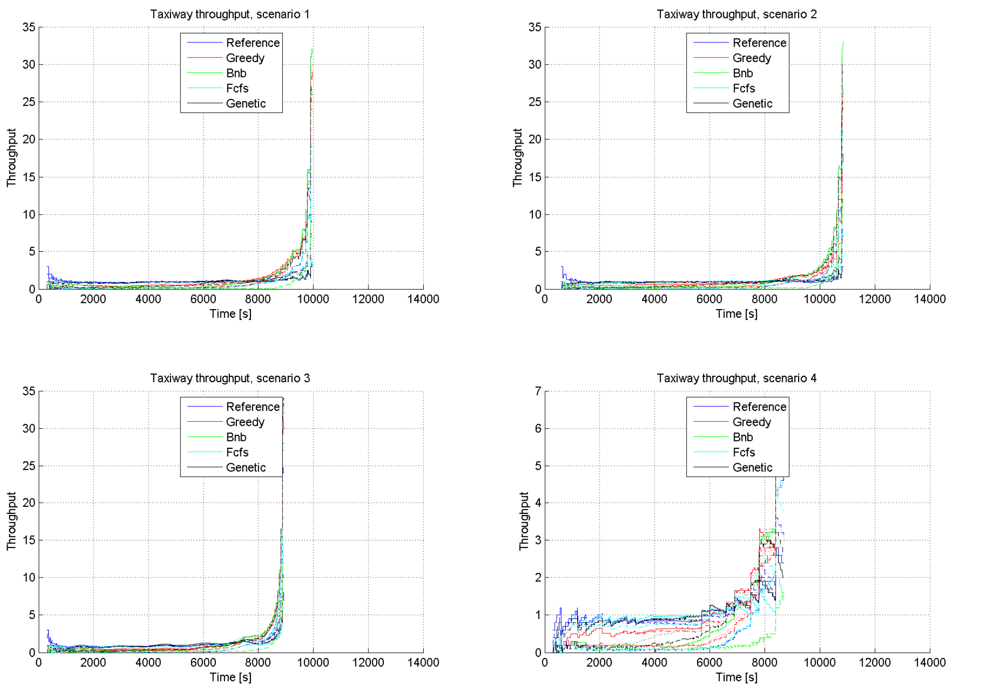

Figure C.40: Taxiway throughput when an average disturbance of 25:00 minutes is added to the system

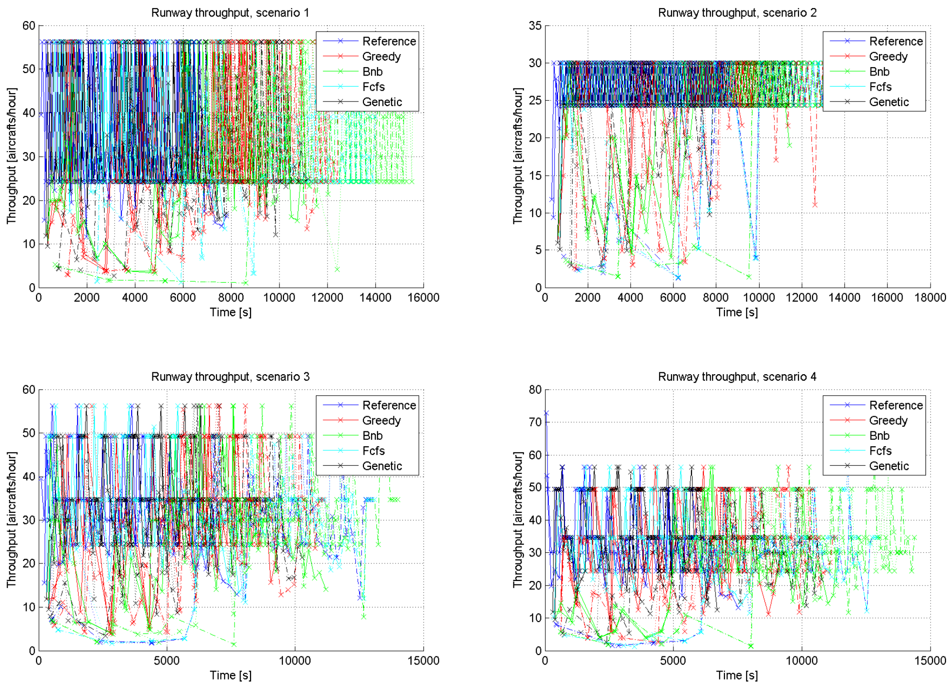

Figure C.41: Runway throughput when an average disturbance of 25:00 minutes is added to the system

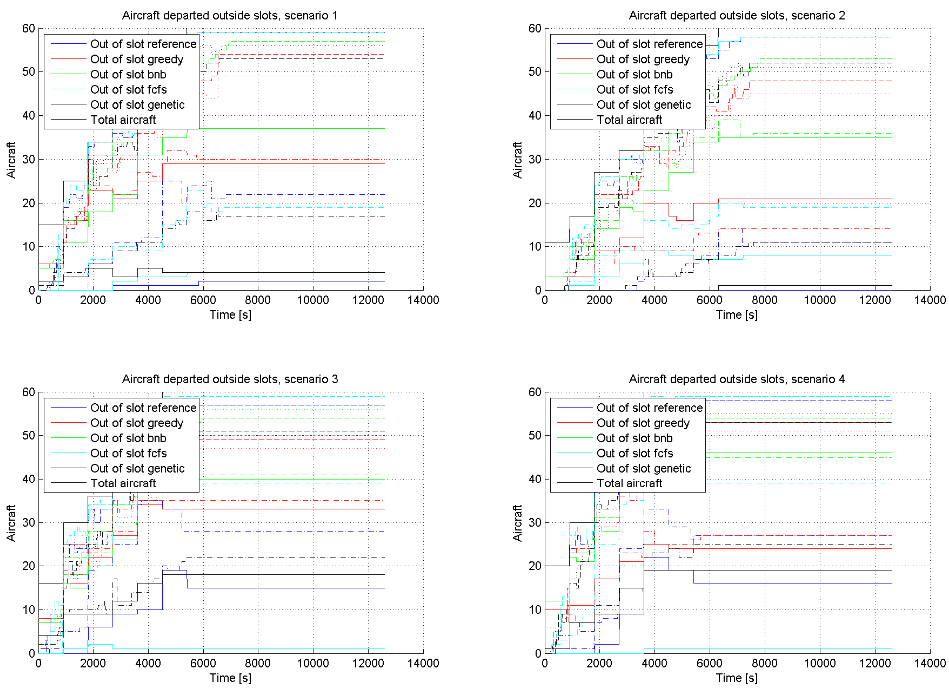

Figure C.42: Aircraft departed out of their slots when an average disturbance of 25:00 minutes is added to the system

Figure C.43 up to Figure C.49 show the output parameters when the boundaries of the departure slots are varied as described in Table 5.3. A slot size of -5 +10 minutes is shown as a solid line, -1 +1 minutes as a dotted line, -1 +2 minutes as a combination of dashes and dots, -5 +5 as a dashed line and -10 +10 and -10 +15 minutes as a solid line again. In Figure C.46, the ideal arrival times as desired by the aircraft are shown as algorithm 0. For each algorithm the lines from bottom to top represent slot sizes of -5 +10, -1 +1, -1 +2, -5 +5, -10 +10 and -10 +15 minutes respectively.

