

Evolutionary-Computation Based Risk Assessment of Aircraft Landing Sequencing Algorithms

Wenjing Zhao, Jiangjun Tang, Sameer Alam, Axel Bender, Hussein A. Abbass

► **To cite this version:**

Wenjing Zhao, Jiangjun Tang, Sameer Alam, Axel Bender, Hussein A. Abbass. Evolutionary-Computation Based Risk Assessment of Aircraft Landing Sequencing Algorithms. 7th IFIP TC 10 Working Conference on Distributed, Parallel and Biologically Inspired Systems (DIPES) / 3rd IFIP TC 10 International Conference on Biologically-Inspired Collaborative Computing (BICC) / Held as Part of World Computer Congress (WCC) , Sep 2010, Brisbane, Australia. pp.254-265, 10.1007/978-3-642-15234-4_25 . hal-01054494

HAL Id: hal-01054494

<https://hal.inria.fr/hal-01054494>

Submitted on 7 Aug 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Evolutionary-Computation Based Risk Assessment of Aircraft Landing Sequencing Algorithms

Wenjing Zhao¹, Jiangjun Tang¹, Sameer Alam¹, Axel Bender², Hussein A. Abbass¹

¹ UNSW@ADFA, Canberra, Australia.

w.zhao@student.adfa.edu.au, j.tang@student.adfa.edu.au, h.abbass@adfa.edu.au

²DSTO, Edinburgh, Australia.

axel.bender@dsto.defence.gov.au

Abstract. Usually, Evolutionary Computation (EC) is used for optimisation and machine learning tasks. Recently, a novel use of EC has been proposed – Multiobjective Evolutionary Based Risk Assessment (MEBRA). MEBRA characterises the problem space associated with good and inferior performance of computational algorithms. Problem instances are represented (“scenario Representation”) and evolved (“scenario Generation”) in order to evaluate algorithms (“scenario Evaluation”). The objective functions aim at maximising or minimising the success rate of an algorithm. In the “scenario Mining” step, MEBRA identifies the patterns common in problem instances when an algorithm performs best in order to understand when to use it, and in instances when it performs worst in order to understand when not to use it.

So far, MEBRA has only been applied to a limited number of problems. Here we demonstrate its viability to efficiently detect hot spots in an algorithm's problem space. In particular, we apply the basic MEBRA rationale in the area of Air Traffic Management (ATM). We examine two widely used algorithms for Aircraft Landing Sequencing: First Come First Served (FCFS) and Constrained Position Shifting (CPS). Through the use of three different problem (“scenario”) representations, we identify those patterns in ATM problems that signal instances when CPS performs better than FCFS, and those when it performs worse. We show that scenario representation affects the quality of MEBRA outputs. In particular, we find that the variable-length chromosome representation of aircraft scheduling sequence scenarios converges fast and finds all relevant risk patterns associated with the use of FCFS and CPS.

Keywords: Algorithms’ Behavior, Aircraft Sequencing, Evolutionary Computation.

1 Introduction

Existing demands on the air traffic system routinely exceed the capacity of airports. This leads to air-traffic imposed ground and airborne delays of aircraft. For the majority of U.S. and European airports such delays are estimated to be over 15 minutes per aircraft [4] costing airlines billions of dollars per year [10]. Thus airports

are proving to be serious bottlenecks in handling rising air traffic densities. Since constructing new airports or additional runways is not a near-term solution, researchers investigate various approaches as how to make the most efficient use of the available runways given safety constraints. Amongst these approaches is the effective scheduling of aircraft landings, which can significantly improve runway throughput capacity as well as safety and efficiency of airports.

It has been shown in the literature that the problem of finding optimal landing sequences – when the constraints of spacing between arrivals depend on the aircraft type as is the case in real-world applications – is NP-hard [6]. Thus it is unlikely that efficient optimisation algorithms exist [6]. Even if there was an accurate schedule optimiser, it would probably lack the speed to respond quickly to operational demands in the high-paced work environment of air traffic controllers (ATC). In the real world, therefore, fast and frugal heuristics are more useful than sophisticated but slow algorithms.

