Searching air sectors for risk

ASHiCS (Automating the Search for Hazards in Complex Systems)

Kester Clegg
Dept. of Computer Science
University of York
York. U.K.
kester@cs.york.ac.uk

Rob Alexander
Dept. of Computer Science
University of York
York. U.K.
rda@cs.york.ac.uk

***Abstract*—ASHiCS permits the automatic discovery of high risk air traffic scenarios. In this paper we describe the evolutionary search used in ASHiCS and present an analysis of the project's Stage 2 solution landscape. We suggest that random or exhaustive search is infeasible given the size of the solution space presented by simple air traffic scenarios, and that standard linear regression modeling is unlikely to find traffic input patterns that indicate the presence of high risk. While ASHiCS successfully targets high risk scenarios, we remain faced with very large search spaces in future Systems of Systems (SoS) models, and hope that our investigations into linear regression modeling will lead to a generic technique of practical value in the dimension reduction of these large search spaces.**

Keywords-component; search, safety, risk, heuristics, rare events, air traffic control simulation.

I. INTRODUCTION

ASHiCS has demonstrated the use of search heuristics on simulation outputs to find scenarios of high risk for a given air sector. Our previous reports[1] have described how weighted heuristics are able to focus on specific incident types, flight paths or aircraft so that the search can effectively target those areas of interest within the solution space. The ASHiCS search harness generates traffic inputs for the RAMS Plus air traffic control fast time simulator and analyses the output of each simulation. Air traffic is generated by creating three text files that specify the characteristics of each aircraft entering the air sector being simulated, namely:

- the aircraft type;
- the aircraft entry time, its entry and exit flight level;
- the navigational aids specifying its flight path and the point at which any level changes occur.

These traffic input files are created with certain restrictions, such as predetermined flight paths and wake turbulence separation. In the project's Stage 2 scenarios, one aircraft is selected at random on a particular flight path to undergo an explosive cabin pressure loss event, requiring the aircraft to carry out an emergency descent to FL100.

Once the input files have been created, a non-graphic version of RAMS Plus is executed and its outputs are analyzed by heuristics in the ASHiCS software. The analysis uses several factors to assign a score that comprises a measure of risk. This estimate of risk outcome for that configuration of input traffic is termed the scenario's "fitness score". The ASHiCS search harness ranks the scenarios being tested in terms of their fitness, selects the "fittest" (i.e. the ones containing the most risk) and mutates their inputs to see if their risk measure increases. The search continues until no higher fitness scores are achieved.

Section I.A describes background and previous work related to the project, Section II describes the search context, parameters and experiment design. Section III contains our analysis of the solution space and conclusions.

A. Background and previous work

ASHiCS came from the realization that as complex systems increased in size and complexity, manual hazard analysis of the systems has become much harder and we may soon reach a point when such analysis must be carried out by machines as part of an automated process. ASHiCS was intended to demonstrate a proof-of-concept approach using automated search within a complex ATM simulation to discover hazards that would have been hard to find using manual analysis.

The use of search techniques with simulated environments is well established, however using search to investigate ATM simulations is still relatively new. Previous work has been done in conjunction with EUROCONTROL on using a co-evolutionary search to discover the causes of delays in dynamic continuous descent arrivals (CDA) scenarios [1]. This technique evolved a pair of "problem and solver" algorithms, pitting one against another to evaluate traffic distributions and ground events that would enable the researchers to identify delay bottlenecks in the system. The same team from the University of New South Wales more recently used genetic algorithms (GA) to find the flight events resulting from an Air Traffic Controller's (ATCo) actions that could lead to higher risk of air-borne collision. Their approach is similar to ASHiCS in that their approach evolves ATC actions that:

[1] Available at http://www.complexworld.eu/projects/

increase collision risk in a given air traffic scenario so that [those] closer to a target level of safety are deemed "fitter individuals", having increased likelihood of survival to the next generation in the evolutionary process of the GA. [2]

This approach of evolving ATM scenarios to contain increasing levels of risk is very similar to ASHiCS, albeit with different scenarios, means of measuring risk and simulation environment. However, the basic approach of using a multi-objective genetic algorithm to search for risk is the same. The principal difference between the two approaches is that ASHiCS uses the RAMS Plus fast time simulator that allows us to use task workload measures and conflict resolutions by the ATCo as part of our risk assessment. The study by Alam et al. does not appear to take a measure of ATCo workload from conflict resolution into account.

II. Search Context

The search context is limited to a single en-route air sector containing a number of predetermined flight paths specified using navigational aids. Scenarios use a sample size of twenty aircraft whose start times are randomly generated over the span of one hour. The aircraft do not arrive or depart from airports: they "appear" outside the sector at cruise speed and do not deviate from their flight path except to resolve a conflict. However, there are several restrictions on the allocation of start times to aircraft (see Section II.A).