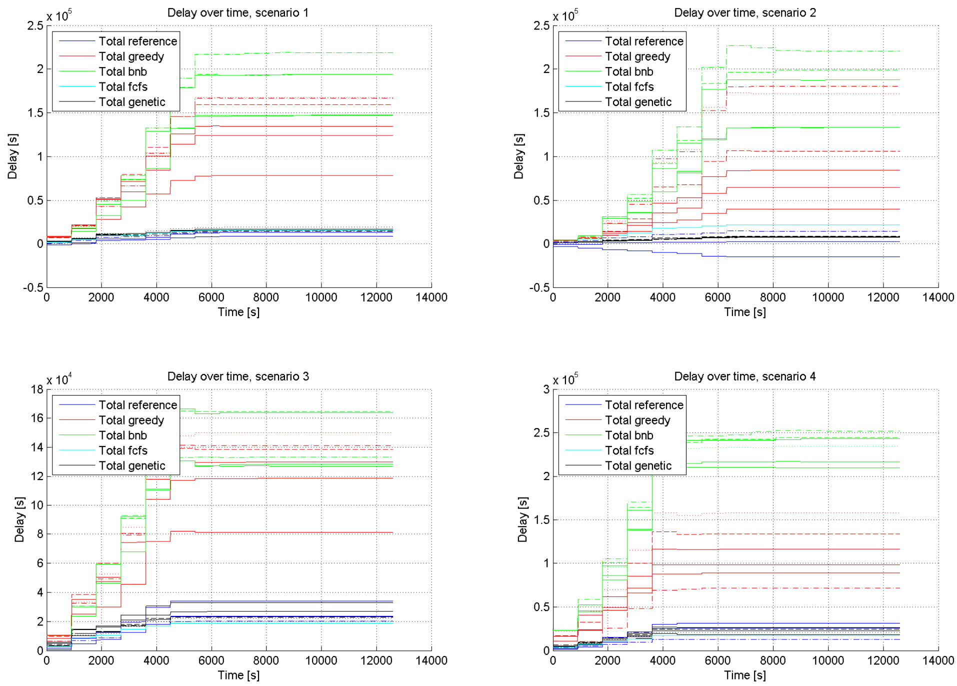

Figure C.43: Total delay over time when the slot boundaries are varied

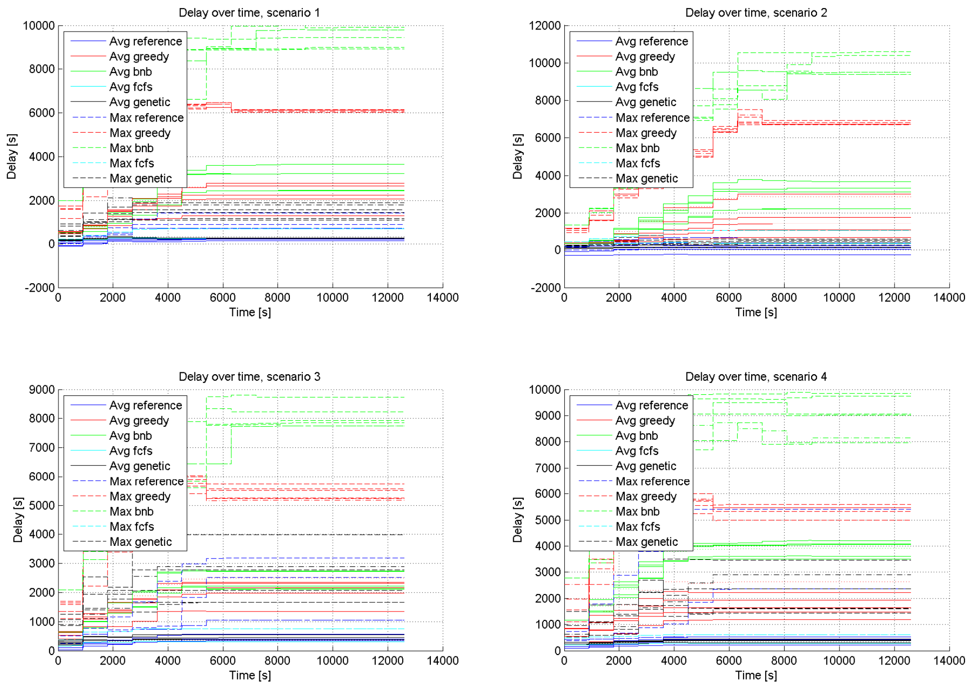

Figure C.44: Maximum and average individual delay when the slot boundaries are varied

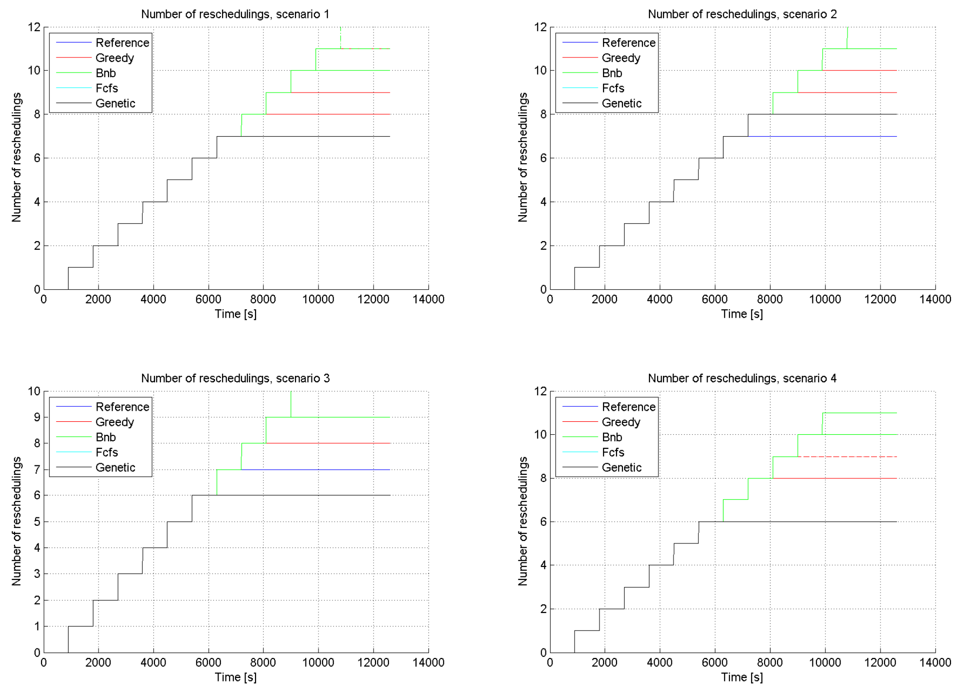

Figure C.45: Number of rescheduling operations when the slot boundaries are varied