The most commonly used heuristics-based algorithm that generates efficient aircraft landing sequences is First Come First Served (FCFS). The basis of this method is the Estimated Time of Arrival (ETA) of aircraft at the runway and the minimum time separation between aircraft [7]. In FCFS, the aircraft land in order of their scheduled arrival times. ATC add suitable separation times to ensure appropriate spacing between aircraft. FCFS is straightforward and favoured by airlines for its fairness and by ATC for its simplicity that puts little demands on ATC workloads. However, its drawback is that it may lead to reduced runway throughput due to unnecessary spacing requirements [8].

Another common approach is Constrained Position Shifting (CPS) [2] in which an aircraft can be moved forward or backward in the FCFS schedule by a specified maximum number of positions. This approach provides ATC with additional flexibility and helps pilots to better predict landing times and positions [8]. However, it also increases the controller's workload in terms of increased ATC-Pilot communication and controller directives.

Both FCFS and CPS thus have their advantages and disadvantages, which express themselves in variations of algorithmic performance depending on problem situation and context of use. Considering the large amount of money lost because of runway congestions, it makes economical sense to investigate in which aircraft landing sequence scenarios (ALSS) CPS performs better (or worse) than FCFS. Such an investigation will enable airports to identify and understand the risks, both negative and positive, when choosing one scheduling heuristic over another.

In this paper, we make use of the recently introduced Multiobjective Evolutionary Based Risk Assessment (MEBRA) framework [1] to identify positive and negative risks associated with the application of a particular algorithm. Rather than optimising an algorithm, MEBRA explores and evaluates the risk profiles of algorithms. These risk profiles are signatures in the problem space and associated with the performance of a computational algorithm. In its risk assessment, MEBRA employs scenario representation, scenario generation, scenario evaluation and scenario mining. Here the term "scenario" refers to a problem instance in which the computational algorithm under investigation is applied.

So far though, MEBRA has only been applied to a limited number of problems. Here we demonstrate its viability by applying it to the Air Traffic Management (ATM)

problem domain. We study performance and identify risks associated with the use of FCFS and CPS in ALSS. Our paper further investigates how scenario representation impacts on algorithm evaluation. We examine three different representations: Fixed Length Chromosome, Variable Length Chromosome, and a Probabilistic Model.

At the start of our application of MEBRA to ATM, random ALSS are generated and encoded in the chromosome representation. Then complex landing sequence scenarios are evolved over many generations by applying genetic operators and using a fitness function that correlates with risk. This imposes selection pressure on the population of scenarios. ALSS that are deemed “fitter individuals” have increased likelihood to survive into the next generation. In the final “scenario Mining” step of MEBRA, the scenario population at the end of evolution is used to identify common characteristics, or “signatures”, of aircraft landing sequences that contribute to schedule delays. This aids in understanding those factors that result in technical risks in the generation of landing sequences when using scheduling heuristics such as FCFS or CPS.

The rest of this paper is organised as follows. In Section 2, we describe the aircraft landing sequencing problem along with details of the FCFS and CPS algorithms. Next, we present the MEBRA framework (Section 3) and how it applies to the risk assessment of aircraft landing sequencing algorithms (Section 4). We illustrate the approach in a simple example and describe our results in Section 5. Conclusions are drawn in the final section.

2 Aircraft Landing Sequencing

The U.S Federal Aviation Administration (FAA) has established minimum spacing requirements between landing aircraft to prevent the turbulence from wake vortices [5]. If an aircraft interacts with the wake vortex of the aircraft landing in front of it, it could lose control. To prevent this risk, a minimum time separation between aircraft is mandated. This separation depends on both the size of the leading aircraft and that of the trailing aircraft. The FAA divides aircraft into three weight classes, based on the maximum take-off weight capability. These classes are:

1. *Heavy Aircraft* are capable of having a maximum takeoff weight of 255,000 lbs or more.
2. *Large Aircraft* can have more than 41,000 lbs and up to 255,000 lbs maximum takeoff weight.
3. *Small Aircraft* are incapable of carrying more than 41,000 lbs takeoff weight.