The two principal flight paths in the sector run broadly north south and west east (hereafter referred to as ns and ew). Each carries a single type of aircraft flying at their typical en-route altitudes. The flight paths intersect obliquely in the center of the sector; however as their flight levels are FL330 and FL190 respectively, no conflicts arise. The high level ns flight path carries an aircraft chosen at random to develop the cabin pressure loss event. This aircraft is referred to as CPLoss. Depending on the navigational aid used to trigger the cabin pressure loss event, CPLoss may descend through traffic on the ew flight path or pass close to traffic on one of the other low level flight paths. CPLoss maintains its planned trajectory on the ns flight path after reaching FL100.

Beyond the two main flight paths that carry the majority of the traffic in the simulation, there are three more flight paths (at FL230) intended to represent the presence of incidental traffic in the sector. These three flight paths also intersect in the centre of the sector but as they carry just a quarter of the traffic it is not intended that these would form the majority of the ATCo's workload. For ease of implementation, we assume the sector is controlled by a single air traffic controller (ATCo) from FL100 to FL600 with a standard minimum separation of 5nm. The aircraft types, flight paths and flight levels are shown in Table 1.

Aircraft type	Flight path	Flight level
A320	ns	330
DH8	ew, r4	230,190
B737	r2,r3,r4	230
C551	r2,r3,r4	230

Table 1: Types of aircraft, flight paths and levels.

A. Initial seeding and distribution of aircraft

The search process essentially uses randomisation of simulation input data to explore the solution space. Our allocation of aircraft to flight paths is intended to represent two busy flight paths vertically separated (the upper flight path containing CPLoss), with potentially conflicting low frequency traffic also entering the sector on a more or less random basis.

The distribution of aircraft between each flight path is decided at random, with the sole restriction that the ns flight path must have at least one aircraft on it to represent CPLoss. The proportion of traffic between the ns and ew flight paths and the lower level paths is split 75:25 in favour of the ns and ew paths. The 25% of traffic on the low level flight paths is allocated one of the r2, r3 or r4 flight paths at random. The entry times of the aircraft are then generated. Aircraft may receive an entry time at any second between the upper and lower bounds; however they may not keep their allotted entry time if they are so close to another aircraft on the same flight path that they may suffer from wake turbulence (i.e. they are less than 120 seconds apart), in which case the aircraft is moved further back in time to enforce sufficient separation.

This can have the effect of pushing some aircraft beyond the time limit of the simulation, in effect preventing any further mutation of their entry times. However, it is rare for aircraft whose entry time is towards the end of the simulation to have much impact on conflicts within the sector. If several aircraft are bunched together, then all those aircraft will be spaced the minimum separation apart. This type of traffic configuration and subsequent separation impinges on the search's ability to explore the start times of aircraft, even though tightly grouped aircraft can represent a hazard as they leave fewer options for the resolution of conflicts.

1) *Mutation of traffic inputs*

The ASHiCS search harness uses a population of 100 scenarios per generation. By selecting a proportion of these and mutating the entry times of the aircraft, we carry out what sometimes termed a 'near neighbour, random hill-climber' search [3]. However, in order to ensure that the search has not been unlucky in its initial seeding of random samples, we continue to allow a proportion of each population to be generated by random sampling. The split in the population is generated by the selection policy; in which the top twenty per cent of the population's scenarios are copied to create three mutants, with each mutant being carried over to the subsequent

generation to see if its fitness improves. The remaining forty per cent of the population is created from new random samples.

Just as in natural evolution, evolutionary search is also generally considered to be a process of gradual optimization, but the reasons for this lie more towards the suspicion that if large mutations are allowed, then the evolutionary search is performing a crude form of *localized random search*, rather than employing the gradual improvement process that evolution and natural selection is famous for. As random search does not perform better than a "brute force" or exhaustive search, we wanted to ensure that the mutation operator was not able to radically change a scenario in a way that made little sense from an evolutionary perspective.

In order to affect this gradual increase in the fitness of a scenario, the easiest method was to alter the aircraft entry times within a fixed range (generally within a few minutes of the previous entry time). Provided such mutations are not radical, we should be guaranteed that a "near neighbour" of the original scenario is created, as all aircraft remain on their flight paths and relatively close to their original entry times.

2) *Evolutionary strategy*

The evolutionary strategy for ASHiCS is based primarily on mutation rather than crossover or other combination methods. Our rationale for not selecting crossover is supported by several studies that suggest that 'destructive' methods for good gene propagation fare less well than methods that allow gradual changes to a phenotype's fitness, with the proviso that this is likely to be dependent on problem type [4].

We earlier outlined some of reasons that the range for the mutation operator should be restricted in order to effect gradual change, and the same reasoning applies to destructive techniques such as crossover for our domain. In pragmatic terms, if we imagine how an ATM planner would investigate peak traffic flows around a certain point in time, we can imagine them looking at how small changes to a particular aircraft's entry times might make a difference with respect to possible conflicts. Our mutation operator replicates this type of manual search, rather than permit investigating variations of a scenario by moving an aircraft or group of aircraft to another flight path. Such a radical change in a scenario configuration would create an entirely new scenario that bore little resemblance to the original, in effect representing a random jump in the search space rather than the evolution of a previous configuration.