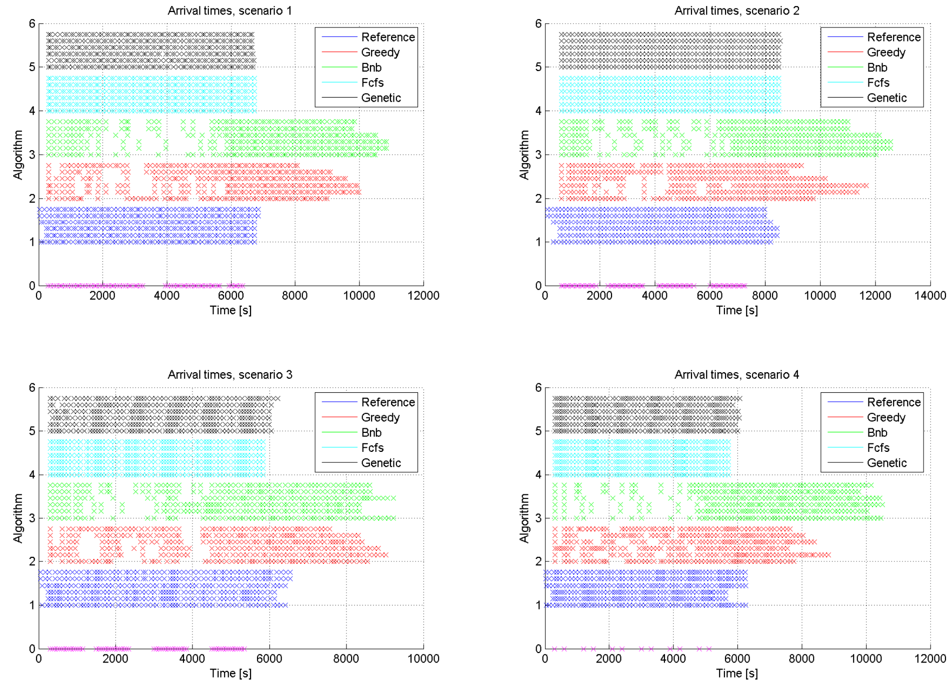

Figure C.46: Arrival times when the slot boundaries are varied

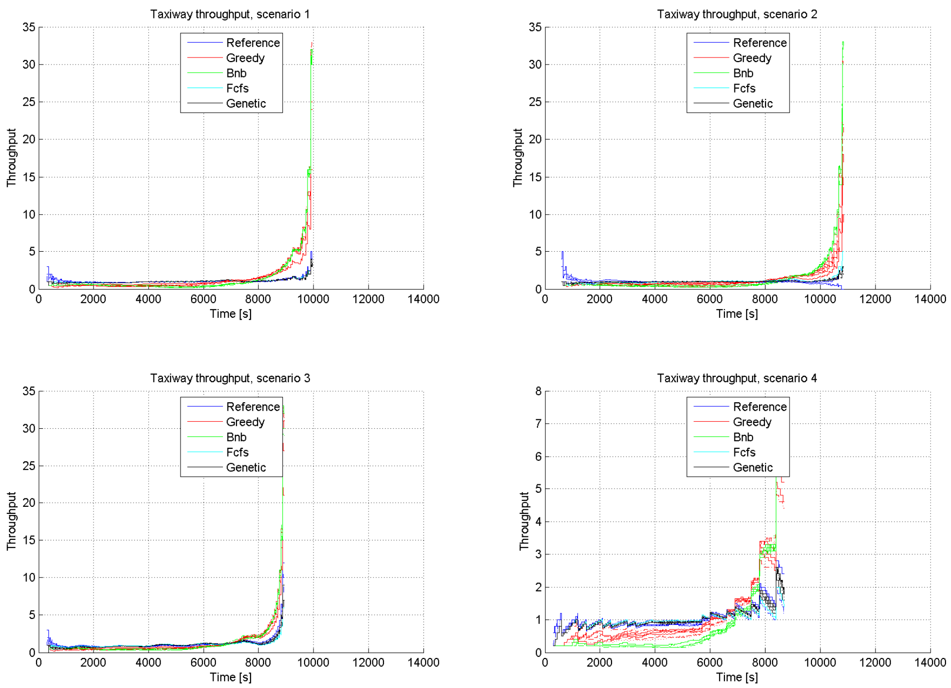

Figure C.47: Taxiway throughput when the slot boundaries are varied

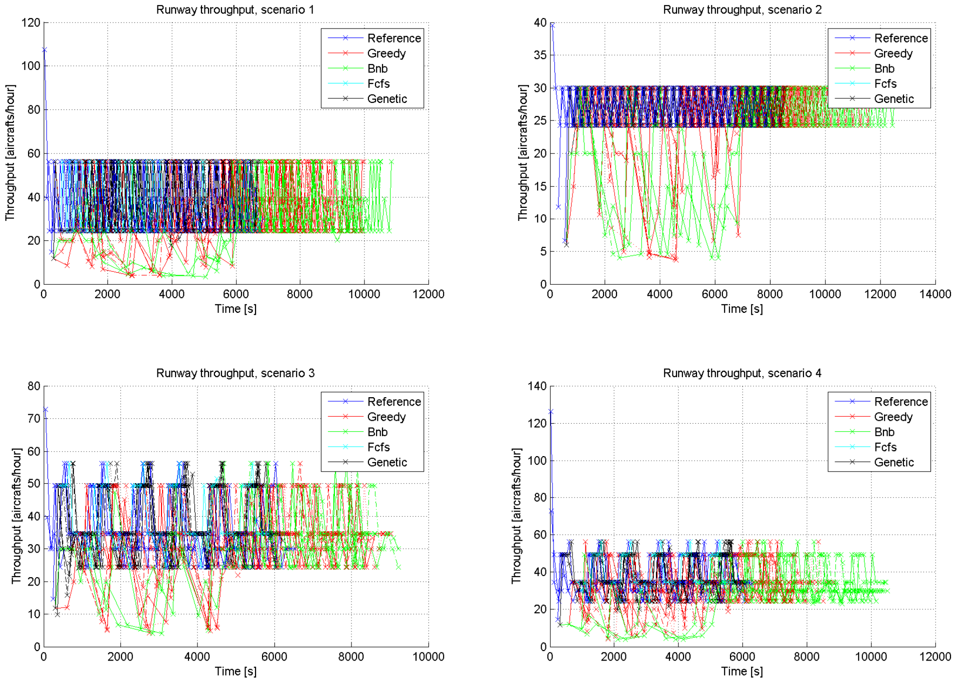

Figure C.48: Runway throughput when the slot boundaries are varied

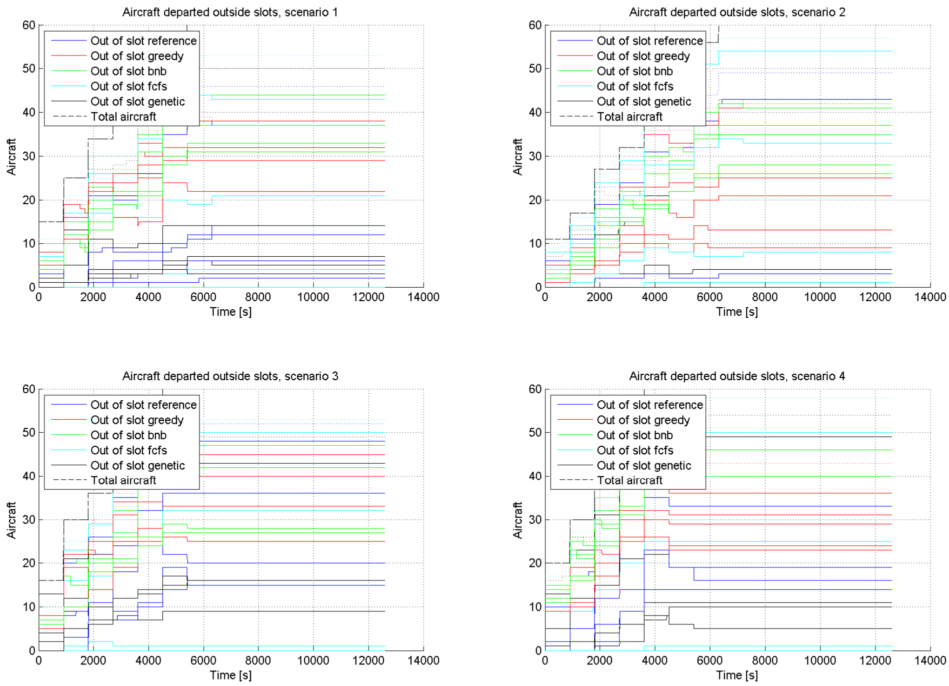

Figure C.49: Aircraft departed out of their slots when the slot boundaries are varied

Figure C.50 up to Figure C.56 show the output parameters when the aircraft based speed constraints of six aircraft (10% of the aircraft) are varied as described in Table 5.3. A disturbance to 10% of the original speed is shown as a solid line, to 50% as a dotted line, to 75% of the original speed as a combination of dashes and dots and no disturbance is shown as a dashed line. In Figure C.53, the ideal arrival times as desired by the aircraft are shown as algorithm 0. For each algorithm the lines from bottom to top represent disturbances to 10%, 50%, 75% of the original speed and no disturbances respectively.

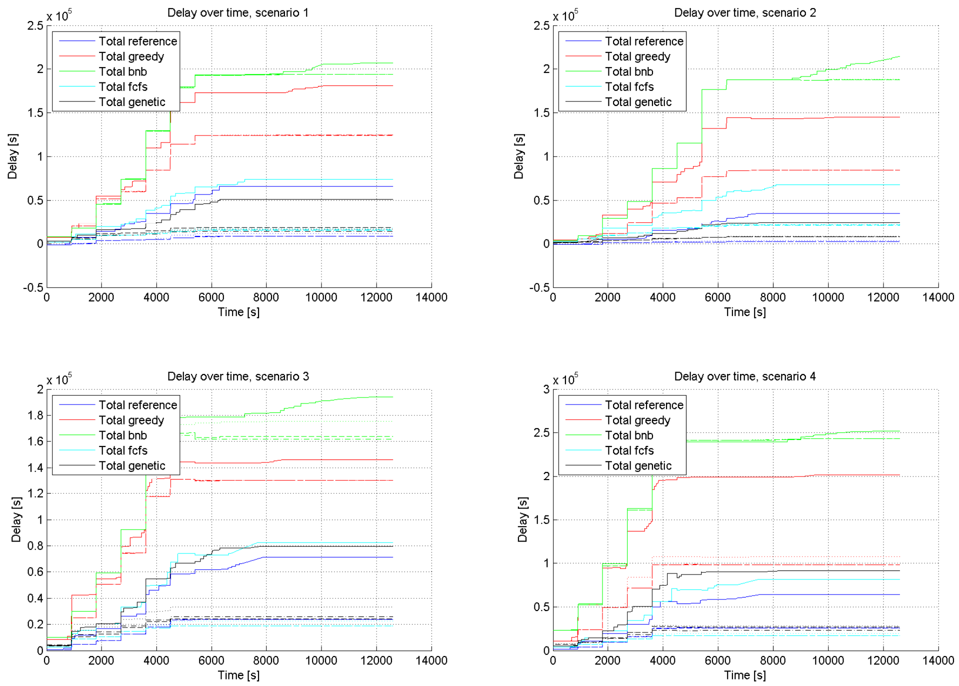

Figure C.50: Total delay over time when the aircraft based speed of six aircraft is disturbed