A matrix of the minimum time separations mandated by the FAA is shown in Table 1.

2.1 First Come First Served

FCFS is a prominent scheduling algorithm in Sequencing Theory [9]. It is the most straightforward method to sequence aircraft arrivals in an airport. Much of present technology has some relationship with it or is even based on it [6].

Table 1. Minimum time separation (in seconds) between landings as mandated by the FAA.

Leading Aircraft	Trailing Aircraft		
	Heavy	Large	Small
Heavy	96	157	196
Large	60	69	131
Small	60	69	82

FCFS determines the aircraft landing sequence according to the order of its estimated time of arrival (ETA) at the runway. ETA is computed by the control center at the time an incoming aircraft crosses the transition airspace boundary. If the difference between the ETA of two successive aircraft violates the minimum separation time constraints, then the Scheduled Time of Arrival (STA) of the trailing aircraft is adjusted accordingly. The following numerical example illustrates this adjustment.

Given seven aircraft, A, B, C, D, E, F, G, each belonging to one of the three weight classes H (heavy), L (large) or S (small). FCFS orders these aircraft according to their ETA, see third row of Table 2. It then adds time to an ETA, when the separation time between two aircraft is smaller than the allowable minima shown in Table 1. For instance, the ETA of the small aircraft C is only 60 sec later than the ETA of the preceding large aircraft B. Thus 71 sec are added to the ETA of C to achieve a separation of 131 sec as required by the FAA (Table 1). In the example, the STA of all aircraft following C are now determined by adding the minimum separation time to the STA of the leading aircraft because all STA calculated this way happen to be later than the ETAs. The makespan (i.e. the difference between final STA and first STA) in this example is 18m59s - 07m51s = 668 sec.

FCFS scheduling establishes the landing sequence based on predicted landing times. It therefore is easy to implement and does not put significant pressure onto ATC workloads. However, it ignores information which can be used to increase runway throughput capacity.

Table 2. FCFS scheduling example

Aircraft	A	B	C	D	E	F	G
Category	L	L	S	H	L	S	H
ETA	07m51s	10m00s	11m00s	12m00s	13m00s	14m00s	15m00s
AC Order	A:1	B:2	C:3	D:4	E:5	F:6	G:7
STA	07m51s	10m00s	12m11s	13m11s	15m48s	17m59s	18m59s

2.2 Constrained Position Shifting

CPS, first proposed by Dear [3], stipulates that the ETA-based schedule can be changed slightly and that an aircraft may be moved up by a small number of positions.

Neumann and Erzberger [8] investigated an enumerative technique for computing the sequence which minimises the makespan, subject to a single position shift (1-CPS) constraint. In the example of Table 2, for instance, the swap of aircraft D and E would result in a reduction of makespan by 23 sec: the STAs of E, D, F and G would be 13m20s, 14m20s, 17m36s and 18m36s, respectively. This is the basic motivation for CPS methods.

Finding the optimal ordering of a set of aircraft through CPS can be seen as a search for the lowest-cost path through a tree of possible aircraft orderings, where the cost is the sum of the time separations required between each pair of aircraft. For the CPS problem, an initial sequence of aircraft is given, along with the list of minimum separation constraints (e.g. Table 1) and the maximum possible time-shifts for each aircraft. In the final sequence shown in Table 3, each aircraft is constrained to lie within one shift from its initial position, and no aircraft must have a time of arrival earlier than permitted by the maximum allowable time shift.