This same argument applies against the use of a crossover operator for scenario chromosomes. Let's say we crossed over two flight path configurations between a pair of scenarios – it is hard to argue that the new scenarios would bear much relationship to the previous ones. It would have the effect not only of radically altering the entry times along the entire flight path, it would also mean we would have to somehow re-balance the numbers of aircraft distributed on the other flight paths. This is not to say that a crossover operator would not discover good scenarios, but given its destructive nature for our type of problem domain we believe it is unlikely it would outperform random sampling.

3) *Population size, crossover and mutation*

Population size is much contested variable that can affect performance and studies exist on the effect of large and small population sizes on search performance from earliest days of evolutionary computation. Much early evolutionary search theory believed that population size had a direct relationship to the amount of information available to the search:

…when the population size is too small for the complexity of a particular search space … it lacks the information capacity to provide accurate sampling (see [5] *for a discussion of population size requirements).* [6]

Proponents of this belief began the era of throwing "heavy metal" at search problems with huge population sizes, using racks of very powerful computers. Koza for example, using GP with crossover operators, regularly used population sizes exceeding 200,000 [7]. It is now more widely recognised that a destructive means of combining chromosomes (such as crossover) requires larger sample sizes to work effectively, whereas mutation can work with smaller populations:

…crossover needs large populations to effectively combine the necessary information, [whereas] mutation works best when applied to small populations during a large number of generations [8]

More recent studies have suggested that the type of problem domain is also related to whether one chooses to use mutation or crossover, and conversely the size of population for each generation. As explained in Section II.A.1), we believe our domain would be particularly sensitive to a destructive type operator, as we feel it could neither produce "near neighbour" hill-climbing to allow gradual improvements from an initial starting point, nor would it be able to trace how a scenario could evolve from one that was relatively low risk to one that contained gradually higher levels of risk[2]. The jumps in the search trajectory from a crossover operator are largely arbitrary.

III. Characterisation of Solution Space

A. Total size of search space

We suspected that our search space was very large even before we tried to calculate the input permutations.[3] Large search spaces are often termed "high dimensionality" problems, i.e. those in which mutation can act on a large number of variables to affect the fitness outcome of individuals. High dimensionality continues to cause the search community considerable difficulty, particularly when trying to demonstrate that an optimal solution has been found in a given

[2] We have not attempted to incorporate traceability into our search harness as yet, but the choice of a non-destructive operator allows us to keep this as an option for future work.

[3] The size of the search space containing all possible permutations would make an exhaustive search impossible (see appendix of the E.02.05-ASHiCS-D2.2-Method Description Technical Report).

search space [9] [10]. For most real-world problems, this sort of "proof" is impossible to achieve, and most practitioners are content to discover a solution which is good enough or better than a previous design.

However from a safety perspective it is important to quantify risk to levels that are deemed acceptable to the regulatory authorities. Using search to discover unexpected risk or hazards is more effective if we are able to say something about the nature of the solution space, i.e. have we found an isolated example of extremely high levels of risk, or are there many such examples out there? What are the average levels of risk and what is the likelihood that a hazardous scenario could develop from a situation that would normally be judged to be safe? These questions are hard to answer unless a quantitative description of the solution space can be given.

B. Reducing the size of the search space

As we became aware of the total size of the search space, it became clear that we might need to reduce the size of the search to more manageable proportions. The quickest, most pragmatic approach to this is to apply domain knowledge to the input variables, often termed 'dimension reduction', so that the search has less work to do. However, dimension reduction runs the risk that you cut out part of the search space that might be profitable to explore, particularly if one is searching for rare events such as near collision. We wanted to find a systematic, evidence-based approach to reducing the overall size of the search space that would provide some guarantee that our search would still cover the areas where high risk scenarios could be found.

1) *What proportion of the total search space is worth searching?*

Our assessment of risk in a given scenario focuses on the principal safety barrier in our model (maintaining separation between aircraft) and how that may be degraded by several factors which we attempt to measure. This combination of risk measures, several of which are weighted to guide the search towards certain outcomes, means that there are many scenarios whose input configurations are unlikely to be of interest to us.

Let us say we have weighted our fitness function to reward those scenarios containing conflicts directly related to CPLoss. If we are only interested in the workload or risk associated with CPLoss in a scenario, it is easy to see that any aircraft that passes out of the sector *prior to CPLoss's entry* cannot impact the controller workload associated with CPLoss. Likewise aircraft that arrive in the sector *after CPLoss has left* cannot come into conflict with CPLoss. So it is clear that there is a relationship between the entry times of aircraft on certain flight paths and the fitness score.