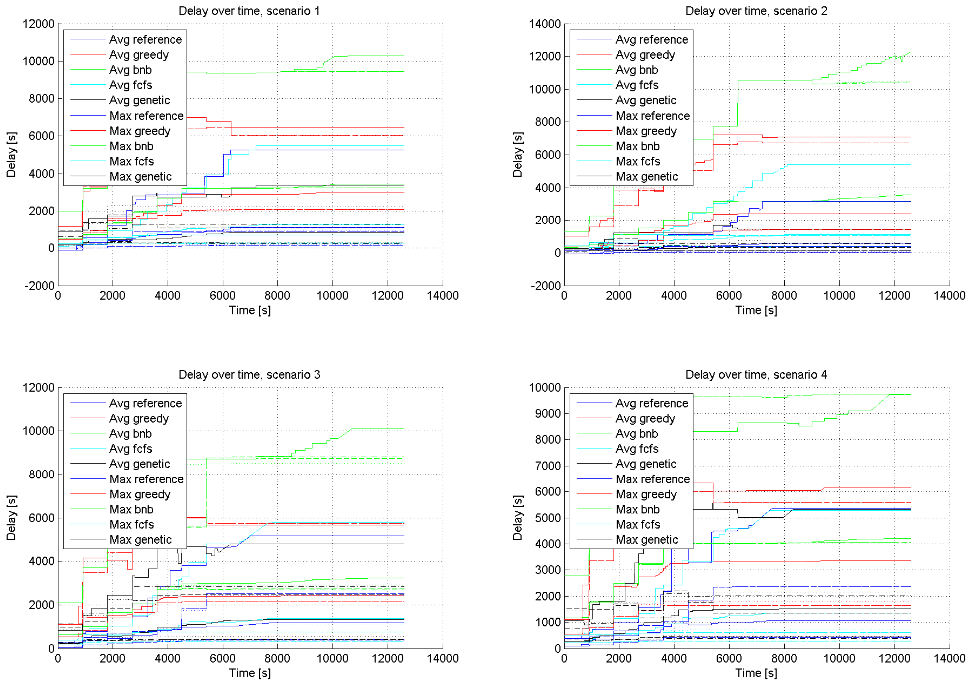

Figure C.51: Maximum and average individual delay over time when the aircraft based speed of six aircraft is disturbed

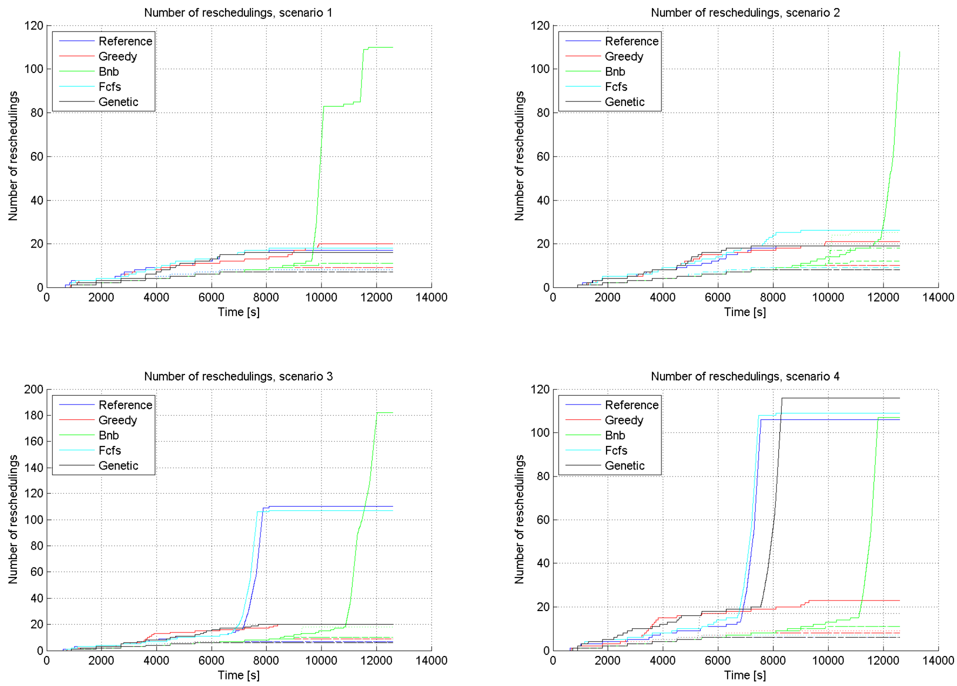

Figure C.52: Number of rescheduling operations when the aircraft based speed of six aircraft is disturbed

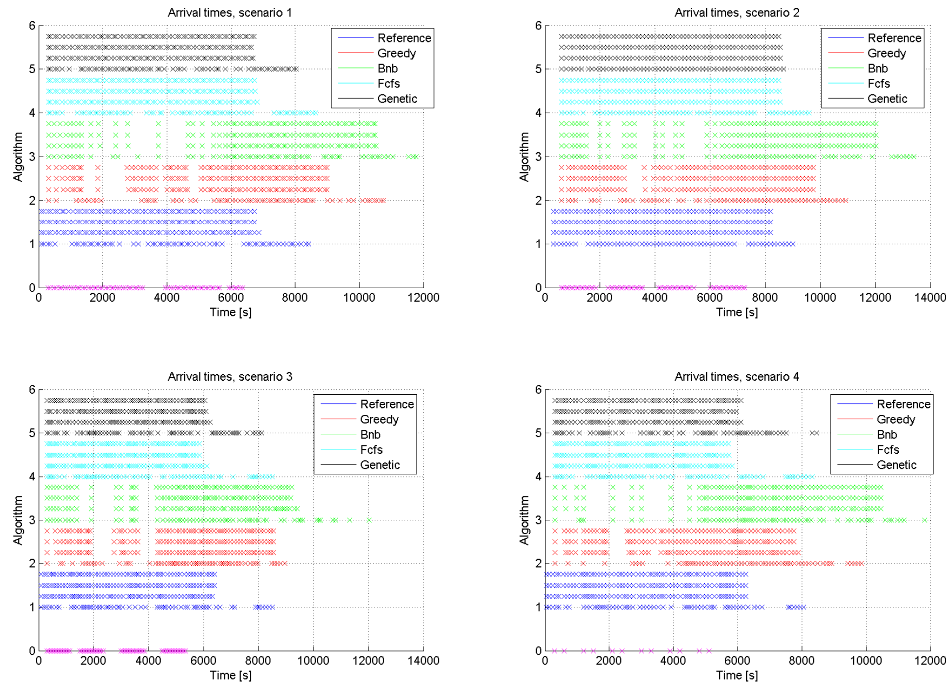

Figure C.53: Arrival times when the aircraft based speed of six aircraft is disturbed

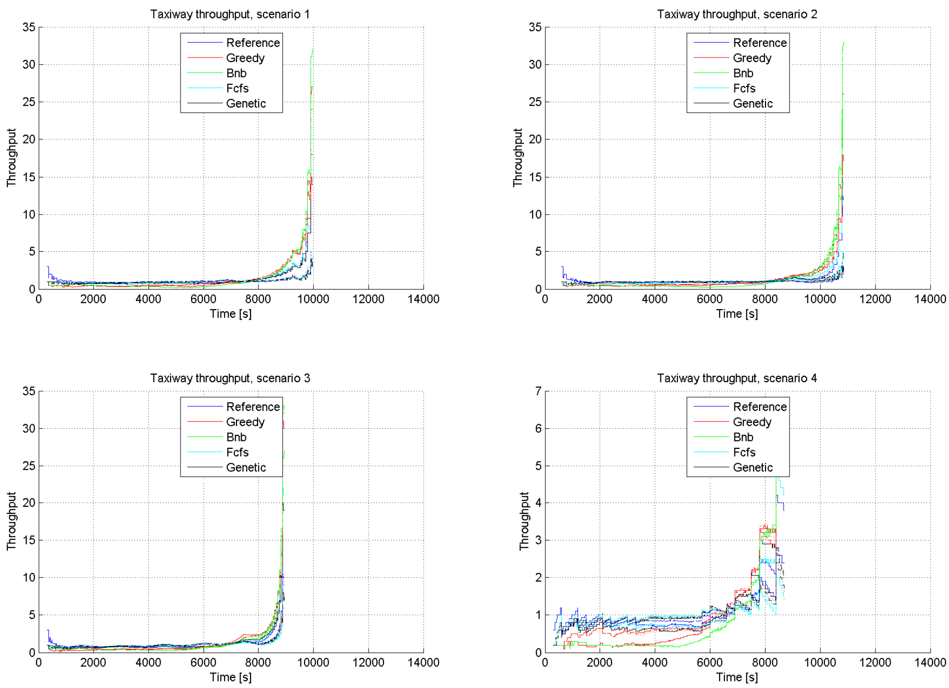

Figure C.54: Taxiway throughput when the aircraft based speed of six aircraft is disturbed