Table 3. CPS scheduling example

Aircraft	A	B	C	D	E	F	G
Category	L	L	S	H	L	S	H
ETA	07m51s	10m00s	11m00s	12m00s	13m00s	14m00s	15m00s
AC Order	A:1	B:2	C:3	E:4	D:5	F:6	G:7
STA	07m51s	10m00s	12m11s	13m20s	14m20s	17m63s	18m36s

3 MEBRA – Multiobjective Evolutionary Based Risk Assessment

The objective of this paper is to demonstrate how evolutionary computation (EC) methods can be used to assess the performance of aircraft landing sequencing algorithms. The approach we take is a simplified version of the Multiobjective Evolutionary Based Risk Assessment (MEBRA) framework that is designed for the purpose of exploring and evaluating computational algorithms under risk [1]. In aircraft landing sequencing problems, risks associated with computational algorithms include production of suboptimal scheduling sequences, i.e. unnecessarily large makespans; computational complexity that results in algorithms taking too long and becoming unresponsive to operational demands; and unnecessary increases of ATC workloads. The occurrence of these risks depends on the specifics of the problem at hand; for instance in an ALSS that requires a large number of aircraft to be scheduled in a very short period of time ATC are more likely to get overloaded than in an ALSS when only a few aircraft need to be sequenced. MEBRA of algorithmic performance is thus concerned with searches on the problem space, also known as the “scenario space”, rather than the solution space.

MEBRA comprises four building blocks: Scenario Representation, Scenario Generation, Scenario Evaluation, and Scenario Mining. MEBRA’s *Scenario Representation* can be as simple as sampling a parameter space that captures quantitative aspects of a problem, or as complex as narratives that try to capture futuristic strategic uncertainties. During *Scenario Generation* MEBRA makes use of

evolutionary computation. Problem instances are evolved over many generations while being exposed to selection pressure. This pressure makes less risky scenarios less likely to survive into the next generation and therefore is part of *Scenario Evaluation*. In this paper, we make use of the single objective version of MEBRA, called SEBRA. In the *Scenario Mining* step, MEBRA identifies risk patterns or “hot spots”, i.e. conditions in scenario space under which risk eventuates. Scenario mining techniques can be as simple as descriptive statistics of the evolved scenario population or as complex as a framework that analyses dynamics and network dependencies to unveil the “rules of the game”.

4 Application of MEBRA to Aircraft Landing Sequencing Algorithms

4.1 ALSS Representation

In order to capture complex patterns of aircraft landing sequences, we use three different chromosome representations: fixed-length sequence representing a problem instance, a variable-length sequence representing a pattern that is repeated in a problem instance, and a stochastic finite state machine representation representing the probability transition matrix to generate patterns. A detailed description of the three representation is as follows:

1. *Fixed-length chromosome*. In the fixed-length chromosome, each gene represents an aircraft type. The position in the chromosome corresponds to the aircraft’s position in the arrival schedule according to ETA. The length of the chromosome is equal to the total number of aircraft whose landing need to be scheduled. In our experiments, the fixed-length chromosome contains 200 genes. At chromosome initialisation, ETA values are spaced with 1 sec and assigned to the aircraft sequence. We use this initialisation condition because having all aircraft arrive “at once” puts the biggest demand on the landing sequencing algorithms and thus will facilitate the search for “hot spots” in ALSS.
2. *Variable-length chromosome*. The variable-length chromosome encodes a pattern. A pattern is a partial sequence of aircraft arrivals. As with the fixed-length chromosome, each gene encodes an aircraft type. With respect to the whole aircraft arrival sequence, the partial sequence has a starting point described by a position in the arrival schedule and a length that is smaller than the total number of aircraft to be scheduled. In our experiments, the starting point is always the first position in the scheduling sequence and the pattern’s length varies between 3 and 50. At the time a pattern is evaluated, it is repeated as many times as needed to generate a 200-gene sequence. For example, a pattern of length 50 would need to be repeated four times. This normalises the scale when comparing-variable length and fixed-length chromosome representations. The evolution based on the variable length

representation is pushed to find those patterns that optimise the fitness function (see Subsection 4.3). A selection pressure is placed automatically to favour shorter patterns since their frequency in the 200-gene sequence increases.