For example, the aircraft on the ew flight path are the same type (DH8), so they travel at the same speed. The same is true for A320 aircraft on the ns flight path. Therefore if a group of aircraft that are bunched up close together on the ew flight path enter the sector at a particular time with respect to the aircraft chosen to be CPLoss, there is a high chance that CPLoss may be in conflict with one of them as it makes its emergency descent.

A similar line of reasoning can be applied to the emergency descent. CPLoss is free to make its descent at any of the navigational aids on the ns flight path. However, only a few of them will force its descent through the ew flight path and close to other flight paths. Similarly if the event is triggered at a navigational aid on the edge of the sector, it is unlikely any aircraft will conflict with CPLoss. It seems reasonable to assume therefore that there is a relationship between the navigational aids and fitness score for the scenario. The following sections look at how we might be able to uncover this relationship to allow some dimension reduction.

C. Determining the nature of the search landscape

Our initial searches suggested that the evolutionary search was not out-performing the random sector of the population. By tagging the scenario with its original index, we could investigate each jump in fitness and track it to see its lineage. Throughout Stage 1 and the early part of Stage 2 it was often the case that even after hundreds of generations, a random scenario could appear late in the run and outscore the previous evolved best. The evolutionary changes appeared as gradual improvements in fitness (as you would expect from a mutation operator). However, large (and unexpected) jumps nearly always came from the 40% of scenarios who bore no relation to the selected part of the population. In order to try and establish what was going on and to learn more about the search landscape, we decided to conduct a sensitivity analysis.

1) *Sensitivity Analysis*

This form of analysis generally takes the fitness score of the near neighbours that are sampled as part of the hill-climbing algorithm. By keeping a record of these, we find out how destructive or beneficial our mutations are to each near neighbour of the original scenario. We had noticed that the search often reached long plateaus of high fitness (when the best of the previous generation is passed on unchanged to the subsequent generation), during which time it appeared that no mutations of high scoring scenarios were able to improve their fitness score. By conducting a sensitivity analysis, we could directly compare different mutation rates and see how the mutations were affecting the average.

If mutations show a significant drop in fitness, it suggests that the original scenario is on or near a peak of high fitness. However, if the summit is very narrow, and the sides of the peak are steep, then the drop off in fitness is rapid and most mutations will fail to find the "sharp" point of highest fitness. How would this appear in a sensitivity analysis? If we look at two typical plots showing a wide variance in the mutation range value, we can see a marked difference not only in the performance of the search, but also in the effect of mutating copies of the best scenarios.[4]

[4] We ran over 30 evolutionary search runs on Stage 2 over two months. Here we show the progression of typical runs for the stated parameters.

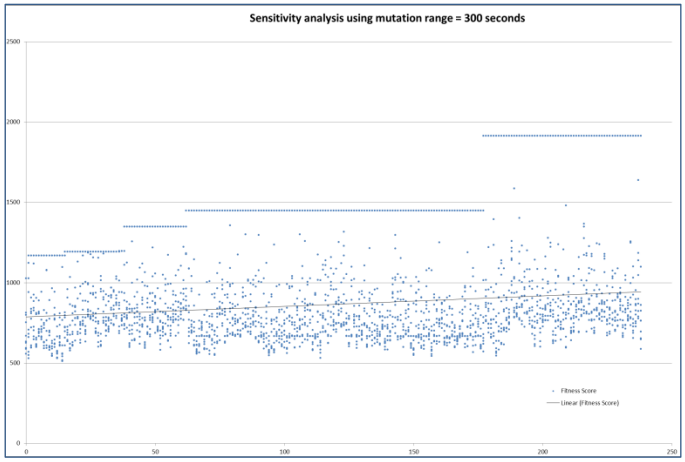

Figure 1: Sensitivity analysis showing fitness of top ten per generation. Mutation range = 300 seconds.

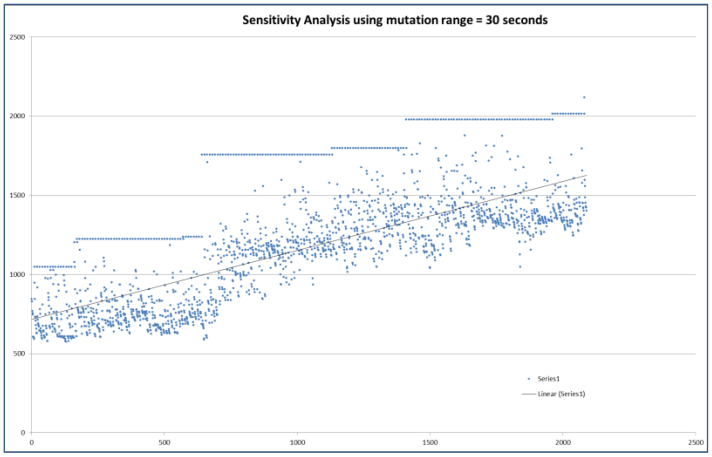

Figure 2: Sensitivity analysis showing fitness of top ten per generation. Mutation range = 30 seconds.