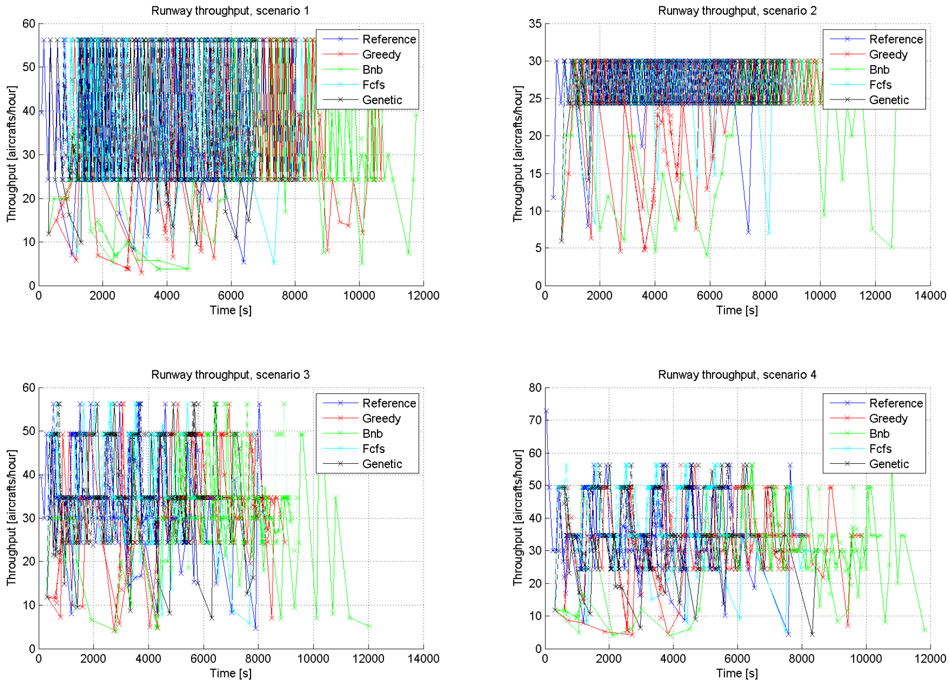

Figure C.55: Runway throughput when the aircraft based speed of six aircraft is disturbed

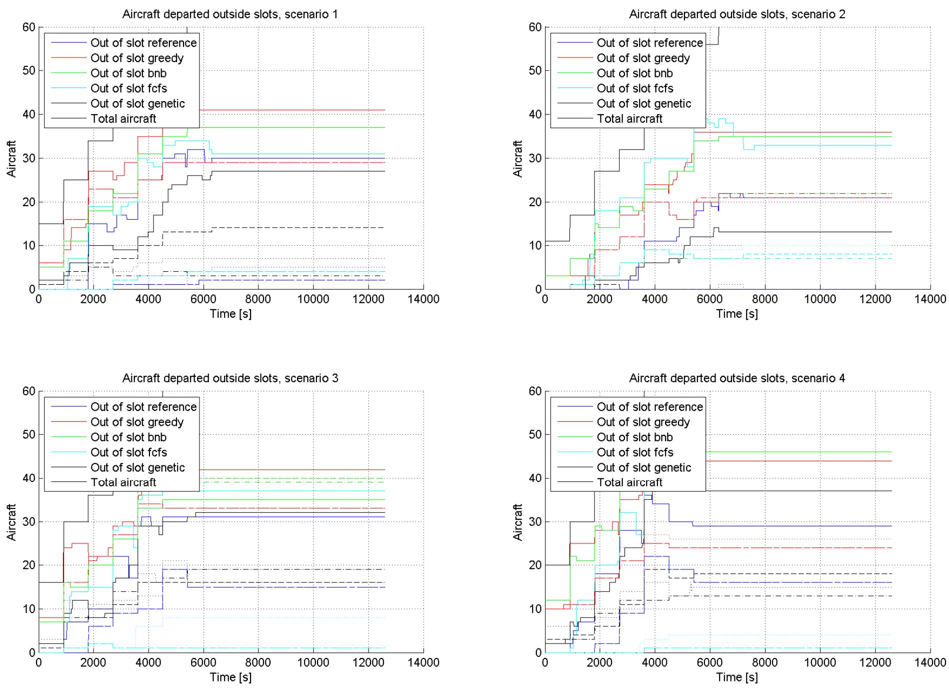

Figure C.56: Aircraft departed out of their slots when the aircraft based speed of six aircraft is disturbed

Figure C.57 up to Figure C.63 show the output parameters when the aircraft based speed constraints of twelve aircraft (20% of the aircraft) are varied as described in Table 5.3. A disturbance to 10% of the original speed is shown as a solid line, to 50% as a dotted line, to 75% of the original speed as a combination of dashes and dots and no disturbance is shown as a dashed line. In Figure C.60, the ideal arrival times as desired by the aircraft are shown as algorithm 0. For each algorithm the lines from bottom to top represent disturbances to 10%, 50%, 75% of the original speed and no disturbances respectively.

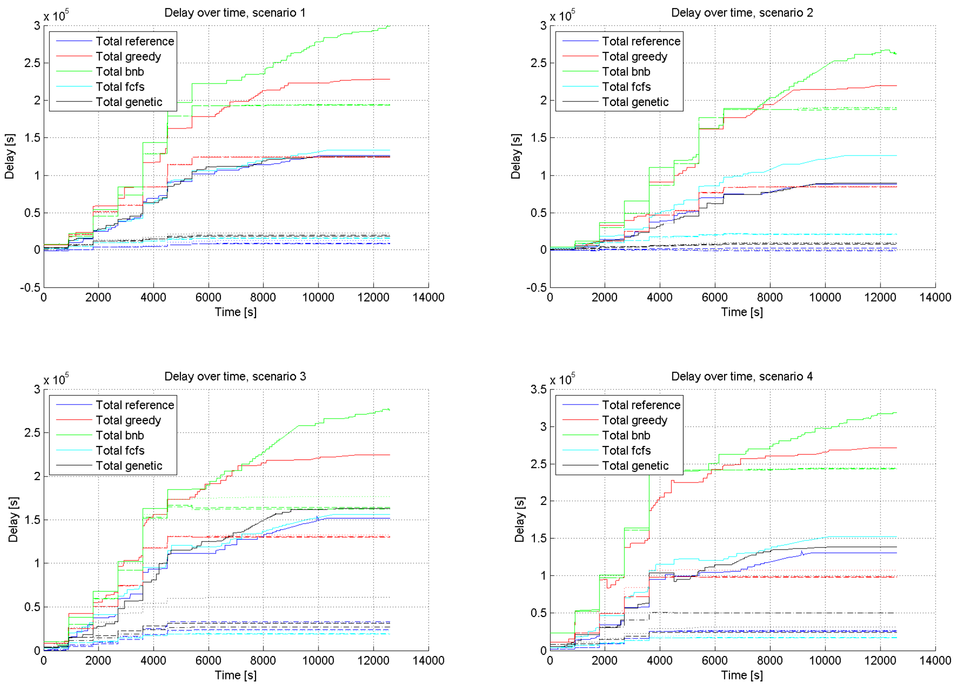

Figure C.57: Total delay over time when the aircraft based speed of twelve aircraft is disturbed

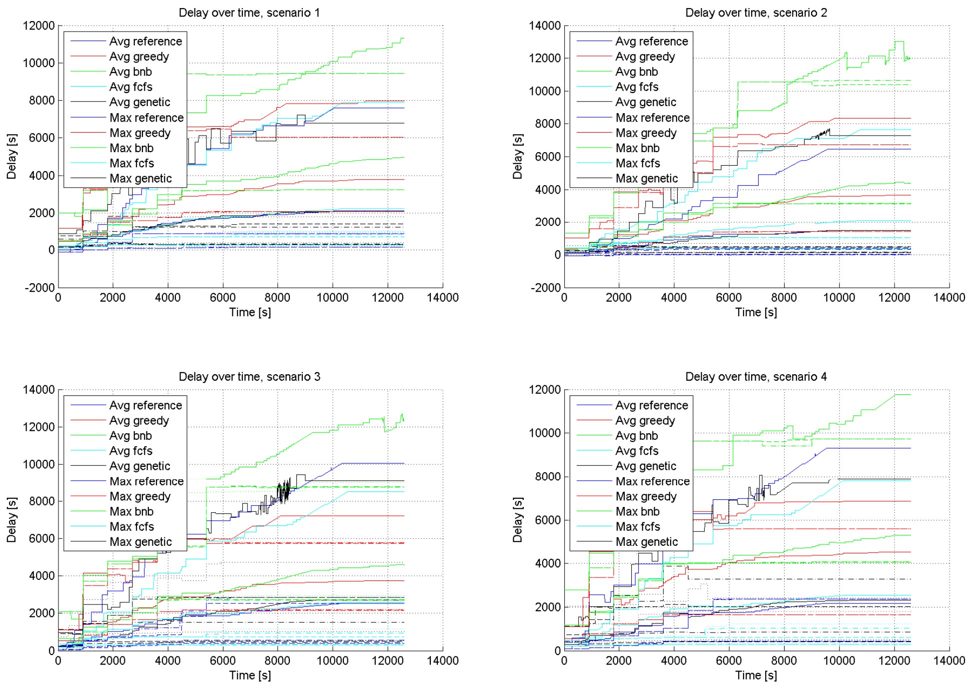

Figure C.58: Maximum and average individual delay when the aircraft based speed of twelve aircraft is disturbed