3. *Stochastic Finite State Machine (SFSM) chromosome.* The SFSM chromosome contains nine genes which encode how likely it is that an aircraft type is followed by another in the schedule. The genes thus represent probabilities of the nine possible SFSM transitions. The initial generation initializes the chromosomes randomly from uniform distributions. Obviously, when the SFSM is used to generate a sequence, transition probabilities out of each node are normalised. Moreover, it is natural that this stochastic representation would require multiple evaluations (30 in our case) of each chromosome to approximate its fitness.

4.2 ALSS Generation

In the generation of ALSS we make use of evolutionary computation (EC) techniques. In EC, a seed population of scenarios is evolved over many generations (implicit parallelism) to explore the space of possible ALSS. From generation to generation, individuals are subjected to single-point crossover and uniform mutation. Evolution (“search”) proceeds to meet a given selection pressure (such as in Equation 2 below) and according to some given rules; e.g. in our experiments (Section 4.5) we apply tournament selection with elitism. Note that evolving ATM problems according to the selection pressure in Equation 2 does not ensure that we always find scenarios for which both FCFS and CPS generate optimal landing schedules. However, for most of the evolved complex scenarios in the final population this actually is the case. It is thus fair to assume that low-risk scenarios evolved with Equation 2 will have features that differ from those of the high-risk problems generated under the selection pressure of Equation 1 (below).

4.3 ALSS Evaluation

To assess both positive and negative risk of inefficiency-based delays in aircraft landing sequencing algorithms we define two fitness functions. The first one is designed to identify those situations where FCFS is inferior to CPS. Therefore, the objective of the first fitness function is to maximise the difference between the FCFS makespan and the CPS makespan. As mentioned earlier, we study worst-case situations, i.e. when all aircraft in a sequence arrive within one second of each other and are ready to be landed. The “negative-risk” objective function can be described formally as follows:

$$\text{Max } \{F = T_{FCFS} - T_{CPS}\} . \quad (1)$$

where T_X denotes the makespan of algorithm X .