Figure 1 shows a sensitivity analysis that plots the fitness values (y axis) of the top ten scenarios of each generation (x axis for this and all following plots) containing a population of 100 scenarios. Bear in mind that the top ten of each generation will contain three copies each of the previous generation's top three scenarios (see Section 2).

As the evolutionary run progress, we can see plateaus of no improvement for the best scenarios becoming longer (these runs were stopped before 250 generations). At the beginning of the run, the distance between the fitness scores that the best of a generation and its mutated copies achieve are reasonably close together. As the run progress and higher fitness scores are achieved by selected scenarios, the gap between the mutated copies and the best individual starts to widen. Once the highest level of fitness is achieved, the gap between the mutated copies and the best individual is so wide it seems barely worth continuing the search, as every mutation seems to radically worsen the fitness score. This type of sensitivity pattern suggests the search landscape is composed of tall, narrow spikes, in which a large mutation is likely to mean the individual is placed beyond the small area of high fitness occupied by the original scenario. The most likely explanation for this is that the near neighbourhood determined by a mutation range of 300 seconds is too large, making it difficult for a random mutation to fall within the range of values that will ensure an improvement.

When we compare this to Figure 2 that shows exactly the same search parameters and risk measures, but with a much reduced mutation range of 30 seconds, it is immediately obvious that the smaller mutation range has produced a much less destructive effect on the fitness of near neighbours. The overall performance is improved, with a higher final fitness score for the best scenario (which had improved gradually over many generations) but also with a better average fitness across the population. The gap between the best and its near neighbours' fitness scores indicates that the Stage 2 solution space is highly sensitive to mutation rates.

Figure 1 and Figure 2 provide us with an intuition of the search space; in that we can see small mutations enable the heuristics to improve the scenario's fitness scores, whereas large mutations appear destructive. But how can we be sure that the best scenarios are being "evolved" and not chosen from the randomly generated part of the population? By tagging the previous index of selected scenarios, we can track their roots after they enter the top ten of a particular generation. If the search is working correctly and effectively, we would expect the proportion of successful scenarios coming from the random sector of the population to decrease over time, as evolved scenarios compete for fitness. If they don't, it suggests that the search is not working, or at least it is not outperforming the random part of the population.

Figure 3 and Figure 4 show the lineage of the top ten from each generation, using the same data as Figure 1 and Figure 2. As before, the scatter plots show generations on the x axis, with the lineage or index of the scenario on the y axis. Any scenario that comes from an index of over 60 (measured on the y-axis) has come from the random part of the population.

We can see that in the case of the poorly performing, large mutation rate, the random part of the population continues to supply almost equal numbers of scenarios in the top ten of that generation. Over generations there is little drop off in their numbers, which indicates the search is both selecting scenarios from the random part of the population and failing to improve on them by mutation. Figure 4 shows the same type of plot but for the small mutation range. It shows how over time the search increasingly selects from mutated scenarios, while the contribution from the random part of the population falls off rapidly as the search progress and tails off to almost zero in the final stages. By using both the sensitivity analysis and tracing the lineage of the scenarios, we can be confident that our evolutionary search outperforms random search and see evidence that supports our decision to use a non-destructive mutation operator. However, while we can make qualitative assessments of the solutions discovered by looking at how sensitive the solution space is to mutation rates, we cannot give

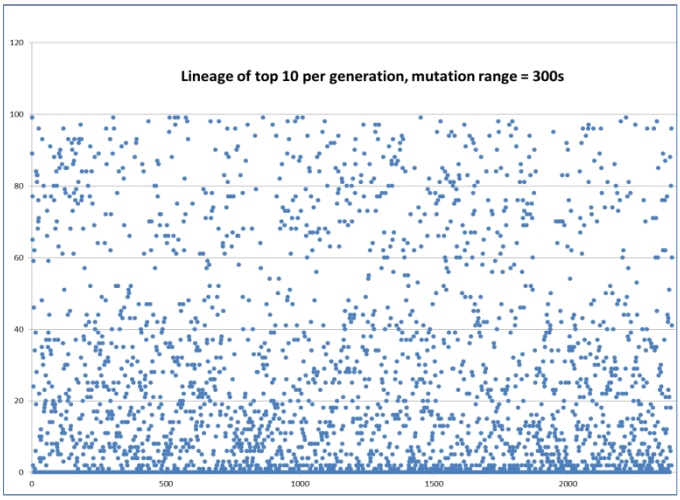

Figure 3: Lineage of top 10 per generation with mutation range at 300s. Individuals with an index (y-axis) greater 60 were created from random sampling