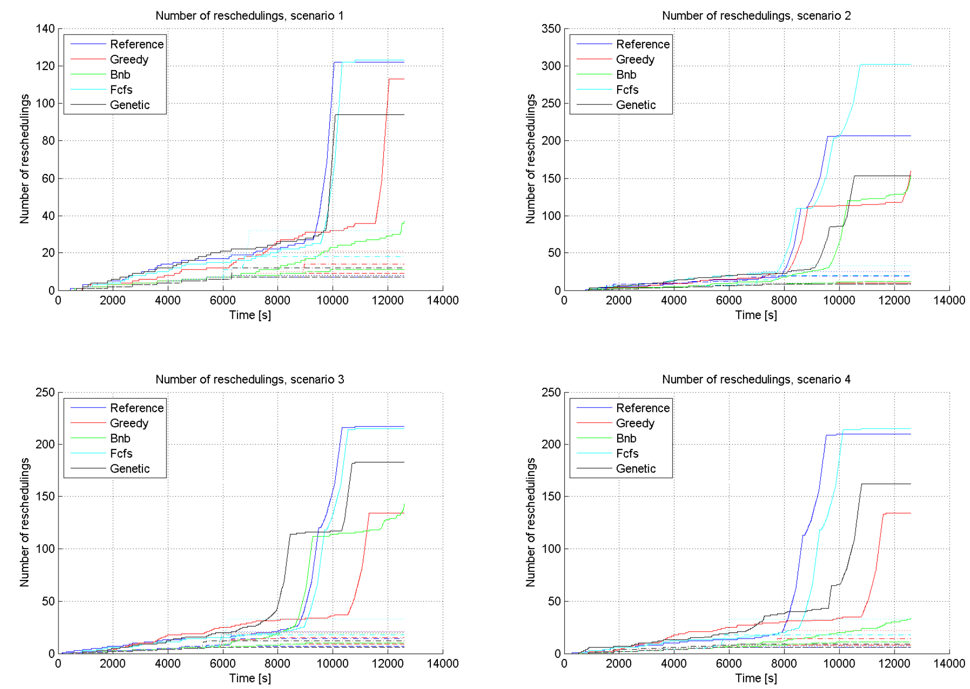

Figure C.59: Number of rescheduling operations when the aircraft based speed of twelve aircraft is disturbed

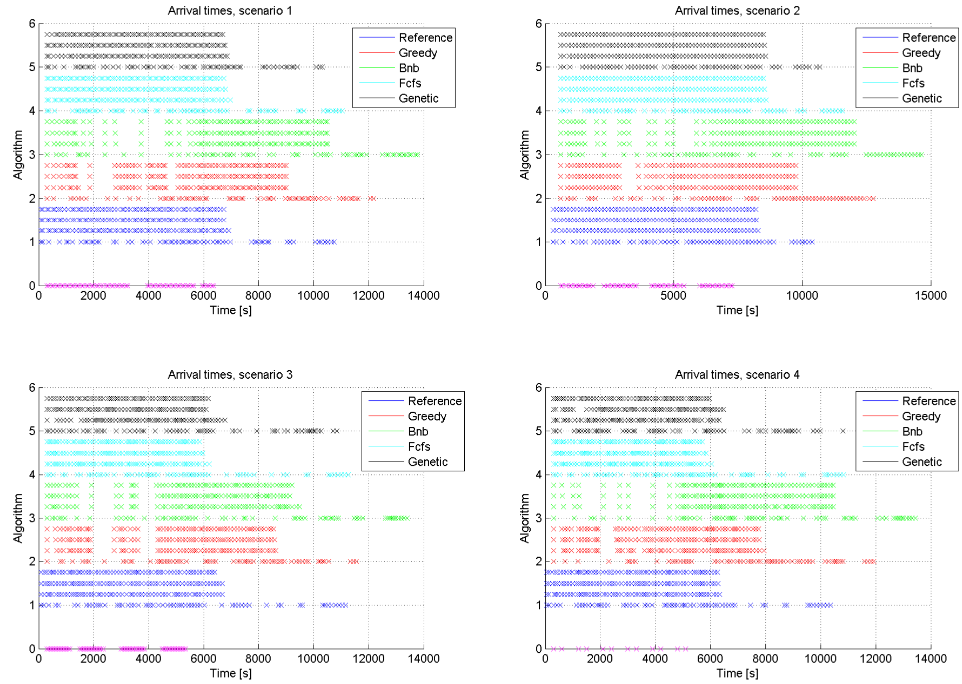

Figure C.60: Arrival times when the aircraft based speed of twelve aircraft is disturbed

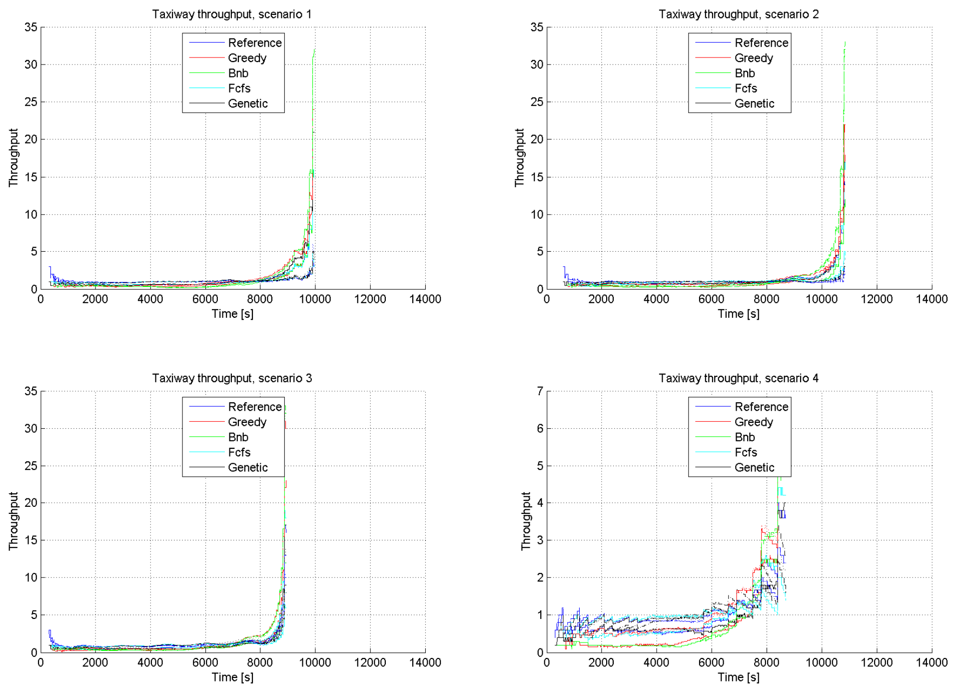

Figure C.61: Taxiway throughput when the aircraft based speed of twelve aircraft is disturbed

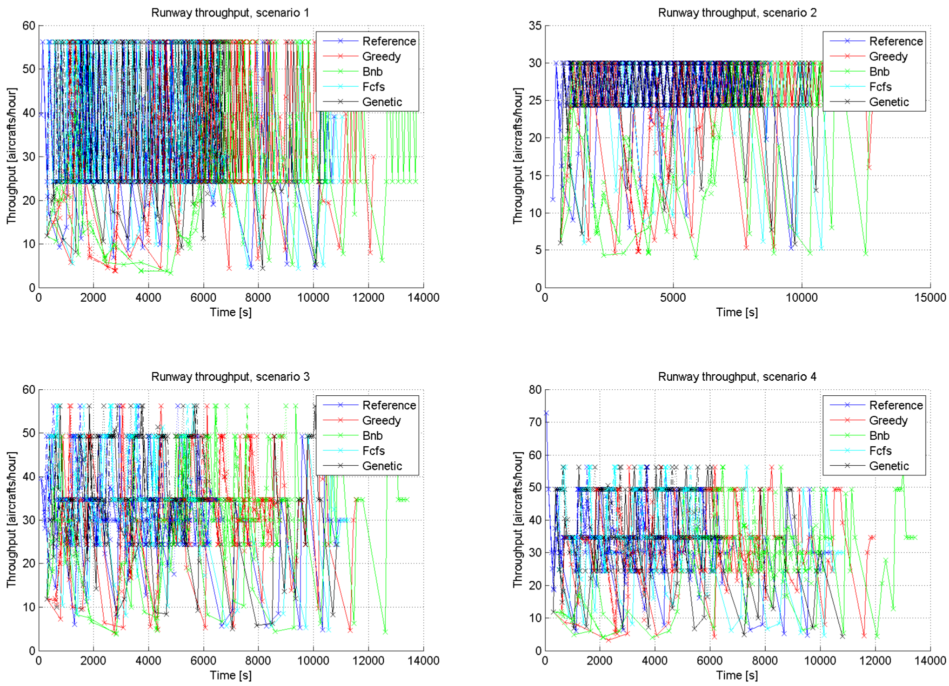

Figure C.62: Runway throughput when the aircraft based speed of twelve aircraft is disturbed