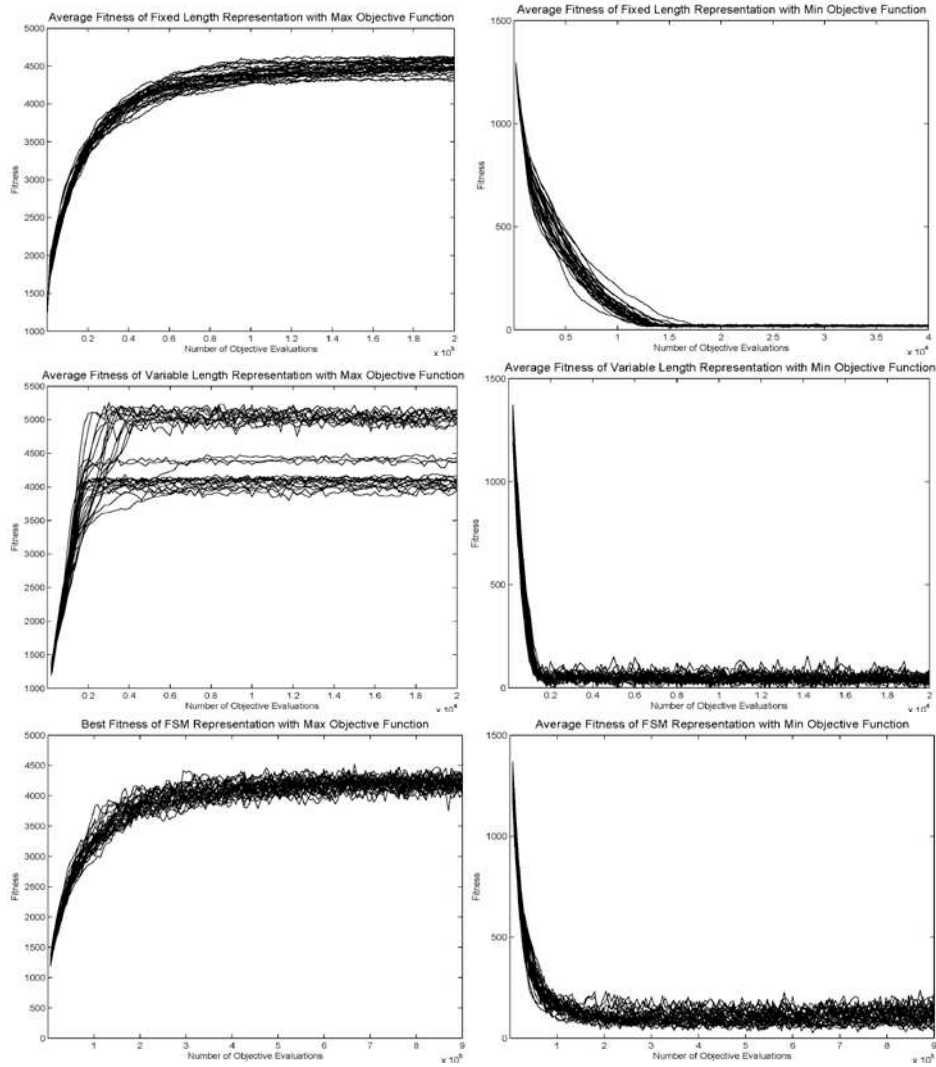


Fig. 1. The progress in fitness values as a function of the number of objective function evaluations. Figures on the left are for the negative-risk objective function while those on right are for the positive-risk objective function. The top figure depicts the evolution of a population of scenarios encoded with fixed-length chromosome representation, the middle one for the variable-length chromosome representation and the bottom one for the SFSM chromosome representation of ALSS.

As described earlier, in any ALSS the CPS method guarantees an equal or better makespan than the FCFS sequencing approach. By evolving solutions that optimise the function in Equation 1, MEBRA will evolve problem instances for which CPS

considerably outperforms FCFS. While we cannot be sure that CPS is a very good algorithm to use in such evolved complex scenarios, we definitely know that FCFS performs very poorly. The evolutionary process thus finds scenario sets for which CPS results in maximum improvements to the FCFS schedule; i.e. we identify scenarios in which FCFS is particularly inefficient.

The second fitness function, the “positive-risk” objective, is to minimise the difference between the two makespans, i.e. we identify low-risk scenarios for which CPS will not result in significantly reduced makespans. Formally,

$$\text{Min } \{F = T_{FCFS} - T_{CPS}\} . \quad (2)$$

4.4 ALSS Mining

To compare among the three representations, we use three two-way 2x2 comparison matrices. Each matrix captures the best-best, worst-best, worst-worst, and best-worst overlaps between the solutions found using each representation. Each cell in the matrix is the comparison result between:

1. Fixed length v.s. Variable Length: the count of matched patterns by sliding the pattern of the variable length and counting its frequency in the fixed length. We start from the first aircraft in fixed length chromosome and slide the variable length chromosome by one aircraft position at each step. We count the number of matches between the partial sequence in fixed length is as same as the whole sequence of variable length chromosome.
2. Fixed length v.s. SFSM: the distance of probabilities of transitions by transforming the fixed length to a SFSM using the frequency of transitions found in the fixed length chromosome. We calculate the frequency of aircraft transitions in the fixed length chromosome and translate these frequencies into the stochastic finite state machine representation. We obtain nine transition probabilities from the fixed length chromosome with the same format as the SFSM chromosome. The Euclidean distance between the two normalized probability vectors is used to calculate similarities.
3. Variable Length v.s. SFSM: the distance of probabilities of transitions by transforming the variable length into the fixed length (by repeating the patterns) then transforming the fixed length to a SFSM using the frequency of transitions found in the 200-gene sequence. The calculations are then done in the same way illustrated in the previous step.