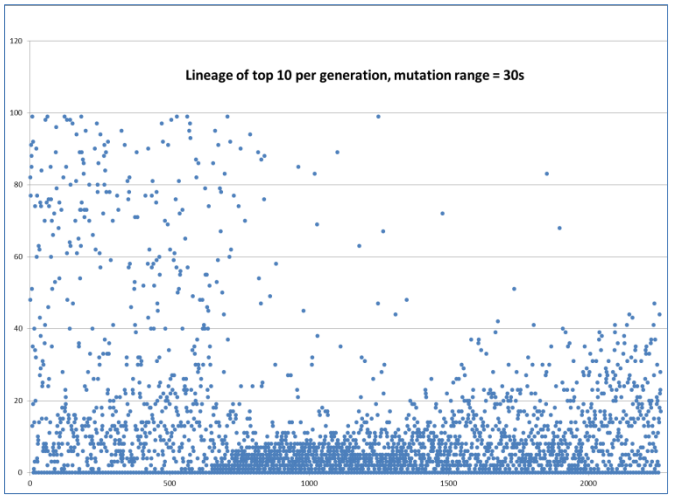

Figure 4: Lineage of top 10 per generation with mutation range at 30s. Individuals with an index (y-axis) greater 60 were created from random sampling.

any form of quantitative assessment of the area we are searching. That is to say we cannot say whether there are many possible solutions of a similar nature or very few.

This is not usually an issue for research into search heuristics; however in the case of searching for risk or safety related factors, it is important as the cost of dealing with risk (usually through the implementation of safety barriers) is often worked out by determining the cost of the outcome multiplied by the frequency of the event occurring. As we are searching for rare events, it would be of interest to not only discover instances of these, but to gain an approximate idea of their frequency for a given configuration of air space.

IV. Analysis of High Risk Scenarios

This section describes our on-going efforts to see whether we can find patterns in the input data or the output logs of the high risk scenarios that could help us to reduce size of the search space. As each search run takes a long time to complete, it has been hard to compile a set of experimental data that could provide us with a statistical measure of certainty with regards to the type of solution most often found.[5] We decided given the limited time and resources available, that we would use a large random sample to collect statistical information. This allowed us to set up a dedicated machine that would conduct extensive random samples of the solution space that could then be harvested for analysis. Our hope is that this information might lead us to a method which could reduce the size and complexity of the solution space.

After discussion with a colleague who specialises in search techniques (Dr Simon Poulding, York), we decided that although random sampling could not represent a viable search method given the size of our solution space, there should be some limit to how long the sampling process should run (other than one simply determined by size). With this in mind, we set an upper limit of a million random samples taken from the Stage 2 solution space. This took about 165 hours to complete (approximately one week).

1) *Random search results*

As expected with such a large solution space, random sampling failed to discover any scenarios approaching the best evolved solutions. However, the best results found by random sampling were within the fitness values achieved by evolutionary search after about 150 generations (15,000 simulations), so we considered these results to be indicative of solutions that could be subsequently mutated to achieve higher fitness scores. By assessing these medium to high fitness scenarios we hoped to uncover any patterns in the input data that would indicate if the scenario is likely to generate high levels of risk.

Intuitively, one would assume there must be some type of linear relationship between the input variables that leads to high levels of risk. Even accepting that different aircraft travel at different speeds, and that the resolution algorithm used by RAMS Plus is unknown to us, it would seem reasonable to assume that the high fitness scenarios can to some extent be predicted from their traffic inputs.

For example, we know that that only one aircraft from the ns flight path will be CPLoss. So the fewer aircraft that are on that flight path, the fewer options there are for CPLoss to be selected. We also know that the main flight path that could conflict with CPLoss is the ew flight path, so again if there are few aircraft on that flight path the less chance there is of conflict. However, we don't know what the optimum distribution of aircraft between the two flight paths should be for a high level of risk.

[5] For example, to get meaningful statistics on the performance of a given parameter setting, we would need to run at least 20 (overnight) searches. Unfortunately there are many potential parameter settings we could explore, so a line has to be drawn on whether to experiment with new settings or to stick with a setting so that a reasonable number of results can be obtained.

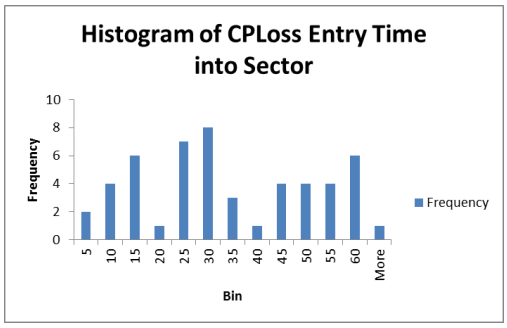

Figure 5: Frequency of CPLoss's entry times within 5 min bin ranges from random samples having a fitness score of over 800.