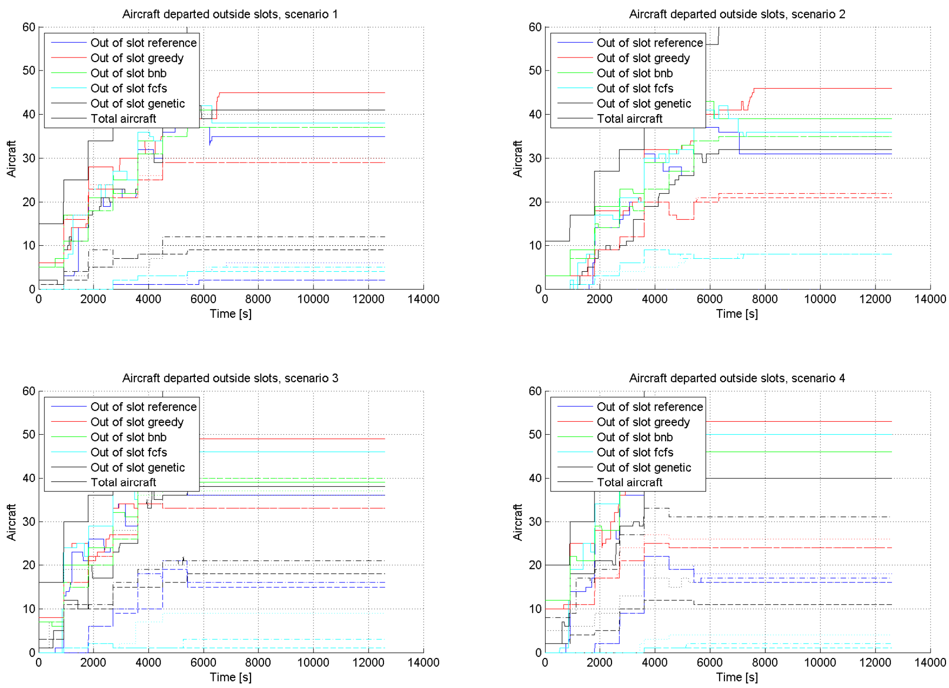

Figure C.63: Aircraft departed out of their slots when the aircraft based speed of twelve aircraft is disturbed

Figure C.64 up to Figure C.70 show the output parameters when the ready to taxi margin is varied as described in Table 5.3. A ready to taxi margin of zero minutes is shown as a solid line, a margin of ten minutes is as a dotted line, twenty minutes as a combination of dashes and dots and a thirty minutes margin as a dashed line. In Figure C.67, the ideal arrival times as desired by the aircraft are shown as algorithm 0. For each algorithm the lines from bottom to top represent ready to taxi margins of zero, ten, twenty and thirty minutes respectively.

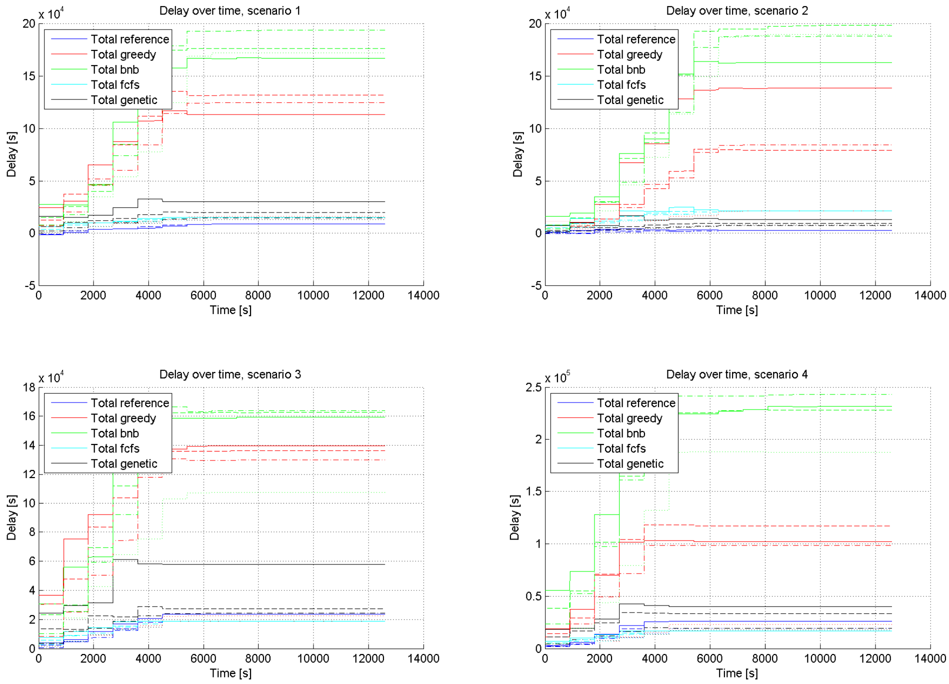

Figure C.64: Total delay over time when the ready to taxi margin is varied

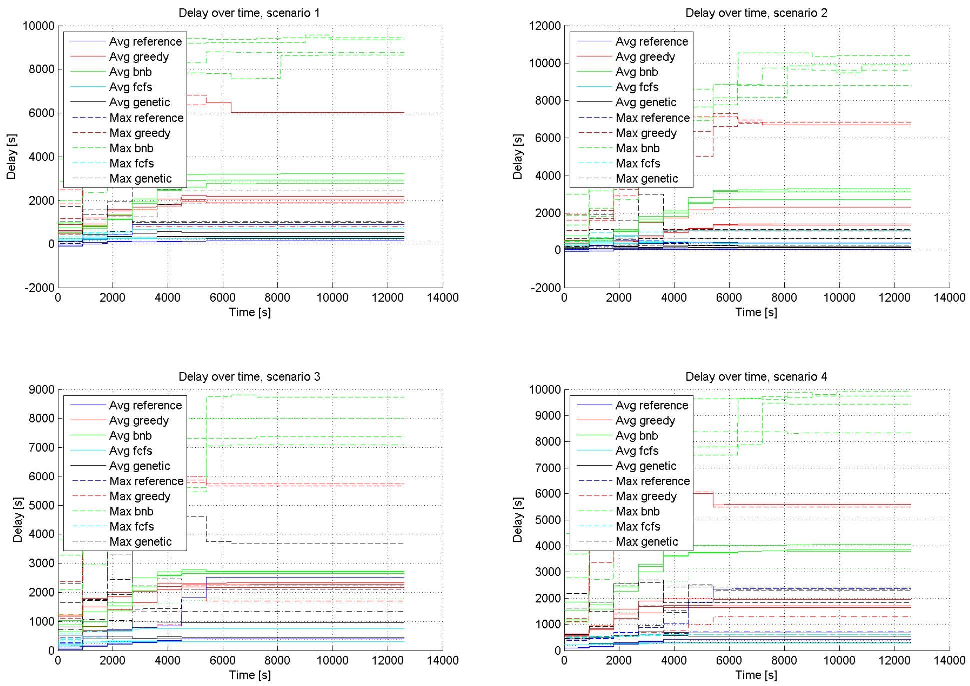

Figure C.65: Maximum and average individual delay when the ready to taxi margin is varied

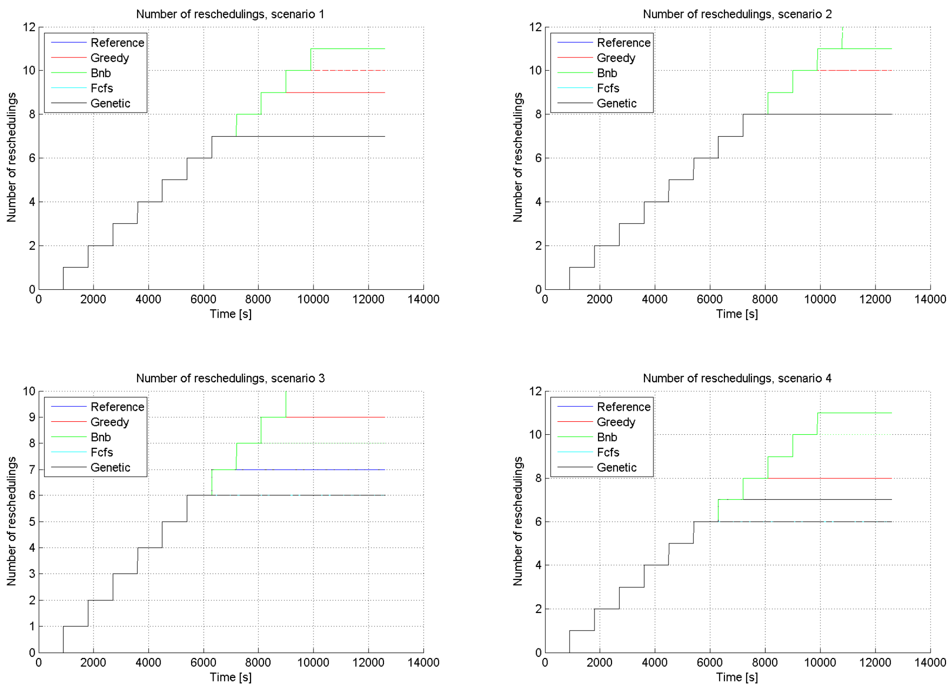

Figure C.66: Number of rescheduling operations when the ready to taxi margin is varied