4.5 Experimental Setup

We ran each of the 6 SEBRA evolutions 30 times with different seeds and a population size of 200. We apply tournament selection with elitism, single-point crossover with probability 0.9 and uniform mutation with probability equal to the reciprocal of the chromosome length. For the variable length chromosome, the mutation is set to 0.02. Those parameters are chosen carefully after a number of

sample runs. We allowed sufficient number of objective evaluations in each run for evolution to become stable (the best solution does not change significantly).

5 Results

The progress in the two fitness functions, “negative-risk” and “positive-risk” objectives, corresponding to each of the six experimental setups and the associated 30 seeds is plotted in Figure 1.

The following observations can be made:

1. Three types of local optima in the negative-risk objectives can be distinguished when we use a variable-length chromosome representation of ALSS – one with a fitness value of around 5000, a second with a fitness of approximately 4400, and a third one with fitness of about 4000.
2. Both fixed-length and SFSM chromosome representations appear to have become stuck between two of the three local optima found by the variable-length chromosome.
3. In the variable-length chromosome representation the number of objective evaluations to convergence is an order of magnitude smaller than the evaluations needed in the other two scenario representations.

This suggests that it is more efficient to evolve pattern (as in the experiments with variable-length ALSS chromosomes) than to evolve whole scenarios.

Table 4. Count of Building Blocks Matches in Fixed-Length vs Variable-Length ALSS

		Fixed	
		best	worst
Variable	best	20196	0
	worst	0	7200

Table 5. Distance of Probabilities for Building Blocks when comparing SFSM vs Fixed-Length and vs Variable-Length ALSS

		Fixed		Variable	
		best	worst	best	worst
SFSM	best	0.718186	1.452852	0.635106	1.827628
	worst	1.898998	0.839643	1.914774	1.079916

We now address the question whether the patterns found by evolving the variable-length representation are also present in the evolved fixed-length and SFSM ALSS. Table 4 shows that the patterns which maximize the difference between FCFS and CPS (“worst”-case scenarios, as of Eq.1) and which are found by evolving variable-

length ALSS can be found with high frequency in the evolved fixed-length ALSS. These patterns are not at all present in fixed-length scenarios that minimize the difference between the two makespans (“best”-case scenarios, as of Eq.2). This indicates that the variable-length patterns are some sort of building blocks in this problem and that it is more efficient to evolve building blocks directly than to evolve the solution vector as a whole.

Similar trends are found in Table 5. The normalized transition probabilities found by the fixed and variable length representations are closer to those found by the SFSM representation in corresponding experiments.

Figure-2 depicts two patterns found by evolving SFSM ALSS. Examples of high frequency patterns found by evolving the variable-length representation when looking for worst-case scenarios include: HSHSHSH, HSHSHSS, HLSHS, and SHSHSH. These patterns are not as simple as they may appear. The HSHSHSH pattern, when used as a building block will generate an HH link. Examples of building blocks found in best-case scenarios include: SLLH, LHSL, HHHH, LHHHH, and HHHSS SLLH, LHSL, HHHH, LHHHH, and HHHSS. It is easy to see why each of these patterns would give an advantage to CPS over FCFS.

In summary, we demonstrated that evolutionary computation can be a powerful framework to evaluate the performance of different algorithms. A deeper analysis of the resulting solutions can shed light on the problem patterns that determine strengths and weaknesses of an algorithm compared with another (baseline) algorithm. In the problem domain investigated in this paper, discovering these patterns allows to balance safety risks that can result from an unnecessary increase of ATC workloads and the (economic and ecological) costs that result from unnecessary delays or holdings of aircraft.