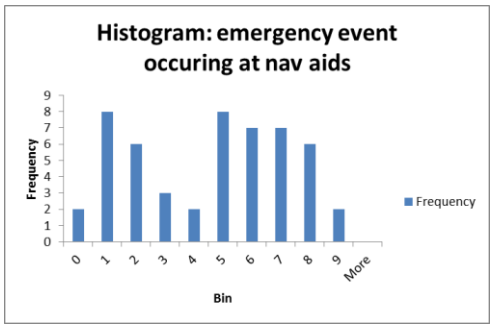

Figure 6: Frequency of high risk scenarios having CPLoss emergency event occurring at a particular navigational aid. The x-axis shows the index of the nav aid along the ns flight path.

We can also make similar assumptions about when CPLoss enters the scenario. If it is too early, then there are very few aircraft for it to come into conflict with, likewise if it is very late. We would expect scenarios with either of these configurations to have low risk levels with respect to conflicts involving CPLoss. There is a similar argument with regard to the navigational aid chosen as the point at which the emergency cabin pressure loss incident occurs. Finally, if CPLoss descends towards a tightly grouped bunch of aircraft on the ew flight path, it would seem reasonable to assume there is an increased risk of conflict with one of the aircraft within the group. Based on these intuitive assumptions from our domain knowledge, we might expect a pattern to emerge from high fitness scenarios that show:

- CPLoss enters the sector outside the very earliest or latest times;
- CPLoss has the emergency pressure loss event close to the ew flight path;
- CPLoss descends towards a group of aircraft on the ew flight path.

Our random sample had just 10 scenarios with fitness scores over 1000, so it would be relatively simple to do a manual analysis of the input files for these 10 scenarios and look for any correlation between them with respect to the points above. However, such a small sample might be misrepresentative. By choosing all scenarios that score over 800, we raise that number to 51 and hopefully get a clearer picture. We must emphasise at this point that we are not attempting to carry out an exhaustive statistical analysis; we are just manually looking for any obvious patterns.

The first place to look is to see if CPLoss's entry times are significant. Using our sample of 51, we can create a histogram of the entry times for CPLoss using a bin size of 5 minutes. The results are shown in Figure 5. We can see that there immediately appears to be very little correlation with CPLoss's entry time to high fitness scores. While we can see a small drop off in the frequency of high risk scenarios which contain CPLoss entering at either end of the simulation, there is otherwise no obvious discernible pattern to emerge. The next step is to see whether a better correlation appears when we look at *where* the CPLoss event occurs. This is coarsely determined according to the position of the 10 navigational aids that make up the ns flight path. Again, as with the entry times, we might expect fewer conflicts are caused if the emergency event occurs very early or late in the simulation. However, as Figure 6 shows there is a similar, weak pattern of lower frequency at either end of the simulation and again in the centre which respect to navigational aid triggers the emergency event.

As a further test, it is a relatively simple matter to run the entry times for CPLoss and navigational aid indices with the associated fitness score through the statistical software "R" to see if there is any correlation:

```
> cor(d$Fitness.Score, d$Index.of.Nav.Aid)
[1] -0.02506953
> cor(d$Fitness.Score, d$Entry.time.of.CPLoss)
[1] 0.1085984[1] -0.02506953
> cor(d$Fitness.Score, d$Entry.time.of.CPLoss)
[1] 0.1085984
```

The figures also suggest a weak correlation between these input variables and the scenario fitness score. These manual analyses suggest that the most probable pattern remaining is the distance in time that separates CPLoss from other aircraft in the scenario and whether aircraft are grouped together on the ew or random flight paths. However it is important to keep in mind the aim of looking for such patterns, which is to reduce the search space either by reducing its dimensionality or reducing the parameter ranges we search across. If we can determine that most high risk scenarios are related to particular groupings of aircraft or the relative entry time of aircraft on different flight paths, then we have no need to generate such large variations of input data. We can focus the search across a tight spectrum of inputs, allowing greater coverage and more efficient searches.

2) *Linear modelling on adjusted inputs*

By looking at the entry times of aircraft on all or selected flight paths and comparing those times to CPLoss's entry time, we can see whether aircraft that enter the sector within a certain time frame relative to CPLoss have an impact on fitness scores. If there is a correlation between them, then we can look to see if the patterns of inputs can be traced to a likelihood of conflicts involving CPLoss. This type of analysis is only possible after the event, i.e. after selecting which aircraft will

represent CPLoss. Prior to the event, we can only say that certain traffic input patterns represent possible risk should one of them incur the emergency cabin pressure loss event.