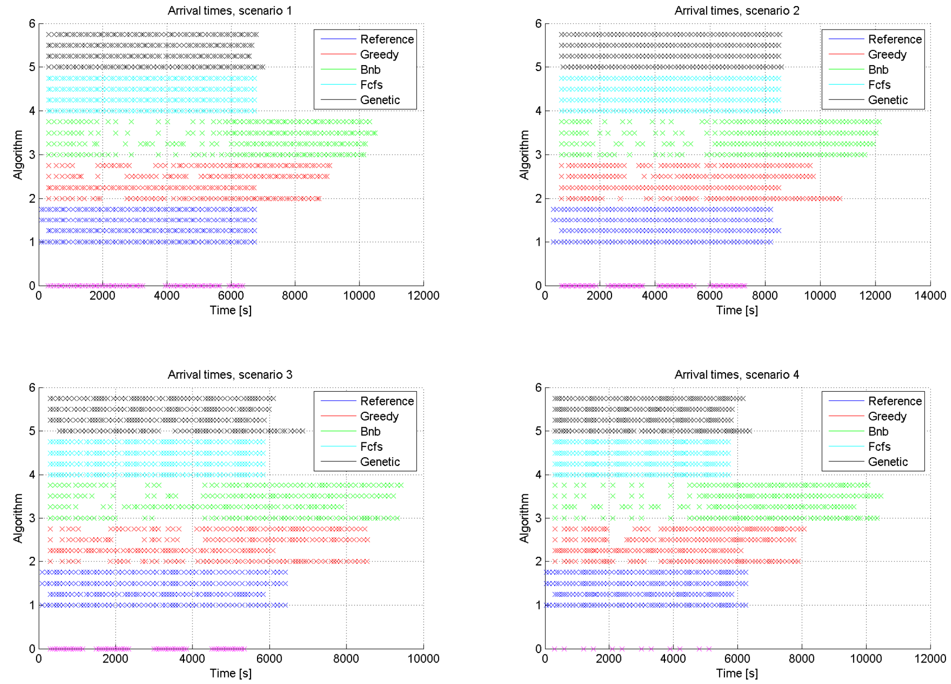

Figure C.67: Arrival times when the ready to taxi margin is varied

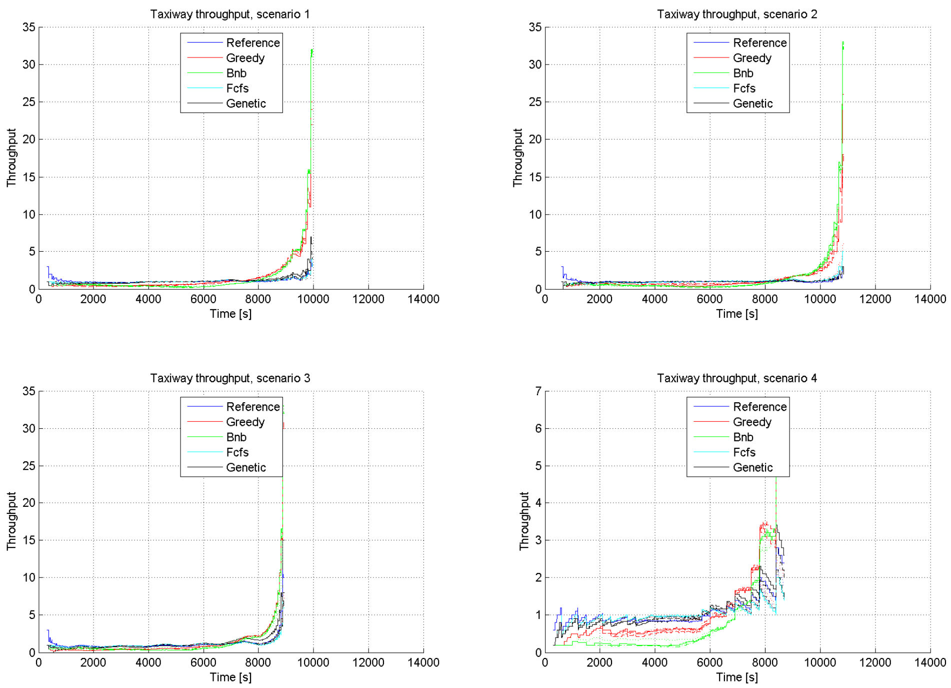

Figure C.68: Taxiway throughput when the ready to taxi margin is varied

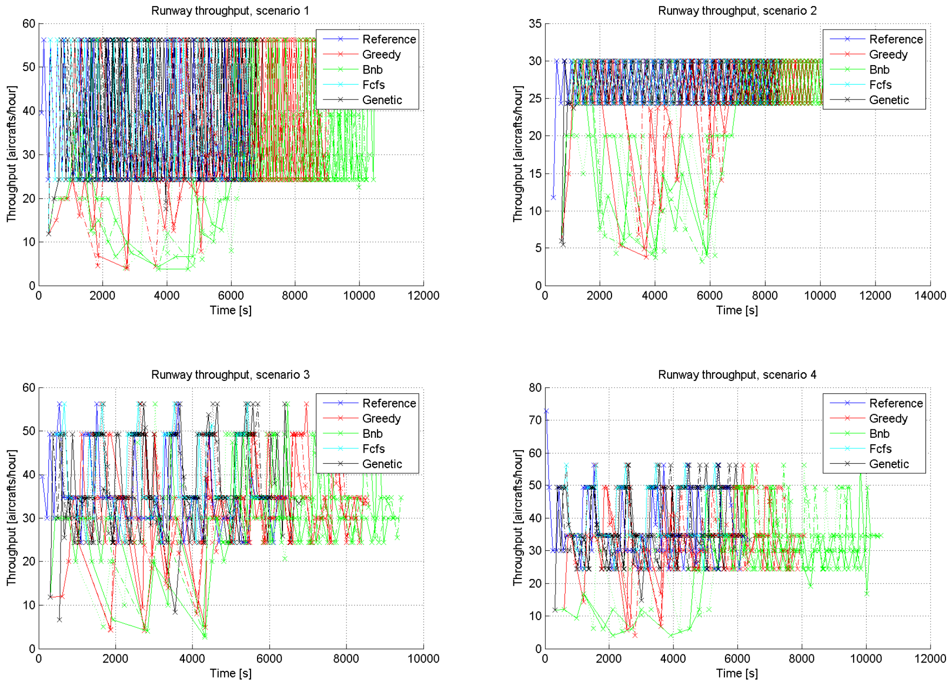

Figure C.69: Runway throughput when the ready to taxi margin is varied

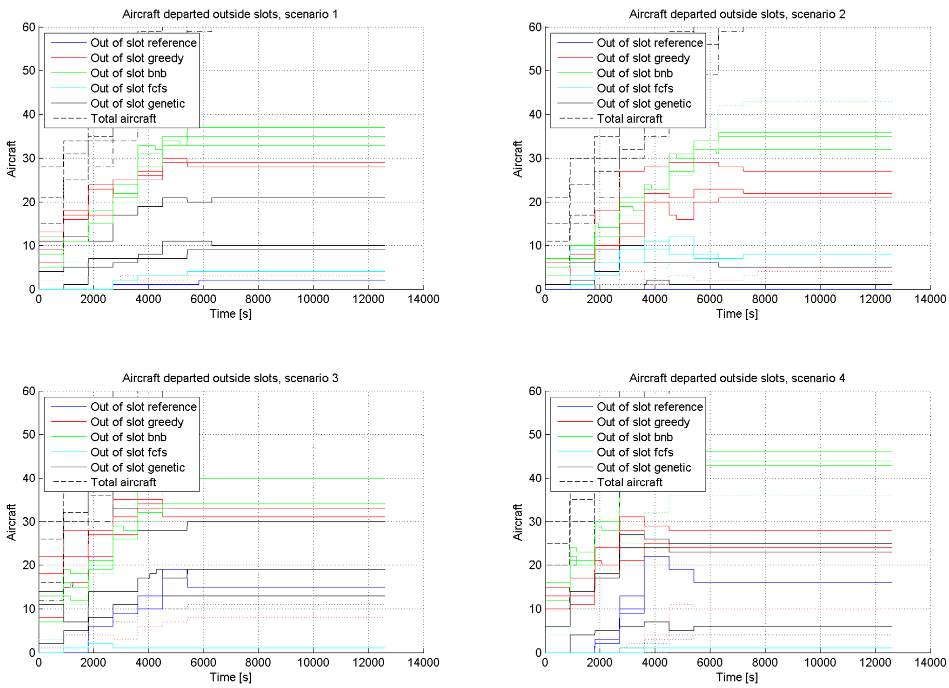

Figure C.70: Aircraft departed out of their slots when the ready to taxi margin is varied