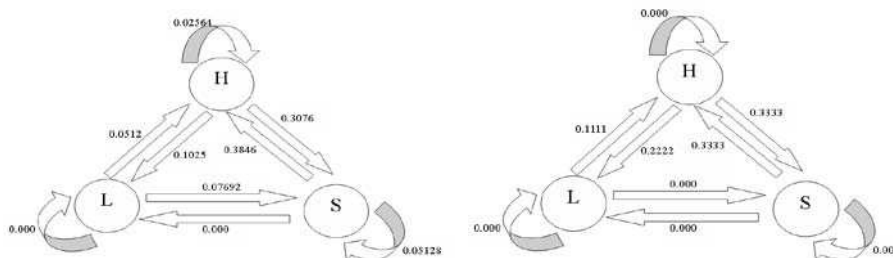


Fig. 2. Two examples of SFSM found in the case of Max objective function.

6 Conclusions

For many years, Evolutionary Computation (EC) has been successfully applied to optimisation problems, although almost exclusively to evolve solutions for such problems. In this paper, we showed that EC techniques can be used in a novel way, namely to assist in the assessment of algorithmic performance. We employed the Multiobjective Evolutionary Based Risk Assessment (MEBRA) concept to evolve

problem instances in which heuristic algorithms perform particularly poorly or particularly well.

MEBRA can be used as a comparative analysis technique. Through the application of clustering methods, pattern analysis and the like to the population of evolved problem instances, or scenarios, it can detect signatures, or “hot spots”, in the scenario space for which an algorithm performs better or worse than a reference algorithm. Thus, MEBRA provides valuable information about when it is best to use one algorithm over another.

We applied a single-objective version of the MEBRA framework – SEBRA – to the comparison of two prevalent heuristics used in the landing sequencing of aircraft arrivals in an airport: the First Come First Served (FCFS) and Constrained Position Shifting (CPS) algorithms. We found indeed that SEBRA could identify hot spots in the problem space for which FCFS performed markedly worse than CPS. We also found patterns in the sequences of estimated time of arrival (ETA), for which FCFS performs equally well as the computationally more complex CPS. The patterns were interesting and could easily be interpreted by making use of the minimum separation time matrix.

Our results indicate that convergence and variance of SEBRA depend on the chromosome representations for the SEBRA problem instances. The fixed-length chromosome and stochastic representations were stable and converged reasonably fast. The variable-length chromosome representation converged the fastest and found all patterns of interest.

References

1. Abbass, H.A., Alam, S., Bender, A.: MEBRA: Multiobjective Evolutionary Based Risk Assessment. *Computational Intelligence Magazine*, August: 29--36 (2009)
2. Balakrishnan, H., Chandran, B.: Scheduling aircraft landing under constrained position shifting. *AIAA Guidance, Navigation, and Control Conference and Exhibit* (2006)
3. Dear, R.: The dynamic scheduling of aircraft in the near terminal area. Technical report, M.I.T (1976)
4. Donohue, G., Laska, W.: United States and European Airport Capacity Assessment using the GMU Macroscopic Capacity Model (MCM). In *Proc. of the 3rd USA/Europe ATM R&D Seminar, Napoli, Italy* (2000)
5. FAA: *Aeronautical Information Manual/Federal Aviation Regulation*. McGraw-Hill (2003)
6. Hu, X., Chen, W.: Genetic algorithm based on receding horizon control for arrival sequencing and scheduling. *Engineering Applications of Artificial Intelligence*, 18:633--642 (2005)
7. Neuman, F., Erzberger, H.: Analysis of sequencing and scheduling methods for arrival traffic (1990)
8. Neuman, F., Erzberger, H.: Analysis of delay reducing and fuel saving sequencing and spacing algorithms for arrival spacing. Technical report, NASA Technical Memorandum 103880 (1991)
9. Pinedo, M.: *Scheduling:theory,algorithms,and systems*. Prentice-Hall, New Jersey (2002)
10. Wickens, C.D.: *The Future of Air Traffic Control*, chapter Human Operators and Automation. National Academy Press, Washington, D.C. (1998)