This time we used an extended selection from the best of the random sample, which resulted in a set of medium to high fitness scenarios (131 scenarios in total), and looked at their entry times relative to CPLoss. We can separate these into separate flight paths and look to see the entry times relative to CPLoss has any correlation to that scenario's fitness score. This transformation was done for 131 input files and the resulting table analysed using the R statistical package to see if either entry time relative to CPLoss or flight path (or both) are correlated to a scenario's fitness score. Unfortunately, this manipulation of the data is not sufficient to show the effect of aircraft "bunching together" on a flight path, which may be why they showed only a weak correlation when we tried to use linear modelling to fit these two factors against the fitness scores of each scenario. To date, we have not managed to show a correlation using the best fitness scores from random sampling run. However, we believe there may be other candidate techniques out there (such as principal components analysis) which will help us to reduce the dimensionality of the search space. Another possibility we are investigating is to use a collection of the best evolved solutions to see if these will provide clearer results.

V. SUGGESTED EXPANSION OF TECHNIQUE TO SYSTEMS OF SYSTEMS (SOS)

ASHiCS is intended as a proof-of-concept approach to the automated discovery of hazards in complex systems. We believe the work we have to date shows that heuristic search is capable of finding high risk scenarios in such systems, and we see no reason why it could not be extended to cover large SoS. As the method of using heuristics to target the search towards those areas of risk that interest us would remain the same, the main question to answer is whether the search could continue to perform in an increased size of search space. Evolutionary search in this domain has already demonstrated its capability to find solutions better than those than can be found using random search. Our calculations indicate that the search space typical for this domain is extremely large. What is interesting is that evolutionary search always appears to discover "solutions" better than those found even in large random samples. This suggests that either:

- the search performs exceptionally well in very large search spaces;
- the search will find some "risk" in almost any part of the search space and can then manipulate the input variables to increase this risk.

We believe that although both of these possibilities are an acceptable outcome for the project's method, it is the second which we think will make the technique useful in a pragmatic fashion for the domain of ATM.

VI. CONCLUSIONS

In this deliverable we have described the heuristic search used by ASHiCS in the search for high levels of risk in our Stage 2 scenarios. We have provided arguments that the use of destructive operators such crossover as opposed to mutation is unlikely to be effective, as small mutation ranges were shown to be far more productive than destructive mutations using a larger range. We have shown that evolved solutions are significantly better than any found using random sampling and that these solutions are found in a much reduced time.

We have attempted to find out quantitative information about the solution space by conducting a large random sample and using dimension reduction techniques on the inputs to see if we could characterise the solution space and therefore target the search more effectively. While we believe this technique holds promise, to date we have been unable to show a correlation between the input variables that would allow us to use it to recognise traffic patterns that indicate scenarios containing high risk. We believe that the techniques such as linear regression modelling can uncover patterns in the input data that would allow effective dimension reduction, something we believe would pay dividends on the much larger search spaces that a SoS domain would present.

VII. BIBLIOGRAPHY

[1] S. Alam, W. Zhao, J. Tang, C. Lokan, H. Abbass, M. Ellejmi and S. Kirby, "Discovering Delay Patterns in Arrival Traffic with Dynamic Continuous Descent Approaches using Co-Evolutionary Red Teaming," in *9th ATM Seminar*, Berlin, 2011.

[2] S. Alam, C. Lokan and H. Abbass, "What can make an airspace unsafe? characterizing collision risk using multi-objective optimization," in *IEEE Congress on Evolutionary Computation (CEC), 2012*, 2012.

[3] S. Luke, Essentials of Metaheuristics, Available online at http://cs.gmu.edu/~sean/book/metaheuristics/, 2009.

[4] D. White and S. Poulding, "A rigorous evaluation of crossover and mutation in genetic programming," in *12th European Conference, EuroGP 2009*, Tübingen, Germany, 2009.

[5] D. Goldberg, Genetic algorithms in search, optimization, and machine learning, Addison-Wesley, 1989, pp. Goldberg, D. (1989). Genetic algorithms in search, optimization, and machine learning. Addison-Wesley.

[6] K. De Jong and W. Spears, "An analysis of the interacting roles of population size and crossover in genetic algorithms," in *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, 1991, pp. 38-47.

[7] J. R. Koza, M. Keane and M. Streeter, "Evolving inventions.," *Scientific American*, p. 52–59., 2003.

[8] C. F. Lima, D. E. Goldberg, K. Sastry and F. G. Lobo, "Combining competent crossover and mutation operators: A probabilistic model building approach," in *Proceedings of the 2005 conference on Genetic and evolutionary computation (GECCO '05)*, New York, 2005.

[9] K. Beyer, J. Goldstein, R. Ramakrishnan and U. Shaft, "When Is "Nearest Neighbor" Meaningful?," in *Database Theory — ICDT'99: Lecture Notes in Computer Science*, Berlin , Springer , 1999, pp. 217-235.

[10] P. Merz and B. Freisleben, "On the effectiveness of evolutionary search in high-dimensional NK-landscapes," in *Evolutionary Computation Proceedings, IEEE World Congress on Computational Intelligence*, 1998.

[11] S. Alam, H. A. Abbass and M. Barlow, "ATOMS: Air Traffic Operations and Management Simulator," *IEEE Transactions on intelligent transportation systems*, vol. 9, no. 2, June 2008.